

MULTI-MODEL SEMI-SUPERVISED LEARNING FOR PERSONALISATION

Božidara Cvetković

Doctoral Dissertation
Jožef Stefan International Postgraduate School
Ljubljana, Slovenia

Supervisor: Dr. Mitja Luštrek, Jožef Stefan Institute, Ljubljana, Slovenia

Co-Supervisor: Prof. Dr. Matjaž Gams, Jožef Stefan Institute, Ljubljana, Slovenia

Evaluation Board:

Asst. Prof. Bernard Ženko, Chair, Jožef Stefan Institute, Ljubljana, Slovenia

Prof. Dr. Zoran Bosnić, Member, Faculty of Computer and Information Science, Ljubljana, Slovenia

Asst. Prof. Stefano Bromuri, Member, Open University of Heerlen, Heerlen, The Netherlands

MEDNARODNA PODIPLOMSKA ŠOLA JOŽEFA STEFANA
JOŽEF STEFAN INTERNATIONAL POSTGRADUATE SCHOOL



Božidara Cvetković

MULTI-MODEL SEMI-SUPERVISED
LEARNING FOR PERSONALISATION

Doctoral Dissertation

POLNADZOROVANO STROJNO UČENJE
Z VEČ MODELI ZA PRILAGAJANJE UPORABNIKU

Doktorska disertacija

Supervisor: Dr. Mitja Luštrek

Co-Supervisor: Prof. Dr. Matjaž Gams

Ljubljana, Slovenia, March 2018

Dedicated to my family who gave me unconditional support and were a constant source of inspiration, wisdom and understanding.

Posvečeno moji družini, ki mi je nudila brezpogojno podporo ter bila stalni vir navdiha, modrosti in razumevanja.

Acknowledgments

First and foremost I want to thank my supervisor Dr. Mitja Luštrek for his contributions in the form of ideas, discussions, reviews of my work and his constructive critique. I would also like to thank my co-supervisor Prof. Dr. Matjaž Gams for many challenges which contributed to my personal and professional growth. I would like to express my sincere gratitude to the members of the evaluation board Asst. Prof. Bernard Ženko, Prof. Dr. Zoran Bosnić and Asst. Prof. Stefano Bromuri for their contributed time in revising this thesis and for constructive comments that made this dissertation better.

I would like to thank all my academic and technical colleagues at the Department of Intelligent Systems. In particular, I would like to thank Domen Zupančič and Violeta Mirchevska for the words of encouragement and Boštjan Kaluža for all academic and non-academic advice. A special thanks are extended to the professors and colleagues at the Department of Knowledge Technologies and at the Department of Computer Systems for their advice and a debate during this process.

There were quite a few people that were supporting me during this journey. I thank all of you for your patience in the last few months in which I cancelled my social life. Your support and encouragement were worth more than I can express on paper. I am especially grateful to Sonja Zlobko for almost lifelong unconditional support, Rok Pečičnik for very much needed distractions and for making even the most casual walk in the park an adventure, and Dejan Radunić who was there when I started this journey and is happily cheering with a bottle of wine at the finish line. I am thankful to Petra Ždrade for being very supportive, and Maša Malovrh for sincere critique and last-minute support.

Last but not least, I would like to thank my parents, my sister, her husband and my awesome nephew Filip for supporting me spiritually throughout writing this thesis and my life in general. Special acknowledgement for keeping my spirits high goes to Archibald.

I acknowledge financial support by the Department of The Intelligent Systems, Jožef Stefan Institute and the Ministry of Education, Science and Sport of the Republic of Slovenia for covering tuition fees.

Abstract

In this dissertation, we propose a novel semi-supervised learning method called Multi-Model Adaptive Training for adaptation and personalisation of machine-learning models. The method is designed to efficiently exploit large amounts of unlabelled data generated by wearable devices while they are in use.

Systems that utilise machine learning are usually deployed with a single or multiple machine-learning models trained with datasets collected in a controlled environment and explicitly labelled for their task. Once such models are used in real life, they may perform well if the statistical properties of the data are stable and similar to the training dataset. If the statistical properties of the data vary or drift, the models might diminish in accuracy since the collected dataset on which they were trained may not include all possible variations. The solution is either to collect and hand-label data containing new statistical properties and train the model in a supervised manner, which is time-consuming and expensive, or to use a methodology that can exploit unlabelled data. The latter approach is cheaper since an overwhelming amount of unlabelled data is generated while such systems are in use. To enable this data to be used for personalisation it needs to be labelled either by a human annotator when required, via active learning, or automatically, via semi-supervised learning (SSL). The latter is the subject of this dissertation, in which we propose a method that builds on the idea of probabilistic methods called Self-training from the SSL low-density separation methods group.

The idea of Self-training methods is to train a model or an ensemble of models in a supervised manner and use them to label unlabelled examples. The decision whether or not an automatically labelled example is included in the training set used for retraining is done with heuristics, which are typically designed to include only the most confident predictions to avoid a potential decrease in accuracy. However, using only highly confident predictions in retraining may not improve the performance compared to the initial model, since the included examples already fit it well.

The proposed method increases the potential for improvement after retraining by also including examples labelled with a higher uncertainty, which is achieved by replacing the heuristics with a machine-learning model. Each unlabelled example is first labelled with two domain models, a general model trained on data of other people than the target person and a person-specific model trained on a small amount of data specific for the target person. The results of the domain models are stacked with a selection model that selects which domain model should provide the final labels for the example. The results of the domain models and the selection model are additionally processed with an inclusion model trained to decide whether the automatically labelled examples should be included in the training set for retraining.

We evaluate the personalisation using the proposed method on three classification tasks and one regression task. We analyse each step of the method and compare it against other SSL approaches. We achieve successful personalisation and show that the proposed method outperforms competing approaches by up to 10 percentage points on average.

Povzetek

Doktorska disertacija predstavlja novo metodo polnadzorovanega učenja, ki je zasnovana tako, da učinkovito izkorišča velike količine neoznačenih podatkov, ki jih ustvarjajo nosilne naprave med uporabo, za prilagajanje in personalizacijo modelov strojnega učenja.

Sistemi, ki uporabljajo strojno učenje, so ponavadi opremljeni z enim ali več modeli strojnega učenja. Ti so naučeni na namensko zbrani in označeni učni množici v nadzorovanem okolju. Takšni modeli lahko uspešno opravljajo svojo nalogo, če so statistične lastnosti v uporabljenem okolju stabilne. V primeru, da se statistične lastnosti podatkov pogosto spreminjajo, pa se lahko zgodi, da modeli začnejo delovati z zmanjšano točnostjo. Razlog je v vsebini učne množice, saj je skoraj nemogoče zbrati takšno učno množico, ki bi vsebovala vse možne oblike podatkov. Točnost lahko popravimo tako, da zberemo in ročno označimo nove podatke ter prilagodimo model s ponovnim učenjem na nadzorovan način. Takšen postopek je zelo dolgotrajen in drag, saj mora vse podatke označiti človek. Alternativen način je uporaba metodologije, ki zna izkoristiti neoznačene podatke, kakršni se dnevno generirajo v velikih količinah. Slednji način je cenejši, a če želimo, da bi ti podatki postali uporabni za prilagajanje in personalizacijo, jih je vendar treba nekako označiti. To lahko storimo tako, da za oznake človeka prosimo po potrebi (aktivno učenje), ali pa uporabimo metodo, ki jih zna označiti samodejno (polnadzorovano). Slednji način je tudi glavna tema doktorske disertacije, v kateri predstavimo novo metodo, ki temelji na idejah polnadzorovanih metod, imenovanih *samo-učeče metode* (Self-training).

V samo-učečih metodah na nadzorovan način naučimo en model ali ansambel modelov ter jih uporabimo za označevanje neoznačenih primerov. Vsak samo-označen primer je ovrednoten s heuristiko, ki določi, ali je primer označen z zadostnim zaupanjem in je s tem primeren za uporabo pri prilagoditvi modela. Pomankljivost tega pristopa je, da so kot primerni navadno določeni le tisti primeri, ki dosežejo visoko zaupanje oznake. Uporaba samo teh primerov lahko vpliva na to, da model ni bistveno izboljššan v primerjavi z začetnim modelom, saj visoko zaupanje oznake kaže na to, da primer že ustreza modelu.

Predlagana metoda poveča možnost izboljšanja, ker pohitri prilagajanje tako, da poleg primerov z visokim zaupanjem oznake vključi tudi tiste z večjo negotovostjo. To dosežemo tako, da heuristiko nadomestimo z modelom strojnega učenja. Vsak neoznačen primer se označi z dvema domenskima modeloma, s splošnim modelom, ki je naučen na podatkih, ki ne vsebujejo podatkov končne osebe, in s specifičnim modelom, ki je naučen na majhni količini podatkov končne osebe. Napovedi obeh modelov ovrednoti izbirni model, ki določi, kateri model bo primeru dodelil končno oznako. Vključitveni model ovrednoti samo-označeni primer in odloči, ali je ta primeren za vključitev v učno množico splošnega modela, ki ga prilagajamo.

Predlagana metoda je preizkušena in ovrednotena na treh klasifikacijskih in enem regresijskem problemu. Analizirali smo prispevek vsakega koraka metode k personalizaciji ter metodo primerjali z najbolj tipičnimi metodami polnadzorovanega učenja. Rezultati kažejo, da dosežemo uspešno personalizacijo in da je naša metoda uspešnejša glede na druge metode tudi do 10 odstotkov.

Contents

List of Figures	xv
List of Tables	xvii
List of Algorithms	xix
Abbreviations	xxi
Symbols	xxiii
1 Introduction	1
1.1 Motivation	1
1.2 Problem Definition	3
1.3 Purpose and Goals of the Dissertation	5
1.4 Hypotheses	5
1.5 Scientific Contributions	6
1.6 Organisation of the Dissertation	8
2 Background	11
2.1 Personalisation	11
2.2 Semi-Supervised Learning	12
2.2.1 Generative Models	13
2.2.2 Graph-Based Methods	14
2.2.3 Low-Density Separation	15
2.2.4 Self-Training Methods	16
2.3 SSL for Real-time Personalisation	18
2.4 Summary	20
3 Multi-Model Adaptive Training	21
3.1 General Framework of Multi-Model Adaptive Training	21
3.2 Implemented Architecture	23
3.2.1 Datasets	25
3.2.2 Domain and Decision Models	26
3.2.2.1 Domain Models	26
3.2.2.2 Decision Models	27
3.2.3 Training Procedure	29
3.2.4 Adaptation procedure	33
3.3 Summary	33
4 Evaluation	35
4.1 Use Case 1: Low-Level Activity Recognition Task – Classification	35
4.1.1 Related Publication	35

4.1.2	Dataset	36
4.1.3	Training Procedure	36
4.1.3.1	Data Partitioning	36
4.1.3.2	Machine-Learning Model Training	36
4.1.4	Adaptation Procedure	36
4.1.5	Summary	37
4.2	Use Case 2: Context Recognition Task – Classification	53
4.2.1	Related Publication	53
4.2.2	Dataset	53
4.2.3	Training Procedure	53
4.2.3.1	Data partitioning	53
4.2.3.2	Machine-Learning Model Training	53
4.2.4	Adaptation Procedure	54
4.2.5	Summary	54
4.3	Use Case 3: Human Energy Expenditure Estimation - Regression	63
4.3.1	Related Publication	63
4.3.2	Dataset	81
4.3.3	Training Procedure	81
4.3.3.1	Data Partitioning	81
4.3.3.2	Machine-Learning Model Training	83
4.3.4	Adaptation Procedure	86
4.3.5	Summary	89
4.4	Use Case 4: Identification Task – Classification	90
4.4.1	Dataset	91
4.4.2	Training procedure	92
4.4.2.1	Data partitioning	92
4.4.2.2	Machine-Learning Model Training	93
4.4.3	Adaptation Procedure	100
4.4.3.1	Single-Person Adaptation	100
4.4.3.2	Two-People Adaptation	105
4.4.4	Summary	109
5	Conclusions and Further Work	111
5.1	Scientific Contributions	111
5.1.1	Design and Implementation of a Novel SSL Method Called Multi-Model Adaptive Training	111
5.1.2	Analysis of the Multi-Model Adaptive Training	111
5.1.3	Demonstration and Evaluation of the Applicability	112
5.2	Discussion	113
5.2.1	Critical Aspects of The Method	113
5.2.2	Conditions Under Which the Method Operates	114
5.3	Further Work	114
	References	117
	Bibliography	123
	Biography	127

List of Figures

Figure 1.1:	Types of concept drifts and a new concept.	2
Figure 2.1:	Self-training methods according to their properties	17
Figure 3.1:	Machine-learning pipeline enhanced with MMAT.	22
Figure 3.2:	Multi-Model Adaptive Training architecture.	23
Figure 3.3:	Dataset partitioning requirements.	26
Figure 3.4:	Data for training the person-specific and general model.	30
Figure 3.5:	Data partitioning for training the selection and inclusion model.	30
Figure 4.1:	Number of examples per person and per type of recognised activity. . .	82
Figure 4.2:	Person data divided for training and evaluation of the models	82
Figure 4.3:	Retraining the general model with correctly labelled examples in EEE. . .	84
Figure 4.4:	Selection model decisions. EEE adaptation in batch mode.	87
Figure 4.5:	Inclusion and labelling rate. EEE adaptation in batch mode.	87
Figure 4.6:	Selection model decisions. EEE adaptation in incremental mode.	88
Figure 4.7:	Inclusion and labelling rate. EEE adaptation in incremental mode. . . .	88
Figure 4.8:	Number of examples in the dataset per person and per type of walking. .	92
Figure 4.9:	Person data divided into datasets for training and evaluation.	93
Figure 4.10:	Retraining the general model with correctly labelled examples.	95
Figure 4.11:	Selection and inclusion model training for single-person adaptation. . .	97
Figure 4.12:	Selection and inclusion model training for two-person adaptation. . . .	98
Figure 4.13:	Selection model decisions. Single-person in batch mode.	101
Figure 4.14:	Inclusion and labelling rate. Single-person in batch mode.	101
Figure 4.15:	Selection model decisions. Single-person adaptation in incremental mode.	101
Figure 4.16:	Inclusion and labelling rate. Single-person in incremental mode.	102
Figure 4.17:	General model accuracy. Single-person adaptation in incremental model.	103
Figure 4.18:	The first two components of the PCA for included and discarded examples.	104
Figure 4.19:	Selection model decisions. Two-people adaptation in batch mode. . . .	106
Figure 4.20:	Inclusion and labelling rate. Two-people adaptation in the batch mode.	106
Figure 4.21:	Selection model decisions. Two-people adaptation in incremental mode.	107
Figure 4.22:	Inclusion and labelling rate. Two-people adaptation in incremental mode.	108

List of Tables

Table 4.1:	Energy expenditure dataset in minutes and number of examples	81
Table 4.2:	Level of intensity according to MET value range.	85
Table 4.3:	Evaluation of ML algorithms for selection model in EEE.	85
Table 4.4:	Evaluation of ML algorithms for training the inclusion model in EEE.	85
Table 4.5:	Feature set per decision model in EEE.	86
Table 4.6:	Evaluation of the condition value for training the inclusion model.	86
Table 4.7:	EEE adaptation results.	89
Table 4.8:	Identification dataset in minutes and number of examples	92
Table 4.9:	General model ML algorithm selection.	94
Table 4.10:	Person-specific model ML algorithm selection in single-person adaptation.	96
Table 4.11:	Person-specific model ML algorithm selection in two-people adaptation.	96
Table 4.12:	Selection model ML algorithm selection in identification task.	97
Table 4.13:	Inclusion model ML algorithm selection in identification task.	99
Table 4.14:	Feature set per decision model in identification task.	99
Table 4.15:	Accuracy results in single-person adaptation.	103
Table 4.16:	Statistical significance in single-person adaptation.	105
Table 4.17:	Accuracy results for two-people adaptation.	108
Table 4.18:	Statistical significance for two-people adaptation.	109

List of Algorithms

Algorithm 3.1:	Multi-Model Adaptive Training algorithm.	24
Algorithm 3.2:	Procedure for extraction of features for selection model.	28
Algorithm 3.3:	Selection model training procedure – classification tasks.	31
Algorithm 3.4:	Inclusion model training procedure – classification tasks.	32

Abbreviations

AL	... Active Learning
AR	... Activity Recognition
CART	... Decision Tree
DSM	... Data Stream Mining
EEE	... Energy Expenditure Estimation
EM	... Expectation Maximisation
G	... General Model
GPS	... Global Positioning System
I	... Inclusion Model
LOF	... Local Outlier Factor Algorithm
LogReg	... Logistic Regression
LOPO	... Leave-One-Person-Out Evaluation
LOSO	... Leave-One-Set-Out Evaluation
MAE	... Mean Absolute Error
MAPE	... Mean Absolute Percentage Error
MCAT	... Multi-Classifer Adaptive Training
MET	... Metabolic Equivalent of Task
ML	... Machine Learning
MLE	... Maximum Likelihood Estimate
MLP	... Multi-Layer Perceptron
MMAT	... Multi-Model Adaptive Training
MRAT	... Multi-Regressor Adaptive Training
MTR	... Multi-Target Regression
OneClassSVM	... One Class Support Vector Machine
P	... Person-Specific Model
PCT	... Predictive Clustering Trees
RF	... Random Forest
RS	... Recommender Systems
S	... Selection Models
SSL	... Semi-Supervised Learning
SVM	... Support Vector machine
SVM ^{light}	... Light Support Vector machine
TSVM	... Transductive Support Vector Machine

Symbols

$Dataset_{\text{general}}$	... dataset for training the general model
$Dataset_{\text{person}}$	... dataset for training the person-specific model
$Dataset_{\text{selection}}$	... dataset for training the selection model
$Dataset_{\text{inclusion}}$	... dataset for training the inclusion model
$Dataset_{\text{evaluation}}$	... dataset for evaluation
$Dataset_{\text{unlabelled}}$	... dataset for adaptation
$Dataset_{\text{generalTemp}}$	... temporary dataset for training the general model
$Dataset_{\text{selectionTemp}}$	... temporary dataset for training the selection model
$Dataset_{\text{inclusionTemp}}$	... temporary dataset for training the inclusion model
$Data_{\text{general}}$	... array of $Dataset_{\text{general}}$ for current P_{training}
$Data_{\text{person}}$	... array of $Dataset_{\text{person}}$ for current P_{training}
$Data_{\text{selection}}$	... array of $Dataset_{\text{selection}}$ for current P_{training}
$Data_{\text{inclusion}}$	... array of $Dataset_{\text{inclusion}}$ for current P_{training}
$label_G$	... label as predicted by the general model
$label_P$	... label as predicted by the person-specific model
$label_S$	... label as predicted by the selection model
$label_I$	... label as predicted by the inclusion model
C_G	... prediction confidence of the general model
C_P	... prediction confidence of the person-specific model
C_S	... prediction confidence of the selection model
$Basic$	... feature defining the labels known by the person-specific model
$Equal$	... feature defining the equality of the predicted labels
$Activity$	... feature for the recognised activity
$Intensity$	... contextual feature for the intensity of the recognised activity
p_{target}	... current target person
p_{temp}	... temporary target person
P_{all}	... set of all people in the dataset
P_{training}	... set of people used in the training procedure
x_i	... example
y_i	... label of the example x_i
X	... set of examples
Y	... set of labels

Chapter 1

Introduction

We are experiencing a flood of artificial intelligence-driven applications composed of single or multiple machine-learning models used for various purposes. The accelerated deployment of such applications is facilitated by research in artificial intelligence and by the development of accessible devices with high computational capabilities. This enables computationally demanding problems to be solved faster, and real-time solutions to be deployed on devices accessible to average users, such as a smartphone. Since smartphones are personal devices, the solutions running on them should be personalised, which is currently not the case. They are usually deployed with a machine-learning model trained in a controlled laboratory environment to fit an average user, namely a general model. Examples of such solutions are low-level activity recognition and estimation of energy expenditure, which are already deployed in most common smartphones. A deployed activity recognition is limited to recognising three to four low-level activities, which are assumed to be the same for everybody (walking, running, rest, and cycling). Energy expenditure estimation is one step closer to personalisation since it requires the user to enter his/her weight to calculate burned calories using accelerometer counts (Crouter, Churilla, & Bassett, 2006). To provide more fine-grained and informative results, one cannot use only a general model, but rather must personalise the solution, desirably in real time.

This chapter first introduces the problem addressed in the dissertation, explains which machine-learning approach was chosen, and defines purposes, goals, and hypotheses. Then, we list the scientific contributions and conclude with the overview of the dissertation structure.

1.1 Motivation

Machine-learning models to be deployed in a real-life environment are usually trained in a supervised manner in a laboratory or real-life controlled environment with good-quality data specifically collected and labelled for the task. Once models are trained and positively evaluated, they are deployed in a target environment to solve tasks they are designed for. If a task has constant parameters and the statistical properties of the data (for example, quality control of products in the industry, robot assembling different parts, etc.), the trained model contains knowledge about all possible states important for the task and can stay as it is until parameters or statistical properties change. Once the change occurs, a new labelled dataset is collected, and the model retrained or incrementally updated in a supervised manner. If the task is diverse and the statistical properties of the data vary and often change (for example, computer vision tasks, monitoring human behaviour, etc.), the model trained in the laboratory environment can face decreased accuracy immediately after the deployment in the real-life environment. This occurs due to differences between

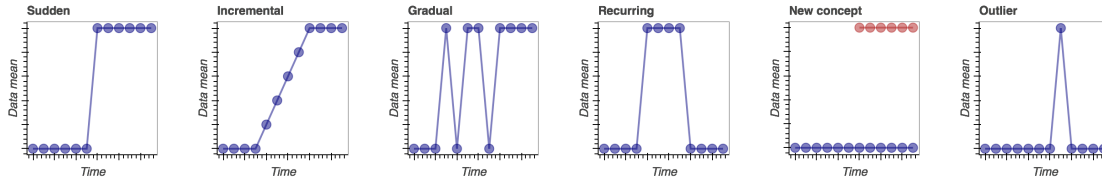


Figure 1.1: Types of concept drifts, introduction of a new concept and an outlier, which is not considered a concept drift (João Gama, Žliobaitė, Bifet, Pechenizkiy, & Bouchachia, 2014).

the knowledge about the task represented in the dataset collected for training the model in the laboratory environment, and the specifics of the target task in the real-life environment since it is not possible to collect all forms of the task.

Data stream mining (DSM) is a field of research focused on processing large amounts of continuous data from single or multiple data streams in real-time (Joao Gama, 2010; Ramírez-Gallego, Krawczyk, García, Woźniak, & Herrera, 2017). It can experience the decrease in accuracy due to similar events as introduced before. Briefly, the general task of DSM is to extract knowledge from the data streams and utilise the data for predictions or estimations in real-time using a machine-learning technique. As in any dynamic and nonstationary environment, the distribution of data may change over time resulting in an event called concept drift or concept evolution which leads to a decreased accuracy of the DSM task. The decrease in accuracy is usually handled by first detecting the change and eventually labelling it, and second incrementally adapting the used model to the new statistical properties of the concept in a supervised manner (João Gama, Žliobaitė, Bifet, Pechenizkiy, & Bouchachia, 2014; Haque, Khan, Baron, Thuraisingham, & Aggarwal, 2016). The categorisation of the types of change in DSM fits well with the changes that occur in person-centred systems we are tackling in this research. Figure 1.1 presents the four types of concept drifts, an introduction of a new concept or a concept evolution and an outlier which is not considered a concept drift. The first two types occur when the statistical properties of the data for the individual concept changes and stays: the sudden concept drift happens abruptly, and the incremental slowly over time. The next two types occur when the statistical properties of the data alternate from changed to baseline properties: the gradual concept drift alternates randomly, and the recurring alternates into one of the states for an extended period. The fifth graph presents a concept evolution or an introduction of a new concept which occurs once a stream contains data with entirely new properties. The last graph presents an outlier event that should not be considered a concept drift. If such events occur in the person-centric system, we will need the ability to collect the data representing the new statistical properties and use it to adapt the deployed model to the changes. Even though we do not address all the types, we adopt the names and the nature of the change in statistical properties of the concept.

A massive amount of unlabelled data that is generated daily enables the data containing new statistical properties to be collected freely. For example, smartphones, wearables, and environmental sensors continuously sense and generate data; speech is sensed while talking to mobile assistants (Siri, Google assistant, Alexa, etc.), a large amount of new content on websites is generated daily, etc. Even though this data is unlabelled, it gives a perfect opportunity to be used for the adaptation of machine-learning models since it does represent the current statistical properties of the task in the environment in which the models are deployed. To enable this data to be used for adaptation purposes, a human annotator is usually employed to hand-label the collected unlabelled data. This approach

usually results in perfectly labelled data; however, it is slow, labour-intensive, and expensive. Less-expensive approaches are called active learning (AL) (Settles, 2012) and semi-supervised learning (SSL) (Chapelle, Schölkopf, & Zien, 2006). In AL, an algorithm is employed to select only those data examples which seem the most informative for later machine-learning model retraining. The selected data are sent to the human annotator for labelling (domains are limited to human understandable data, for example, images, video, speech). In SSL, a human annotator is asked to label a small amount of data, which are used for training the machine-learning model that can automatically label the rest of the generated unlabelled data. The SSL approach is by far the cheapest way to retrieve labels for unseen data and is the focus of this dissertation because even though the idea and need for automatic labelling emerged decades ago (Scudder (1965), Vapnik and Chervonenkis (1974)), the area of SSL is still not mature enough to be generally applied.

We will focus and explore personalisation algorithms with an emphasis on the SSL technique. We will explore the possibilities and limitations of using SSL technique for the task. We will design an SSL algorithm that can efficiently perform the personalisation of machine-learning models in the person-centric environments.

1.2 Problem Definition

The problem of decreased machine-learning model accuracy outside of the laboratory environment can occur in any domain. If the system in which the model's accuracy is decreased generates large amounts of unlabelled data, it provides an opportunity for adaptation. In artificial intelligence applications that learn and reason about human behaviour, one must take into consideration each person's individuality, and since it is impossible to collect labelled data for each person the model is deployed for, automatic personalisation using unlabelled data is needed. One of the main challenges in the development of person-centric models is that we can expect different types of concept drift immediately after deployment and occasional introduction of a new concept.

We focus on person-centric environments in this dissertation in which we deploy an initial general model trained on data of people that are not the target person. Once the user starts using the system in which the general model is deployed, the method gradually personalises it and adapts it to the target person. The models are trained either to predict a categorical value, namely classification or to predict a continuous value, namely regression. We tackle two types of problems, examples of which are briefly presented below. The types are: (i) Change of statistical properties of labels due to a concept drift; and (ii) The introduction of a new label. First three examples address the change in statistical properties and the last example addresses both the introduction of a new class and the change in statistical properties.

Basic activity recognition (AR) is a very popular classification task in person-centric systems using wearable devices. Regardless of what sensors one uses for solving this task, it is an example in which we can expect a sudden concept drift (presented in Section 1.1). If localisation sensors are used for AR (sensors sense the coordinates in three dimensions), the height of the person represents a significant parameter in relation to the performed activity. In this case, the sudden concept drift might manifest as follows: a shorter person's standing activity might be similar to a taller person's sitting; if the target person is lying in a sitting position, this might be misclassified as sitting; a taller person's walking might be similar to a shorter person's running, etc. If inertial sensors are used for the AR, the sudden concept drift manifests as follows: the impact of the walking activity of a tall person might be similar to the impact of running of a shorter person, or the walking activity of the short person might not have a sufficient magnitude that it can easily be misclassified as standing

and moving a bit. In both cases, some personalisation of the model is needed.

A more complex classification example is the recognition of a person’s context in terms of whether they are eating, at home, at work, exercising, out doing errands, out for leisure etc., from smartphone sensor data (for example, accelerometer, gyroscope, sound, light, GPS, WIFI, etc.). The model trained in a laboratory can recognise general patterns of each context. The location, time, habits, and another person’s specifics play an important role in recognising all contexts accurately. In this particular task, one context may occur in multiple guises (multiple work locations, shops, etc.), so, in addition to the sudden concept drift, we are also faced changes that have a nature of a recurring concept drift.

Problems solved by regression often need personalisation, too. For example, a very popular topic in person-centric systems is the estimation of human energy expenditure (burned calories while performing the physical activity). The general rule for solving this task is that the expended energy is correlated with the person’s body mass index, age, heart rate, and accelerometer signal while performing activities. However, the metabolic system differs in people of the same age and body mass index group; thus, personalisation of the model is beneficial for more accurate estimations.

The introduction of a new class is addressed in the identification task in which we aim at identifying a person from his or her walking signature. Each person has a different signature walk, but since there are several types of walking (normal, fast, uphill, carrying a burden, Nordic walk, etc.), the identification model may be confused about the identification prediction. The goal is to personalise the identification model so it can identify the person during different types of walking. The model starts without any knowledge about the new person (new label). Once the new person starts using the model, the model detects a new person and uses the walking data for adapting itself. The change in the walking type of a person to be identified is a change of the labels’ statistical properties and has a nature of recurring concept drift (presented in Section 1.1).

Introduced tasks could be adequately personalised in a supervised machine-learning manner by labelling enough person-specific data and training a person-specific model. This approach is, apart from being labour intensive and expensive, sometimes even impossible. It may be hard to design a scenario in which the collection of all events is feasible and safe (elderly person running, falling, etc.). It is also assumed as intrusive and unacceptable to video record a persons’ lifestyle, so the recording can be used later to label otherwise not understandable data. Personalisation using SSL seems an excellent approach, considering that it utilises unlabelled data and the amount of unlabelled person-specific data generated while using such systems.

There are three general categories of SSL methods: generative models, graph-based methods, and low-density separation methods. The former two were successfully used in many domains, but they have some important drawbacks described in Section 2. Low-density separation methods contain a group of probabilistic methods called self-training methods which we adopt as suitable for personalisation. Briefly, the self-training methods evaluate automatically labelled examples in real time and use label probability as a prediction confidence measure to decide on the inclusion of each automatically labelled example in the dataset used for retraining the model. The drawback of these methods is that the inclusion criterion is usually set in such a way that only the most confident predictions are included in the training set, which may cause the retrained model not to improve over the initial model since the examples already fit the model correctly.

In this dissertation, we aim to design an SSL method that fits the idea of self-training. The designed method should automatically label unseen data and, when appropriate, use it to retrain a general machine-learning model, thus adapting and personalising it. The method needs to allow personalisation in real time on unseen data. We explore SSL

approaches and define the steps and design the building blocks needed for successful adaptation. We develop a methodology for the selection of the final label and the inclusion of the newly labelled example in the training dataset. This is achieved by designing and training two machine-learning models, one for the selection of the label, namely the selection model, and one for the decision on the inclusion of the newly labelled example, namely the inclusion model. The task of the selection model is to choose the model which will provide the final label. It chooses between the general model, trained on the general dataset that contains data of people that are not the target person, and the person-specific model, trained on a few examples of the target person. The results of the domain models and the selection model are evaluated by the inclusion model with a task to decide whether the newly labelled example is suitable for inclusion. These examples are added into the general dataset, and the general model is retrained, thus gradually adapted. With this methodology, we include examples with higher prediction uncertainty in addition to examples with highly confident prediction, thus introducing diversity to the model, which will result in a more efficient adaptation.

1.3 Purpose and Goals of the Dissertation

The purpose of this dissertation is to solve the problem of decreased machine-learning model accuracy when deployed in real life by exploiting a large amount of unlabelled data that are generated daily while using a machine-learning enhanced system.

To achieve this, we will investigate if and how SSL can be used for the task and design an SSL method that enables automatic labelling of unlabelled data produced by wearable sensors, thus decreasing the need for human annotation. The SSL method will utilise the automatically labelled data for personalisation of the machine-learning models

The goals are:

Goal 1. *Design a novel self-training SSL method for real-time adaptation and personalisation of models.*

Goal 2. *Analyse different aspects of designing the structure of a self-training SSL method in terms of the number of machine-learning models, as well as the selection and inclusion procedures.*

Goal 3. *Empirically evaluate the designed method on classification and regression domain where personalisation is required.*

Goal 4. *Compare the proposed method against other personalisation and SSL methods that can operate in real time.*

1.4 Hypotheses

The hypotheses of the dissertation are:

Hypothesis 1. *SSL can efficiently exploit unlabelled data generated by wearable sensors for personalisation of machine-learning models.*

Hypothesis 2. *Employing an additional model in a self-training method to aggregate the outputs of the introduced domain models for the selection of the final label improves the performance, as opposed to the use of a simple heuristic.*

Hypothesis 3. *Employing an additional model in a self-training method for the decision on the inclusion of self-labelled examples enables inclusion of more diverse examples into the dataset used for retraining the general domain model. This enables efficient adaptation and personalisation in comparison to the heuristic approach that includes only the examples with high labelling confidence.*

1.5 Scientific Contributions

Contribution 1. *Design and implementation of a novel SSL method called Multi-Model Adaptive Training for personalisation of models.*

The contribution is composed of:

- Architecture of the method
- Design of the models composing the method
- Training procedure for the individual models

The work related to the development of the method architecture, training procedure, and reference implementation is covered in Chapter 3 and the journal paper included in Section 4.1. The work was published in the following publications:

Journal Article

Cvetković, B., Kaluža, B., Gams, M., & Luštrek, M. (2015). Adapting activity recognition to a person with Multi-Classifer Adaptive Training. *Journal of Ambient Intelligence and Smart Environments*, 7(2), 171–185.

Conference Papers

Cvetković, B. & Luštrek, M. (2014). Analiza možnosti zaznavanja podobnosti med uporabniki. In *Intelligent systems : Proceedings of the 17th International Multiconference Information Society - IS 2014*. Ljubljana, Slovenia: Jožef Stefan Insitute.

Cvetković, B., Kaluža, B., Luštrek, M., & Gams, M. (2012). Multi-Classifer Adaptive Training: Specialising an activity recognition classifier using semi-supervised learning. In F. Paternò, B. de Ruyter, P. Markopoulos, C. Santoro, E. van Loenen, & K. Luyten (Eds.), *Ambient intelligence* (pp. 193–207). Springer Berlin Heidelberg.

Cvetković, B., Luštrek, M., Kaluža, B., & Gams, M. (2011). Semi-supervised learning for adaptation of human activity recognition classifier to the user. In *IJCAI Workshop on Space, Time and Ambient Intelligence*.

Contribution 2. *Analysis of the Multi-Model Adaptive Training.*

The contribution is composed of:

- Evaluation of all individual models composing the method and evaluation of the contribution of each model to the final performance of the method
- Analysis of the selection model output in successive iterations of adaptation
- Analysis of the inclusion model performance in terms of the number of included and discarded examples in successive iterations of adaptation

The work related to the evaluation of individual models, contribution of each model to the performance of the method, as well as the analysis of the models, is covered in Chapter 4 and in journal papers included in Section 4.1 and Section 4.2. The work was published in the following publications:

Journal Article

- Cvetković, B., Janko, V., Romero, A. E., Kafali, O., Stathis, K., & Luštrek, M. (2016). Activity recognition for diabetic patients using a smartphone. *Journal of Medical Systems*, 40(12), 256.
- Cvetković, B., Kaluža, B., Gams, M., & Luštrek, M. (2015). Adapting activity recognition to a person with Multi-Classifer Adaptive Training. *Journal of Ambient Intelligence and Smart Environments*, 7(2), 171–185.

Contribution 3. *Demonstration and evaluation of the applicability of the method on three classification domains and one regression domain.*

The case studies are:

- Classification case study 1: Personalisation of a low-level activity recognition classification model using localisation sensor data
- Classification case study 2: Personalisation of a context recognition classification model utilising data fusion of multiple smartphone sensors
- Regression case study 1: Personalisation of a human energy expenditure estimation regression model utilising physiological and accelerometer sensor data
- Classification case study 3: Personalisation of a classification model for identifying people from their walking signature using smartphone accelerometer sensor data

The contribution is composed of:

- Development, training, and evaluation of the best-performing domain model per case study in a supervised manner
- Evaluation of the Multi-Model Adaptive Training (Multi-Classifer Adaptive Training - MCAT) for personalisation of the classification model in the classification case studies
- Evaluation of the Multi-Model Adaptive Training (Multi-Regressor Adaptive Training - MRAT) for personalisation of the regression model in the regression case study
- Comparison of the Multi-Model Adaptive Training (Multi-Classifer Adaptive Training - MCAT) against most common personalisation and self-training SSL approaches.

The work related to the development, training, and evaluation of the best-performing domain models in a supervised manner and their evaluation using the Multi-Model Adaptive Training is covered in Chapter 4 and in the journal paper included in Section 4.3. The work was published in the following publications:

Journal Article

- Cvetković, B., Szeklicki, R., Janko, V., Lutomski, P., & Luštrek, M. (2017). Real-time activity monitoring with a wristband and a smartphone. *Information Fusion*. In press. doi:10.1016/j.inffus.2017.05.004
- Cvetković, B., Janko, V., Romero, A. E., Kafali, O., Stathis, K., & Luštrek, M. (2016). Activity recognition for diabetic patients using a smartphone. *Journal of Medical Systems*, 40(12), 256.
- Cvetković, B., Milić, R., & Luštrek, M. (2016). Estimating energy expenditure with multiple models using different wearable sensors. *IEEE Journal of Biomedical and Health Informatics*, 20(4), 1081–1087.
- Cvetković, B., Kaluža, B., Gams, M., & Luštrek, M. (2015). Adapting activity recognition to a person with Multi-Classifer Adaptive Training. *Journal of Ambient Intelligence and Smart Environments*, 7(2), 171–185.
- Kaluža, B., Cvetković, B., Dovgan, E., Gjoreski, H., Gams, M., Luštrek, M., & Mirchevska, V. (2014). A multi-agent care system to support independent living. *International Journal on Artificial Intelligence Tools*, 23(01), 1–30.

Conference Papers

- Cvetković, B., Janko, V., & Luštrek, M. (2015). Demo abstract: Activity recognition and human energy expenditure estimation with a smartphone. In *13th IEEE International Conference on Pervasive Computing and Communication, PerCom Workshops 2015* (pp. 193–195).
- Cvetković, B., Mirchevska, V., Janko, V., & Luštrek, M. (2015a). Recognition of high-level activities with a smartphone. In *Adjunct Proceedings of the 2015 ACM International Joint Conference on Pervasive and Ubiquitous Computing and Proceedings of the 2015 ACM International Symposium on Wearable Computers, UbiComp/ISWC'15 Adjunct* (pp. 1453–1461). Osaka, Japan: ACM.
- Cvetković, B. & Luštrek, M. (2014). Analiza možnosti zaznavanja podobnosti med uporabniki. In *Intelligent systems : Proceedings of the 17th International Multiconference Information Society - IS 2014*. Ljubljana, Slovenia: Jožef Stefan Insitute.
- Cvetković, B., Kaluža, B., Luštrek, M., & Gams, M. (2012). Multi-Classifer Adaptive Training: Specialising an activity recognition classifier using semi-supervised learning. In F. Paternò, B. de Ruyter, P. Markopoulos, C. Santoro, E. van Loenen, & K. Luyten (Eds.), *Ambient intelligence* (pp. 193–207). Springer Berlin Heidelberg.
- Cvetković, B., Luštrek, M., Kaluža, B., & Gams, M. (2011). Semi-supervised learning for adaptation of human activity recognition classifier to the user. In *IJCAI Workshop on Space, Time and Ambient Intelligence*.

1.6 Organisation of the Dissertation

The dissertation is organised as follows. Chapter 1 presents the shortcomings of models trained in a supervised manner when used in a dynamic environment and places the dissertation in the context of SSL with a problem definition. We declare the purpose and goals of the dissertation as well as the hypotheses and scientific contributions.

Chapter 2 further elaborates the problem and introduces the background and the related work on personalisation as is usually performed. It introduces the SSL approaches,

their usage and their shortcomings. It also identifies the steps to be taken to develop an SSL approach suitable for personalisation.

Chapter 3 introduces the proposed SSL method called Multi-Model Adaptive Training. First, it describes the architecture in general and the purpose of each model in the method. Second, it describes the models in detail in terms of the type of dataset and the set of features used for training. Third, it introduces an algorithm that divides the labelled dataset into folds and uses them for training the models: the training algorithm is designed with special attention to fully exploiting the data and to dividing the data into non-overlapping folds, thus providing fair training circumstances.

Chapter 4 presents the evaluation of the personalisation using Multi-Model Adaptive Training on four domains—three classification and one regression. The chapter features three journal papers. Two papers cover the adaptation of classification models in two classification domains and one paper presents a starting point for the adaptation of the regression domain. The results are also compared against simple personalisation techniques and other SSL approaches.

Chapter 4 concludes the dissertation with a summary and discussion. The chapter also points out possible improvements and directions for further research.

Chapter 2

Background

This chapter presents the most common approaches to personalisation of machine-learning models in Section 2.1 and introduces the SSL in Section 2.2. Special attention is given to self-training, that is, a group of SSL methods that are closely related to the method proposed in this dissertation. We finally define the challenges to be tackled to design an SSL method to be used for efficient personalisation.

2.1 Personalisation

Personalisation is very important in the field of recommender systems (RS) and some of the ideas are also used in SSL, so we briefly introduce it. The personalisation in RS is divided into personalisation on a group and individual levels with the goal of providing the most suitable advice to a person, usually in the form of ranked items. The most common approach to group-level personalisation is collaborative filtering (Koren & Bell, 2015), which groups together users with similar parameters (interests, demography, etc.) using an unsupervised machine-learning technique with the rationale that similar users are interested in or need similar items. The approach to individual personalisation is called content-based filtering (Lops, de Gemmis, & Semeraro, 2011), which learns from past explicit interests of a particular user to estimate the interest for a new, unseen item.

The approaches to personalise machine-learning models in systems that use data from wearable sensors differ according to how the target person’s labelled data (person-specific data) is used.

The basic approach is to train the model only with the person-specific data or to include person-specific data in the general training dataset that contains data of other people and use the merged dataset for training the model (Weiss & Lockhart, 2012; Medrano, Plaza, Igual, Sánchez, & Castro, 2016). Some personalisation can also be done by utilising contextual and anthropomorphic person-specific features for the normalisation of important parameters (Altini, Casale, Penders, & Amft, 2015).

Advanced approaches are often done either offline by boosting the general model with the person-specific data (Bleser et al., 2015; Zinnen, Blanke, & Schiele, 2009), or online by having the person label important events as they occur, after which the algorithm uses this labelled data to update and personalise the general model (Parkka, Cluitmans, & Ermes, 2010). Reiss and Stricker (2013) proposed two approaches, in which they trained an ensemble of models on a general dataset and obtained the final label by the majority vote. In the first approach, the person-specific data were used for adapting the weights in a majority vote for each model in the ensemble. Each model in the ensemble labelled the example from the person-specific dataset. If it was labelled correctly, the model’s weight stayed the same, and if it was mislabelled it was decreased by half. In the second approach,

the person-specific data were evaluated by each model in the ensemble to find a matrix of weights, where each weight corresponded to one model per majority vote class. Both approaches resulted in an unchanged ensemble of models with personalised weights in the majority vote. They also explored how the amount of labelled data used for adapting and calculating the weights affects the adaptation accuracy of the model. An important conclusion was that in general, a larger amount of labelled data is better, but a significant improvement is also achieved with a limited amount of data, which indicates that a system that uses a small amount of labelled data for adaptation is feasible.

Maekawa and Watanabe (2011) explored the possibility of personalising a model in an unsupervised manner without any person-specific data, similar to collaborative filtering in RS. They explored a general training dataset to find similar users to a new user according to physical characteristics, with only the resulting data employed to train the model. This approach is the cheapest way to personalise models; however, to use this approach efficiently, the general dataset from which the training examples are chosen would need to be fairly large to include people with diverse characteristics, which is difficult.

Huynh and Schiele (2006) showed that a small amount of labelled data from wearable sensors can be used to label unlabelled data. In their approach, the unlabelled data are firstly clustered using a multiple eigenspaces approach. Next, they used a Support Vector Machine (SVM) model trained on a small amount of labelled examples to label the clustered data. The reported accuracy of their approach is comparable to or even higher than that when a model is trained on all data in a supervised manner. Jänicke, Sick, Lukowicz, and Bannach (2014) used a similar idea for model self-healing that occurs in environments in which the sensor type is often changed. The task was to adapt the machine-learning model to work with new sensor type data.

These approaches show that a small amount of data labelled specifically for the person or a task can improve a general model. Supervised approaches use person-specific data to train a new model, boost or normalise a general model, or adjust the weight per model in the majority vote procedure in case an ensemble of models is used. The unsupervised approaches build on the idea of similarity between users and use physical characteristics to find a subset of similar users whose data is used for training a new model. The last two approaches utilise both labelled and unlabelled data, which is a common ground with SSL methods, and achieve better results regarding classification accuracy after personalisation.

The purpose of this dissertation is to use unlabelled data for real-time retraining and adaptation of machine-learning models trained on general datasets. We explore the SSL methodology that deals with automatic labelling of unseen examples which can be used for retraining the machine-learning models.

2.2 Semi-Supervised Learning

Learning techniques can be categorised according to labelled data availability into supervised learning (Definition 2.1) and unsupervised learning (Definition 2.2). SSL occupies the middle ground between supervised and unsupervised learning (Definition 2.3), and deals with learning from unlabelled data using a small amount of labelled data. The idea of learning from unlabelled data appeared in the mid-60s (Scudder, 1965) with the idea of self-labelling and took off in the 70s, first with transductive inference (Vapnik & Chervonenkis, 1974) and later with estimating Fisher’s linear discriminant rule with unlabelled data and a mixture of Gaussian distributions (Hosmer, 1973), as well as with a mixture of the multinomial distribution.

Definition 2.1 (Supervised learning). *Supervised learning* requires labelled data from which it infers a function that maps input to output. The training set is given in pairs

(x_i, y_i) , where x_i is an example ($x_i \in X$) and y_i is its label ($y_i \in Y$). When Y contains a finite set of discrete values, the task is called classification, and when Y contains continuous numeric values, the task is called regression.

Definition 2.2 (Unsupervised learning). *Unsupervised learning* requires unlabelled data, which are assumed to be drawn from identical distribution on X . Unsupervised learning tasks infer a hidden structure of the unlabelled data. The training set is composed from examples ($x_i \in X$) from which the structure of X is inferred.

Definition 2.3 (Semi-supervised learning). *Semi-supervised learning* learning requires labelled data ($X_l \subseteq X$, for which Y_l is known) and unlabelled data ($X_u \subseteq X$, for which Y_u is not known). The SSL utilises all data to infer a function $f : X \rightarrow Y$ in transductive (Definition 2.4) or inductive setting (Definition 2.5).

Definition 2.4 (Transductive setting). In a *transductive setting*, an SSL method uses all given data (X_l, Y_l and X_u) to train a function $f : X_{l+u} \rightarrow Y_{l+u}$ to give good predictions of Y_u for unlabelled data X_u .

Definition 2.5 (Inductive setting). In an *inductive setting*, an SSL method uses all given data (X_l, Y_l and X_u) to train a function $f : X \rightarrow Y$ to give good predictions of Y for any unlabelled example $x \in X$, beyond X_{l+u} .

The goal of SSL is to extract useful information from X_u to make efficient inference on $p(y|x)$. If useful information is not extracted, the inference might not achieve improvement over supervised learning and may even degrade the accuracy with erroneous inference. To make semi-supervised learning work, at least one of the following assumptions must hold for the data (Chapelle et al., 2006):

- The smoothness assumption — if we take into consideration the density of the inputs, then the two points x_1, x_2 in a high-density region that are close have corresponding labels y_1, y_2 that are also close. If two points x_1, x_2 are separated by a low-density region, the labels should differ.
- The cluster assumption — we assume that points of each class form a cluster and if points x_1, x_2 are in the same cluster, they are likely to have the same labels $y_1 = y_2$.
- The manifold assumption — high-dimensional input data lie on a manifold of lower dimension. In this case, the dimensionality can be decreased, and the training can be done on the low-dimensional manifold.

SSL approaches can be divided into three general categories (Chapelle et al., 2006; Zhu & Goldberg, 2009): a) Generative models, b) Graph-based methods, and c) Low-density separation methods. In the following subsections, we briefly introduce the categories with an emphasis on self-training approaches from the low-density separation category, since they are closely related to the method proposed in this dissertation.

2.2.1 Generative Models

Generative models are mostly used in transductive setting and assume that a given dataset has an identifiable class distribution, on which class-conditional density $p(x|y)$ of X_u can be estimated. A predictive density $p(y|x)$ specifies the likelihood of each class label for a given example, which can be calculated with a probability rule (e.g., Bayes rule). Examples of generative models are Gaussian and Multinomial mixture models and Hidden Markov Models (Marin, Mengersen, & Robert, 2005).

In supervised learning, a generative model learns the class distribution from the labelled training set $\mathfrak{D} = (X_l, Y_l)$ by estimating the parameters θ (e.g., mean, covariance matrix in case of Gaussian distribution) that describe the class distribution. The estimation of θ is an optimisation problem with the objective to find θ under which the data likelihood $p(\mathfrak{D}|\theta)$ is the largest. This procedure is solved analytically most commonly with the logarithmic maximum likelihood estimate (MLE) method (Harris & Stocker, 1999).

In SSL, the training set consists of labelled and unlabelled examples $\mathfrak{D} = (X_{l+u}, Y_{l+u})$, which makes the optimisation problem difficult to solve analytically due to the existence of unlabelled data. Finding the local maxima of the class distribution needs to be solved with an approach that can handle both labelled and unlabelled data. The most common approach for finding the parameters θ is the Expectation Maximisation algorithm (EM) (Dempster, Laird, & Rubin, 1977). EM iteratively clusters the unlabelled data into the number of clusters defined by the number of classes in the given dataset. The procedure iterates until convergence (finds the local maxima of the θ). Once the procedure stops, it is assumed that the examples in the same cluster have the same labels (cluster assumption in Section 2.3); therefore, the labelled examples are used to associate each cluster with the class.

An advantage of the generative models approach is that if the class distribution is identifiable, the data structure can be modelled by learning. The shortcoming is that if the class distribution is not identified correctly from the unlabelled data, the use of labelled examples may not result in an accurate model. Besides, even if the resulting model is correct and yields high accuracy for the given data, it may degrade once updated with a larger amount of previously unseen unlabelled examples.

2.2.2 Graph-Based Methods

Graph-based methods are one of the most active areas of SSL research and are done mostly in transductive setting. The idea is to construct a graph $g = (V, E)$ using the entire dataset $(X_l, Y_l \text{ and } X_u)$, where the examples X_l and X_u are vertices V , and the edge e between x_i and x_j represents the pairwise distance (similarity) between the vertices. The similarity is assessed with a weight $(w_{i,j})$, where a higher value indicates similarity and a lower value dissimilarity, while the absence of the edge equals infinite distance or zero similarity. Labels Y_l are used for label propagation in order to label all the nodes. Graph-based methods agree with the manifold assumption (Section 2.3), if the distance between two vertices is computed as the minimal aggregated path distance over all connecting paths (manifold of data points).

The distances between vertices can be defined as the Euclidean distance between k -nearest neighbours. The result is the weight matrix where $w_{i,j}$ is non-zero if x_i and x_j are among k -nearest neighbours. Another popular approach to calculate the weight matrix is with Gaussian function.

There are two general approaches for label propagation, one uses graph structure to propagate the labels from labelled vertices to unlabelled vertices (Zhu & Ghahramani, 2002; D. Zhou, Bousquet, Lal, Weston, & Schölkopf, 2003; Szummer & Jaakkola, 2002b) and the other is based on graph regularisation using a Laplacian matrix to pass the information to unlabelled vertices (Belkin & Niyogi, 2002; Belkin, Matveeva, & Niyogi, 2004; Delalleau, Bengio, & Le Roux, 2005).

Zhu and Ghahramani (2002) proposed an algorithm which propagates a label iteratively. Each iteration is composed of two steps. The first step is label propagation to their first neighbours and the second step is re-labelling the initially labelled vertices X_l with the original labels Y_l . The procedure starts with the labelled vertices propagating the label to their first neighbours and iterates until all vertices are labelled, and the labels converge.

Y. Zhou and Goldman (2004) proposed a procedure which uses the normalised Laplacian matrix in label propagation and allows label change of the initially labelled vertices. In each iteration, the labelled vertices propagate the label to the neighbouring vertices in the form of a contribution to their current label value (instead of a fixed label as in the previous example).

The process iterates until all vertices are labelled and it reaches convergence. Szummer and Jaakkola (2002b) proposed Markov random walks through the graph. The transition probabilities between vertices are calculated using EM (Dempster et al., 1977) and used as label propagation probabilities. The performance of such an approach depends on the length of the walk, which can be decided with cross-validation (if we have enough labelled data) or with a heuristic, which makes this a point of failure of the approach.

Graph regularisation approaches deal with the trade-off between the consistency of the partial labelling and the geometry of the data. The cost function is composed of a value that measures the change of the initially labelled data and a penalty value. The penalty value is used to penalise quick changes of the labels in the vertices that are close together according to the W matrix. Zhu, Ghahramani, and Lafferty (2003) proposed an approach where the labels are first propagated to the neighbouring vertices as in other approaches; second, the initially labelled vertices are re-labelled with original labels; third, the penalty function is used on newly labelled vertices (unlabelled portion of the data) to smooth the transitions and maintain the geometry of the data (smoothness assumption in Section 2.2).

2.2.3 Low-Density Separation

The main idea of these methods is that the decision boundary between classes needs to lie in a low-density region (Chapelle & Zien, 2005). In general, low-density separation methods agree with either the smoothness or cluster assumption presented in Section 2.2.

The intuition behind low-density separation was introduced by Vapnik (1998) as a transductive inference approach called transductive support vector machine (TSVM). The general goal of the low-density separation is to find the most informative examples in the unlabelled dataset with the aim of pushing the class decision boundary of one class as far away from the other class as possible, thus separating the data well. This is achieved by employing a maximal-margin algorithm that separates the data by a hyperplane that has a maximal margin to the nearest labelled example for all labels. One such algorithm is SVM (Cortes & Vapnik, 1995). Finding maximal margin using both labelled examples (X_l, Y_l) and unlabelled examples (X_u) with SVM is difficult and computationally complex for large datasets (Vapnik & Sterin, 1977; Bennett & Demiriz, 1999).

Joachims (2002) proposed the SVM^{light} algorithm which uses the labelled portion of the dataset (X_l, Y_l) to train an SVM model to be used to predict the labels (Y_u) of the unlabelled examples (X_u). It uses the coordinate descent algorithm (Wright, 2015) to perform a local search on unlabelled examples as labelled by the trained SVM in a predefined number of iterative steps, each of which retrains the SVM model with the newly labelled examples. The number of examples to be labelled in one step is defined by the user or estimated according to the balance of the training set (the number of positive examples and the number of negative examples). A similar approach was proposed by Demiriz and Bennett (2001) in which they recalculate the number of included examples in each iteration to speed up the adaptation. The last two approaches can be used in an inductive setting.

Unlike TSVM, which is a non-probabilistic model, an alternative approach is to put the problem into a discriminative Bayesian framework and concentrate on modelling the posterior probability $p(y|x)$ (Neil D. & Michael I., 2006) or to minimise the entropy (as a measure of class overlap) and increase the posterior probability $p(y|x)$ in high-density regions (Zhu et al., 2003; Szummer & Jaakkola, 2002a). Grandvalet and Bengio (2006)

proposed a minimum-entropy method in an inductive setting. The method trains a probabilistic model with a minimum entropy criterion applied (Hu, Fan, Wu, & Zhou, 2013) on labelled examples and uses it to label the unlabelled examples in iterations, each of which retrains the model.

Probabilistic approaches are convenient for the evaluation of their performance since they output label probability. This opens the possibility to design methods that can perform in an inductive setting. The method can label the unlabelled example and use the label probability as a prediction confidence measure. This enables faster adaptation of the model. Such methods are called self-training, and since they are closely related to the method proposed in this dissertation, we dedicate a separate section (Section 2.2.4) to a detailed overview.

2.2.4 Self-Training Methods

Self-training methods fit into the low-density separation methods category. The goal of their task is to push the decision boundary between classes further apart in each iteration.

Briefly, self-training methods are wrapper methods in which, first, one or multiple models are trained on a small number of labelled examples in a supervised manner to predict labels for unseen unlabelled data; second, the most confident predictions are used to retrain the model(s). The advantage of these methods is that the models do not make any specific assumptions about the input data. The shortcoming is that if the models are retrained with erroneous labels, the performance decreases, while if only the most confident predictions are used for retraining, the performance might not surpass the performance of the model trained in a supervised manner.

There are several properties that distinguish between the self-training methods that were well defined by Triguero, García, and Herrera (2015):

- Number of domain models used for label prediction defines how many models label the example and how the final label is selected.
- Type of machine-learning algorithm used for training the domain models defines whether the same algorithm is used for all of them.
- Number of data views defines whether the feature vector structure of the domain models is the same.
- Retraining property defines how the retraining is performed.

The self-training algorithms are presented in the form of a hierarchy according to their properties in Figure 2.1. According to the number of domain models used for predicting the label, a method can be a single-model method if only one model predicts the label and a multiple-model method if two or more models predict the label. In case of a multiple-model method, we differentiate between single-learner and multiple-learner methods. In single-learner methods, the models are trained using the same machine-learning algorithm as opposed to the multi-learner methods where each model is trained using a different machine-learning algorithm. The design of the feature space per model in multiple-model methods defines whether the method is a single-view method or a multi-view method. Single-view methods use the same feature space in all models. In multi-view methods, each model has orthogonally independent feature subset. The retraining property defines whether the model(s) are retrained in an incremental, batch, or amending mode. Incremental mode evaluates the label confidence of each newly labelled example and selects for retraining only the examples with the most confident label predictions. Batch mode evaluates the prediction confidence of the newly labelled examples and adds into a batch

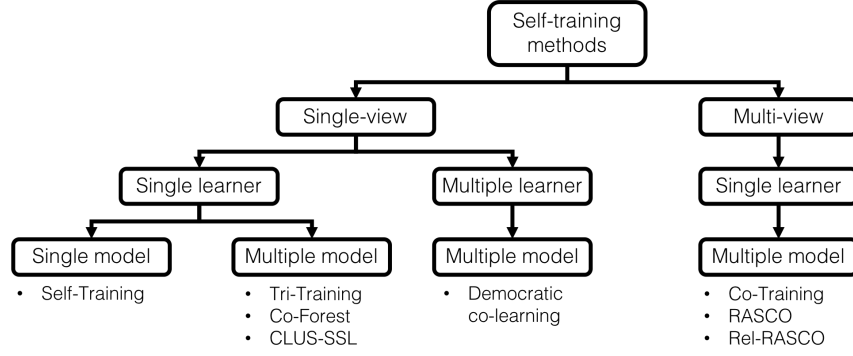


Figure 2.1: Self-training methods according to their properties (Triguero, García, & Herrera, 2015).

each that meets the confidence criteria. The examples in the batch are used for retraining at once. The amending mode is either in incremental or batch mode and takes into consideration that the example in the dataset can get outdated. The amending mechanism handles the evaluation of an example's obsolescence and removes it from the dataset used for retraining if required.

Two algorithms that established the area of self-training and are adopted as a standard are Self-Training proposed by Yarowsky (1995) and Co-Training proposed by Blum and Mitchell (1998). Self-Training is an incremental, single-model, and single-learner algorithm, and Co-Training is an incremental, multiple-model, single-learner, and multi-view algorithm. Later algorithms are mostly enhancements or extensions of these two algorithms with different approaches to confidence evaluation and inclusion criteria.

The most common single-view method is Self-training (Yarowsky, 1995), which is composed of one model trained with any machine-learning algorithm that outputs probability. The model predicts the labels of the unlabelled examples and retrains itself with the newly labelled examples that achieve confidence criterion (usually high label probability). Examples of multiple-model methods are Tri-Training (Z. H. Zhou & Li, 2005), Co-Forest (Li & Zhou, 2007) and CLUS-SSL proposed by (Levatić, Ceci, Kocev, & Džeroski, 2015). All are single-learner and single-view methods. Tri-training is composed of three models, and the retraining is performed in incremental mode. It first divides the training dataset into three data subsets, each of which is used to train one of three models with the Bagging algorithm (Breiman, 1996). A majority vote obtains the final label of the unlabelled example from the predictions of the three models. The newly labelled example is added to the dataset used for retraining and temporary models are retrained. A heuristic evaluates the performance of the temporary models on the initial labelled dataset. If the performance stays the same or is improved, the example stays in the dataset used for retraining the models; otherwise, it is omitted. The Co-Forest uses Random Forest (Breiman, 2001) to train an ensemble of trees and performs the adaptation in a batch mode. The ensemble predicts the labels for the unlabelled examples, and the Random Forests majority vote decides on the final label. If the confidence of the Random Forest surpasses a predefined threshold, the newly labelled example is included in the dataset used for retraining. CLUS-SSL uses predictive clustering trees (PCT) in the Random Forest method to train an ensemble of trees for multi-target regression (MTR). The ensemble is trained on the labelled dataset and used to predict the labels for the unlabelled examples. Since the task is to adapt the model for MTR task, they propose a heuristic to be used as a confidence measure for the prediction. If the label prediction surpasses the proposed heuristic threshold, the newly

labelled example is included in the training set used for retraining.

Democratic co-learning (Y. Zhou & Goldman, 2004) is composed of three models and the retraining is performed in incremental mode. This method is a multiple-learner method, which means that the three models are trained with a different machine-learning algorithm. The selection of the final label is obtained by a majority vote from the labels as predicted by the three models. Whether the newly labelled example is added to the dataset used for retraining or not is decided by a heuristic.

Multi-view methods are necessarily composed of more than one model, which categorises them into multiple-model methods. Most commonly, the models are trained with the same machine-learning algorithm (single-learner methods). As mentioned, Blum and Mitchell (1998) pioneered the multi-view methods with Co-training composed of two independent models that retrain in incremental mode. The Co-training method has two prerequisites. First, the feature space needs to be divided into two orthogonal subsets, one for training each model. Second, each trained model needs to be self-sufficient for the task. The first prerequisite ensures that the trained models are independent of each other, thus providing individual views on the task (this is where the multi-view name comes from) and complementary information. The second prerequisite ensures that the models can accurately label the unlabelled examples and decrease the possibility of retraining with incorrectly labelled examples. Both models label unlabelled examples, and the most confident predictions of each model are included in the training set used for retraining.

The main obstacle to implement an efficient Co-training method is the orthogonality prerequisite since it is not always possible to achieve it. RASCO (J. Wang, Luo, & Zeng, 2008) and Rel-RASCO (Yaslan & Cataltepe, 2010) are both upgrades of the Co-training method regarding how to choose the feature subsets for the trained models to comply with the rule of subset orthogonality. RASCO stands for RANdom Subspace Co-Training: the proposed approach randomly divides the feature subspace into a predefined number of subspaces and trains the same number of models. In contrast to the original Co-training which requires self-sufficient models, RASCO requires models that return accuracy better than a random guess to efficiently adapt the models and outperform the original Co-training. Majority voting obtains the final label. Rel-RASCO stands for Co-training with relevant random subspaces, which is an upgrade of both mentioned methods. The features are evaluated using mutual information between the feature and the label as a relevance score. To maintain the randomness, the features are selected into subspaces according to probabilities proportional to the relevance scores, which are calculated on the labelled portion of the dataset. The number of models composing the method is predefined with the number of feature subspaces. Majority voting obtains the final label.

Self-training methods with more than one model mostly obtain the final label using a majority vote. The inclusion criterion is defined with either the selection of the number of most confident predictions from the given set of unlabelled examples or with a heuristic that evaluates whether the prediction confidence meets the requirements for inclusion. The heuristic is usually set to select only the examples with high prediction confidence. This may result in the inclusion of examples that do not contribute new information to the model since they already perfectly fit the models trained in a supervised manner. Models would benefit if examples with a higher uncertainty were included for retraining.

2.3 SSL for Real-time Personalisation

Overview of personalisation techniques showed that certain amount of person-specific data is always required. These data are used to personalise a larger general dataset that contains data of other people than the target person. The personalised dataset is afterwards

used for training a machine-learning model personalised for the target person. Overview of the SSL techniques showed that unlabelled data can be used to improve the model trained using only labelled data and that it can be used in real-time. Since the behaviour, habits, physical characteristics and similar change over time, the model personalised with common personalisation technique may soon become less accurate and would require further personalisation. On the other hand, a large amount of unlabelled data generated while using such system can be used in SSL manner to personalise the model without the need for human annotator.

We aim at designing a method that is simple to use and can be deployed in real life for real-time adaptation and personalisation of models in the inductive setting. The personalised model will need to be able to efficiently predict labels on unseen data (Definition 2.5) and use them for retraining. Additionally, we would like to fully exploit the general dataset(s) available for a task in addition to a small amount of labelled person-specific data.

Generative models were successfully applied on many domains, mainly in the transductive setting, and even though they have appealing features connected with identifiable class distribution, they also have some major drawbacks. Modelling $p(x|y)$ is more demanding than modelling $p(y|x)$, as is the goal of discriminative methods. It is too computationally demanding to be used in real time on high-frequency data. Besides, if the dataset has complex class distribution (multinomial problem), it may be easily misspecified, which results in an incorrect model, thus compromising accuracy.

The graph-based approaches are currently a very popular area of research since graph theory proved to have a high potential in many domains from natural language processing (Mills & Bourbakis, 2014) and bioinformatics (Yan, Zhao, & Pang, 2017) to the analysis of social networks (Truong, Truong, & Dkaki, 2016). These methods are done in a transductive setting and propagate the label to unlabelled examples through neighbouring vertices defined by similarity. Their task is to label the unlabelled examples while maintaining label consistency and data geometry. Each time a vertex is labelled, all neighbouring nodes are revisited and labels re-evaluated, which means that the entire graph may need to be updated. In case of large datasets with highly variable data, this may cause too many re-evaluations for real-life tasks in an inductive setting.

Low-density separation methods are based on an idea of transductive inference, which uses SVM to find the maximal low-density area as a boundary between classes. Since finding the boundary using both labelled and unlabelled data with SVM is a hard optimisation problem, the methods transited to achieving the same results using probabilistic models. The initial model is trained on the labelled portion of the data in a supervised manner and used to label the unlabelled examples, which are then used for retraining the model. This approach opened a new area of methods called self-training, which can be used in real time and their performance measured with the label probability output. These methods do not require assumptions about the input data and were successfully used in inductive setting on many domains, which makes them suitable for personalisation.

The performance of the self-training methods depends on the quality of single or multiple models and the heuristics used for the selection of the final label and the inclusion of the labelled examples. If the inclusion heuristic includes only the most confidently labelled examples, the retrained model might not perform better compared to a model trained in a supervised manner.

We aim to improve the adaptation by also including examples labelled with higher uncertainty, which can be achieved by replacing the heuristic with a machine-learning model. Additionally, we would like to exploit general labelled datasets that can be freely obtained or were previously collected and labelled for other people. These data can be

used to train a general model, and the SSL technique can be used to adapt this model to the target person. To design an SSL method that takes into consideration all the above-mentioned parameters and to replace the heuristic with machine-learning models, we will address the following challenges:

- Define the design of the domain models trained to solve the domain task in terms of feature vectors or views, datasets, and a machine-learning algorithm used for training.
- Define the criterion procedure for the selection of label of newly labelled examples, preferably in the form of a machine-learning model.
- Define the criterion procedure for the inclusion of newly labelled examples in the training dataset, preferably in the form of a machine-learning model.
- Define a stopping criterion — when does the retraining stop.

In the next chapter, we answer the challenges and propose a multi-model, single-learner, self-training method with an upgrade of the selection and inclusion heuristics as they are used in the current self-training approaches. The method uses two machine-learning models: one to tackle the selection of the final label instead of the majority vote, and one to tackle the inclusion task. The goal is to include more diverse examples in the training set used for retraining the models compared to approaches that include only the examples with high prediction confidence. The introduced diversity should be beneficial for achieving better adaptation results.

2.4 Summary

The chapter presents the related work on most common personalisation techniques and overview of SSL methods. The overview of the SSL methods focuses on exploring the SSL categories to highlight the one that can be used for personalisation in real-time. We concluded that the self-training methods from the low-density separation category are the most suitable approach for further exploration and reviewed them in detail. The chapter ends with the section presenting the positive aspect and limitations of the personalisation and self-training approaches and points out the challenges to be addressed in order to design the self-training SSL method for real-time personalisation successfully.

Chapter 3

Multi-Model Adaptive Training

Most common approaches to personalisation are efficient for short-term improvement of the models (Section 2.1) since they consider one-time-only personalisation and do not employ any adaptive mechanisms that would enable further refinement of the models. Self-training SSL approaches (Section 2.2.4) seem to facilitate implementation of such mechanism using the same person-specific data as in the common personalisation approaches. However, the shortcoming of the self-training methods is in the heuristic that evaluates the reliability of the label prediction to be used for personalisation (retraining) of the model. If the heuristic is too conservative, the included example may not contribute to the improvement of the model, and if it is too liberal it may include misclassified examples which will inevitably degrade the accuracy. We present the SSL method named Multi-Model Adaptive Training that introduces the appealing feature of common personalisation such as using larger publicly available labelled datasets for the task into SSL self-training algorithm enhanced with machine-learning models instead of heuristic for the evaluation of the reliability of the predicted label for the inclusion in retraining. Since we use multiple models that need to be trained, we also provide a training procedure dedicated to training all composing machine-learning models by ensuring fair data division and full exploitation of the available labelled data. The method and the training procedure was published in the paper Cvetković, Kaluža, Gams, and Luštrek (2015) included in Chapter 4.1, but here we put the method into a framework, extend it and present it in more detail.

The chapter is structured as follows. First, we introduce the general framework of the method and explain the connection to the general machine-learning pipeline and purpose of each module of the method. Next, we introduce the implemented and evaluated architecture and the structure of each composing model. Finally, we present the training procedure that ensures training of all models with a limited amount of data.

3.1 General Framework of Multi-Model Adaptive Training

Multi-Model Adaptive Training follows the idea of self-training methods for personalisation of machine-learning models. It is designed to connect to the general machine-learning pipeline presented in Figure 3.1 (Fayyad, Piatetsky-Shapiro, & Smyth, 1996). The pipeline is modified to emphasise the procedure after the deployment of the trained machine-learning models.

The general pipeline is composed of a data input module, a data preprocessing module, and a module in which a machine-learning task is solved. The data input can be set to retrieve the raw data from a database or real-time stream to the preprocessing module in which the data are fused, synchronised, filtered, and transformed into a feature vector forming an example ready to be fed into one or more machine-learning models in the

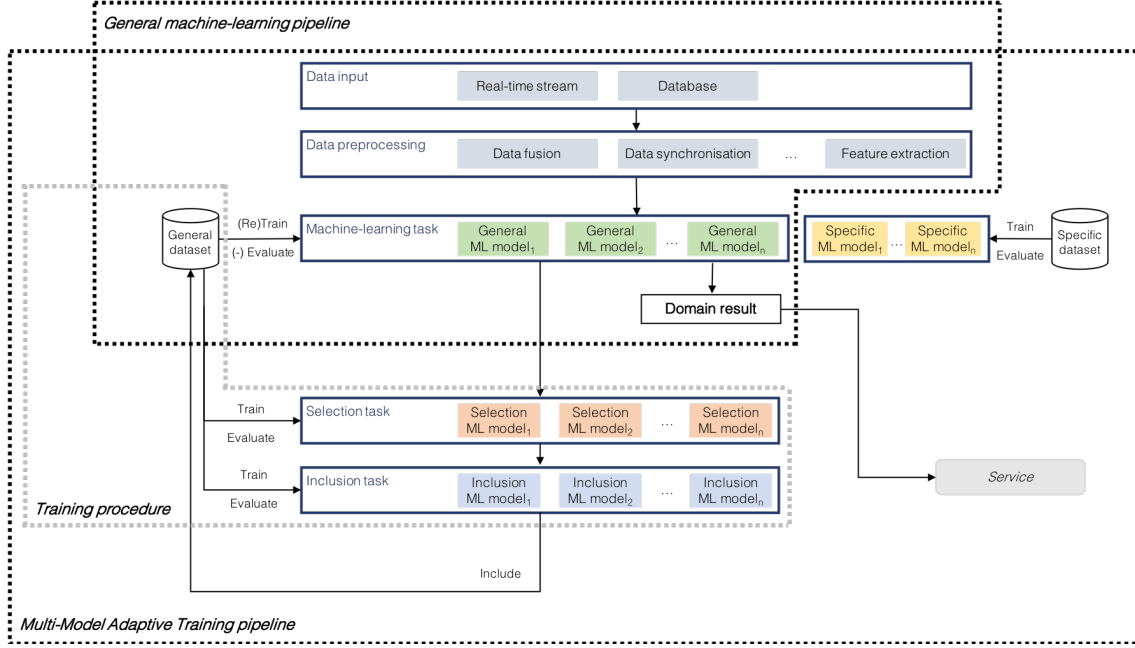


Figure 3.1: General machine-learning pipeline enhanced with the Multi-Model Adaptive Training SSL pipeline.

machine-learning task module. The models in the general machine-learning pipeline are general domain models (green models in Figure 3.1) trained in a supervised manner on a general dataset (left-side dataset in Figure 3.1), which contains sufficiently diverse data to yield high accuracy in a laboratory evaluation. Once the models are deployed, they process the example to make a prediction and obtain a domain result in the form of a label that can be used by an application-specific service (e.g., recognised activity used is activity monitoring service). A general dataset can be obtained from public repositories or can be previously collected for the domain task. These data are perceived as general domain and do not contain any specific knowledge about the target person.

Multi-Model Adaptive Training adds to the general pipeline one or multiple specific machine-learning domain models that belong to the target person (yellow models in Figure 3.1). The specific machine-learning models are trained and evaluated in a supervised manner with a small amount of person-specific data (right-side dataset in Figure 3.1) for the same task as the general models in the general machine-learning pipeline. The domain results from each domain model are fed into the selection module, which is based on an idea of stacking (Wolpert, 1992; Z.-H. Zhou, 2012) and uses one or multiple stacked machine-learning models to predict the final label by choosing one of the domain models for labelling the example (red models in Figure 3.1). The selection models are trained and evaluated using the data from the general dataset in a specifically designed training procedure introduced later in Section 3.2.3. In case of multiple selection models, these can be additionally stacked for a final selection result. The outputs of the selection models, together with the results of the general and specific models, are fed into the inclusion module, which uses one or multiple machine-learning models (blue models in Figure 3.1) to decide whether the newly labelled example is appropriate for inclusion in the general dataset used for retraining the general machine-learning models. The inclusion models are trained and evaluated using the data from the general dataset in the same training procedure as selection models, introduced later in Section 3.2.3. Again, in case of multiple

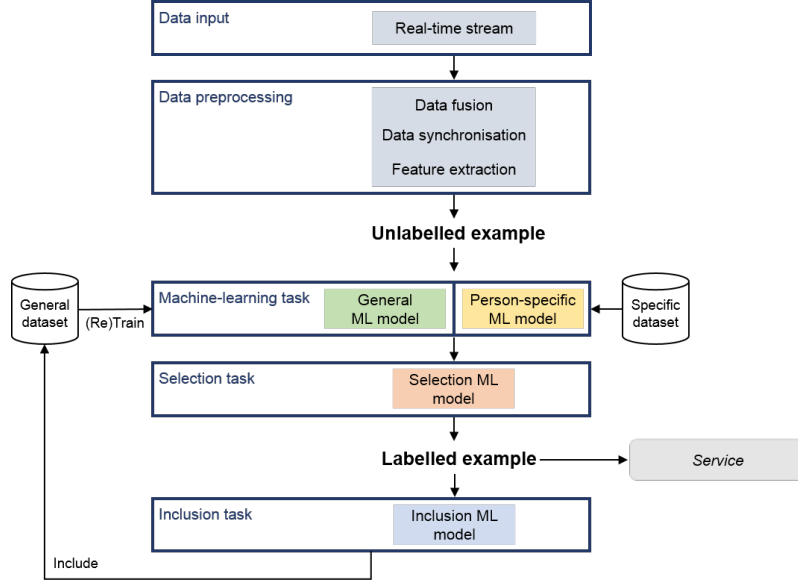


Figure 3.2: Multi-Model Adaptive Training architecture.

inclusion models, these can be additionally stacked to obtain a final result.

There are two main differences between the approach to SSL and self-training as seen in the Section 2.2.4 of Chapter 2 and the Multi-Model Adaptive Training method. The first difference originates from SSL methods usually utilising only data collected specifically for the target person for training the models and for the adaptation. In our method, in addition to person-specific data, we also exploit general datasets that already exist and can be easily obtained. We use general datasets to train general models for solving the task. The second difference originates from the use of heuristics for selection and inclusion criteria in self-training methods, which in general selects and includes only the examples with the most confidently predicted labels. Retraining only with the most confident predictions might result in low improvement of the model performance since these examples already fit the model. In Multi-Model Adaptive Training, we replace the heuristics with machine-learning models: one type of models are trained to select the final label and the second type are trained to decide on the inclusion of the newly labelled examples in the general dataset for retraining. This enables the method to include, in addition to the most confidently labelled examples, examples that were labelled with higher uncertainty, which will introduce more diverse examples specific for the target person, thus better and faster adaptation of the model.

3.2 Implemented Architecture

Following the pipeline presented in Figure 3.1, we construct and implement the method with a single general, single person-specific model and single selection and inclusion model. The architecture is presented in Figure 3.2 and as pseudo-code in Algorithm 3.1. The method is meant to be deployed as a part of a system that already uses machine-learning models for a domain task and requires each example to be labelled. In addition to labelling each example coming into the system, the Multi-Model Adaptive Training decides whether the labelled instance should be used for retraining the models or not.

The method operates as follows. Line numbers in brackets correspond to lines in Algorithm 3.1. The data are obtained from a real-time stream, preprocessed into a feature vector

Algorithm 3.1: Multi-Model Adaptive Training algorithm.

Models : General (G), Person-specific (P), Selection (S) and Inclusion (I) model
Dataset: $Dataset_{general}$: General dataset
Data: $Queue_{examples}$: Queue containing examples in a form of a feature vectors
Result: G : Adapted general model

```

/* Personalise if the queue contains examples and if the stopping
   criterion is not met */
1 while size( $Queue_{examples}$ ) > 0 and stop == FALSE do
    /* Retrieve example from the queue */
    2  $example \leftarrow getExample(Queue_{examples})$ ;
    /* Label the example with general and person-specific model */
    /* Extract prediction confidence for all labels and labels */
    3  $C_G, label_G \leftarrow label(G, example)$ ;
    4  $C_P, label_P \leftarrow label(P, example)$ ;
    /* Extract features and create feature vector for Selection model
       */
    5  $features_S \leftarrow extractFeaturesSelection(example, C_G, C_P, label_G, label_P)$ ;
    6  $C_S, select \leftarrow label(S, features_S)$ ;
    7 if select == G then
    8 |  $label_{example} \leftarrow label_G$ ;
    9 else
    10 |  $label_{example} \leftarrow label_P$ ;
    11 end
    /* Extract features and create feature vector for Inclusion model
       */
    12  $features_I \leftarrow extractFeaturesInclusion(features_S, C_G, C_P, C_S)$ ;
    13  $include \leftarrow label(I, features_I)$ ;
    14 if include == YES then
    15 | /* Include labelled example and retrain */
    16 |  $Dataset_{general} \leftarrow Dataset_{general} + (example, label_{example})$ ;
    17 |  $G \leftarrow train(Dataset_{general})$ ;
    18 end
    /* Evaluate if the conditions for the stopping criterion are met */
    18 if stoppingConditions () then
    19 | stop  $\leftarrow$  TRUE;
    20 end
21 end

```

that represents the unlabelled example ($example$) and inserted in a queue ($Queue_{examples}$). If the queue is not empty and the predefined stopping criterion is not met, the method proceeds with the personalisation (line 1). The unlabelled example is retrieved from the queue (line 2) and labelled with the general and the person-specific model at the same time (lines 3 and 4). The results are an array of prediction confidence values (C_{model}) for all labels and the labels per model ($label_{model}$). The prediction confidence values (label probability) and labels per model are processed with a procedure which extracts the selection model features and forms a selection model feature vector (line 5). The feature

vector is fed into the selection model which outputs the decision regarding which model should be trusted with final labelling the example and its prediction confidence value (line 6). The example is labelled according to the result of the selection model (lines 7 to 11). The next step of the Multi-Model Adaptive Training is the evaluation of the inclusion suitability of the automatically labelled example. The array of prediction confidence values of all models is processed with the feature extraction procedure to form a feature vector for the inclusion model to obtain a decision whether the automatically labelled example should be included into the general dataset or not (lines 12 and 13). If the output of the inclusion model returns a positive decision, the automatically labelled example is added to the general dataset, and the general model is retrained (lines 14 to 17). Before a new repetition, the procedure for evaluation of stopping criterion is called (lines 18 to 20).

To implement and use the Multi-Model Adaptive Training method, one must train three machine-learning models on a limited amount of data and one model on data specific for the target person. The dataset and the training procedure are described in the next subsections.

3.2.1 Datasets

Figure 3.2 presents the method work flow composed of two datasets that contain raw data: the general dataset (left-side dataset in Figure 3.2), and the specific dataset (right-side dataset in Figure 3.2).

The **general dataset** ($Dataset_{\text{general}}$) is either obtained from public repositories or collected and labelled for the task. It is required that the dataset is of good quality since it is used for training and evaluation of the domain models in the laboratory environment. Data collection is usually controlled with the intention to collect as diverse data as possible. This is achieved by instructing the people whose data are collected to try to perform the tasks as naturally as possible (in activity recognition, context recognition), to try to perform the task to the limit (e.g., running or cycling vigorously for a longer period of time in human energy expenditure estimation), or even produce specific states (running vigorously before measuring blood pressure in case of blood pressure estimation). These datasets are therefore rich in information, but lack the information on the target user.

Along with the requirement of quality labelled $Dataset_{\text{general}}$, two additional requirements need to be fulfilled for the dataset to be suitable for the training procedure introduced in Section 3.2.3. The dataset should enable partitioning by a person, e.g., identification number (columns in Figure 3.3), and some semantic parameter, such as label, scenario, or context of the task (rows in Figure 3.3). For example, the data of person A with labelled daily activities can be semantically grouped by the type of activity. The partitioning per person enables us to exclude a person for which the training is performed and to evaluate the trained model in the Leave-One-Person-Out (LOPO) approach. The division per semantic group enables control over the creation of representative folds required for the training procedure of the models (Section 3.2.3).

The **specific dataset** ($Dataset_{\text{person}}$) is briefly collected and labelled onsite with the target person. This dataset is small and contains just a few examples and does not necessarily contain all the labels that are present in the $Dataset_{\text{general}}$.

In most common machine-learning enhanced applications, the $Dataset_{\text{general}}$ is the dataset used for training a domain model in a supervised manner; after evaluation in a laboratory environment, the model is deployed in real life to be used with the target user. Due to physical, behavioural, habit, etc., differences between people and the lack of information about the target person, the model can experience a decrease in accuracy. One could merge $Dataset_{\text{person}}$ and $Dataset_{\text{general}}$ and retrain the domain model in a supervised manner; however, the amount of data is small and not all labels are represented, so it

	Person (e.g., user ID)					
	A	B	C	D	E	F
a						
b						
c						
d						
e						
f						

Figure 3.3: Required dataset partitioning of the $Dataset_{general}$ to be suitable for the training procedure. The columns represent partitioning by a person (identification number), and rows by a semantic parameter (label, scenario, or context of the task).

may not contribute significantly to performance improvement of the domain model. The goal of the Multi-Model Adaptive Training is to include person-specific examples into the $Dataset_{general}$ automatically, thus personalise the model trained on this dataset.

3.2.2 Domain and Decision Models

This section introduces the design of the domain and decision models.

3.2.2.1 Domain Models

General model (G) is the main domain model. When the task is to adapt to the new statistical properties of the data (concept drift), the general model is trained only on $Dataset_{general}$. When the personalisation task is to adapt to the newly introduced label, the general model is trained on the union of $Dataset_{general}$ and $Dataset_{person}$. The general model is obtained with the common procedure for the development of best performing machine-learning models in a supervised manner. One must train the model in a supervised way and evaluate different feature sets and machine-learning algorithms. The feature set and machine-learning algorithm that yields the best results on the given dataset in the LOPO approach for the domain task is chosen as a design of the final general model. It is required that the machine-learning algorithm used for training the general model outputs label probabilities or any other measure that can be used as prediction confidence.

The general model is being retrained, thus personalised, with automatically labelled examples included into the $Dataset_{general}$ by the Multi-Model Adaptive Training method.

Person-specific model (P) is a helper domain model trained on $Dataset_{person}$. We evaluated different approaches to person-specific model design (Cvetković & Luštrek, 2014) and selected the following. The design of the model is adopted from the general model and is thus constructed from the same set of features and trained with the same algorithm as the general model. An exception occurs in classification tasks in which person-specific model is a one class model. In this case it needs to be trained with a machine-learning algorithm that can handle one-class classification. As previously introduced, the $Dataset_{person}$ contains fewer labels than the $Dataset_{general}$, so the person-specific model trained on this data can provide a limited number of labels as an output. The data contained in the person-specific model can represent chosen labels (Section 4.1 and Section 4.2) or a chosen person-specific context (Section 4.3 and Section 4.4). Nevertheless, the content is called basic labels or basic context (*Basic*).

3.2.2.2 Decision Models

The architecture presented in Figure 3.2 is composed of two meta-level decision models: the selection model trained to select one of the domain models for the final labelling of the example, and the inclusion model trained to decide whether or not the labelled example is suitable for inclusion to the $Dataset_{\text{general}}$. The method for the extraction of selection features in classification task is presented in Algorithm 3.2.

Selection model (S) is a stacked decision model trained to tackle a classification problem. It is required that the machine-learning algorithm used for training the selection model has an ability to output the label probabilities or any other measure that can be used as a prediction confidence value. The selection model is trained to output the instruction on how to choose the final label. For classification tasks, it is trained to output one of the domain models (G or P) as trustworthy for labelling the example. In the regression task, it is trained to output either one of the domain models (G or P) as trustworthy for labelling the example or an indicator of the equation to be used on the output values of the domain models (e.g., average) to calculate the final label.

We evaluated different feature sets to be used for the design of the selection model (Cvetković, Luštrek, Kaluža, & Gams, 2011). The most common approach, which uses the domain model feature vector together with outputs of the domain models, did not yield good results, so we chose the feature set with features extracted only from the outputs of the general and person-specific model (Algorithm 3.2). The features can be contextual, statistical, or raw outputs of the domain models. Although the selection features in different systems might not be exactly the same, we propose the initial set that should be used in classification and regression domains:

- $C_G(label_G)$: The prediction confidence of the general model in its prediction.
- $C_P(label_P)$: The prediction confidence of the person-specific model in its prediction.

In case the domain models are solving a classification task, we additionally propose:

- $C_G(label_P)$: The prediction confidence of the general model in the label predicted by the person-specific model.
- $C_P(label_G)$: The prediction confidence of the person-specific model in the label predicted by the general model.
- $Basic(label_G)$ binary feature set to describe whether the general model predicted a basic label. As previously introduced, the basic labels are those that are contained in the person-specific model.
- $Equal(label_G, label_P)$ binary feature set to describe whether the general model and person specific model predictions are equal.

We also propose using single or multiple contextual features in both classification and regression domains:

- $Context(example)$: Contextual information of the example.
- $Context(label_G)$: Contextual information of the label predicted by the general model.
- $Context(label_P)$: Contextual information of the label predicted by the person-specific model.

Algorithm 3.2: Procedure for extraction of features for selection model.

```

Data:  $C_G, C_P, label_G, label_P, example$ : prediction confidences and labels
Result: labels: labels, features: decision features

/* Extract prediction confidences per prediction */
1  $C_G(label_G)$ ;
2  $C_P(label_P)$ ;

/* Extract contextual features */
3  $Context(label_G) \leftarrow \text{extractFeaturesContext}(label_G)$  ;
4  $Context(label_P) \leftarrow \text{extractFeaturesContext}(label_P)$  ;
5  $Context(example) \leftarrow \text{extractFeaturesContext}(example)$  ;

/* Extract prediction confidences per label as predicted by other
   model */
6  $C_G(label_P)$ ;
7  $C_P(label_G)$ ;

8 if  $label_G == label_P$  then
9   |  $Equal \leftarrow \text{YES}$  ;
10 else
11   |  $Equal \leftarrow \text{NO}$  ;
12 end
13 if  $label_G$  is basic then
14   |  $Basic(label_G) \leftarrow \text{YES}$  ;
15 else
16   |  $Basic(label_G) \leftarrow \text{NO}$  ;
17 end
18  $features \leftarrow (C_G(label_G), C_P(label_P), C_G(label_P), C_P(label_G), Equal, Basic(label_G),$ 
    $Context(label_G), Context(label_P), Context(example))$  ;
19 return features
  
```

Extending the feature set with a contextual type of features depends on the domain problem itself, so we cannot propose a general contextual features. In the regression domain, a contextual feature is discretisation of the estimated value by the general model and the person-specific model into a context. For example, estimated blood pressure can be discretised into a context of blood pressure: low, normal, or high blood pressure. We elaborate and propose contextual features in Section 4.3 of Chapter 4 for tackled regression domain (energy expenditure estimation).

Inclusion model (I) is a decision model trained to tackle a binary classification problem. It is required that the machine-learning algorithm used for training the selection model has an ability to output the label probabilities or any other measure that can be used as a prediction confidence. The output of the model is a decision (yes or no) whether the example should be included in the $Dataset_{\text{general}}$ for retraining or not. The features for the model are copied from the selection model with additional features extracted from the results of the selection model. To the list of features extracted for the selection model, we propose adding the following two:

- $label_S$: The result of the selection model.
- $C_S(label_S)$: The prediction confidence of the selection model in its prediction.

3.2.3 Training Procedure

As introduced previously, the three models, that is, the general model, selection model, and inclusion model, need to be trained using the labelled data from the $Dataset_{\text{general}}$. To perform this procedure correctly, we need to divide the dataset in such a way that the data we use for training an individual model do not overlap with those used for training other models and we still keep enough data for training them all. The training procedure for a case in which there are more repetitions of a task per person in a $Dataset_{\text{general}}$ is presented in the paper included in Section 4.1 of Chapter 4. Since it is common to have a dataset with single repetition of a task per person, we extend the presented training procedure to such cases in this section.

The $Dataset_{\text{general}}$ requirements were presented in Section 3.2.1. Briefly, apart from quality labelled data, the dataset needs to be dividable per person and per semantic groups (these can be labels if no other division is possible). These requirements are needed so we can allocate a representable subset of data for training the selection and inclusion model, as well as for a fair evaluation of the models and the method. The latter is explained in the next chapter (Chapter 4) and the former on an example as follows. Let us select Person A from the dataset as if this person were the new target person, the $Dataset_{\text{general}}$ as partitioned in Figure 4.9, and train all models needed for the Multi-Model Adaptive Training for the given person. We define P_{all} as a set of all people contained in the dataset (Eq. 3.1) and $P_{\text{target}} = \{A\}$ as the target person A.

To train the person-specific model for person A, we need to define the content of $Dataset_{\text{person}}$ regarding a number of labels and an amount of data per label specific to the p_{target} . The number of labels is defined by a number of labels that can be easily and safely collected (e.g., in activity recognition it is easy to collect walking, sitting and lying, but not safe to collect falls). The amount can be arbitrarily defined according to the easiness of the collection (e.g., 15 seconds of sensor data per label). These labels are called basic labels. Once we define the amount of data to be allocated for $Dataset_{\text{person}}$, we identify the semantic partitions in which the basic labels occur. The examples for $Dataset_{\text{person}}$ are selected either according to some natural boundary (first n examples), according to the repetition (if the dataset has multiple repeats of the same partition), or arbitrarily (randomly selected examples). In Figure 3.4 we have three basic labels that appear in the partitions a, b and c and are marked green (column A data of p_{target}). We exclude the rest of column A from all further training. The remaining data from all other persons (yellow samples in Figure 3.4) are defined as P_{general} (Eq. 3.2), which is used to train the general model for target person P_{target} .

$$P_{\text{all}} = \{A, B, C, D, E, F\} \quad (3.1)$$

$$P_{\text{general}} = P_{\text{all}} \setminus P_{\text{target}} \quad (3.2)$$

The training of selection and inclusion model is done using only the data of the remaining five people previously allocated for training the general model (P_{general}). Figure 3.5 shows that we exclude the p_{target} and use the remaining data for the training procedure in five repetitions. In each repetition, we select another person to be a temporary target person p_{temp} . As explained previously, we allocate the data for the temporary $Dataset_{\text{person}}$ (green samples per repetition in Figure 3.5) and for $Dataset_{\text{general}}$ (yellow samples per repetition in Figure 3.5). Next, we divide the remaining data of p_{temp} in two datasets, the dataset for training the selection model $Dataset_{\text{selection}}$ (blue samples per repetition in Figure 3.5) and the dataset for training the inclusion model $Dataset_{\text{inclusion}}$ (red samples per repetition in Figure 3.5). This procedure of dividing the datasets can be done

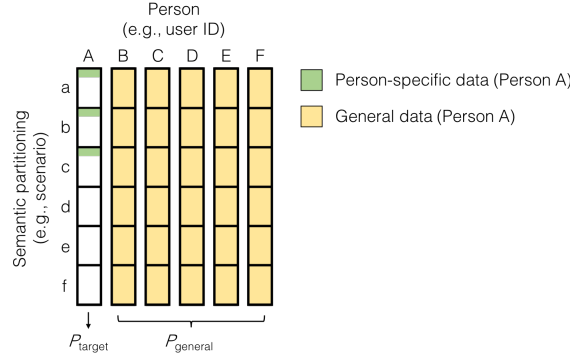


Figure 3.4: Data used for training the person-specific and general model for Person A.

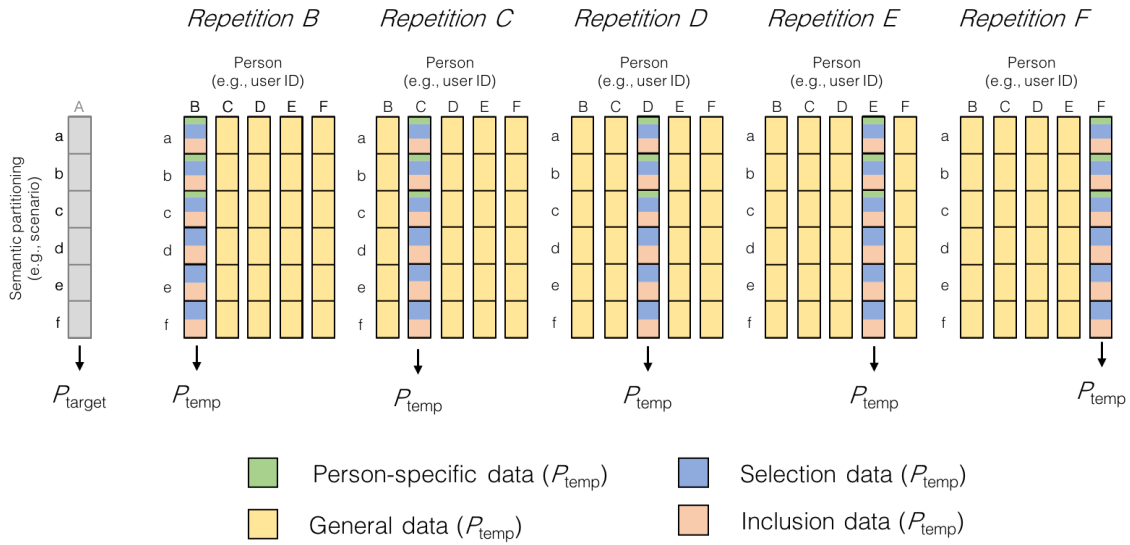


Figure 3.5: Data partitioning for training the selection and inclusion model for Person A.

multiple times in loops in each selecting new random portion the datasets (Cvetković, Kaluža, Gams, & Luštrek, 2015; Cvetković, Kaluža, Luštrek, & Gams, 2012; Cvetković, Janko, Romero, et al., 2016). Here we explain one loop.

Once we have all datasets ready, we first run the procedure for training the selection model presented in Algorithm 3.3. The line numbers in this paragraph correspond to this algorithm. The input to the procedure are arrays of required datasets per person. The arrays are denoted as *Data* to distinguish them from the datasets per person. The procedure iterates over the data of people in P_{general} . For each temporary target person p_{temp} , a general model and a person-specific model are trained (lines 3 and 4). Every example from *Dataset*_{selection} (extracted from *Data*_{selection} for p_{temp}) for current p_{temp} is labelled with the general and person-specific model (lines 6 to 8). The predicted labels are fed into the procedure presented in Algorithm 3.2 (line 9), which extracts the features composing the feature vector of the selection model. The selection feature vector is labelled with person-specific (P) if the person-specific model labelled the example correctly, or general model (G) otherwise (lines 10 to 14). In the regression task, the selection feature vector is labelled with the option that returned the lowest estimation error (one of the domain models or a value calculated from the outputs of the domain models, e.g., average).

Algorithm 3.3: Selection model training procedure – classification tasks.

Dataset: $Data_{\text{general}}$, $Data_{\text{person}}$, $Data_{\text{selection}}$: Arrays of per person datasets
Data: P_{general} : set of people in the general dataset
Result: S : Selection model trained with $Dataset_{\text{selectionTemp}}$ per person

```

/* Process selection datasets of all remaining people */
1 for i ← 0 to length( $P_{\text{general}}$ ) do
    /* Simulated temporary target person */
    2  $p_{\text{temp}} \leftarrow P_{\text{general}}[i]$ ;
    /* Train models for temporary target person */
    3  $G \leftarrow \text{train}(Data_{\text{general}}(p_{\text{temp}}))$ ;
    4  $P \leftarrow \text{train}(Data_{\text{person}}(p_{\text{temp}}))$ ;
    5 for j ← 0 to length( $Data_{\text{selection}}(p_{\text{temp}})$ ) do
        6  $example \leftarrow Data_{\text{selection}}(p_{\text{temp}})[j]$ ;
        7  $C_G, label_G \leftarrow \text{label}(G, example)$ ;
        8  $C_P, label_P \leftarrow \text{label}(P, example)$ ;

        /* Extract features with Algorithm 3.2 */
        9  $features \leftarrow \text{extractFeaturesSelection}(example, C_G, C_P, label_G, label_P)$ ;

        /* Select the model with correct classification label */
        10 if  $label_P == label_{\text{true}}$  then
            11 |  $label_S \leftarrow \text{person-specific}$ ;
        12 else
            13 |  $label_S \leftarrow \text{general}$ ;
        14 end
        15  $Dataset_{\text{selectionTemp}} \leftarrow Dataset_{\text{selectionTemp}} + (features, label_S)$ ;
    16 end
17 end
    /* Train selection model */
18  $S \leftarrow \text{train}(Dataset_{\text{selectionTemp}})$ ;
19 return  $S$ 

```

The labelled examples are stored in $Dataset_{\text{selectionTemp}}$ (line 16), which is the dataset for training the final selection model (line 18).

Once we trained the final selection model, we run the procedure for training the inclusion model presented in Algorithm 3.4. The line numbers in this paragraph correspond to this algorithm. The input to the procedure are arrays of required datasets per person. The procedure iterates over the data of people in P_{general} taking one as a temporary target person p_{temp} per repetition. For each p_{temp} , a general model and a person-specific model are trained (lines 3 and 4). To train the selection model, we first exclude the current temporary person p_{temp} from the P_{general} (line 5) and feed the remaining set of people ($P_{\text{selection}}$) into Algorithm 3.3 to train the temporary selection model (line 6). Every example from $Dataset_{\text{inclusion}}$ (extracted from $Data_{\text{inclusion}}$ for p_{temp}) for current p_{temp} is labelled with the general and person-specific model (lines 9 and 10). The predicted labels are fed into the procedure presented in Algorithm 3.2 (line 9), which extracts the features composing the feature vector of the selection model (line 11) that are fed into the selection model trained in line 6. The results of the selection model are used for the construction of the inclusion model feature vector (line 13). It is essentially selection model feature vector with additional three features, the prediction of the selection model and the confidence in its prediction. The prediction of the selection model is also used to define the label for the

Algorithm 3.4: Inclusion model training procedure – classification tasks.

Dataset: $Data_{general}$, $Data_{person}$, $Data_{inclusion}$: Arrays of datasets per person

Data: $P_{general}$: set of people in the general dataset

Result: I : Inclusion model trained with $Dataset_{inclusionTemp}$

```

1 for i ← 0 to length( $P_{training}$ ) do
2    $p_{temp} \leftarrow P_{training}[i]$ ;
3    $G \leftarrow \text{train}(Data_{general}(p_{temp}))$ ;
4    $P \leftarrow \text{train}(Data_{person}(p_{temp}))$ ;

   /* Remove current target person from the  $P_{training}$  */
5    $P_{selection} \leftarrow P_{training} \setminus p_{temp}$ ;
   /* Train selection model with Algorithm 3.3 */
6    $S \leftarrow \text{trainSelectionModel}(P_{selection})$ ;

7   for j ← 0 to length( $Data_{inclusion}(p_{temp})$ ) do
8      $example \leftarrow Data_{inclusion}(p_{temp})[j]$ ;
9      $C_G, label_G \leftarrow \text{label}(G, example)$ ;
10     $C_P, label_P \leftarrow \text{label}(P, example)$ ;

    /* Extract features with Algorithm 3.2 */
11     $features_S \leftarrow \text{extractFeaturesSelection}(example, C_G, C_P, label_G,$ 
       $label_P)$ ;
12     $C_S, label_S \leftarrow \text{label}(S, features_S)$ ;

    /* Extract features for inclusion model */
13     $features_I \leftarrow \text{extractFeaturesInclusion}(features_S, C_S, label_S)$ ;

    /* Label for inclusion if the selected model label is correct */
14    if  $label_S == \text{general}$  then
15      if  $label_G == label_{true}$  then
16         $label_I \leftarrow \text{YES}$ ;
17      else
18         $label_I \leftarrow \text{NO}$ ;
19      end
20    else
21      if  $label_P == label_{true}$  then
22         $label_I \leftarrow \text{YES}$ ;
23      else
24         $label_I \leftarrow \text{NO}$ ;
25      end
26    end
27     $Dataset_{inclusionTemp} \leftarrow Dataset_{inclusionTemp} + (features_I, label_I)$ ;
28  end
29 end
30  $I \leftarrow \text{train}(Dataset_{inclusionTemp})$ 
31 return  $I$ 

```

inclusion model feature vector (lines 16 to 26). In classification tasks, we evaluate if the result of the selection model (which domain model should label the example) corresponds to the correct label. If the selection model was correct, the label of the inclusion feature vector is set to YES (for inclusion), otherwise, it is set to NO (against inclusion). With

this, we hope to train the decision model to detect the potential misclassification and omit falsely labelled examples from inclusion in the general dataset and retraining. In case of a regression task, we evaluate the mean absolute error of the selection models prediction (error between the true value and the value as predicted by the selected domain model). One can choose any error that seems appropriate (e.g., mean absolute error, root mean squared error, mean absolute percentage error, etc.). If the error of the prediction is in an acceptable interval, we label the feature vector for inclusion (YES), and if it is not, we label it against inclusion (NO). The labelled feature vectors are stored in *Dataset_{inclusionTemp}* (line 27). Once all the people are processed, this dataset is used for training the final inclusion model (line 30).

It may not be needed to include all correctly labelled examples in the training set due to redundant information, but since it is hard to detect the utility of a newly labelled example, we choose to train the model with the mentioned heuristic.

3.2.4 Adaptation procedure

The adaptation or personalisation procedure can start once all models are trained. The procedure takes as an input a stream of raw data for the task or data from the database. It preprocesses it and transforms it into domain feature vectors that can be used by the domain models for predictions which are evaluated by the decision models for the inclusion. The personalisation can be performed either in a batch or incremental mode. Batch mode is more appropriate for utilising data from the database since it first evaluates a larger amount of data for inclusion and then uses all positively evaluated predictions for retraining. In the incremental mode, the retraining occurs each time prediction is positively evaluated for inclusion.

The adaptation procedure is performed until a stopping criterion is met. The stopping criterion can be either set arbitrarily or as a measure calculated from the performance of the method. Example of an arbitrarily defined stopping criterion is a predefined period of time or a preset number of included examples. Example of a performance measure is the ratio between the included and discarded examples in an interval which we use as a stopping criterion in the evaluation chapter in Section 4.3 and Section 4.4.

3.3 Summary

The chapter introduces the proposed method called Multi-Model Adaptive Training and associated procedures that enable its implementation. We start with the description of the general framework and its connection to the general machine-learning pipeline and proceed to a more detailed description of the implemented architecture composed of four machine-learning models, two of which are designed to replace the heuristics commonly used in self-training approaches and improve the personalisation. We present the dataset requirements and the training procedures for each of the composing machine-learning models. We end with the description of the overall adaptation procedure and potential stopping criteria.

Chapter 4

Evaluation

We evaluate the Multi-Model Adaptive Training (MMAT) on three classification tasks and one regression task. In each task, we train a general model and adapt it to the new person. In Sections 4.1, 4.2 and 4.4, we tackle the three classification tasks for which we use Multi-Classifier Adaptive Training (MCAT), a customised MMAT to work with classification domain models. In Section 4.3, we tackle a regression task for which we use Multi-Regressor Adaptive Training (MRAT), a customised MMAT to work with regression domain models.

We start with the evaluation of MCAT on a simple domain, an identification task in which we introduce a new label and adapt a model to more accurately identify people. Next, we proceed to two complex classification tasks, the adaptation of a low-level activity recognition model and adaptation of a context recognition model both enclosed as published journal papers. The goal of the low-level activity recognition task is to adapt the classification model to better recognise activities as performed by the new person, similarly, in the second task, the aim is to adapt the classification model to better recognise contexts of the new person.

To evaluate MRAT, we tackle the human energy expenditure estimation regression task. We enclose a published journal paper in which we describe the supervised regression task and point out the problem of increased estimation error due to the difference between people. In Section 4.3, we extend the supervised approach from the paper with MRAT to adapt the regression model to the new user.

4.1 Use Case 1: Low-Level Activity Recognition Task – Classification

Intelligent systems aimed at monitoring persons' daily lifestyles usually employ a module that tackles low-level activity recognition to detect atomic activities (e.g., standing, walking, running, lying, sitting, etc.) of the person. An activity-recognition model is usually trained in a laboratory on data of individuals performing predefined activities. Even though the evaluation in a laboratory environment shows high recognition accuracy, once the model is deployed in real life, the accuracy often decreases, since people have specific characteristics and ways of performing activities.

4.1.1 Related Publication

In the included journal paper (Cvetković, Kaluža, Gams, & Luštrek, 2015), we use MCAT to personalise the low-level activity recognition model to a new user. Here we summarise the content of the paper.

4.1.2 Dataset

The dataset used in this paper contains data of ten volunteers wearing localisation tags on the chest, waist, and both ankles. The volunteers were performing a predefined set of scenarios reflecting typical daily activities (standing, sitting, lying, sitting on the ground, crawling on all fours, falling, going down, and standing). The scenario was repeated five times which provided 2.8 hours of data per person. The localisation data are obtained in the form of a three-dimensional vector describing the location in x, y, and z coordinates (Kaluža et al., 2014).

4.1.3 Training Procedure

The dataset is in line with the general requirements presented in Section 3.2.1. It can be partitioned by person and by scenario, which is needed for the training procedure presented in Section 3.2.3.

4.1.3.1 Data Partitioning

First, we identified the labels that will be included in the person-specific model. These were lying, sitting and standing (30 seconds per activity) which were randomly selected from the scenarios in which the label occurred. The rest of the data was divided to be used for training the selection model (called Meta-A) and inclusion model (called Meta-B).

4.1.3.2 Machine-Learning Model Training

The initial general model is trained on preprocessed data to recognise eight activities: standing, sitting, lying, sitting on the ground, crawling on all fours, falling, going down, and standing up as presented by Kaluža et al. (2014). The person-specific model is trained on a small amount of data (30 seconds per activity) representing the lying, sitting, and standing activity of the current target person. The selection model (called Meta-A) and inclusion model (called Meta-B) are trained with the training procedure in which the data for the models are selected according to the repetitions of a scenario and cross-validated on the same dataset (Section 4.2 in the enclosed paper and Section 3.2.3 in the dissertation).

4.1.4 Adaptation Procedure

The paper evaluates the procedure for personalisation of the activity-recognition model on 10 people. We start with the initial general activity-recognition model with the mean accuracy of 70%. Using the MCAT for personalisation increases the overall mean accuracy by 12.7 percentage point to 82.7 %.

The personalisation results are compared against four baseline supervised approaches and three baseline SSL approaches. The baseline supervised approaches are the accuracy of the initial general model, the person-specific model, a model trained on their merged data of both models (most common personalisation approach), and the selection model without adaptation. The baseline SSL approaches are the Self-training (Yarowsky, 1995), slightly modified Democratic co-learning (called the Majority vote in the paper) (Y. Zhou & Goldman, 2004) and a threshold-based MCAT (a threshold rule replaces the inclusion model). We also compared the results against Co-training (Blum & Mitchell, 1998), in which we strived to divide the dataset into two orthogonal views and fulfil the prerequisite of having two efficient models, but the results per model were so poor that we did not report them. All compared methods increased the accuracy of the initial model (by up to six percentage points) but did not outperform the MCAT approach.

4.1.5 Summary

There are two main conclusions of the paper. First is that the MCAT procedure can be used for personalisation of low-level activity recognition models; second is that employing the two additional models (selection and decision model) significantly improves the personalisation procedure. In comparison with other approaches, MCAT turned out to be superior in terms of increased accuracy of the general model. Additionally, the dataset on which the adaptation was performed reflects the distribution of typical daily activities which is unbalanced (Table 3 in the included paper). Persons' day is composed of large amount lying, sitting and standing and a smaller amount of all other activities. We showed in previous research on the same dataset that the method can adapt the general model also to the classes that do not occur often (Cvetković et al., 2011). However, it would be beneficial to employ a mechanism that will evaluate the structure of the dataset after personalisation to ensure that none of the classes prevail.

The future work consisted of an evaluation on an additional classification and on a regression domain and employment of an amending mechanism for removing outdated examples. The additional classification domain is presented in Section 4.2 and regression domain in Section 4.3.

Adapting activity recognition to a person with Multi-Classifer Adaptive Training

Božidara Cvetković^{a,b,*}, Boštjan Kaluža^{a,b}, Matjaž Gams^{a,b} and Mitja Luštrek^{a,b}

^a *Department of Intelligent Systems, Jožef Stefan Institute, Jamova cesta 39, 1000 Ljubljana, Slovenia*

E-mails: bostjan.kaluza@ijs.si, matjaz.gams@ijs.si, mitja.lustrek@ijs.si

^b *Jožef Stefan International Postgraduate School, Jamova cesta 39, 1000 Ljubljana, Slovenia*

Abstract. Activity-recognition classifiers, which label an activity based on sensor data, have decreased classification accuracy when used in the real world with a particular person. To improve the classifier, a Multi-Classifer Adaptive-Training algorithm (MCAT) is proposed. The MCAT adapts activity recognition classifier to a particular person by using four classifiers to utilise unlabelled data. The general classifier is trained on the labelled data available before deployment and retrieved in the controlled environment. The specific classifier is trained on a limited amount of labelled data belonging to the new person in the new environment. A domain-independent meta-classifier decides whether to classify a new instance with the general or specific classifier. The final, second meta-classifier decides whether to include the new instance into the training set of the general classifier. The general classifier is periodically retrained, gradually adapting to the new person in the new environment. The adaptation results were evaluated for statistical significance. Results showed that the MCAT outperforms competing approaches and significantly increases the initial activity-recognition classifier classification accuracy.

Keywords: Adaptation, semi-supervised learning, adaptation to the person, MCAT – Multi Classifier Adaptive Training, activity recognition

1. Introduction

Applications that classify highly variable data with machine learning are often faced with the problem of decreased classification accuracy when deployed in a real-world situation. For example, an activity-recognition classifier may show high classification accuracy when tested in a controlled environment, yet be significantly less accurate with an end-user. This issue could be resolved if enough person-specific data could be labelled in the real-world situation in which the classifier is deployed. However, since this is often impractical and labour-intensive, semi-supervised learning can be used to automatically label the data of specific end-user.

In semi-supervised learning, one or multiple classifiers usually label each instance. A mechanism selects the final class based on their outputs. If the confidence

in this class is sufficient, it is used to label the instance, which is then added into the training set of the classifiers. This approach raises the following three challenges: (i) how to design the classifiers and what data to use for training each of them; (ii) how to choose the final class during the selection process; and (iii) how to decide if an instance will be added to the training set of the classifiers.

This paper introduces a novel algorithm for adapting an activity-recognition classifier to a person. This algorithm is referred to as Multi-Classifer Adaptive Training (MCAT). It addresses all of the above-mentioned challenges. The algorithm is based on the following key contributions: (i) it introduces two classifiers for classifying activities, a general one (trained on general data labelled in a controlled environment), and a specific one (trained on a limited number of labelled instances belonging to a specific person); (ii) the selection of the final class is handled by a meta-classifier, which uses the outputs of both the specific and gen-

*Corresponding author. E-mail: boza.cvetkovic@ijs.si.

eral classifiers; and (iii) the decision about which instance to include in the training set of the general classifier is tackled by an additional meta-classifier. The two meta-classifiers combine the general activity-recognition model with the end-user model in the environment in which the classifiers are deployed. The final contribution of the paper is the training procedure, which takes maximum advantage of all the available data to properly train each of the four classifiers.

The MCAT algorithm was implemented and evaluated on an activity-recognition domain based on Ultra-Wideband (UWB) localisation technology. The individuals had four wearable sensors attached to their clothes (chest, waist and both ankles). The general activity-recognition domain knowledge was induced from the data of the individuals performing activities while wearing the sensors in the controlled laboratory environment. The specific data was obtained from an additional person to whom the system was adapted. The proposed approach was compared to three adaptive approaches: the initial version of the proposed approach [5], the self-learning algorithm [32], and the majority-vote algorithm [20,21].

The experimental results show that the MCAT algorithm successfully increases the classification accuracy of the initial activity-recognition classifier and significantly outperforms all three competing methods.

The rest of the paper is structured as follows. Section 2 reviews the related work on semi-supervised learning approaches and their use in adapting activity recognition. Section 3 introduces the motivating domain. Section 4 explains the MCAT algorithm. Section 5 describes the experimental setup and the results. Finally, Section 6 concludes the paper.

2. Related work

Semi-supervised learning is a technique in machine-learning that uses both labelled and unlabelled data. It is gaining popularity because the technology makes it increasingly easy to generate large datasets, whereas labelling still requires time-intensive human effort. The main idea of the semi-supervised approach is to extend classifiers either trained in supervised or unsupervised mode to label unlabelled data and include additional information into classifiers.

A similar approach is active learning, which also uses supervised learning for the initial classification. However, when the classifier is less confident in its output, a human annotator is consulted [6,27]. Human

interaction is needed when high-risk data must be labelled, such as the data of patients with degenerative diseases. The focus of this paper is on low-risk data, where the cost of active learning is too high.

Four dimensions have been proposed along which semi-supervised learning can be categorised [32]: (i) single- and multiple-classifier; (ii) single- and multi-view; (iii) inductive and transductive; and (iv) classifier and database approaches. The single-classifier methods use only one classifier for the classification task, whereas multiple-classifier methods use two or more classifiers. The key characteristic of multi-view methods is multiple classifiers with different features and data for one classification problem. Single-view methods use classifiers with the same feature vector, but differentiate in the algorithm used for learning. Inductive methods first produce labels for unlabelled data and then train a classifier to use this self-labelled data. Transductive methods only produce labels and do not generate a new classifier. Classifier-based approaches start from one or more initial classifiers and enhance them iteratively. The database approaches discover an inherent geometry in the data and exploit it to find a good classifier. This paper will focus on multiple-classifier, single-view, inductive, classifier-based semi-supervised learning.

Self-training is the most common method that uses a single classifier [32]. After an unlabelled instance is classified, the classifier returns a confidence in its own prediction, or the class probability. If a class-probability threshold is reached, the instance is added to the classifier's training set, and the classifier is retrained. Bicocchi et al. [1] have successfully applied the self-training method to activity-recognition domains. Their activity-recognition classifier was initially trained on camera data and could recognise four atomic activities. The classifier was later used to label accelerometer data, which was intended to be used autonomously. Reported results are 80 percent classification accuracy for the self-trained accelerometer classifier, while the initial camera-based classifier achieves 88 percent classification accuracy. The self-training method can only be used if the initial classifier alone achieves a high classification accuracy, since misclassified instances used for retraining can decrease the classifier accuracy. The self-training has also been successfully applied to several other domains, such as handwriting recognition [10], natural language processing [13], and protein-coding gene recognition [9].

Democratic co-learning [31] is a single-view technique with multiple classifiers. All the classifiers have

the same set of features that are trained on the same labelled data, but with different algorithms. When an unlabelled instance enters the system, all classifiers provide an output. The final label is based on a weighted majority vote among the classifiers. If the voting results have sufficient confidence, the instance is added in the training set of all the classifiers.

Co-training [2] is a multi-view method with two independent classifiers. To achieve independence, the features are split into two feature subspaces, one for each classifier. The classifier that surpasses a confidence threshold for a given instance can classify the instance. The instance is then added to the training set of the classifier that did not surpass the confidence threshold. A major problem of this algorithm is that the feature space of the data cannot always be divided orthogonally. If the features are randomly split, it is possible that classifiers do not satisfy the self-sufficiency requirement [8]. In other words, each classifier must achieve sufficient classification accuracy.

A modified co-training algorithm, En-Co-training [12], was used in the activity-recognition domain. The method uses information from 40 sensors. There are 20 sensors on each leg to identify the posture. The multi-view approach was changed into single-view by using all the data for training three classifiers with the same feature vector and a different learning algorithm. This is similar to democratic co-learning. The final label is chosen by majority voting among the three classifiers. The classified instance is added into the training set of all the classifiers. The reported results show an average classification improvement of 14.5 percentage points.

This paper extends our previous research on MCAT [4]. The MCAT method uses two classifiers. Both are trained with the same algorithm but on different data. A meta-classifier is used to make the final class prediction. The decision whether or not to put an instance into the training set is solved by using another meta classifier, rather than a threshold as seen in all the mentioned methods. In contrast to co-training and en-co-training, our two domain classifiers have the same feature set. Therefore, we do not have the problem of dividing the sets.

3. Motivating domain

The MCAT algorithm is applied to activity recognition, which is a very common task in ambient intelligence. An activity-recognition classifier is usually trained using a machine-learning algorithm on the data

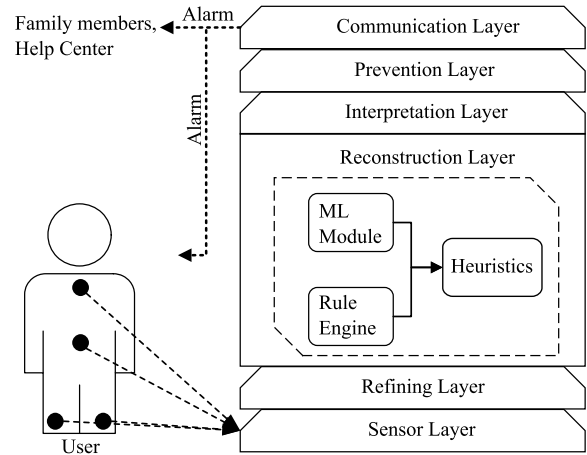


Fig. 1. The multi-layer system architecture of the Confidence system. It consists of six layers. The reconstruction layer is the focus of this paper, since it contains the activity-recognition mechanism.

retrieved from individuals performing predefined activities in a controlled environment, such as a research laboratory. The classifier trained in this fashion will typically report a high classification accuracy when tested in the same environment. However, it is likely that the classification accuracy will decrease when deployed in a new environment with an end-user, since each person tends to have specific characteristics and mannerisms in performing the activities. We faced this problem during the development and evaluation of the Confidence system [7,25] in different environments.

The Confidence system is developed for real-time activity monitoring and detection of abnormal-event (such as falls), or detection of long-term unusual behaviour that results from a developing disease. It is based on a six-layer architecture shown in Fig. 1. The sensor layer serves raw localisation data to the next layer. The refining layer filters out the noise and interpolates the missing data. The reconstruction layer determines the location and reconstructs a person's activity. The interpretation layer interprets the state of the person and provides emergency information. The prevention layer observes the person and detects possible behavioural deviations [18]. The communication layer interacts with the user, relatives, or emergency services. Detailed description of the Confidence system can be found in [16,19]. The paper at hand focuses on the reconstruction layer.

System inputs from the sensor layer are the coordinates of Ubisense location tags [28] (based on UWB localisation technology) attached to the person's waist, chest and ankles. Since the Ubisense system is

noisy, three filters are used, implemented in the refinement layer, to attenuate it. The data is first processed with a median filter. Afterwards the data is processed with an anatomic filter, which applies anatomic constraints. Finally, the data is processed with the Kalman filter. Detailed information on the filters were given in [17].

Tag positions in a specific timespan are represented as snapshots, created with 4 Hz frequency. Each snapshot is augmented with positions and various features for activity recognition and other purposes, such as detection of abnormal behaviour. For each tag the following features are computed: the z coordinate of the tag, the velocity of the tag, the velocity of the tag in the z direction. For each pair of tags the following features are computed: the distance in the z direction, the distance in the xy plane and the total distance. This results in 30 features per time point. To capture the pattern of movement, the snapshot contains a sequence of features across 10 consecutive time points. Each feature vector thus contains 300 features, which is further in the paper referred as a snapshot. The reader is referred to [22] for more details about the features and the snapshot.

The reconstruction layer, which is responsible for activity recognition, can recognise eight atomic (elementary) activities: standing, sitting, lying, falling, on all fours, sitting on the ground, standing up, and sitting down. Activity recognition is done by combining two individual activity-recognition modules, as shown in Fig. 1. The snapshots are processed by a machine-learning classifier and a rule engine working in parallel. Both return the activity that is fed into heuristic Bayesian rules, which provide the final snapshot label. Each module was evaluated in a controlled environment on data of five individuals, using the leave-one-person-out approach. The classification accuracy of the machine-learning (ML) module was 82 percent, while the accuracy of the rule-engine was 80 percent. When combined by heuristics, activity-recognition accuracy increases to 86 percent. However, if machine-learning module classification accuracy decreases, the overall accuracy will also decrease. The classification accuracy is the percentage of correctly classified instances in the dataset. The reader is referred to [23] for more details on the design and evaluation of the activity-recognition modules. The high activity-recognition classification accuracy is essential, since the entire reasoning of the system (detection of falls and abnormal behaviour) is based on it. Adapting activity recognition to each person helps improve the

overall performance of the system. Accurate recognition of atomic activities also contribute to better recognition of complex activities [15]. The rest of the paper is focused on the ML module (Fig. 1).

4. The Multi-Classifier Adaptive Training method (MCAT)

This section describes the MCAT method that improves the classification accuracy of an initial activity-recognition classifier using unlabelled data and auxiliary classifiers. In addition to the general approach, a small amount of labelled data from the new real-world environment and new person is obtained. This is usually done when the system is introduced to the person for the first time.

The initial classifier is trained on activities performed by several people. When using this classifier on a new person, whose physical characteristics are different, the recognition classification accuracy can be low, since each person has a specific way of moving. The MCAT method uses a few activities performed by the new person to learn such specifics. The knowledge of the specifics is later used to adapt the initial classifier to further improve its performance.

4.1. The MCAT algorithm

The proposed MCAT algorithm is shown in Fig. 2 and presented as pseudo code in Algorithm 1. The *general classifier* is the initial classifier trained on the general domain data. This would be the only classifier deployed in a typical application that does not use the proposed method. In Fig. 1, this would be the ML module inside the reconstruction layer. To improve the general classifier, a set of three auxiliary classifiers are proposed: (i) the *specific classifier*, which is trained on a small subset of manually labelled data specific for the person; (ii) the *meta-A classifier*, which decides which classifier (specific or general) will classify an instance; and (iii) the *meta-B classifier*, which decides whether the instance should be included in the training set of the general classifier.

The general classifier is trained on a general set of labelled data available for the activity-recognition domain in a controlled environment. The features are domain-specific. They are listed in Section 3 and denoted as AR (activity recognition) in Table 1. The machine-learning algorithm is chosen based on its performance on the domain.

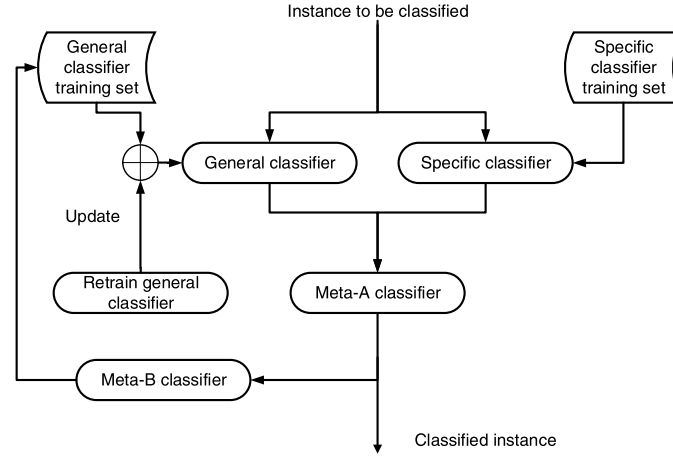


Fig. 2. The work flow of the algorithm proceeds as follows: The method contains two activity-recognition classifiers (general and specific) and two additional meta-classifiers. The meta-A classifier decides on the final class of the instance, and the meta-B classifier decides whether the instance is to be included in the training set of the general classifier.

Algorithm 1 Multi-Classifier Adaptive Training

<pre> 1: Load classifiers G, S, MA, MB; 2: $timer \leftarrow n$ 3: $set \leftarrow \{\}$ 4: procedure MCAT($G, S, MA, MB, inst$) 5: $C_G \leftarrow G.classify(inst)$ 6: $C_S \leftarrow S.classify(inst)$ 7: $ma_{attr} \leftarrow \text{generate features for } MA$ 8: if $MA.classify(ma_{attr})$ is G then 9: $class \leftarrow C_G$ 10: else 11: $class \leftarrow C_S$ 12: end if 13: $mb_{attr} \leftarrow \text{generate features for } MB$ 14: if $MB.classify(mb_{attr})$ is $TRUE$ then 15: $set \leftarrow \text{add}(inst)$ 16: end if 17: $timer \leftarrow timer - 1$ 18: if $time$ is 0 then 19: $G_{dataset} \leftarrow \text{add}(set)$ 20: $G.train(G_{dataset})$ 21: $timer \leftarrow n$ 22: $set \leftarrow \{\}$ 23: end if 24: return $class$ 25: end procedure </pre>	▷ Load all four classifiers ▷ Adaptation time ▷ Set of instances that will be added into the training set ▷ Classify with general classifier ▷ Classify with specific classifier ▷ Generate meta-A features ▷ Select which classifier will classify the instance ▷ Generate meta-B features ▷ Should the instance be included into the G dataset ▷ Start of adaptation when timer expires ▷ Adaptation of the general classifier ▷ Set timer to the initial value ▷ Empty the set ▷ Return class for the current instance
--	---

The specific classifier is trained on a limited amount of labelled data that is specific to the new environment in which the classifiers are deployed (the new person). Note that this limited dataset does not necessary contain all the classes that are present in the

dataset of the general classifier. This may be due to an unbalanced distribution of class labels. For example, in the activity-recognition domain, quick and short movements such as falls are rare. The classes the specific classifier knows are termed basic. The features and the

Table 1

Features per classifier. The first row represent features used for activity recognition (AR), rows from 2-10 are the features to be used for training the meta-A classifier. The feature in the last row is an additional feature for the meta-B classifier, the confidence of the meta-A classifier in its prediction

Features	Classifiers			
	General	Specific	Meta-A	Meta-B
AR	x	x		
C_G			x	x
C_S			x	x
$P_G(C_G)$			x	x
$P_S(C_S)$			x	x
$P_G(C_S)$			x	x
$P_S(C_G)$			x	x
$Basic(C_G)$			x	x
$Equal(C_G, C_S)$			x	x
C_{meta-A}				x

machine-learning algorithm should preferably be the same as those used in constructing the general classifier as shown in the second column of Table 1.

After both classifiers return their outputs (Algorithm 1, lines 5, 6), the meta-A classifier is activated. The meta-A classifier's decision problem is to select one of the two classifiers to classify a new instance (Algorithm 1, lines 8–12). The meta-A classifier should be trained with the machine-learning algorithm performing best on the domain. The features for the meta-A classifier should describe the outputs of the general and specific classifiers as completely as possible, while remaining domain-independent. If domain features are added to the meta-features, the decision of the meta-A classifier will also be based on the specifics of the training data available prior to the deployment of the classifiers. This may be different from the specifics of the situation in which the classifiers are deployed. Our previous work [5] experimentally confirmed that domain features do not contribute to higher classifier accuracy of the classifier. Instead, they can result in overfitting to the specifics of the training data.

Although the features in meta-A classifiers, deployed in different systems need not be exactly the same, the following set of features (also shown in the third column of Table 1) are proposed:

- The class predicted by the general classifier (C_G)
- The class predicted by the specific classifier (C_S)
- The probability assigned by the general classifier to C_G ($P_G(C_G)$)
- The probability assigned by the specific classifier to C_S ($P_S(C_S)$)

- The probability assigned by the general classifier to C_S ($P_G(C_S)$)
- The probability assigned by the specific classifier to C_G ($P_S(C_G)$)
- Is C_G one of the basic classes? ($Basic(C_G)$)
- Are C_G and C_S equal? ($Equal(C_G, C_S)$)

After the selection process, the meta-B classifier then solves the problem of whether or not an instance should be included in the general classifier's training set (Algorithm 1, lines 14–16). The meta-B classifier's output should determine if the current instance contributes to a higher classification accuracy of the general classifier. This question is not trivial and there are several approaches that specifically address it [30]. We use a heuristic that answers the question: "Did the meta-A classifier select the correct class for the current instance?" The heuristic performs well and is computationally inexpensive, so the investigation of more complex approaches will be left for the future work. The features used in the meta-B classifier are the same as those in the meta-A classifier, with one addition: The confidence of the meta-A classifier in its prediction. The features of the meta-B classifier can be observed in the fourth column of Table 1. The meta-B classifier should be trained with the machine-learning algorithm performing the best on the domain.

The instances selected as eligible for inclusion into the dataset of the general classifier are stored in a set (Algorithm 1, line 15). The adaptation is performed when the predefined timer expires (Algorithm 1, lines 18–23). The stopping criteria of the algorithm can either be running out of unlabelled instances in offline adaptation or (in the case of online adaptation) a duration of adaptation in days, hours, weeks or even never.

4.2. Training procedure

The training of the four classifiers requires the data to be divided into subsets so that no classifier is used to classify the data on which it was trained, while maximally utilising all the data. In particular, the meta-A classifier needs data classified by the general and specific classifiers, neither of which should be trained on these data. Furthermore, the meta-B classifier needs data classified by the meta-A classifier, but the meta-A classifier should not be trained on this data.

We propose a training procedure that divides the data into subsets for training both the general classifier and the meta-classifiers. The general classifier is

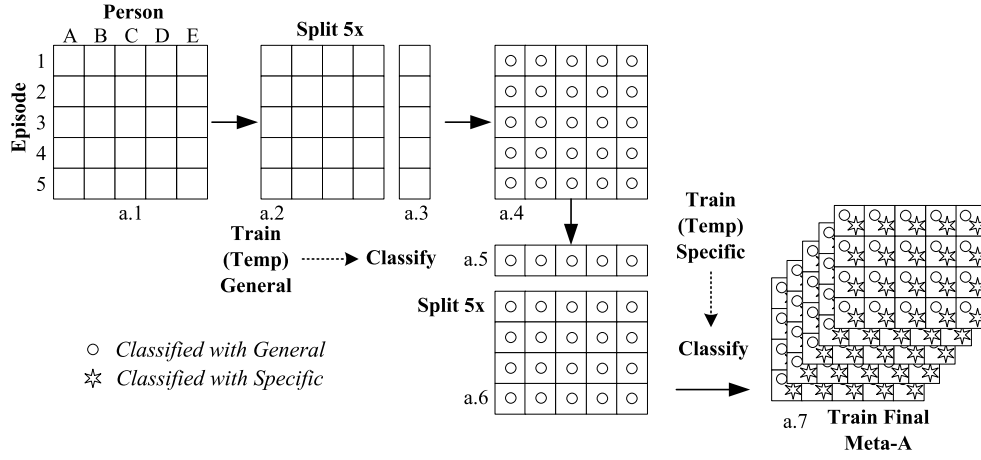


Fig. 3. Preparing the data for meta-A classifier training.

Algorithm 2 Prepare data for meta-A

```

1: Load dataset data;
2: numberOfpersons;
3: dataDot
4: dataMetaA
5: procedure PREPAREDATAMETAA(data)
6:   for i = 0 to numberOfpersons do
7:     G ← train(data[not i])
8:     dataDot[i] ← G.classify(data[i])
9:   end for
10:  for i = 0 to numberOfpersons do
11:    S ← train(data[i][0])
12:    dataMetaA[i] ← S.classify(dataDot[i][not 0])
13:  end for
14:  return dataMetaA
15: end procedure

```

▷ Data classified with general classifier
▷ Data prepared for meta-A training

▷ Figure 3 Step a.2
▷ Figure 3 Step a.3

▷ Figure 3 Step a.5
▷ Figure 3 Step a.6

▷ Result is data for training meta-A, Figure 3 Step a.7

simply trained on the complete dataset, while meta-classifier training is explained in the following paragraphs. Steps are labelled a.1 through a.7 for the meta-A classifier, and b.1 through b.10 for the meta-B classifier. The procedure is general, but for the purposes of this paper the activity-recognition domain is used. Specifically, the publicly available Localisation Data For Person Activity dataset from the UCI Machine Learning Repository [29] will be used. The dataset consists of recordings of five people performing a predefined scenario five times. The scenario is a continuous sequence of activities that represent typical daily activities. Each person had four Ubisense [28] tags attached to the body (neck, waist, and both ankles). Each tag returns its current position. The goal is to assign one of 8 activities to each time frame. The dataset can

be divided five times by the person, and five times by the scenario repetition, or episode (steps a.1 and b.1).

Meta-A classifier training. The procedure is shown in Fig. 3 and Algorithm 2.

Four people are selected for training a temporary general classifier (step a.2), which is used to classify the fifth person (step a.3). This is repeated five times, once for each person. The resulting complete dataset is classified with the temporary general classifier (step a.4). The data classified with the temporary general classifier is represented with dots in Fig. 3. This data is then split five times by the episode. Since the specific classifier should be trained on a small amount of data, one episode for each person is used for training a temporary specific classifier (step a.5). The remaining four episodes are classified with the temporary specific

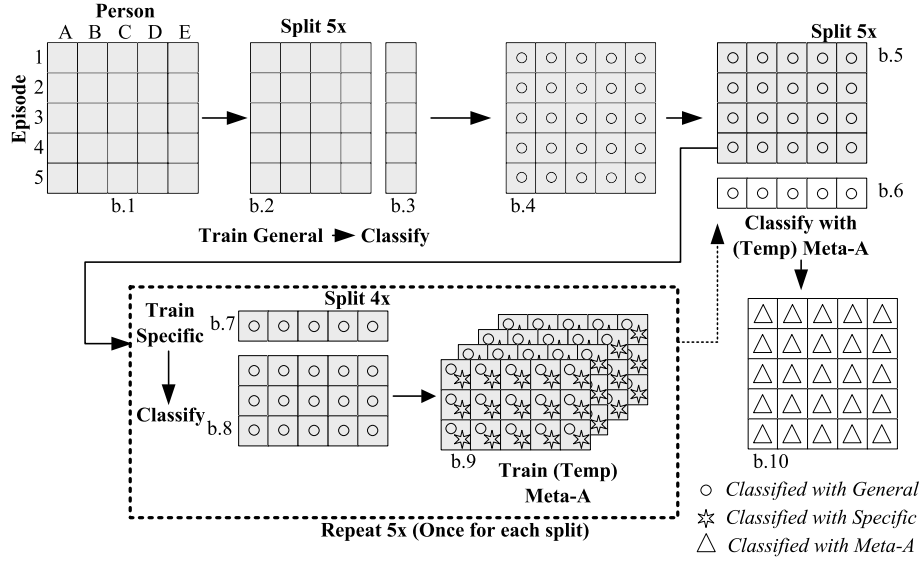


Fig. 4. Preparing the data for meta-B classifier training.

Algorithm 3 Prepare data for meta-B

```

1: Load dataset data;
2: numberOfpersons;
3: numberOfEpisodes;
4: dataTemp
5: dataDot
6: tempDataMetaA
7: dataMetaB
8: procedure PREPAREDATAMETAB(data)
9:   for i = 0 to numberOfpersons do
10:    G ← train(data[not i])
11:    dataDot[i] ← G.classify(data[i])
12:  end for
13:  for j = 0 to numberOfEpisodes do
14:    dataTemp ← dataDot[not j]
15:    for i = 0 to numberOfpersons do
16:      S ← train(dataTemp[i][0])
17:      tempDataMetaA[i] ← S.classify(dataTemp[i][not 0])
18:    end for
19:    tempMetaA ← train(dataTempMetaA)
20:    dataMetaB[j] ← tempMetaA.classify(dataDot[j])
21:  end for
22:  return dataMetaB
23: end procedure

```

▷ Temporary data
 ▷ Data classified with general classifier
 ▷ Data prepared for temporary meta-A training
 ▷ Data prepared for meta-B training

▷ Figure 4 Step b.2
 ▷ Figure 4 Step b.3
 ▷ Result is *dataDot* Figure 4 Step b.4
 ▷ Figure 4 Step b.5
 ▷ Figure 4 Step b.7
 ▷ Figure 4 Step b.8
 ▷ Figure 4 Step b.9
 ▷ Figure 4 Step b.6
 ▷ Result is data for training meta-B, Figure 4 Step b.10

classifier (step a.6). The data classified with the temporary specific classifier are represented with stars in Fig. 3. There is a total of five temporary specific classifiers (one per person), each of which classified four episodes. This finally gives us the data in the step a.7,

which represents the training set for the final meta-A classifier.

Meta-B classifier training. Figure 4 and Algorithm 3 show this procedure. The steps from b.1 to b.4 are identical to the steps from a.1 to a.4 in the meta-

A classifier training procedure. Steps that are the same for meta-A and meta-B training have shaded background in Fig. 4. The data classified with the temporary general classifier (dotted data in step b.4) is then divided by the episode. Four episodes are used to train a temporary meta-A classifier (step b.5), while the remaining episode is kept aside to be classified by this classifier (step b.6). The training then proceeds in essentially the same way as the training of the final meta-A classifier (steps a.5 through a.7). One episode for each person is used to train a temporary specific classifier (step b.7), and the remaining three episodes are classified with the temporary specific classifier (step b.8). This gives five specific classifiers (one per person), each of which was used to classify three episodes, yielding the data in step b.9. This data is used to train the temporary meta-A classifier. The data that was kept aside in step b.6 is now retrieved and classified with the temporary meta-A classifier (b.10). Steps b.5 through b.9 are repeated five times to classify all five episodes, yielding the training set for the final meta-B classifier (step b.10).

4.3. Computational complexity

The computational complexity of instance labelling introduces only a constant increase in computational load, thus only computational complexity for training procedure and retraining of the classifier is analysed. Since MCAT approach may use various algorithms as the core classifiers, only relative changes in computational complexity are analysed.

Let p be the number of people and let e be the number of episodes per person. Suppose a base classifier with the training time complexity O_b is chosen. The training time complexity using $p \cdot e$ data is defined by Eq. (1).

$$O_b(p \cdot e) \quad (1)$$

The meta-A classifier has three levels of training. First, it builds p temporary general classifiers using $(p - 1) \cdot e$ data. Second, it builds $p \cdot e$ temporary specific classifiers using less than 1 episode unit data. And third, the final meta-A classifier is built using $p \cdot (e - 1) \cdot e$ data. The total complexity of training the meta-A classifier is defined by Eq. (2).

The meta-B classifier has four levels of training. Similarly, it first builds p temporary general classifiers using $(p - 1) \cdot e$ data. Second, $p \cdot e$ temporary specific classifiers are built using less than 1 episode unit data. Third, e temporary meta-A classifiers are built using

$p \cdot (e - 2) \cdot (e - 1)$ data. Finally, the meta-B classifier is built on $p \cdot e$ data. The computational complexity is defined by Eq. (3). The upper limit for total training computational complexity is given by Eq. (4).

$$\begin{aligned} O_{metaA} &\leq p \cdot O_b((p - 1) \cdot e) \\ &\quad + p \cdot e \cdot O_b(1) \\ &\quad + O_b(p \cdot (e - 1) \cdot e) \\ &\leq p \cdot O_b(p \cdot e) + p \cdot e \cdot O_b(1) \\ &\quad + O_b(p \cdot e^2) \end{aligned} \quad (2)$$

The MCAT training procedure introduces polynomial increase (of power two) in the number of trained classifiers as well as in the amount of trained data. Since instance labelling introduces only a constant increase, that is, an instance is labelled with four classifiers instead of one, the main source of complexity increase remains in the classifier training procedure. Since the training procedure is run only when the MCAT is initialised, the complexity increase does not affect real-time labelling. Occasional retraining of the general classifier, which is triggered during labelling, requires $O_b(p \cdot e + \epsilon)$ steps, where ϵ is the number of newly arrived instances.

$$\begin{aligned} O_{metaB} &\leq p \cdot O_b((p - 1) \cdot e) \\ &\quad + e \cdot p \cdot (e - 1) \cdot O_b(1) \\ &\quad + e \cdot O_b(p \cdot (e - 2) \cdot (e - 1)) \\ &\quad + O_b(p \cdot e) \\ &\leq (p + 1) \cdot O_b(p \cdot e) \\ &\quad + p \cdot e \cdot (e - 1) \cdot O_b(1) \\ &\quad + e \cdot O_b(p \cdot e^2) \end{aligned} \quad (3)$$

$$\begin{aligned} O_{upper} &\leq [p \cdot O_b(p \cdot e) + p \cdot e \cdot O_b(1) \\ &\quad + O_b(p \cdot e^2)] \\ &\quad + [(p + 1) \cdot O_b(p \cdot e) \\ &\quad + p \cdot e \cdot (e - 1) \cdot O_b(1) \\ &\quad + e \cdot O_b(p \cdot e^2)] + [O_b(p \cdot e)] \\ &\leq (2 \cdot p + 2) \cdot O_b(p \cdot e) \\ &\quad + p \cdot e^2 \cdot O_b(1) \\ &\quad + (e + 1) \cdot O_b(p \cdot e^2) \end{aligned} \quad (4)$$

5. Experimental evaluation

The experimental evaluation focuses on the activity recognition discussed in the previous sections. The main reason why the general classifier will likely not perform well in a real-world environment on a particular person is that each person has their own specifics, such as the height and movement characteristics. The general classifier, which is trained on several people, can then recognise the activities of a “general person”. Obtaining enough training data for a particular end-user is difficult, so the MCAT method is well-suited for solving this problem.

5.1. Experimental setup

The experimental data was collected using the Confidence system described in Section 3. Two different sets of data were collected, one for the classifier training (*the training dataset*) and one for testing the semi-supervised adaptation to the individuals (*the test dataset*).

The training dataset was retrieved from the UCI Machine Learning Repository [29]. The data was divided into segments as discussed in Section 4.2 and used for training the classifiers.

The test dataset consists of the recordings of 10 people performing typical daily activities. All individuals were different than in the training dataset. Each person repeated the scenario five times, which provided an average 2.8 hours of data per person, and 28.5 hours altogether. The scenario was designed to reflect the distribution of the activities during one day of an average person. The scenario contains eight activities: standing, lying, sitting, sitting down, standing up, falling, on all fours, and sitting on the ground. Compared to the training dataset, the scenario in the test dataset contains an additional episode of sitting on the ground, with the respective transitions.

The training dataset contains more labels than the test dataset, so the transition activities were merged as follows. The activities lying down and sitting down were merged into the going down activity. The standing up from lying, standing up from sitting, and standing up from sitting on the ground activities were merged into the standing up activity. The walking activity was merged into the standing activity.

The test dataset was divided as follows in order to create datasets needed for the proposed method: (i) the basic activity dataset had 10 people performing the basic activities (30 seconds per activity), which were

Table 2
Information about the people used in the test dataset

Person	A	B	C	D	E	F	G	H	I	J
Gender	M	M	F	M	M	M	M	F	F	F
Age	27	26	28	28	24	27	25	34	29	27
Height (cm)	194	181	160	184	188	178	187	160	172	168

lying, standing, and sitting; it was used to train specific classifier for each person and (ii) the unlabelled test dataset had 10 people performing the scenario five times (125 minutes per person on average, data for the basic activity dataset is excluded); it was used for the semi-supervised adaptation, which is the core of MCAT.

Demographic information on the people used for MCAT adaptation is shown in Table 2. Additional information about the datasets can be seen in Tables 3 and 4, which show the number of instances, duration and distribution of classes per person of the unlabelled test dataset and the basic activity dataset, respectively, in comparison to the training dataset.

The classifier training used the training dataset, as described in Section 4.2. The meta-A classifier was trained with the Support Vector Machines algorithm [24] and the meta-B classifier used the C4.5 machine learning algorithm [26]. The classifiers could be trained with other machine-learning algorithms, but these two algorithms proved satisfying in experiments. Both were used with the default parameter values as implemented in the Weka suite [14], and the features presented in Section 4.2 and in Table 1. The general classifier was trained with the Random Forest algorithm [3] and the features presented in Section 3 and in Table 1. This algorithm was selected according to the results from previous research [11]. The classifier is 82 percent accurate (as seen in Section 3) when evaluated using the leave-one-person out approach. This implies that one can expect the same classification accuracy when the classifier is deployed in the new environment. However, this is not the case, as Table 5 shows in the column labelled with G (initial general classifier). There, the average classification accuracy is 70.03 percent. The specifics of each person when performing the activities explain this decreased classification accuracy.

The specific classifier was trained with the Random Forest algorithm on the basic activity dataset for each person in the test dataset. The unlabelled test dataset of that person was processed with the MCAT method. The general classifier was retrained after each episode of the test scenario (five episodes per person), to take

Table 3

Number of instances, duration in hours and distribution of classes per person of the unlabelled test dataset in comparison to training dataset used to train the general classifier

	Training dataset	Person									
		A	B	C	D	E	F	G	H	I	J
# Instances ($\times 10000$)	4.1	2.5	2.6	3.0	3.1	3.2	3.4	3.2	3.2	3.5	3.2
Duration (hours)	3.80	2.28	2.36	2.76	2.84	2.93	3.18	2.97	2.96	3.23	2.96
Standing (%)	19.56	30.27	31.56	29.40	31.87	28.84	29.02	24.98	29.27	28.40	31.03
Sitting (%)	16.67	15.82	16.10	16.11	15.82	16.06	14.69	16.19	14.96	14.67	14.96
Lying (%)	33.14	31.57	31.18	30.52	30.62	31.22	30.25	32.40	33.43	33.38	31.43
Sitting on the ground (%)	7.20	15.26	15.69	15.46	15.36	16.65	14.69	19.00	16.18	17.31	15.59
On all fours (%)	3.06	0.49	0.54	0.71	0.57	0.67	0.72	0.66	0.69	0.44	0.65
Falling (%)	1.82	1.23	0.92	0.88	1.17	0.64	0.84	0.87	0.62	0.62	0.79
Going down (%)	4.80	1.42	0.96	1.08	1.25	1.46	1.75	1.37	1.12	1.01	1.20
Standing up (%)	13.75	3.94	3.05	5.85	3.34	4.46	8.04	4.52	3.74	4.17	4.34

Table 4

Number of instances, duration in minutes and distribution of classes per person of the basic dataset used to train the specific classifier in comparison to training dataset used to train the general classifier. The classes that are not present in the basic dataset are omitted from the table

	Training dataset	Person									
		A	B	C	D	E	F	G	H	I	J
# Instances	40968	304	303	303	307	305	304	302	305	306	307
Duration (minutes)	228	1.69	1.68	1.68	1.71	1.69	1.69	1.68	1.69	1.70	1.71
Standing (%)	19.56	33.88	33.66	33.66	33.22	33.44	34.21	34.44	32.79	34.31	33.22
Sitting (%)	16.67	32.57	32.34	33.33	33.55	34.10	32.24	32.78	33.44	32.35	33.55
Lying (%)	33.14	33.55	33.99	33.00	33.22	32.46	33.55	32.78	33.77	33.33	33.22

Table 5

Classifier classification accuracies and comparison of the MCAT method to the transductive approaches: baseline (G and S merged G&S) and MCAT without adaptation (only meta-A); and semi-supervised approaches: self-training (ST) and majority vote (MV); and our: threshold-based MCAT (TB) and full MCAT

Person	Initial classifier (classification accuracy %)			Method Comparison (gain/loss in pp against G)				
	Initial G	Specific S	G&S	Meta-A	ST	MV	TB	MCAT
A	75.28	76.82	−10.57	+3.62	+2.36	+0.38	+3.82	+9.18
B	76.28	60.06	+1.17	+0.58	+0.28	+0.64	+0.70	+7.38
C	62.87	70.85	+8.52	+12.82	−1.18	+3.07	+13.46	+20.58
D	69.55	76.17	−0.21	+8.99	+2.23	+2.10	+9.77	+12.57
E	68.13	74.23	+0.20	+5.78	−1.81	+6.73	+9.76	+16.22
F	73.57	68.18	−4.42	+2.62	+6.07	+3.67	+8.08	+8.86
G	65.42	67.72	5.97	+6.99	−0.21	+8.57	+9.76	+12.60
H	73.45	67.46	−8.02	+0.01	+4.31	+0.41	+4.51	+10.53
I	62.09	68.08	+7.75	+16.81	−0.18	+11.03	+16.98	+17.08
J	73.67	74.17	+1.18	+3.15	+6.87	+5.98	+8.73	+11.65
Average	70.03	70.37	+0.16	+6.14	+1.87	+4.26	+8.56	+12.66

advantage of the instances from the unlabelled dataset that were included in its training dataset.

The MCAT method was compared to five competing methods: two transductive approaches that only

labelled the instance and did not perform any adaptation; and three inductive semi-supervised learning approaches. The transductive approaches are: (i) the baseline approach; and (ii) the MCAT without meta-

B approach. The inductive approaches are: (i) self-training; (ii) majority vote; and (iii) threshold-based MCAT.

The baseline approach merges the training data of the general and the specific classifier into one training set and then trains a new general classifier.

MCAT without meta-B is MCAT without the general classifier adaptation. It builds both the general and specific classifier and uses the meta-A classifier to decide which one to trust.

Self-training is a well-known method for semi-supervised learning [10]. It uses the general classifier for classification. The instances in the unlabelled test dataset with 100 percent classification confidence are added to its training set. A 100 percent confidence is most commonly used as a self-training threshold in related work. Self-training was not specifically adapted to be used with our dataset.

The majority vote is slightly modified Democratic co-learning [31]. The modification is done on the level of decision, since the original paper does not contain enough information for effective reimplementation. The majority vote uses three classifiers trained on the same training set with different machine-learning algorithms. The algorithms that achieved the highest classification accuracy in the evaluation of general classifier with the leave-one-person out approach were used. These were: Support Vector Machines, Random Forest, and C4.5. The instances in the unlabelled test dataset with 100 percent classification confidence were included in the training set of all classifiers as seen in Democratic co-learning.

The threshold-based MCAT is the previous version of the current MCAT algorithm [5]. It uses a threshold rule of 100 percent instead of the meta-B classifier to select the instances to be included in the training set of the general classifier. It was included in the comparison to show the benefits of the meta-B classifier.

The experiment was done in two steps: The classifier training and then the MCAT as a semi-supervised adaptation of the initial general classifier.

The data for each person was processed two times by all the methods. In the first run, the instances selected for inclusion in the training set were assigned a weight of 2 to accelerate the adaptation and in the second run the weight was decreased to 1 to avoid overfitting. If an instance already existed in the training set and was selected for the inclusion again, it was discarded.

5.2. Results

Table 5 shows the absolute difference in classification accuracies between the initial general classifier and the general classifier after adaptation. The left side of Table 5 shows the classification accuracy of the initial general classifier G and the specific classifier S. The right side of Table 5 shows the gain/loss in classification accuracy of the MCAT and the competing methods, compared to the initial general classifier.

The results for the general classifier show a decrease in classification accuracy when used in a different environment than the controlled environment, which had a classification accuracy of 82 percent (cross-validated on the training set). The average classification accuracy on 10 people was 70.03 percent (Table 5, column Initial G). The results of the specific classifier (Table 5, column Specific S) show that it achieved a slightly higher classification accuracy than the initial general classifier in several individual cases, even though it was able to predict only three basic classes (lying, standing, sitting).

The worst results were with the baseline method, where the merged datasets (general and specific) were used for training. The classification accuracy for four people decreased compared to the general classifier. This is because some instances from the specific and general classifier were similar but differed in the label. The results of using the meta-A classifier to select the final class showed that this method outperformed the general classifier by 6.14 percentage points (pp). The higher classification accuracy is due to the knowledge of the basic activities representing 76.67 percent of the dataset. This increase in the classification accuracy reveals that using a classifier for class selection is one reason for the success of the proposed approach. Another reason is the semi-supervised adaptation.

The results of the self-training show that in a few cases, where the general classifier has a low initial classification accuracy, the method performs poorly and further weakens the classifier. The average classification accuracy after adaptation was 71.91 percent. The results of the majority-vote method show that introducing extra classifiers contributes to a gain in classification accuracy compared to the initial classifier. The average classification accuracy after the adaptation with the majority-vote method was 74.29 percent. The average classification accuracy of the general classifier after adaptation using the previous version of MCAT was 78.59 percent. The average gain in the classification accuracy was 8.56 percentage points.

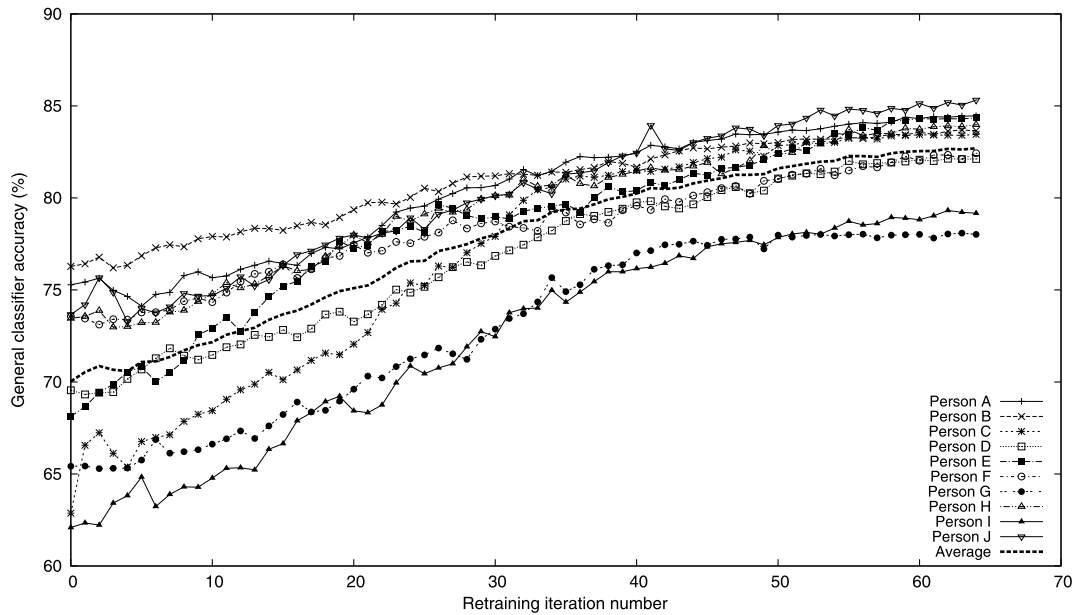


Fig. 5. Adaptation process. Increase in classification accuracy per person and on average.

The results for the MCAT method show the average gain in the classification accuracy of 12.66 percentage points, and the average classification accuracy of the classifier was 82.70 percent. In the best case (Person C), it achieves an increase in the classification accuracy of 20.58 percentage points. The worst case (Person B) had an increase in the classification accuracy of only 7.38 percentage points. Person J achieved the highest classification accuracy of 85.32 percent. The MCAT method outperformed the self-training method (by 10.79 percentage points), the majority vote (by 8.41 percentage points), and the previous version of MCAT (by 4.11 percentage points).

Figure 5 shows the trend of changing the classification accuracy with the adaptation time. The dashed lines represent the general-classifier classification accuracy for each person, and the solid line is the average general-classifier classification accuracy line for the entire process. We observed that the average line increased rather monotonously. The reported final classification accuracy was measured after all the data was used. We speculate that if more data was fed to the MCAT method, we could gain more classification accuracy before achieving convergence.

To further evaluate the results of the MCAT method, a paired two-tailed t-test for statistical significance of the algorithm was performed. The t-test was performed for initial classifiers (general and specific) and for each algorithm the MCAT method was compared to. The re-

sults of the test returned a probability of $6 \times 10^{-4} < p < 4.8 \times 10^{-6}$. The conclusion is that MCAT statistically outperformed all other methods.

6. Conclusion and discussion

The paper is focused on the problem of improving an initial activity-recognition classifier using unlabelled data. The two main contributions of this paper are: (i) a novel approach for specialising the activity-recognition classifier by semi-supervised learning referred to as MCAT; and (ii) a procedure for training multiple classifiers from a limited amount of labelled data that fully utilises the data. The MCAT method for the semi-supervised learning uses auxiliary classifiers to improve the classification accuracy of the initial general classifier. It uses a specific classifier trained on a small amount of labelled data specific to a particular person, in addition to the general-knowledge classifier. The two additional meta-classifiers decide whether to trust the general or the specific classifier on a given instance, and whether or not the instance should be added to the training set of the general classifier.

The MCAT method was compared to two transductive approaches and three inductive semi-supervised approaches. The MCAT method significantly outperformed the baseline approach (by an average 12.51 percentage points) and the MCAT without meta-B (by an average 6.53 percentage points). The MCAT

method also significantly outperformed inductive approaches. It outperformed the self-training (by an average 10.79 percentage points), the majority vote (by an average 8.41 percentage points), and the threshold-based MCAT (by an average 4.11 percentage points). On average, the classification accuracy of the initial classifier improved by 12.66 percentage points. The classification accuracy trend analysis suggest that if there was more data to feed to the MCAT algorithm, a higher classification accuracy could be achieved before convergence.

The outcomes of the experiment were tested for statistical significance with a paired two-tailed t-test. The results show that MCAT statistically significantly ($p \ll 0.01$) outperformed all other methods and the initial classifiers.

To verify if MCAT can significantly contribute to further development of the ambient intelligence applications, many more tests will have to be performed. It is not clear yet if the method is successful only for recognising an individual's activities or if it can be used for general learning tasks. Real-time systems produce large amount of unlabelled data that can be used for adapting the modules to a specific environment or person. Each person has specific mannerisms in performing activities. This can be used to achieve higher classification accuracy on activity recognition. MCAT offers several potential benefits, including accurate recognition of atomic activities, which can also contribute to more reliable recognition of the complex activities.

In addition to more concrete analysis of the algorithms performance, overfitting of the classifiers and drift, the algorithm will have to be evaluated on several other domains. Moreover, the future work includes further investigation of the MCAT algorithm to work on regression domains, for example, estimation of human energy expenditure.

Acknowledgments

The research leading to these results was partly supported by the ARRS under the Research Programme P2-0209, partly by the European Community's Framework Programme FP7/2007–2013 under grant agreement N 214986.

References

- [1] N. Biccocchi, N. Mamei, A. Prati, R. Cucchiara and F. Zambonelli, Pervasive self-learning with multi-modal distributed sensors, in: *2nd IEEE International Conference on Self-Adaptive and Self-Organizing Systems*, IEEE Press, Washington, 2008, pp. 61–66.
- [2] A. Blum and T. Mitchell, Combining labeled and unlabeled data with co-training, in: *11th Annual Conference on Computational Learning Theory*, Morgan Kaufmann Press, 1998, pp. 92–100.
- [3] L. Breiman, Random forests, *Machine Learning* **45** (2001), 5–32.
- [4] B. Cvetković, B. Kaluža, M. Luštrek and M. Gams, Multi-classifier adaptive training: specialising an activity recognition classifier using semi-supervised learning, in: *Aml 2012*, F. Paterno, B. Ruyter, P. Markopoulos, C. Santoro, E. Loenen and K. Luyten, eds, LNCS, Vol. 7683, Springer, Heidelberg, 2012, pp. 193–207.
- [5] B. Cvetković, B. Kaluža, M. Luštrek and M. Gams, Semi-supervised learning for adaptation of human activity recognition classifier to the user, in: *IJCAI Workshop on Space, Time and Ambient Intelligence*, Barcelona, 2011, pp. 24–29.
- [6] S. Dasgupta, Two faces of active learning, *Theoretical Computer Science* **412** (2011), 1767–1781.
- [7] E. Dovgan, M. Luštrek, B. Pogorelc, A. Gradišek, H. Burger and M. Gams, Intelligent elderly-care prototype for fall and disease detection from sensor data, *Zdravniški Vestnik – Slovenian Medical Journal* **80** (2011), 824–831.
- [8] J. Du, C.X. Ling and Z.H. Zhou, When does cotraining work in real data? *IEEE Transactions on Knowledge and Data Engineering* **23** (2010), 788–799.
- [9] G. Feng-Biao and Z. Chun-Ting, ZCURVEV: a new self-training system for recognizing protein-coding genes in viral and phage genomes, *BMC Bioinformatics* **7** (2006).
- [10] V. Frinken and H. Bunke, Self-training strategies for handwriting word recognition, in: *Advances in Data Mining. Applications and Theoretical Aspects*, P. Perner, ed., LNCS, Vol. 5633, Springer, Heidelberg, 2009, pp. 291–300.
- [11] H. Gjoreski, A daptive human activity/posture recognition and fall detection using wearable sensors, Master thesis, Jožef Stefan International Postgraduate School, 2011.
- [12] D. Guan, W. Yuan, Y.K. Lee, A. Gavrilov and S. Lee, Activity recognition based on semi-supervised learning, in: *IEEE International Conference on Embedded and Real-Time Computing Systems and Applications*, IEEE Press, Washington, 2007, pp. 469–475.
- [13] R. Guzmán-Cabrera, M. Montes-Y-Gómez, P. Rosso and L. Villaseñor-Pineda, A web-based self-training approach for authorship attribution, in: *6th International Conference on Advances in Natural Language Processing*, Springer-Verlag Press, 2008, pp. 160–168.
- [14] M. Hall, E. Frank, G. Holes, B. Pfahringer, P. Reutemann and I.H. Witten, The WEKA data mining software: an update, *SIGKDD Explorations* **11** (2009), 10–18.
- [15] L. Kalra, X. Zhao, A.J. Soto and E. Milios, Detection of daily living activities using a two-stage Markov model, *Journal of Ambient Intelligence and Smart Environments* **3**(5) (2013), 273–285.
- [16] B. Kaluža, B. Cvetković, E. Dovgan, H. Gjoreski, M. Gams and M. Luštrek, Multiagent care system to support independent living, *International Journal on Artificial Intelligence Tools* **1**(23) (2014).
- [17] B. Kaluža and E. Dovgan, Glajenje trajektorij gibanja cloveškega telesa zajetih z radijsko tehnologijo, in: *Proc. of the*

- 12th International Multiconference, Information Society – IS, Ljubljana, Slovenia, 2009, pp. 97–100.
- [18] B. Kaluža and M. Gams, Analysis of daily-living dynamics, *Journal of Ambient Intelligence and Smart Environments* **4**(5) (2012), 403–413.
- [19] B. Kaluža, V. Mirchevska, E. Dovgan, M. Luštrek and M. Gams, An agent-based approach to care in independent living, in: *AMI 2010*, B. Wichert, R. Keyson, D. Markopoulos, P. Streitz, N. Divitini, M. Georgantas, N.M. Gomez and A. De Reuter, eds, LNCS, Vol. 6439, Springer, Heidelberg, 2010, pp. 177–186.
- [20] L.I. Kuncheva, C.J. Whitaker and R.P.W. Duin, Limits on the majority vote accuracy in classifier fusion, *Pattern Analysis and Applications* **8** (2003), 22–31.
- [21] B. Longstaff, S. Reddy and D. Estrin, Improving activity classification for health applications on mobile devices using active and semi-supervised learning, in: *4th International Conference on Pervasive Computing Technologies for Healthcare*, IEEE Press, 2010, pp. 1–7.
- [22] M. Luštrek and B. Kaluža, Fall detection and activity recognition with machine learning, *Informatica* **33** (2009), 197–204.
- [23] V. Mirchevska, M. Luštrek, I. Velez and N.G. Vega, Classifying posture based on location of radio tags, in: *AMIF*, P. Cech, V. Bures and L. Nerudova, eds, Ambient Intelligence and Smart Environments, Vol. 5, IOS Press, 2009, pp. 85–92.
- [24] J.C. Platt, *Fast Training of Support Vector Machines Using Sequential Minimal Optimization*, MIT Press, Cambridge, MA, 1999.
- [25] Project Confidence FP7, 2013, <http://www.confidence-eu.org/>.
- [26] Q.J. Ross, *C4.5: Programs for Machine Learning*, Morgan Kaufmann, San Francisco, 1992.
- [27] B. Settles, Active learning literature survey, CS Technical report, University of Wisconsin-Madison, 2009.
- [28] UbiSense, 2013, <http://www.ubisense.net>.
- [29] UCI ML Repository, 2013, <http://archive.ics.uci.edu/ml/>.
- [30] C. Yanhua, R. Manjeet, D. Ming and H. Jing, Non-negative matrix factorization for semi-supervised data clustering, *Knowledge and Information Systems* **17** (2009), 355–379.
- [31] Y. Zhou and S. Goldman, Democratic co-learning, in: *16th IEEE International Conference on Tools with Artificial Intelligence*, IEEE Press, Washington, 2004, pp. 594–602.
- [32] X. Zhu, Semi-supervised learning literature survey, CS Technical report, University of Wisconsin-Madison, 2005.

4.2 Use Case 2: Context Recognition Task – Classification

In the last decade, we witnessed a rapid increase in the development and availability of sensors that are currently embedded into almost all personal electronic devices. One such device is the smartphone, which is very convenient for context monitoring since we rarely leave home without it and its processing unit is capable of real-time computing. The contexts can be seen as high-level activities, such as the person being outdoors doing errands or leisure activities or being indoors doing home chores, leisure activities, eating, preparing food, working, etc. The contexts are drawn from the time, sound, location, performed low-level activity, and intensity of the performed activity (Cvetković, Mirchevska, Janko, & Luštrek, 2015).

If one wants to use person-specific data such as sound, location, and time component for analysis of the target person's context, the general model trained on data of other people may not be sufficient for accurate context recognition.

4.2.1 Related Publication

In the enclosed paper (Cvetković, Janko, Romero, et al., 2016), we use MCAT to personalise the model for the recognition of mentioned contexts. Here we summarise the content of the paper.

4.2.2 Dataset

The dataset in the paper contains smartphone and ECG data of nine volunteers who were instructed to carry a smartphone with the data-collection application for two weeks. They were asked to lead their life as usual and to label the following events that occur at home as home (home-chores, home-leisure, food preparation), eating, exercise, work, events that occur outdoors as out (out-errands, out-leisure) , sleep and transport.

4.2.3 Training Procedure

The dataset is in line with the general requirements presented in Section 3.2.1. It can be partitioned by person and by context, which is needed for the training procedure presented in Section 3.2.3.

4.2.3.1 Data partitioning

First we identified the labels that will be included in the person-specific model (called the User-specific model). These were exercise, home and eating. Since we had two weeks of data, we selected the first week for training and the second for adaptation and evaluation. The data for the person-specific model was in this case selected on first occurrence in the first week (the user would be instructed to label the first occurrence of the events). The rest of the first week data was divided to be used for training the selection model (called Meta model selection) and inclusion model (called Meta model selection).

4.2.3.2 Machine-Learning Model Training

The general model, called Person-independent (PI) model, is trained on general data with the location data preprocessed into a semantic location feature (home, work, out). The person-specific model is trained on the dataset of first occurrences of the defined events. The selection and inclusion models are trained with the training procedure with repetitions (Section 3.2.3) presented in the paper enclosed in the Section 4.1 in which a day is

considered a repetition. The evaluation of the models is done with cross-validation during the training procedure.

4.2.4 Adaptation Procedure

The paper evaluates the use of MCAT for personalisation of the context-recognition model on nine people using data from sensors embedded in an average smartphone. The personalisation results are compared against one person-independent approach, two single-model personalisation approaches, and one multi-model personalisation approach. The person-independent approach uses a person-independent model (PI method in the paper) trained on general data of other people (with the semantic location extracted). The single-model personalisation approaches are using a person-dependent model (PD method in the paper) for which we take the collected week of data (labelled) and train it in a supervised manner, and the most common approach to personalisation, merging a small amount of person-specific data (which is labelled at the first occurrence) with the general dataset (PIA method in the paper). The multi-model approach is MCAT for which we exclude the inclusion model and replace the selection model with a heuristic (PIAH method in the paper). The results show that MCAT is superior to other compared approaches and even surpasses the accuracy of the person-dependent approach (trained on one week of labelled data) by nine percentage points, since, for some people in the dataset, the first week differed from the second week, which could not be captured from the data of the first week, but could be adapted during the second week with the use of MCAT.

4.2.5 Summary

The main conclusion is that SSL should be used in tasks such as context recognition since daily patterns differ between people and even change in the case of the same person; therefore, the used models need to be personalised accordingly.

The experiment was done on a dataset that reflects the distribution of contexts as they occur in a normal daily life of a person (unbalanced dataset). We showed that we can also successfully adapt to the classes that do not happen often. We evaluated the accuracy and F-measure of the general model and showed that none of the classes experience a decrease in any of the measures (Table 2 and Table 3 in the included paper). However, we did not explore the performance of the general model if certain class never happens during the personalisation. We assume that this would result in other classes prevailing the missing one, which is a shortcoming. This needs further investigation and employment of an amending mechanism that will ensure a proper structure of the dataset.

Activity Recognition for Diabetic Patients Using a Smartphone

Božidara Cvetković^{1,2}  · Vito Janko^{1,2} · Alfonso E. Romero⁴ ·
Özgür Kafalı³ · Kostas Stathis⁴ · Mitja Luštrek^{1,2}

Received: 29 February 2016 / Accepted: 8 September 2016 / Published online: 8 October 2016
© Springer Science+Business Media New York 2016

Abstract Diabetes is a disease that has to be managed through appropriate lifestyle. Technology can help with this, particularly when it is designed so that it does not impose an additional burden on the patient. This paper presents

an approach that combines machine-learning and symbolic reasoning to recognise high-level lifestyle activities using sensor data obtained primarily from the patient's smartphone. We compare five methods for machine-learning which differ in the amount of manually labelled data by the user, to investigate the trade-off between the labelling effort and recognition accuracy. In an evaluation on real-life data, the highest accuracy of 83.4 % was achieved by the MCAT method, which is capable of gradually adapting to each user.

This article is part of the Topical Collection on *Mobile Systems*.

Electronic Supplementary Material The online version of this article (doi:[10.1007/s10916-016-0598-y](https://doi.org/10.1007/s10916-016-0598-y)) contains supplementary material, which is available to authorized users.

Keywords Activity recognition · Smartphone · Lifestyle · Diabetes

✉ Božidara Cvetković
boza.cvetkovic@ijs.si

Vito Janko
vito.janko@ijs.si

Alfonso E. Romero
alfonso.romero@rhul.ac.uk

Özgür Kafalı
rkafali@ncsu.edu

Kostas Stathis
kostas.stathis@rhul.ac.uk

Mitja Luštrek
mitja.lustrek@ijs.si

Introduction

Currently around 415 million people suffer from diabetes, and by 2040 their number is expected to increase to 642 million [3]. Diabetes is a metabolic disease in which the body's cells do not respond properly to insulin or the insulin production is inadequate, so glucose is not removed from the blood to be used for energy. Since diabetes has no cure, it has to be managed through medication and appropriate lifestyle. Key lifestyle activities for diabetic patients are eating and exercise – eating causes glucose in the blood to rise while exercise speeds up its absorption.

Technology for activity monitoring can help the patients better manage their lifestyle, and provide their physicians an insight into the patients' life. In the COMMODITY12 project [2], we developed a personal health system for diabetes intended both for the patients and their physicians. In order not to burden the patients with unnecessary devices, it primarily relies on a smartphone. The phone can be augmented with an ECG monitor, which was introduced for

¹ Jožef Stefan Institute, Jamova cesta 39, Slovenia

² Jožef Stefan International Postgraduate School, Jamova cesta 39, Slovenia

³ North Carolina State University, Raleigh, NC 27695-8206, USA

⁴ Royal Holloway, University of London, Egham, TW20 0EX, UK

the management of cardiovascular co-morbidities [13] and diagnosis [15], but is also used for activity recognition.

Activity recognition is a common task in telemonitoring, because physical activities characterise the patients' lifestyle and provide the context for more direct health-related observations. For example, in diabetes blood glucose needs to be interpreted in the context of the two most important activities which influence the blood glucose level: eating and exercise. The recognition of other activities gives an insight into the patients' lifestyle which makes it possible to give more appropriate lifestyle advice and monitor the progress of the disease and its complications.

Activity recognition is usually done by utilising environmental or wearable sensors or even both [17]. This paper focuses on wearable sensors. Most approaches for activity recognition with smartphone sensors and other wearable devices tackle low-level activities such as Walking, Sitting and Lying. In contrast, this paper describes the recognition of high-level activities such as Work, Eating, Exercise and Home activities.

Low-level activities are commonly recognised with machine learning from a sliding window passed over a stream of acceleration data [4, 5]. Dernbach et al. [8] used the low-level approach for high-level activities and reached the accuracy of barely 50 %. Lee & Cho [18] applied hierarchical hidden Markov models to accelerometer data to first determine low-level activities, and from those high-level activities (Shopping, Taking_bus, Walking). They reached the precision of around 80 %, but their set of activities was very limited and the users carried the phone in their hand.

Using other smartphone sensors, such as the GPS, improves activity recognition. This way, Lin [19] classified Work, Sleep, Leisure, Visit, Driving and Other with conditional random fields. He achieved the accuracy of 86 %. Wang et al. [21] additionally used the microphone and light sensor. They determined the users' state (Working, Home_talking, Place_speech etc.) with a rule-based system. They achieved the accuracy of around 90 %, but the users' home and office wi-fi names were known to the system in advance, and some states were defined expressly in terms of ambient sounds.

Helal et al. [11] developed a platform specifically for monitoring diabetic patients. They used hidden Markov models to recognize activities such as Washing_hands, Cooking and Eating_oatmeal. They relied on ambient sensors, restricting their platform to the patients' homes. Amft et al. [1] detected eating from video, and recognised different types of eating by analysing body movement while eating. However, this was done with fairly intrusive inertial, EMG and other sensors.

Unlike in most related work, the system described in this paper attempts to recognise all the users' activities, including ambiguous ones (e.g., cycling can be exercise, transport

or a part of shopping), and was tested in a completely natural setting. The core methods are based on machine learning. We investigated various options differing in the number of models used, and the amount of data labelled by the person using the system. The recognised activities are refined by rules encoded in Event Calculus [16] and optimised for speed and real-time purposes.

Activity Recognition Approach

Our activity recognition approach is composed of three main steps: feature extraction, core activity recognition, and refinement of the recognised activities. Feature extraction transforms raw sensor data into features that can be used in the next two steps. The core activity recognition mainly uses machine-learning to recognise high-level lifestyle activities. Finally, we refine the recognised activities with symbolic reasoning.

Features for activity recognition

The features are extracted from smartphone sensors and optionally from an accelerometer-equipped chest-worn ECG monitor. They are computed over one-minute windows and belong to five groups: sound, location, acceleration, heart-rate and respiration-rate.

Sound features are extracted from the ambient sound recorded with the smartphone's microphone using the jAudio library [12]. We record 100 ms of sound out of each second in a minute (to preserve the users' privacy) and further split it to 20 ms sub-windows. The computed features are the average spectral-centroid, zero-crossing, mel-frequency-cepstral-coefficient, method-of-moments values and linear-predictive-coding for each one-minute window [6].

Location features are extracted from two sources: (i) the smartphone's GPS receiver, from which we take the geographical coordinates, and from (ii) the wi-fi module, from which we take information about the visible access points and signal strength. The features from both sources are clustered into 20 clusters each with the hierarchical clustering as implemented in the Weka machine-learning suite [10]. The clusters are afterwards assigned one of three semantic locations – residence, work and elsewhere – with the following algorithm:

- The importance of each cluster is computed as the fraction of time the user spends in the cluster in a day (heatmap in Fig. 1).
- Clusters with the highest free-day importance, that also have workday importance above 0, belong to the location residence (blue semantic location in Fig. 1).

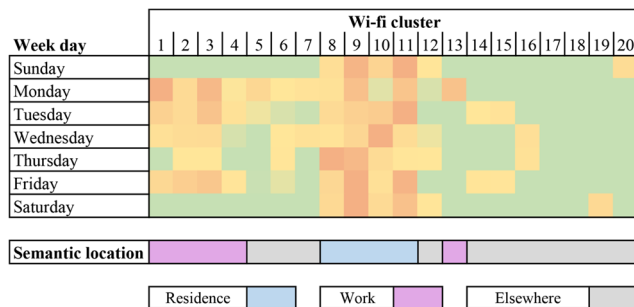


Fig. 1 Heatmap for wi-fi clusters and the transformation into semantic locations for an example user

- Clusters with highest workday importance above 0, which are never visited on free days, and aggregate into 50 % of the time, belong to the location work.
- All other clusters belong to the location elsewhere.

Each wi-fi cluster is assigned the correct semantic location with the accuracy of 84 % and each GPS cluster with 81 %. In Fig. 2 we present a GPS plot (latitude, longitude) with GPS clusters (the size of the circle represents the size of the cluster) and their corresponding semantic locations. Since GPS can only be used outdoors due to its poor signal indoors, the wi-fi signal clustering is needed for more fine-grained location indoors. Fig. 3 shows workplace clusters from Fig. 2 (GPS clusters 1, 2 and 20) and Fig. 4 the residence cluster from Fig. 2 (GPS cluster 18) as a composition of wi-fi clusters, assigned semantic location and the distribution of activities per cluster. We can observe that wi-fi clusters enable detection of more fine-grained semantic locations (in brackets after each cluster number).

GPS work location (GPS cluster 1, 2 and 20) is composed of five wi-fi clusters (Fig. 3): wi-fi-clusters 1 and 2 are clean work clusters, wi-fi cluster 2 is a mixed cluster corresponding to a cafeteria close to the office, the wi-fi cluster 4 is probably assigned to the parking lot close to the office (Exercise activity represents the running to the car), and wi-fi cluster 13 represents the restaurant near the workplace.

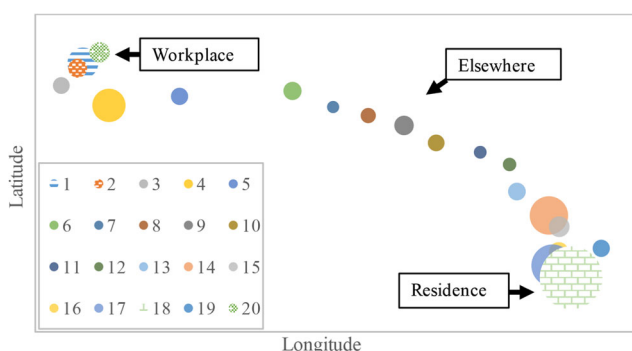


Fig. 2 All 20 GPS clusters and their corresponding semantic locations for an example user



Fig. 3 Wi-fi clusters composing GPS clusters 1, 2 and 20 (GPS work semantic location). Each wi-fi cluster is labelled with the recognised wi-fi semantic location and presented with the distribution of activities

GPS residence location (GPS cluster 18) is composed of five wi-fi clusters (Fig. 4): wi-fi cluster 7 represents the outdoors of the residence, while the remaining four may represent different parts of the house. Wi-fi cluster 8 and 10 probably represent the dining room, wi-fi cluster 9 the bedroom and wi-fi cluster 11 the living room.

Additional location features are the wi-fi availability determined by the strength of the wi-fi signal, GPS availability determined by the visibility of satellites, velocity, and location category according to the Foursquare location API [9].

Acceleration features are extracted from the smartphone's and/or ECG monitor's accelerometer. They are the user's most common low-level activity, the percentage of each low-level activity within the one-minute window, the user's average estimated expended energy and the percentage of the estimated low, medium and vigorous intensity of the activity. They are computed with our recent method [5] that can seamlessly use the phone, ECG monitor or both.

Heart-rate features and *respiration-rate features* are extracted from the ECG monitor if present. The features are the (i) minimum, (ii) maximum and (iii) average heart-rate or respiration-rate within each one-minute window.

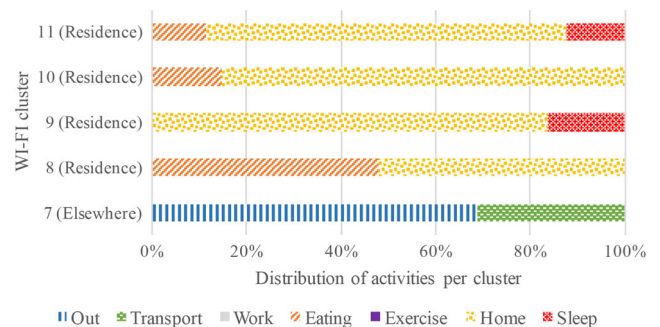


Fig. 4 Wi-fi clusters composing the GPS cluster 18 (GPS residence semantic location). Each wi-fi cluster is labelled with the recognised wi-fi semantic location and presented with the distribution of activities

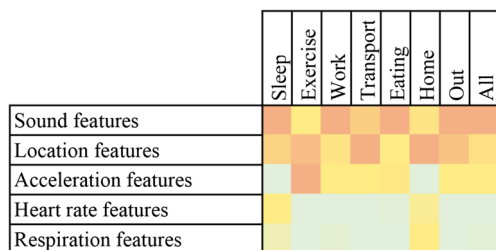


Fig. 5 Correlation heatmap between groups of features and activities

The last three categories of features use the ECG monitor. The quality of the acceleration features is not much degraded without the ECG monitor [5], and according to the heatmap presented in Fig. 5, the heart-rate and respiration features are not very important, so the ECG monitor is not essential.

Core methods for activity recognition

Since each person performs the daily activities differently, our activity-recognition system can benefit from labelled activities of each user. However, since labelling represents a burden for the user, our goal was to investigate the trade-off between the amount of labelled data and the recognition accuracy. We developed five methods, some of which use a single model trained with machine learning (Fig. 6) and some multiple models (Fig. 7):

- *Person-dependent method (PD, Fig. 6a)* is a single-model method, where the model (PD model) for each user is trained on that user's data only (PD dataset). This requires the user to label all the activities to be recognised during the training phase – one week in our experiments. This can be fairly burdensome, and the quality of the recognition strongly depends on the user's conscientiousness.
- *Person-independent method (PI, Fig. 6b)* is a single-model method, where the model (PI model) for each user is trained on data of people other than the user (PI dataset). This approach does not require any labelling from the user.
- *PI method with person-specific data (PIA, Fig. 6c)* is a single-model method, where the PI dataset is augmented with user-specific data of Exercise and Eating

(these two activities are the most relevant for diabetic patients, but any activities could be included in principle). This approach requires the user to label Eating and Exercise during the training phase, after which an augmented model is trained (PIA model).

- *PI and person-specific model combined with heuristics method (PIAH, Fig. 6a)* is a two-model method. The PI model is trained on the PI dataset as in the PI method. The user-specific model is trained on data of Exercise and Eating labelled by the user, requiring the same effort as the PIA method. The outputs of the two models are merged with a heuristic method into the final activity. The reader is referred to [6] for more details.
- *Multi-Classifier Adaptive Training (MCAT, Fig. 6b)* is a semi-supervised learning method that can adapt to a user while the system is in use. It is composed of two domain-dependant models and two domain-independent meta-models. The domain-dependant models are the PI model and a small user-specific model. The user-specific model is trained on only the first occurrences of Eating, Exercise and Home activity, requiring minimal labelling effort from the user (again, any activities could be used in principle). The outputs of the two models are merged with a meta-model that selects which of them to trust. The MCAT algorithm also utilises a meta-model that decides whether each instance should be included in the PI dataset or not. The PI model is then periodically retrained, becoming ever more personalised. While the MCAT method does not need much user-labelled data, it does need a fairly large amount of unlabelled data to gradually adapt to the user. The reader is referred to our earlier work on a different problem [7] for more details.

All five methods are summarised in Table 1 according to the labelling effort needed from the user.

Refinement of recognised activities

Activity recognition using a mobile phone is error prone because mobile sensors are noisy. Sensor noise fed in classification algorithms propagate errors in predictions and as a result an activity being recognised can be incorrect. On top of this, classification algorithms are oblivious to common sense knowledge, thus it is possible for example to confuse

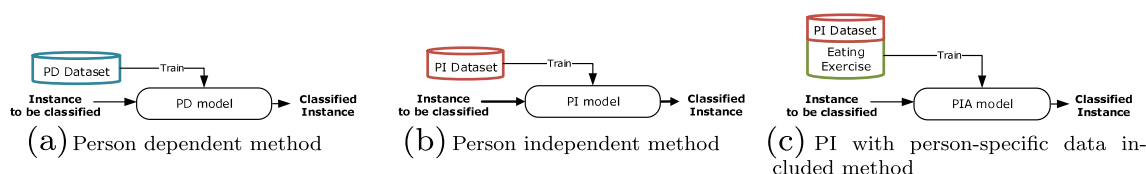


Fig. 6 Single-model methods

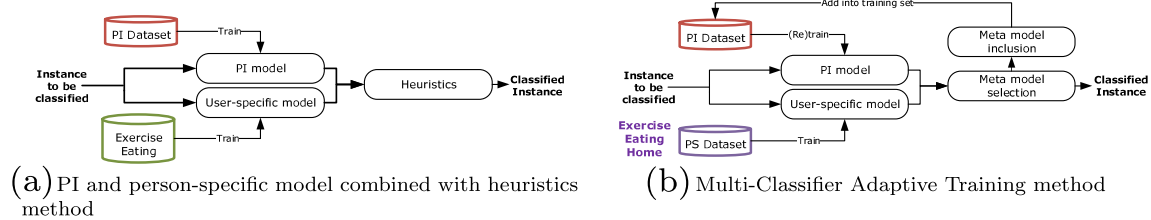


Fig. 7 Multi-model methods

that a person is eating while the person is actually exercising, even if it is commonly known that eating and exercising to be incompatible.

To refine confused predictions we study how symbolic rules can be used with classification algorithms. The idea is to refine the classification models and to act as interfaces with techniques used to monitor a diabetic patient and reason about his condition as in the agent-oriented system discussed in [13]. More specifically, Kafalı et al [14] show how activity recognition from sensors can be reasoned upon with symbolic rules to arrive at more complex conclusions, for example, that while a person was walking home from work has fallen because he had a hypoglycemic attack. This framework is based on a specific representation of activities where events (or observations) that happen at different times start, suspend, resume and achieve activities according to a user's goals specified at the mobile phone's settings.

We combine the Kafalı et al framework together with predictions provided by a classifier to eliminate misclassified activities using symbolic rules of the form: *Conclusion* $\leftarrow$ *Conditions* where the *Conclusion* is a predicate that holds to be true if the set of predicates in the *Conditions* hold to be true. A predicate name must be a constant, while each argument within a predicate can either be a constant or a variable. Constants are represented by identifiers starting with a lower-case letter and variables by identifiers starting with an upper-case letter. A constant stands for a specific entity, and different constants stand for different entities. A variable can stand for any entity, and different variables can stand for the same entity. A single underscore '_' denotes an anonymous variable and means 'any term'.

Table 1 Comparison of the amount of manually labelled data by the user in the core methods for activity recognition

Method	Labelled data by the user	
	Activities	Duration
PD	All	1 week
PI	None	None
PIA	Eating, Exercise	1 week
PIAH	Eating, Exercise	1 week
MCAT	Eating, Exercise, Home	First occurrence

Removal of orphan activities (R1). If the classifier predicts an activity A at a stage identified by time T+1, an activity B at the previous stage identified by time T, and the activity A at a previous stage identified by T-1, then the activity B is replaced with activity A at the stage identified by T. Such a generic rule eliminates predictions such as a person being at home, then being at work, then being back at home in three consecutive and very close to each other times, and replaces them with the person being at home throughout these times.

```
possible_at(activity(A, P)=active, T)  $\leftarrow$ 
  predicted(A, _, T+1),
  predicted(B, _, T),
  A  $\neq$  B,
  predicted(A, _, T-1).
```

As an example of how this rule can be applied, we show the following set of predicted activities taking from the dataset: (12:01:48, Eating), (12:03:00, Transport), (12:04:12, Transport). In that case, the second prediction will be changed into Eating, as it seems unlikely that there is one minute of Transport.

Learning person-dependent activity intervals (R2) captures whether an activity is valid at a given time. For instance, we know that an average person usually eats at least three times a day. If we know that a person eats at specific times, we validate the predicted eating activity. Moreover, we learn such valid activity intervals for each activity using the training datasets available for each individual person.

```
possible_at(activity(A, P)=active, T)  $\leftarrow$ 
  member(A, [sleep, work, eating]),
  valid_interval(P, A, Interval),
  member(Ts-Te, Interval),
  Te  $\geq$  T, T  $\geq$  Ts.

valid_interval(p1, sleep, [30-730]).
valid_interval(p1, work,
  [915-1200, 1300-1730]).
valid_interval(p1, eating,
  [845-900, 1200-1230, 1800-1830]).
```

For a certain person of the dataset, for example, the maximal sleep interval across all the training data is [00:16:12, 10:12:12]. A prediction of Sleep found at 20:40:48 will be

changed into the second more likely prediction (Home, in this case).

We apply the rules in the following order: R1 then R2. R1 applies to all activities, and R2 applies to Sleep, Work and Eating.

Experiments

The activity-recognition approach was evaluated on the recordings of nine volunteers who wore the smartphone and the ECG monitor for two weeks each. All the methods used the first week of recordings for training and the second week for testing. Person-independent methods (PI, PIA, PIAH and MCAT) were evaluated with the leave-one-person-out approach, meaning that models were trained on the data of eight people (first week) and tested on the ninth (second week), repeated once for each person. The first week of each test person was used to determine the semantic locations (residence, work and elsewhere), and to train user-specific classifiers where applicable (see Table 1).

The models in the core activity-recognition are ensembles of base models trained with SVM, J48, Random Forest, JRip, AdaBoost and Bagging algorithms with default parameters as implemented in the Weka machine-learning suite [10]. The exception are the user-specific and two meta-classifiers in the MCAT method, the former of which is trained with the Random Forest algorithm and the latter two with the J48 algorithm.

Dataset

The nine healthy volunteers (eight male, one female, aged from 24 to 36) were asked to carry the smartphone with the data-collection application as much as possible for two weeks, in any pocket they wanted (or in a bag), and to wear the ECG monitor each day until the battery ran out. They were asked to lead their life as usual. While some of them had fairly regular daily routines, others had unusual eating patterns, were staying at a different place during the week and weekend, took trips etc., so the resulting dataset is quite challenging for activity recognition. On average, we collected 7.5 hours of recordings per day with the ECG monitor and 11 hours with the phone.

The smartphone application was collecting the sensor data and was used for labelling the activities: Home-chores, Home-leisure, Food preparation, Eating, Exercise, Work, Out-errands, Out-leisure, Sleep and Transport. We later merged Home-chores, Home-leisure and Food preparation into Home, and Out-Errands and Out-leisure into Out, since these activities proved impossible to distinguish.

Table 2 Average classification accuracy (%) and micro-averaged f-score for the evaluated core methods and after correction with symbolic reasoning

Methods	Core methods		Refining method	
	Accuracy	F-score	Accuracy	F-score
Default	35.9	0.35	35.9	0.35
PD	72.9	0.73	74.3	0.73
PI	71.3	0.71	73.7	0.72
PIA	66.4	0.66	69.9	0.67
PIAH	72.0	0.72	73.7	0.72
MCAT	81.8	0.82	83.4	0.82

Results

Table 2 shows the results of all the evaluated methods in terms of micro-averaged classification accuracy and f-score, both before and after the refinement with symbolic reasoning. In Table 3 we present the f-scores per activity after the refinement. The default model, which serves as a baseline for comparison, always outputs the most common class, which is the work activity.

We first evaluated the two simplest methods: PD, which requires the user to label all the activities during the training week, and PI, which requires no user labelling. The advantage of the PD method is that it can adapt to the user, but it has less training data available. The advantage of the PI method is more training data (from eight people vs. one), but it is not adapted to the user. The results of both methods proved similar, so apparently their advantages and disadvantages balanced out. The PI method was more successful at recognising the Transport and Out activities, which reflects their generality, while the PD method was more successful at other activities, particularly Eating and Exercise, which suggests they are more user-specific.

Table 3 Average f-score for the activities after machine-learning + symbolic approach approach

Activity	Methods				
	PD	PI	PIA	PIAH	MCAT
Sleep	0.80	0.72	0.71	0.72	0.80
Exercise	0.62	0.40	0.58	0.40	0.91
Work	0.88	0.85	0.81	0.85	0.90
Transport	0.58	0.69	0.71	0.69	0.78
Eating	0.34	0.22	0.24	0.22	0.68
Home	0.81	0.75	0.70	0.75	0.83
Out	0.35	0.51	0.51	0.51	0.69

Table 4 Confusion matrix of the MCAT method refined with the symbolic reasoning

TRUE	PREDICTED						
	Sleep	Exercise	Work	Transport	Eating	Home	Out
Sleep	661 (-2)	0 (0)	0 (0)	0 (0)	0 (0)	292 (2)	0 (0)
Exercise	0 (0)	859 (11)	3 (-1)	5 (-3)	8 (-5)	53 (-1)	2 (-1)
Work	31 (0)	18 (0)	11322 (68)	173 (-4)	110 (-6)	922 (-59)	49 (1)
Transport	9 (-2)	13 (-1)	244 (6)	2927 (11)	18 (-3)	235 (-2)	142 (-9)
Eating	1 (0)	1 (0)	50 (-3)	10 (-2)	1329 (15)	497 (-11)	200 (1)
Home	164 (-67)	42 (2)	495 (-11)	71 (0)	138 (-18)	9377 (115)	60 (-21)
Out	0 (0)	20 (-2)	390 (-1)	757 (-5)	197 (-12)	822 (-10)	2884 (30)

The brackets contain the contribution of the refining method

We continued with the evaluation of methods combining user-independent and user-specific data and models. The PIA method, which simply adds user-specific Exercise and Eating data to its model, significantly improved the recognition of Exercise, but had less success with Eating and was the least accurate overall. The PIAH method, which heuristically merges a person-independent and person-specific model, proved almost identical to the PI method overall. It did yield a higher accuracy for four people (81 % – 85 %), but a lower one for the rest (50 % – 76 %). This indicates that some recorded people had substantially different training and test weeks.

At the end we evaluated the MCAT method [7], for which the user was required to label a single Eating, Exercise and Home activity. The MCAT method gradually adapts the PI model to a current user by retraining it on automatically labelled and selected data. It was retrained three times in our experiments. The adaptation improved the recognition of all the activities, not only those for which it had user-specific data. The confusion matrix of the MCAT method with the symbolic refinement is presented in Table 4. Most of the activities were miss-classified as the Home because they occurred at the residence location (the location being the most important feature for activity recognition). For example, most of the recorded people exercised at their residence, and all of them ate and slept there. The miss-classification of Transport occurred during transitions from/to Home, Out and Work. Eating was confused not only with Home, but also Work, Out and even Transport. This happened because people were eating in the same locations and postures as doing home activities (e.g., eating at the kitchen table or on the sofa where they also read and watch TV), working (e.g., eating at their office desk), eating while shopping or commuting.

Conclusion

In this paper we presented an approach that combines machine learning and symbolic reasoning to recognise high-level lifestyle activities of diabetic patients using sensor data obtained primarily from the patients' smartphones. Our motivation was that diabetes has to be managed through appropriate lifestyle – particularly eating and exercise, which directly affect blood glucose. Lifestyle is important for other diseases as well, so our work has applications beyond diabetes. The main challenge was that patients' lifestyles and consequently ways they perform activities are very diverse, so activity recognition should be adapted to each individual. But since labelling represents a burden for the user, we focused on investigating the trade-offs between the amount of labelled data and the recognition accuracy.

Our first results showed that the two extreme cases – relying only on user-labelled data and completely eschewing user-labelled data – are comparable. This suggested that the strengths of both methods should be combined. We did this in increasingly sophisticated fashion: by simply merging general and user-specific training data (PIA method), by heuristically merging the outputs of general and user-specific classifiers (PIAH method), and by utilising semi-supervised learning approach (MCAT method). The last method required the least labelled user-specific data, but nevertheless produced the best results. This is because it could take advantage of unlabelled data, which is easily obtained in large quantities, since it requires no effort from the user other than carrying a smartphone with an activity-monitoring application. Since not all knowledge about activities is captured by phone sensors, the machine-learning methods benefited from symbolic refinement based

on human understanding of the problem, albeit the benefit was not very large.

We believe that the accuracy of the MCAT method is sufficient to be of real benefit for diabetic patients. The overall accuracy of 83.4 % and f-score of 0.82 are comparable to related work, even to examples that tackle easier problems. The recognition of Exercise with the f-score of 0.91 is even better. Only the recognition of Eating with the f-score of 0.68 is somewhat lacking. Since unobtrusive monitoring with smartphone sensors cannot possibly recognise the quantity of food eaten, it can only prompt the user to input the quantity or act (e.g., inject insulin) according to his/her own judgement anyway. In this capacity, even a somewhat lower accuracy is useful. The detection of Sleep can be used to recognise sleep problems, which are common in diabetes. The overall activity recognition can be used to recognise diabetes fatigue and depression by monitoring deviations from the daily routine and the degree of activity. The recognition of Home, Work and Transport activities enables quality advice about exercise (e.g., remind the users of their daily exercise during Home activity rather than work or suggest commuting on foot or by bicycle).

In the future, we plan to use the activity recognition methods described in this paper for just such lifestyle advice, as well as advanced diseases monitoring. We will also improve the recognition of Eating by replacing the ECG monitor with a wrist-worn device. Such devices are also preferable because they are more comfortable and can be worn at all times. Another area for future work is sound processing, since the sound is a general feature with the potential to improve the recognition of eating as well as other activities.

Acknowledgments This paper significantly extends the work of Luštrek et al. [20], which was presented in the PHSCD 2015 Workshop. The work was partially supported by the EU FP7 project COMMODITY₁₂ (www.commodity12.eu).

References

- Amft, O., Ambient, on-body, and implantable monitoring technologies to assess dietary behavior. *Handbook of Behavior, Food and Nutrition*. Springer, 2011.
- Commodity12, <http://www.commodity12.eu/>, 2016.
- Diabetes Atlas, <http://www.diabetesatlas.org/>, 2016.
- Cvetković, B., Milić, R., and Luštrek, M., Estimating energy expenditure with multiple models using different wearable sensors. *IEEE J. Biomed. Health Inf.* 20(4):1081–1087, 2016.
- Cvetković, B., Janko, V., and Luštrek, M., Demo abstract: Activity recognition and human energy expenditure estimation with a smartphone. In: *PerCom 2015*, pp. 23–27. St. Louis, USA, 2015.
- Cvetković, B., Mirchevska, V., Janko, V., and Luštrek, M., Recognition of high-level activities with a smartphone. In: *Adjunct Proceedings of the 2015 ACM International Joint Conference on Pervasive and Ubiquitous Computing and Proceedings of the 2015 ACM International Symposium on Wearable Computers (UbiComp/ISWC'15 Adjunct)*, pp. 1453–1461, 2015.
- Cvetković, B., Kaluža, B., Gams, M., and Luštrek, M., Adapting activity recognition to a person with Multi-Classifer Adaptive Training. *Journal of Ambient Intelligence and Smart Environments* 7(2):171–185, 2015.
- Dernbach, S., Das, B., Krishnan, N. C., Thomas, B. L., and Cook, D. J., Simple and complex activity recognition through smart phones. In: *8th International Conference on Intelligent Environments*, pp. 214–221, 2012.
- Foursquare API. <https://developer.foursquare.com/>, 2016.
- Hall, M., Frank, E., Holmes, G., Pfahringer, B., Reutemann, P., and Witten, I. H., The WEKA data mining software: An update. *SIGKDD Explorations* 11(1):10–18, 2009. doi:10.1145/1656274.1656278.
- Helal, A., Cook, D. J., and Schmalz, M., Smart home-based health platform for behavioral monitoring and alteration of diabetes patients. *Journal of Diabetes Science and Technology* 3(1):141–148, 2009.
- jQuery library. <http://jquery.com/>, 2016.
- Kafalı, Ö., Bromuri, S., Sindlar, M., Van der Weide, T., Pelaez, E. A., Schaechtel, U., Alves, B., Zufferey, D., Rodriguez-Villegas, E., Schumacher, M. I., and Stathis, K., COMMODITY₁₂: A smart e-health environment for diabetes management. *Journal of Ambient Intelligence and Smart Environments* 5(5):479–502, 2013. doi:10.3233/AIS-130220.
- Kafalı, Ö., Alfonso, E. R., and Stathis, K.: Activity Recognition for an Agent-oriented Personal Health System. *Principles and Practice of Multi-Agent Systems - 17th International Conference, PRIMA 2014*, Springer, 2014.
- Kafalı, Ö., Schaechtel, U., and Stathis, K. *Hydra: A hybrid diagnosis and monitoring architecture for diabetes*. *16th IEEE International Conference on e-Health Networking, Applications and Services, Healthcom 2014*, pp. 531–536. Natal-RN, Brazil, 2014.
- Kowalski, R., and Sergot, M., A logic-based calculus of events. *New Generation Computing* 4(1):67–95, 1986.
- Lara, O. D., and Labrador, M. A., A Survey on Human Activity Recognition using Wearable Sensors. *IEEE Communications Surveys & Tutorials* 15(3):1192–1209, 2013.
- Lee, Z. S., and Cho, S. B. *Activity recognition using hierarchical Hidden Markov Models on a smartphone with 3D accelerometer*, pp. 460–467. Berlin Heidelberg: Hybrid Artificial Intelligent Systems, Springer, 2011.
- Lin, L., Location-Based Activity Recognition. Ph.D. Dissertation University of Washington, 2006.
- Luštrek, M., Cvetković, B., Mirchevska, V., Kafalı, Ö., Romero, A., and Stathis, K., Recognising lifestyle activities of diabetic patients with a smartphone. *Pervasive Health 2015: Workshop on Personal Health Systems for Chronic Diseases (PHSCD 2015)*.
- Wang, Y., Lin, J., Annavaram, M., Jacobson, Q. A., Hong, J., Krishnamachari, B., and Sadeh, N., A framework of energy efficient mobile sensing for automatic user state recognition. In: *Proceedings of the 7th International Conference on Mobile Systems, Applications, and Services*, pp. 179–192. New York, USA: ACM, 2009.

4.3 Use Case 3: Human Energy Expenditure Estimation - Regression

The human energy expenditure estimation is a physiological measure of the intensity of the physical activity and is usually expressed in a unit called MET (metabolic equivalent of task). The MET values range from 0.9 MET, which equals the energy expended while sleeping to 23 MET, which equals the energy expended during extreme exertion like running at 22.5 km/h. The standardised MET values per physical activity are listed in Compendium of Physical Activities primarily intended as a reference value to be used in epidemiological studies (Ainsworth et al., 2011). When these values are used on an individual level, it often occurs that the true values deviate from the reference value because of the difference in persons' fitness level and metabolic rate.

Methods to measure true MET values on the individual level (direct- or indirect calorimetry, doubly labelled water) are not applicable in everyday life since they require big and expensive equipment (Levine, 2005), so the research moved to using machine-learning methods on sensor data from consumer wearable devices such as wristbands, smartphones, or other accelerometer and heart-rate-equipped devices to estimate the MET values (Cvetković, Szeklicki, Janko, Lutomski, & Luštrek, 2017; Cvetković, Milić, & Luštrek, 2016; Cvetković, Janko, & Luštrek, 2015; Altini et al., 2015). These approaches achieve low average estimation error, but the error for some individuals is incomparably large, which shows the need for personalisation of the models.

Since fitness centres, medical centres and similar facilities usually own an indirect calorimeter, the individual could measure the real expended energy for a limited set of activities and use them for personalisation.

The Multi-regressor adaptive training (MRAT) is an adapted MCAT algorithm to tackle the regression domain problems such as the human energy expenditure estimation domain. The MRAT is evaluated and analysed in the same manner as MCAT. We present and compare the individual accuracies of each step of the procedure in the batch and incremental mode.

4.3.1 Related Publication

The included paper presents training and evaluation of the general regression model which is used for adaptation and personalised using the MRAT. Here we summarise the paper and extend the adaptation and analysis in Section 4.3.4.

The trained general model for the estimation of energy expenditure is compared to consumer devices, and the results show that our approach outperforms them all. Even though the overall mean error is low, the regression models still achieve high error for some of the evaluated people, which can be seen from the positions of the upper whiskers per box plot in the first chart of Figure 12 in the enclosed paper. The box plots present the estimation errors over all people per recognised activity. The coloured box plots represent 50% of all estimated values (second and third quartile), and upper whisker represents the estimation errors in the fourth quartile. We can observe that 25% of errors (fourth quartile) can be twice or three times as high as the mean absolute error. The analysis of the error shows that the same subgroup of people contributes to such a high deviation of error; therefore, personalisation could be used to decrease the per person and overall mean absolute error.



Contents lists available at ScienceDirect

Information Fusion

journal homepage: www.elsevier.com/locate/inffus

Real-time activity monitoring with a wristband and a smartphone

Božidara Cvetković^{a,b,*}, Robert Szeklicki^c, Vito Janko^{a,b}, Przemysław Lutomski^c,
Mitja Luštrek^{a,b}

^a Jožef Stefan Institute, Department of Intelligent Systems, Jamova cesta 39, 1000 Ljubljana, Slovenia

^b Postgraduate School Jožef Stefan, Jamova cesta 39, 1000 Ljubljana, Slovenia

^c Poznań University of Physical Education, Królowej Jadwigi 27/39, 61-871 Poznań, Poland

ARTICLE INFO

Article history:

Received 18 January 2017

Revised 7 April 2017

Accepted 28 May 2017

Available online xxx

Keywords:

Wristband sensors

Smartphone sensors

Activity recognition

Estimation of energy expenditure

Machine learning

ABSTRACT

Activity monitoring is a very important task in lifestyle and health domains where physical activity of a person plays an important role in further reasoning or for providing personalized recommendations. To make such services available to a broader population, one should use devices that most users already have, such as smartphones. Since trends show an increasing popularity of wrist-worn wearables we also consider a sensor-rich wristband as an optional device in this research. We present a real-time activity monitoring algorithm which utilizes data from the smartphone sensors, wristband sensors or their fusion for activity recognition and estimation of energy expenditure of the user. The algorithm detects which devices are present and uses an interval of walking for gravity detection and normalization of the orientation of the devices. The normalized data is afterwards used for the detection of the location of the smartphone on the body, which serves as a context for the selection of location-specific classification model for activity recognition. The recognized activity is finally used for the selection of one or multiple regression models for the estimation of the user's energy expenditure. To develop the machine-learning models, which can be deployed on the smartphone, we optimized the number and type of extracted features via automatic feature selection. We evaluated each step of the algorithm and each device configuration, and compared the human energy expenditure estimation results against the Bodymedia armband and Microsoft Band 2. We also evaluated the benefit of decision fusion where appropriate. The results show that we achieve a $87\% \pm 5\%$ average accuracy for activity recognition and that we outperformed both competing devices in the estimation of human energy expenditure by achieving the mean absolute error of 0.6 ± 0.1 MET on average.

© 2017 Elsevier B.V. All rights reserved.

1. Introduction

Activity monitoring using body sensor network is a mature and important area of research due to vast range of domains that can benefit from it. Accurate activity monitoring is required in domains where further reasoning or person-specific recommendations rely on the user's physical activity. These range from lighter topics such as sports and lifestyle to more sensitive topics such as health [1,2]. In sports and lifestyle, the activity monitoring gives the user an insight into the amount of their physical activity

either to support better sports training, improve self-awareness and help maintaining a healthier lifestyle, whereas in health this information is used to provide better recommendations on how to manage pathologies of a particular disease and thus improve the quality of life of the patient.

It is a known fact that engagement in physical activity has a positive influence on body and mind [3]. The ones that are the most aware of this fact are already using the service of activity monitoring to track their sports activities such as running and cycling either with a smartphone application [4] or dedicated devices [5]. These services are more popular among young active population, who are aware of the long-term risk imposed by the sedentary lifestyle (e.g., office work). Elderly population is more difficult to reach with technology even though they needed it more due to chronic diseases such as diabetes and coronary heart diseases [6] and generally less robust health. Moreover, the services which include activity monitoring mostly target the younger population and do not include specific features and knowledge

* Corresponding author.

E-mail addresses: boza.cvetkovic@ijs.si (B. Cvetković), szeklicki@awf.poznan.pl (R. Szeklicki), vito.janko@ijs.si (V. Janko), lutomski@awf.poznan.pl (P. Lutomski), mitja.lustrek@ijs.si (M. Luštrek).

URL: <http://www.bozacvetkovic.net> (B. Cvetković), <http://www.awf.poznan.pl/> (R. Szeklicki), <http://dis.ijs.si> (V. Janko), <http://www.awf.poznan.pl/> (P. Lutomski), <http://dis.ijs.si/mitjal> (M. Luštrek)

that applies for the elderly, especially not the ones who have been already diagnosed with some disease. Aim of this research is to develop activity monitoring models specialized for the elderly, which can be used either for lifestyle or health applications, so we develop our methods on data of elderly active population [7] consisting of most common every day and sports activities.

To support the lifestyle self-awareness or disease self-management in terms of activity monitoring, two individual tasks have to be solved: activity recognition to recognize the current activity which is being performed, and the estimation of human energy expenditure to quantify the performed activity. Previous research has shown that both of these tasks can be solved fairly accurate when dedicated devices have static placements (are attached to predefined location on the body, with predefined orientation) [8,9]. The activity recognition is usually set as a classification machine-learning problem, where the recognized activities include sedentary activities (e.g., sitting, standing, lying), ambulatory activities (e.g., walking, jogging, running) and sports activities (e.g., cycling). The energy expenditure is usually set as a regression machine-learning task where the regression model estimates the expended energy in a unit called MET (Metabolic equivalent of task). METs range from 0.9 MET, which is equal to the energy expended while sleeping, to over 20 MET, which is expended during extreme exertion [10]. METs are also used to describe the intensity of the physical activity where activities up to 3 MET are considered light activities, from 3 MET to 6 MET moderate activities and above 6 MET vigorous activities. The advantage over the more common kilo-calories is that MET does not depend on users weight, and can be easily translated into kilo-calories.

To make the activity monitoring more widely applicable, and to make it available to more people, we need to design a body sensor network from the devices most people already have. The most natural choice is a smartphone, which can serve as a processing unit and as a graphical interface for interaction with the user. There are already many smartphone applications available on the market, whose scope is limited to step counters or to activity-specific monitoring. Application which provide step counters and step-related estimation of energy expenditure [11] are suitable for tracking general ambulatory physical activities which do not require very accurate activity quantification. The activity-specific applications are intended for sports-active people. Such applications are the run trackers with run-related estimation of energy expenditure [4], which monitor the specific activity on users request and are therefore very accurate for the quantification of that exact activity. In general, the step counters are the most popular since we carry our smartphone only when we are headed somewhere (walking, running) and not at home so it captures the expended energy only during ambulatory activities and count the time when smartphone is not carried as rest. To catch every day activities and non-ambulatory exercise in addition to ambulatory activities, we need to improve the recognition and estimation of the models used by the smartphone or to use a commodity device that is worn by the user most of the day. In the past several years, the development of sensor-rich wrist-worn devices (smartwatches, sensor-equipped wristbands) has increased, and so did their popularity. Similarly to smartphones, the wrist-worn devices have activity monitoring methods already integrated into the device and can monitor physical activity even while sedentary. The accuracy of such activity monitoring is usually low but satisfactory for getting an insight into the movement pattern of the day. To get more accurate activity quantification, the user still has to explicitly input the type of workout performed (walking, running, cycling, etc.). Additionally, none of the devices fuse data/information from multiple devices but merely override the activity data with the analysis done by the preferred device.

The objective of our research was to develop a real-time continuous activity monitoring algorithm that utilizes sensor data from a smartphone or wrist-worn wearable alone and can fuse the data and decisions of both devices if both are present on the body (active in the body sensor network). The algorithm should be optimized to run on the smartphone to perform continuous data analysis in real-time. To develop such a method the following requirements should be fulfilled:

1. The method has to have the ability to detect which devices are present on the body, thus active in the body sensor network
2. The smartphone and wrist-worn device can be worn freely on the body at most common locations (trousers pocket, jacket pocket, bag) and in any orientation
3. The method has to have the ability to normalize the orientation of the present devices and recognize their location
4. To achieve maximum classification and regression accuracy with any combination of devices, they should be fused on multiple levels: the data level, feature level and decision level [12]
5. Procedures for decreasing the computational complexity should be used to optimize the machine-learning models to be deployable on the smartphone

In this paper we present a method that fulfills these requirements. The main contribution is the automatization of the complete activity-monitoring pipeline: the detection whether single or multiple devices are present on the body, normalization of the orientation of the devices, detection of the location of the smartphone relative to the users body (trousers pocket, jacket pocket, bag), and adaptation of the activity recognition and energy expenditure estimation to the location. Individual steps in this pipeline have been done in related work for activity recognition, but not for energy expenditure estimation. To our knowledge, this paper presents the first implementation of the complete pipeline, and it experimentally shows its benefits in terms of accuracy.

The rest of the paper is structured as follows: related work on body sensor network, pipeline procedures, activity recognition and estimation of energy expenditure is presented in Section 2. Section 3 introduces the datasets and the methods we use to develop the activity monitoring algorithm. The evaluation of the methods is presented in Section 4. We discuss the research in Sections 5 and 6 concludes the paper.

2. Related work

Activity monitoring as seen in this research is composed of human activity recognition and estimation of human energy expenditure, both very common in applications of body sensor networks. We focus on accelerometer-based wearables such as smartphones and wristbands and their combinations, where we pay particular attention to varying orientations and locations of accelerometer sensors, which is the key distinction of our approach compared to the related work.

2.1. Body sensor networks

Body sensor networks are composed of sensor nodes attached to a persons body, which are able to sense one or more physiological or motion signals. These are usually dedicated sensor enclosures or wearables with single or multiple sensors, and often perform preprocessing and storing before transmitting data to a base station [13] it is analysed. There are many challenges and opportunities in the research area that can be roughly divided into sensor design, network communication and data fusion [14]. Sensor design tackles the problems of sensor hardware design (power consumption, fault detection, etc.) and enclosure ergonomics [15], network research focuses on the network challenges (topology,

security, routing algorithms, etc.) [16], and data fusion research is focused on data manipulation (filtering, feature extraction, classification, computational complexity, etc.) [12]. Since we use consumer devices in this research (smartphone and wristband), which already have the final design and their own communication protocol, we will focus on data fusion pertaining to activity monitoring.

One of the main concerns is the development of efficient algorithms in terms of computational complexity and power consumption when preprocessing and classification is performed on sensor nodes. Ghasemzadeh et al. [17] evaluated the trade-off between the number of sensors (using only 1-D signal from accelerometers to conserve power) and accuracy for the detection of transition events (sit to stand, sit to lie, jump, etc.). The goal was to set the minimum number of sensors (attached to predefined locations in predefined orientation) which can still accurately recognize these events. They concluded that different sensors should be active for different events (from 1 to 17), and showed that decreasing the number of active sensor nodes can decrease the power consumption by up to 98 %. To perform continuous activity monitoring it is crucial to extract complex features rather than raw signals, and to sample the data with sufficient frequency. Fortino et al. [18] gathered the requirements for developing a power-efficient body sensor network, and proposed a SPINE framework for sensor node configuration in terms of sampling frequency, feature extraction and window size, which can speed-up the prototyping of applications and evaluate the trade-off between power-consuming tasks (feature extraction) and accuracy.

In our research, we decreased the size of the body sensor network to one (smartphone or wristband) or two (smartphone and wristband) active sensor nodes (wearables), which can be worn freely in any orientation and in the case of the smartphone on three locations on the body. This brings additional challenges, which we review in the following subsections. To further reduce the power consumption of the designed algorithm, we performed feature selection to reduce the number of features to be calculated, thus reduced computational complexity and related power consumption.

2.2. Sensor placement

Sensor placement is composed of the location relative to the users body, and the orientation of the sensor. Most often the activity monitoring research in body sensor network uses single or multiple accelerometers attached to predefined locations and in predefined orientations. Attal et al. [8] thoroughly reviewed the research done until 2015 in the activity recognition domain and showed that the number of recognized activities increases with the number of sensors attached to the users body. Since all the reviewed work used static placements, the authors doubt about the acceptance of such approaches since they require to carefully follow the instructions about the placement and orientation of the sensors.

The research about sensors' orientation and location varying during use was probably pioneered by Kunze et al. [19] in 2005, who used supervised machine learning on accelerometer data to first recognize walking in a manner independent of the sensor location, and afterwards used the walking segment to recognize the sensor placement on the body. In later work [20] they explored the detection of the orientation of an accelerometer-equipped smartphone around the vertical axis by relying on the walking segment. Other researchers explored the possibility to use rotation to normalize the orientation. Mizell [21] proposed a vector calculation to rotate acceleration axes to a canonical orientation and Tundo et al. [22] used quaternion rotations for the same task. The results on the impact of the orientation normalization on activity recognition vary. Tundo et al. [22] reported improvement only for the recognition of the sitting activity. In the same year,

Ustev et al. [23] published results where orientation normalization increased the overall accuracy of the activity recognition.

Since smartphone (one of the sensor nodes in our research) can be worn in any orientation and in various locations on the body, we combined and extended the work of Kunze et al. [20] and Tundo et al. [22] to develop a procedure for real-time orientation normalization and location detection.

2.3. Activity recognition

Most of the research done on activity recognition with the smartphone in different locations was done using machine-learning techniques with predefined knowledge about the placement of the smartphone [24] [25]. In 2016 Shaoib et al. [26] reviewed the research done on activity recognition with the smartphone in terms of the used smartphone-embedded sensors, recognized activities, extracted features, sampling frequency and orientation and location dependency. Lu et al. [27] and Thiemjarus et al. [28] used the Mizell [21] approach to orientation normalization. The former did not take into consideration varying location of the smartphone and required the user to leave the smartphone on the table for the calibration of the orientation. The latter used location-independent features and required the person to perform a set of predefined activities for the calibration. Anjum et al. [29] and Guo et al. [30] used the Tundo et al. [22] approach to orientation normalization. The former used location-independent features, and the latter used predefined locations of the smartphone. Martin et al. [31] took into consideration varying orientation and location of the smartphone. In addition to the smartphone-embedded accelerometer, they used the proximity, gyroscope, magnetometer, gravity and linear acceleration data to estimate the orientation as a relative angle, and to detect the location of the smartphone with a rule-based approach. For each location they trained a location-specific activity-recognition model, which increased the accuracy of the recognition. In our previous research [32], we proposed a real-time method that first normalizes the orientation, recognizes the location and afterwards uses a location specific machine-learning model to recognize five activities with 91% average accuracy over all locations. We build upon this approach in this paper.

The research on activity recognition with wrist-worn devices has started with the accelerometer sensor placed on a persons wrist as one of the sensor nodes in the body sensor network [8]. The activity recognition using only the wrist-worn sensor is more popular for the recognition of activities that involve hand movement [33]. In 2009 Siirtola et al. [34] explored the recognition of sports activities with a single wrist-worn accelerometer. The activity recognition was performed in two steps. In the first step, a sample was clustered into one of six clusters (one activity can be included in more clusters). In the second step, a decision tree model was used to recognize the sports activity. This approach achieved 85 % accuracy. In 2011 Chernbumroong et al. [35] achieved the accuracy of 94 % for five activities (sitting, standing, lying, walking, running) using a decision tree machine-learning algorithm.

Weiss et al. [36] explored the activity recognition accuracy when recognizing the same activity with a smartwatch or smartphone alone (placed in the trousers pocket with a predefined orientation). The scenario of activities was divided into non-hand-oriented activities and hand-oriented activities. The overall accuracy was better with the smartwatch alone, well over 70%, since the majority of the activities included hand movement (eating, folding clothes, handwriting, etc.) which cannot be recognized with the smartphone placed in the trousers pocket. The smartphone activity recognition achieved rather poor accuracy of 30 % since it was able to recognize only the non-hand-oriented movements (walking, jogging, etc.). Ramos et al. [37] combined the smartphone

(trousers pocket) and the wristband to achieve 80% accurate activity recognition of four activities (walking, sitting, standing, driving), but did not report if the orientation of the phone was predefined.

In summary, several researchers investigated activity recognition with a varying orientation and location of the smartphone (or a dedicated sensor) on the users body. For the most part, they found that taking this into account increases the accuracy, although the gains varied. Only a few of these researchers developed activity recognition systems that automatically take the orientation or the location of the smartphone into account. To the best of our knowledge, the system described in this paper is the first that normalizes the orientation of the smartphone and the wristband, detects the location of the smartphone (trousers, jacket and bag) and fuses the data of both devices if present for activity recognition.

2.4. Estimation of energy expenditure

Methods for accurate measurement of the human energy expenditure (direct and indirect calorimetry, doubly labeled water) are expensive, cumbersome and not applicable in everyday life [38]. The alternative is to estimate the expended energy using the smartphone and/or other sensor-rich wearables.

The pioneers of pervasive technology in the estimation of the energy expenditure used a single accelerometer attached to the users body, usually waist, and tried to correlate the motion intensity (activity counts) with the energy expenditure using a single count-based regression equation. This proved insufficient for the energy expenditure estimation of light and vigorous activities [39], so Crouter et al. [40] refined the count-based approach by using different regression equations according to the performed activity (sedentary, ambulatory, lifestyle) recognized from the number of activity counts. For the sedentary activity they assign the static value of 1 MET and used regression equations for others. The shortcoming of this approach, apart from the simplicity of the equations, is that the sensor placement (waist) was unsuitable for the energy expenditure estimation of activities involving only upper or lower limbs (cycling was omitted from the evaluation). Later works introduced activity recognition based on machine learning as an essential part of the estimation of the energy expenditure [41] [42] [43], also known as activity-specific estimation of energy expenditure.

Research utilizing smartphone or wristband accelerometers mostly rely on the research done with dedicated sensors. Pande et al. [44] developed a machine-learning approach which fuses the orientation-independent features from accelerometer sensor attached to a persons waist with demographic data. Altini et al. [45] fused the smartphones' accelerometer data with the heart-rate data, which most of the wristbands have to achieve more accurate estimations.

The research in which multi-sensor fusion is used together with the activity information for the energy expenditure estimation showed a decrease of estimation error. Tapia [43] fused the heart rate information to decrease the estimation error. Altini et al. [46] proposed a combined approach where they used two groups of activities, the sedentary (three activities) and active group (four activities). For the sedentary group they used a MET lookup table [10], the values of which were adjusted using the users heart rate. For other activities they used one regression model per activity. Vyas et al. [47] fused the data from accelerometer and temperature-related physiological sensors, where each sensor represented one or more contexts upon which a context-specific regression model was used. In 2016 we proposed an approach which fuses the data from accelerometer, heart rate and near-body temperature sensor [9] for the estimation of energy expenditure. We used three regression models, each for a set of

recognized activities. This approach is revisited in this paper and upgraded to be used with the smartphone and wristband.

Duclos et al. [48] developed an approach that utilizes smartphone and smartwatch accelerometers for activity-specific estimation of energy expenditure. They differentiated between four activity categories according to intensity (sedentary, low, medium, vigorous intensity) and used a method based on acceleration vector variance, essentially a predefined equation, to estimate the energy expenditure. They did not account for the varying location of the smartphone, nor for the orientation of both devices. They compared the results against static MET values [10] and not the real expenditure which is usually measured with indirect calorimeter.

There are dedicated devices for the estimation of human energy expenditure in the market in form of wristbands [5] [49] and armbands [50], but they mostly monitor sports activities and lack the information about everyday activities. According to Lee et al. [51], who compared several consumer devices in free-living conditions, BodyMedia's armband clearly outperformed the other devices, so we use it for comparison against our results.

In summary, the research on this topic mostly covers the estimation of energy expenditure with accelerometers attached to different parts of the body. The researches evaluated how the number of accelerometers and additional modalities influence the accuracy of energy expenditure estimation. Very little work has been done using smartphone and wristband sensors, and we did not find any that would continuously fuse the data of a smartphone and a wristband. Finally, we are not aware of any research taking varying location and orientation of devices into account when estimating human energy expenditure.

3. Materials and methods

To develop and evaluate the activity monitoring algorithm we used the datasets presented in Section 3.1 and methods presented from Section 3.2 on.

3.1. Datasets

We acquired one dataset to develop and evaluate activity monitoring methods and one dataset to evaluate the presence of the smartphone on the body.

3.1.1. Activity monitoring dataset

The difference between elder and younger population (which was the topic of our previous research) is reflected in the range of popular activities, the manner of performing the activities and most importantly in the metabolic rate (the energy cost of physical activities). The metabolic rate of a person depends on the person's body mass index (BMI) and age. To model the activities and the energy expenditure for the target population we had to collect an appropriate dataset which contains their representative activities.

Prior to designing the scenario for the data collection, we surveyed the population characteristics regarding the health and fitness status through a questionnaire. We acquired the response from 277 persons, aged between 50 and 75 (five European countries), which gave us an insight into the most popular activities we should include into the data collection scenario. The final scenario is composed of ten tasks as presented in Table 1. The scenario contains normal everyday activities (lying, eating, cleaning, etc.) as well as ambulatory and exercise activities (walking slowly, walking normally, Nordic walk, running, etc.). The sequence of tasks was ordered by increasing energy expenditure, from light lying at the beginning building up to more intensive activities such as running and cycling at the end. This ensured that the body processes stimulated by the more intense activities did not distort the data collected during the less intense activities. For the same

Table 1

Data collection scenario per task, its duration and average measured MET with the indirect calorimeter. In task Walking carrying a burden, females were carrying 2 kg burden and males 4 kg burden. In task Walking uphill the inclination was set to 5%.

	Activities	Time	Avg MET
Lying	Lying left side	1'	1.2
	Lying front	1'	1.5
	Lying right side	1'	1.4
	Lying back	7'	1.2
Basic activities	Walking slowly	10''	3.5
	Sitting down at the desk		
	Sitting still	4'	1.2
	Sitting doing light activities (reading, writing, leafing through a book, using computer, knitting, Rubik's cube, playing cards)	4'	1.2
	Standing up		
	Walking slowly	10''	3.5
	Standing still	4'	1.3
	Standing talking & gesticulating	2'	1.7
	Walking slowly	10''	3.5
	Standing washing hands	2'	2.3
	Walking slowly	10''	3.5
	Home chores (cooking, serving food, washing dishes, sweeping floor, washing windows)	6'	2.5
	Eating with cutlery	2'	1.9
	Eating with hands	2'	1.5
	Planting seedlings, digging, raking, weeding	6'	2.2
	Rest	3'	
Walking	Walking slowly (4 km/h)	4'	3.5
	Walking normally (6 km/h)	4'	4.2
	Rest	3'	
Nordic walking	Walking normally (6 km/h)	6'	4.5
	Rest	3'	
Walking carrying a burden	Walking slowly (4 km/h)	6'	4.2
	Rest	3'	
Walking uphill	Walking slowly (3 km/h)	6'	4.4
	Rest	3'	
Running	Running normally (8 km/h)	6'	7.1
	Rest	3'	
Stationary cycling	Cycling lightly (60W)	6'	4.2
	Cycling normally (100W)	6'	5.0
	Rest	3'	

reason mandatory 3 minute rest was imposed between the tasks containing moderate and vigorous activities.

Data collection was done in the laboratory environment at the Faculty of Physical Education, Sport and Rehabilitation in Poznan University of Physical Education, Poland, under the supervision of physiology and sports experts. We have recruited ten healthy volunteers: six male and four female, aged from 51 to 66 (59 ± 4.6) with different fitness levels, BMI from 22 to 29 (25.8 ± 2.3). All volunteers refrained from eating and drinking (except for water) in the 12 h prior the experiment.

The volunteers were explained the process of data collection and instructed to perform the tasks as they would do them in real life, so we could capture the variability of the tasks. They were also instructed to indicate if performed task is too vigorous (fast walking was omitted from the scenario for safety reasons). The volunteers were equipped with four smartphones (2 Samsung Galaxy S4, 2 Samsung Galaxy S2), two wristbands (Microsoft Band 2 [49] and Empatica E4 [52]), BodyMedia Fit Advantage armband [50] and Oxycon mobile [53] indirect calorimeter. The smartphones were put into the volunteers' trousers pocket, jacket pocket and bag (each with a random orientation), while the fourth smartphone was put in torso pocket and was used for collecting the data from Microsoft Band 2, which was worn on the left hand. The second wristband Empatica E4 was worn on the right hand. The Bodymedia FIT Advantage [50] data was recorded for comparison of our results and the Oxycon mobile for the ground truth measurements of the human energy expended. The Oxycon Mobile indirect calorimeter has good indicators of measurement reliability [54,55] and is commonly used in validation studies of other devices [56,57]. It is FDA (Food and Drug Administration) approved

and the error in measurements are specified to not be higher than 3%. Its flow range is from 0 to 15 liters per second and can be used in environments where temperature ranges from -10°C to 50°C and air pressure from 500 mbar to 1.100 mbar. Since it is mobile it can be used both in laboratory environment as well as in free-living conditions during every-day activities which made it suitable for our research.

The users were instructed to wear the Microsoft Band 2 on the left hand in any orientation they thought correct. Two users wore it in the opposite direction from what we considered correct orientation and two people changed the orientation in the middle of the scenario. This data was used to test the normalization of the orientation of the wristband. Fig. 1 shows two volunteers equipped with all devices performing activities from the scenario.

We acquired approximately 110 minutes of data from each sensor for each volunteer.

3.1.2. Presence detection dataset

Since the volunteers did not remove the smartphones from their location during the data collection procedure, we had to acquire separate dataset for evaluation of the smartphone's presence. We prepared a fifty-minute scenario designed to test corner cases of the smartphone use. It was carried out by five volunteers with Samsung Galaxy S6 smartphone.

The scenario included wearing the smartphone in all relevant locations for this research: trousers pocket, two types of torso pockets (jacket, breast pocket) and bag while the volunteer was moving and while the volunteer was sedentary. We also collected the data when the smartphone was in the hand, with screen either on or off, where it was in a bag but the bag was not



Fig. 1. Volunteers equipped with the wearable devices and the indirect calorimeter performing everyday activities.

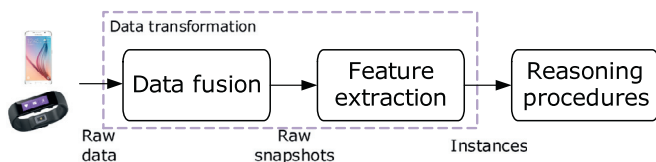


Fig. 2. Data transformation procedure.

Table 2
Frequencies per device [49,50,53] sensor.

Device	Sensor	Frequency
Smartphone	Accelerometer	45–50 Hz
	Proximity sensor	Reports a change
Microsoft Band 2	Accelerometer	40–45 Hz
	Heart rate	1 Hz
	RR interval	2 Hz
	Skin temperature	12 Hz
	GSR	5 Hz
	Calories	0.1 Hz
BodyMedia	Estimated MET	1 per minute
Oxycon mobile	Measured MET	1 per 10 seconds

worn, and where it was placed on some surface. Both horizontal and non-horizontal surfaces were tried, with smartphones screen facing either up or down, and for both still and shaking table (someone was typing and moving mouse nearby).

3.2. Data fusion

The goal of data fusion is to combine and preprocess the data into a form suitable for further reasoning. The data fusion is the first step of data transformation procedure presented in Fig. 2 in which the raw data is fused into time-aligned raw snapshots, which is not trivial due to different sensor frequencies. Table 2 presents the modality and frequency details per device sensor. To fuse the data into structure which contains raw data (raw snapshots) we first divided the sensors in two groups with different modes: the high frequency sensor data (accelerometers) are assigned the “necessary” mode, and the low frequency sensor data (other sensors) are assigned the “duplication” mode.

The construction procedure of the raw snapshots expects at least one sensor reading from the sensors in the necessary mode to release a snapshot as complete for further processing. The sensors in the duplication mode attach the last received sensor reading to the raw snapshot under the condition that it is time-aligned with the data in the necessary mode currently contained by the raw

snapshot. The time between the duplication and necessary data should not exceed one second to be included into the same raw snapshot. The exceptions are the Oxycon mobile data which is used in the energy expenditure estimation method, the proximity data, which is used for smartphone presence detection, Bodymedia data and Microsoft Band 2 calories data, which are used for comparison of the results. The Oxycon mobile data is duplicated if the time between the duplication and necessary data does not exceed ten seconds, which matches the needs of the energy expenditure estimation method (Section 3.5.6). Proximity sensor reports only when the change in proximity is detected, this data is duplicated without any specific conditions. The Bodymedia data is duplicated if the time between duplication and necessary data does not exceed one minute, and is not used for final real-time monitoring but merely for the comparison of the energy expenditure estimation method performance. The Microsoft Band 2 calories were duplicated without any specific conditions.

This data fusion procedure decreases the amount of missing data compared to an approach without duplication and gives us approximately 45 raw snapshots per second if only smartphone data is used, and 40 raw snapshots per second if only wristband is used or both devices are used. The raw snapshots are sent to the feature extraction module for further processing, where additional fusion is done as explained in the next subsection.

3.3. Feature extraction

The feature extraction procedure is the second step of data transformation procedure presented in Fig. 2. It transforms the raw snapshots into feature vectors which are used by machine-learning and rule-based algorithms. The raw snapshots are collected into windows, the length of which depends on the task to be solved. In this research we address four machine-learning tasks for which we use two window lengths. We use 2-second windows for activity recognition and walking detection and 10-second window for smartphone location detection and estimation of energy expenditure (explained in detail in Section 3.5). For each window, a number of features are extracted forming a feature vector ready to be used with the machine-learning algorithm.

Raw acceleration data is low-pass (removing noise) and band-pass (removing noise and gravity) filtered, giving us three values for each axis. Low-pass filtered values are used in features describing orientation, while band-pass filtered values are used in features describing movement. The magnitude of the three-axis vector is also computed. Intuitively, magnitude gives us the intensity of motion, useful for distinguishing running from walking and sedentary activities. On the other hand, features that use one of the axis components give us more information about orientation and direction, which is especially useful at distinguishing between sedentary activities, such as sitting, standing and lying. We extract 90 features from acceleration data when single device is present, 13 of which are orientation-independent. Some features come from statistics and describe the intensity and “shape” of the signal: the mean, variance, Pearson’s correlation between axes, their covariance, skewness, kurtosis, quartile values and range between them. Other have a more physics-based interpretation, such as velocity and kinetic energy. The rest came from expert knowledge of the domain: the number of peaks in the signal, their height, their mean and their sum of the acceleration size, the number of times the signal crosses its mean value, sum of all data from an axis and squared sum of this data. Note that since most of the listed methods can work with low-pass or band-pass filtered data, with any axis or with magnitude, they each generally generate more than one feature (mean of the x-axis, mean of the y-axis, etc.).

Some features are calculated from the filtered raw signals and some are constructed with feature fusion, for example the mean



Fig. 3. Gain ratio and correlation heatmaps of all extracted features when single device is used.

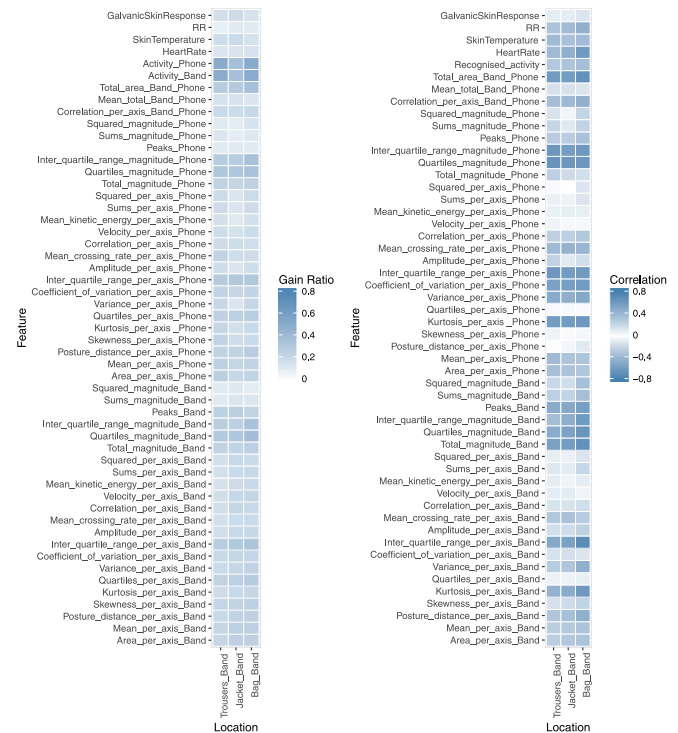


Fig. 4. Gain ratio and correlation heatmaps of all extracted features when both devices are used and the smartphone is worn in the trousers pocket.

kinetic energy is calculated from the velocity feature and the person's BMI; skewness, kurtosis, crossing rates and correlations are calculated using the feature representing sum for each axes, etc.

When both devices are present we calculate 192 acceleration features: 90 from each device and 12 features that use data fused from both accelerometer sensors. They are the mean of all data, its sum, area below the magnitudes signal (marked with keyword “total” on Figs. 3 and 4) and correlations between axes of the devices. Additionally, we add decision features where possible. Decision features are the outputs of machine-learning models used as features and as contexts for decision fusion. When both devices are present, we use the outputs of activity recognition model per individual device as two decision features (phone activity and band activity) in the activity recognition feature vector for the device combination and in the energy estimation task we add the recognized activity as a feature into the energy expenditure estimation feature vector. These are also used for decision fusion in the activity monitoring procedure.

Physiological sensor data retrieved from the wristband are transformed into the average, maximum, minimum and standard deviation value for the current window, which gives us altogether 16 features from the four sensors in the wristband.

One energy expenditure feature is extracted from Bodymedia and one from the Oxycon mobile device. Former is used for comparison and the latter is used as the ground truth.

All features for a given device combination represent a feature vector which is used to build models with machine learning. Since the goal of this research is to design models to be used on a smartphone, the number of features should be decreased and the ones with high computational complexity omitted. The set of features for a given task is selected with the feature selection procedure which also omits the non-informative features, thus improve the accuracy of the models.

Note that we use raw values of proximity sensor and calories as estimated by the Microsoft Band 2, but these are not included into the feature vector.

3.4. Feature selection

The feature selection procedure is composed of two steps: a single-feature evaluation, and a wrapper-based feature selection, which evaluates combinations of features and chooses the final feature set. Feature selection is performed only once for each pipeline procedure which uses machine-learning.

The single-feature evaluation procedure starts with entire feature set. It uses Gain Ratio measure to evaluate the information about the class carried by each feature in the classification tasks. In regression tasks, it evaluates the importance of features as Pearson's correlation with the class.

Figs. 3 and 4 show heatmaps of the results of the evaluation procedure for the activity recognition (classification) and energy expenditure estimation tasks (regression) for individual devices and for the combination of devices. For a more compact representation we merged the features which are calculated for individual axis or are similar into single features with averaged mutual information value. Left side of Figs. 3 and 4 presents the gain ratio value for the features used in activity recognition. The higher the value, the more information is carried by a feature. The right side of Figs. 3 and 4 present the Pearson's correlation coefficient for features used in estimation of energy expenditure task. The correlation values around 0 indicate low correlation and values towards 1 and -1 high positive and negative correlation respectively. Both, gain ratio and correlation were calculated with procedures implemented in the Weka machine-learning suite [58].

The features in the figures are listed as follows. In Fig. 3 the physiological sensor features are at the top, next are the decision features (in the case of the energy expenditure estimation), followed by the magnitude features combining all axes, and per-axis features are at the bottom. It is similar in Fig. 4: the physiological sensor features are at the top, next are the decision features followed by the features extracted from the wristband's accelerometer (first magnitude features, then per-axis features)

and features extracted from the smartphone's accelerometer (first magnitude features, then per-axis features).

The single-feature evaluation procedure in activity recognition (left sides of Figs. 3 and 4) prefers the magnitude features and decision features, some significance is shown for the interquartile range and quartiles features which are orientation-dependent. Common sense tells us that per-axis features can provide more information about the sedentary activities, where not much motion is performed and the orientation of the smartphone is important (sitting, lying, standing, etc.). This justifies our decision to use the wrapper approach, which can remove redundant magnitude features and add important per-axis features to be used in the feature vector.

The single-feature evaluation procedure in energy expenditure estimation (right sides of Figs. 3 and 4) shows low correlation for some per axis features and high (positive or negative) correlation for magnitude and decision features as well as for the features extracted from the physiological sensors. Using only the single-feature evaluation procedure would result in too many features, which may be redundant and would increase computational complexity so we perform the wrapper-based feature selection to reduce the number of features.

The feature selection wrapper is rather simple but efficient. The input to the procedure are the list of features ranked by the value retrieved from the single-feature evaluation procedure, the machine-learning algorithm for which the feature selection is performed and manually created folds for testing. In our case, we evaluate the models with the leave-one-person-out approach (LOPO) so we have n folds, where the data of $n - 1$ people are used as the training set and one person as the testing set (fold). The loop iterates over the list of features and adds them one by one, by decreasing rank, and performs the LOPO evaluation in terms of accuracy for classification and in terms of mean absolute error (MAE) for regression (any other measure can be evaluated). For each new feature set the accuracy/error is compared to the prior one and in case it is higher/lower the last added feature is kept.

3.5. Activity monitoring

Activity monitoring is designed to track the users activity in real-time with the devices currently present on the body. To achieve that, we developed the activity monitoring algorithm presented in Fig. 5 with pipeline of six consecutive modules: four modules to manage the device configuration (left side in Fig. 5) and two modules to analyze the activity (right side in Fig. 5).

In brief, the sensor data received is first evaluated for the presence of the devices with simple heuristics, then the data is fused and features extracted in the data transformation module (Sections 3.2 and 3.3). The algorithm waits for ten seconds of walking for gravity detection, the data of which is used for normalizing the orientation of each device. While the algorithm waits for the normalization data, the modules for activity analysis use machine-learning models trained to be used without orientation and location information for activity recognition. Once the sensor data is normalized, it is used in all further machine-learning tasks. The first is to detect the location of the smartphone, which is then used for the selection of the machine-learning models for activity recognition and estimation of energy expenditure.

Note that the tasks with gray boxes in Fig. 5 use machine learning. All machine-learning tasks use the feature set which was defined with feature selection. The evaluation is in Section 4.

3.5.1. Presence of the devices

Presence of the devices is needed for identifying which data is available to be used for activity monitoring. The presence of the Microsoft Band 2 is straightforward since it has an ability to

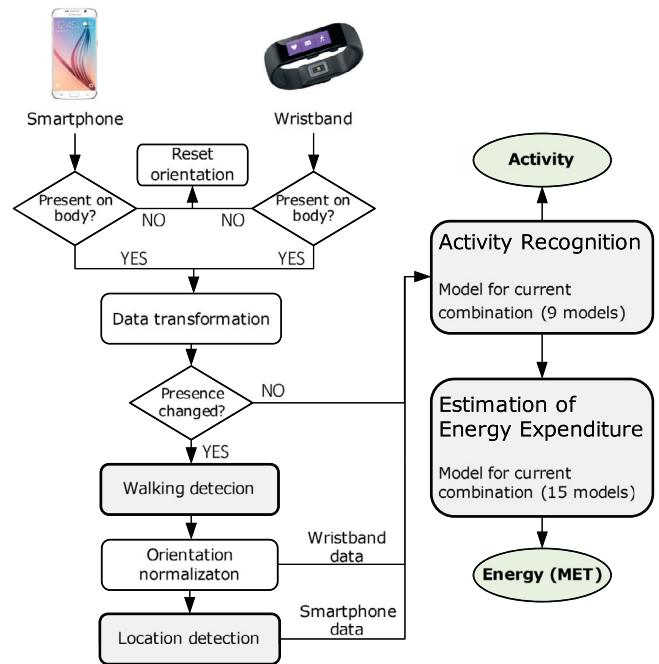


Fig. 5. Activity monitoring algorithm work flow. The modules on the left side manage the device configuration and the right side performs the activity analysis (activity recognition and energy expenditure estimation).

self-report its presence, however the detection of the smartphone presence is a bit more difficult.

We assume that the smartphone is present if it is placed in a pocket or a bag (while the bag is worn). To detect that, we use the smartphone-embedded proximity sensor, the accelerometer data and the information about the phone state as shown in Fig. 6. First, we check if the smartphone's screen is on, if there is an on-going call or if the phone is charging, all of which indicate that the smartphone is not worn. Second, we detect the amount of movement in last two minutes (looking at the accelerometer data), no movement indicates that the smartphone is not present on the body. Even if the user is sedentary, we expect some movement in two-minute time interval. Third, we check the value of the smartphone proximity sensor, which tells us if there is something close to the smartphone which could mean that it is in a pocket. In case there is something in proximity, we check if the smartphone is completely horizontal (easily determinable from accelerometer data), to eliminate the common false positive of the smartphone being on the table face down. Nothing in proximity implies that the smartphone is not in a bag or a pocket, but we again check for one exception - the smartphone being in a breast pocket with its top end looking out of the pocket. We do so by checking if the smartphone's orientation is roughly vertical.

Data from the present devices is sent into the data transformation module (Section 3.2). If the presence was not changed, the data from the data transformation module is transformed into feature vectors for activity analysis and are feed directly into activity analysis modules, otherwise the orientation is reset and data is transformed into feature vector needed for the walking detection.

3.5.2. Walking detection

Walking detection is essential for normalization of the orientation of the devices. It gives us information about the direction of gravity and under assumption that the average acceleration during walking corresponds to the Earth's gravity we can use it to normalize the orientation.

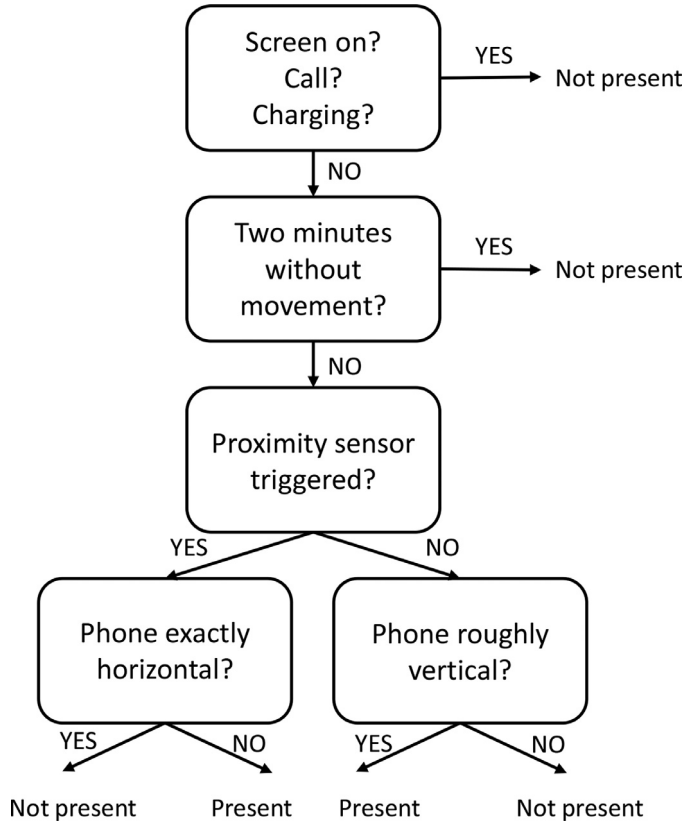


Fig. 6. Heuristics for detection of the smartphone's presence.

The walking detection is a machine-learning tasks done essentially in the same way for both devices. The data transformation procedure segments the raw data of each present device into separate 2-second window and extracts the orientation-independent features creating a feature vector. The walking detection model is a binary classifier trained to distinguish between walking activity and all other activities. While the model classifies 2-second intervals, the final decision – based on which orientation normalization is performed – is made for ten second interval. We assume that walking is detected if the four out of five consequent classifications are walking. This satisfies the smartphone's prerequisites for orientation normalization and prerequisite to detect the wristband gravity along the x-axis (buttons up/down).

3.5.3. Orientation normalization

The normalization of the orientation is based on the gravity detection. Prerequisite for the normalization is thus the detection of walking, which is described in the previous subsection. Let $\vec{a}_{walk} = (x_{walk}, y_{walk}, z_{walk})$ be the acceleration vector consisting of the average accelerations along the three accelerometer axes during ten seconds of walking.

Orientation normalization is needed for both devices. We have seen from practical experience that even if the wristband is always worn on the left wrist, it sometimes occurs that the users wear it upside down (buttons down), which switches acceleration x-axis in opposite direction and therefore inflicts errors in extracted orientation-depended features and consequently classification and regression errors.

Wristband orientation normalization uses simple heuristics to detect vertical direction. We assume that the vector $\vec{a}_{band} = (g, 0, 0)$ is the preferred acceleration vector when the wristband is correctly worn in the vertical direction, while walking. To normalize the orientation it is sufficient to check if the



Fig. 7. Wrist postures. Left is standing posture (A), in the middle is the hand down flat on the table posture (B) and on the right is hand down on the side posture (C).

Table 3
Wristband axis at different orientations.

Position	Buttons up		Buttons down	
	Inside	Outside	Inside	Outside
A	(g, 0, 0)	(g, 0, 0)	(-g, 0, 0)	(-g, 0, 0)
B	(0, 0, -g)	(0, 0, g)	(0, 0, -g)	(0, 0, g)
C (Left)	(0, -g, 0)	(0, -g, 0)	(0, g, 0)	(0, g, 0)
C (Right)	(0, g, 0)	(0, g, 0)	(0, -g, 0)	(0, -g, 0)

x-axis is negative and if absolute value exceeds 0.8g and if it is, all further x-axis data from the wristband are multiplied by -1 , otherwise the orientation in vertical direction is considered as correct.

In Fig. 7 we present the wrist postures with perfect gravity detected. On left is the posture while standing/walking (A), middle is the posture while the hand is flat on the table (B) and on the right is the posture while hand is on the side on the table (C). Table 3 presents the values per-axis in different orientation (buttons up/buttons down and inside/outside) and different wrist postures presented in Fig. 7. We can observe, that it is also possible to normalize z- and y-axis. Since our dataset did not include such data we only evaluated the x-axis orientation normalization.

The normalization of the smartphone is more difficult since the smartphone can be worn in any orientation. Let $\vec{a}_{smartphone} = (0, 9.81, 0)$ be the preferred acceleration vector, which would be obtained if the smartphones longest side was perfectly aligned with the gravity. To normalize the smartphone's orientation it has to be rotated so that the y-axis of $\vec{a}_{walk}$ corresponds the $\vec{a}_{smartphone}$. The rotation between these two vectors is represented with the quaternion matrix R , such that $\vec{a}_{smartphone} = R * \vec{a}_{walk}$. To compute R , we adopted the approach by Tundo et al. [22]. The procedure is as follows. First, we compute the axis-angle pair $\vec{a}_{pair}$ as the cross product between the preferred vector and the walking vector, i.e. $\vec{a}_{pair} = \vec{a}_{smartphone} \times \vec{a}_{walk}$. Second, we normalize $\vec{a}_{pair}$ by dividing it by its magnitude giving us $\vec{a}_{norm}$, which is needed for the quaternion construction. Third, we use the dot product to compute the angle α between the vectors $\vec{a}_{smartphone}$ and $\vec{a}_{walk}$ as presented in Eq. (1).

$$\alpha = \arccos \frac{\vec{a}_{smartphone} \cdot \vec{a}_{walk}}{\|\vec{a}_{smartphone}\| \|\vec{a}_{walk}\|} \quad (1)$$

Finally, the quaternions are calculated according to Eq. (2) and the matrix R is calculated according to Eq. (3).

$$\begin{aligned} q_0 &= \cos\left(\frac{\alpha}{2}\right) \\ q_1 &= \sin\left(\frac{\alpha}{2}\right) \vec{a}_{norm} \cdot x \\ q_2 &= \sin\left(\frac{\alpha}{2}\right) \vec{a}_{norm} \cdot y \\ q_3 &= \sin\left(\frac{\alpha}{2}\right) \vec{a}_{norm} \cdot z \end{aligned} \quad (2)$$

Table 4

Recognized activities (most left) per wristband and smartphone when orientation information and location is missing (second group of columns) and wristband and smartphone location (third group of columns) and combination of devices (most right) when orientation normalized and location detected. O_w = wristband without orientation, O_s = smartphone without orientation and no location, W = wristband, T = trousers location, J = jacket location, B = bag location. In combinations it is assumed that the wristband is also present.

Activity	Single device						Combination		
	O_w	O_s	W	T	J	B	T	J	B
Rest	✓	✓	✓	✓	×	✓	×	×	✓
Standing	✓	✓	✓	✓	×	×	✓	✓	×
Lying	×	×	×	×	✓	×	✓	✓	×
Sitting	×	×	×	×	×	×	✓	✓	×
Upright	×	×	×	×	✓	×	×	×	×
Movement	✓	×	✓	×	×	×	×	×	×
Chores	×	✓	×	✓	×	×	✓	✓	×
Walking	✓	✓	✓	✓	✓	✓	✓	✓	✓
Nordic	×	×	✓	×	×	×	✓	✓	✓
Running	✓	✓	✓	✓	✓	✓	✓	✓	✓
Cycling	×	×	×	✓	✓	✓	✓	✓	✓
#	5	5	6	6	5	4	8	8	5

$$R = \begin{bmatrix} 1 - 2(q_2^2 + q_3^2)2(q_1q_2 - q_0q_3)2(q_0q_2 + q_1q_3) \\ 2(q_1q_2 + q_0q_3)1 - 2(q_1^2 + q_3^2)2(q_2q_3 - q_0q_1) \\ 2(q_1q_3 - q_0q_2)2(q_0q_1 + q_2q_3)1 - 2(q_1^2 + q_2^2) \end{bmatrix} \quad (3)$$

Once the matrix R is computed, each sensed acceleration vector $\vec{a}_{original}$ can be normalized to the preferred orientation in real time, thus creating the normalized acceleration vector $\vec{a}_{normalized}$ using the Eq. (4).

$$\vec{a}_{normalized} = R * \vec{a}_{original} \quad (4)$$

3.5.4. Location detection

Once the data is normalized we use machine learning to detect the location of the smartphone.

The data transformation procedure segments the normalized data from walking detection into 10-second window and extracts the orientation-dependent and orientation-independent features creating a feature vector ready to be used in the machine-learning task. The location model is trained to distinguish between three classes representing the location of the smartphone relative to the user's body. These are the trousers pocket, jacket pocket and bag. Location is assumed immediately.

The wristband location detection is not a subject of this paper, but we can conclude that it is feasible to do so according to the y-axis when worn on left or right wrist (Table 3).

3.5.5. Activity recognition

Activity recognition is a classification machine-learning task. The normalized data is processed with data transformation procedure and segmented into 2-second windows from which the orientation-independent and orientation-dependent features are extracted forming a feature vector which is fed into the machine-learning classification model. We trained nine location- and combination-specific classification machine-learning models, one for each device configuration (7 models) and one for each device (2 models) to be used before orientation is normalized. The model selection therefore depends on the currently present devices and on the recognized location of the smartphone if present (device configuration). The main reason for differentiating between the locations is that the type of motion detected by the accelerometer embedded into the smartphone depends on the location where it is attached, e.g., the smartphone in trousers pocket does not detect the motion of the upper body as much as if the smartphone is in the jacket pocket. It is similar when combinations of devices are used. It is important to adapt the set of activities to be recognized to the location and combination accordingly and to exploit as much information as possible.

All recognized activities are listed in Table 4. Each row corresponds to one activity we can recognize, and each column to one device configuration. First are the no orientation models followed by the seven models according to the device, location and combination. We can observe that walking and running are recognized with all device configurations, whereas Nordic walking is recognized only with the wristband or in combination with the wristband when the orientation is known. Upright activity is one of the activities recognized with a single device, the smartphone located in the jacket pocket. It is composed of sitting and standing which cannot be distinguished only with an accelerometer placed on the persons torso (jacket pockets) due to the same orientation and amount of movement. When we add the wristband to the jacket-located smartphone, we can distinguish sitting from standing, and the number of recognized activities increases from five to eight. Since bag is usually not worn indoors (while lying, sitting), we can observe that the combination of the wristband and the smartphone located in a bag can recognize fewer activities as than the wristband alone. However with this combination we can recognize cycling which cannot be recognized with the wristband only. The data in the dataset presented in Section 3.1 contains 33 activity labels which were first merged into 10 activities (row labels in Fig. 8) and afterwards we merged these activities (colors in Fig. 8) to optimize the recognition according to the device configuration (columns in Fig. 8). For example, for the device configuration T (smartphone placed in trousers pocket) sitting and lying were merged into rest because the position of the smartphone is the same in both activities, thus impossible to distinguish accurately.

3.5.6. Estimation of energy expenditure

Estimation of energy expenditure is a regression machine-learning task. The data is segmented into 10-second windows, which was adopted as the minimal sensible interval based on our previous research. The data transformation module extracts the orientation-independent and orientation-dependent features from the data stream and includes the recognized activity to form a feature vector which is fed into a regression machine-learning algorithm. The output is the estimated energy expenditure expressed in MET.

We have developed and evaluated two approaches:

1. Single model approach
2. Multiple-model approach which utilizes recognized activity for decision fusion about model selection

In single model approach we developed seven regression models, one for each device configuration with features extracted from all available data.

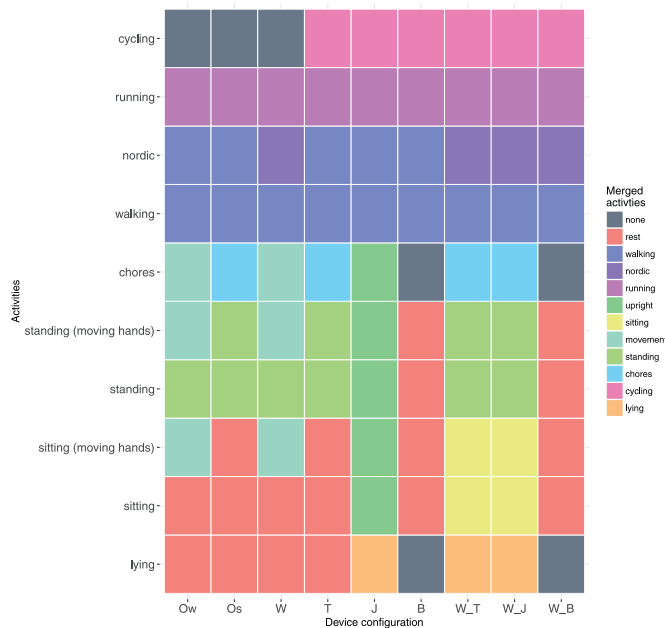


Fig. 8. Merged activities per device configuration. O_w = wristband without orientation, O_s = smartphone without orientation and no location, W = wristband, T = trousers location, J = jacket location, B = bag location, W_T = wristband and smartphone in trousers location, W_J = wristband and smartphone in jacket location, W_B = wristband and smartphone in bag location.

In the multiple-model approach we developed multiple regression models for the device configurations where different sensor modalities are available. Our previous research [9] showed that energy expenditure estimation performance is improved if similar activities with a large range of possible MET values (ambulatory activities) and activities with similar range of possible MET values have their own regression models, so we trained three models per device configuration where different modalities are present. The device configurations where only smartphone is present use single model approach

All together we trained 15 models which are as follows:

- Ambulatory regression models: four models are trained to be used when the recognized activity is either walking or running
- Low regression models: four models were trained to be used when low-intensity activities such as lying, sitting, rest are recognized
- Other regression models: seven models were trained to be used with all other recognized activities and in case only smartphone is present

Feature vectors of the ambulatory regression models and the other regression models are composed of features extracted from all available modalities. The feature vector of the low regression models are composed of features extracted from the accelerometers only and not physiological sensors, even if the wristband is present.

The evaluation of both approaches is presented in the Section 4.7.

4. Evaluation results

For the comprehensive understanding of the performance of our algorithm, we evaluated each task separately in the same sequence as implemented. The errors in orientation normalization and activity recognition are carried over to subsequent tasks, while we evaluated each location separately. The tasks were evaluated on the dataset presented in Section 3.1 in the LOPO manner. We

Table 5

Results of the AR without orientation and location (W =wristband, S =smartphone, # = number of features after feature selection).

Device	Accuracy	Kappa	F-score	#
Wristband	79.2	0.67	0.63	8
Smartphone	79.4	0.67	0.50	6

experimented with six machine-learning algorithms for classification tasks (J48 - decision trees, Support Vector Regression, JRip - decision rules, Random Forest, Naive Bayes and IBK - clustering) and five for regression task (Support Vector Regression, Multi-Layer Perceptron - neural network, Linear Regression, Random Forest and RepTree - decision tree) all as implemented in the Weka machine-learning suite [58]. We report the results of the best performing algorithm only, which was the Random Forest algorithm in all classification and regression tasks.

The results for classification are presented in terms of the classification accuracy (the percentage of correctly classified feature vectors), the kappa statistic (the homogeneity of the classifications) and the F-score measure. The results of regression are presented in terms of mean absolute error (MAE), mean absolute percentage error (MAPE) and root mean squared error (RMSE).

4.1. Presence detection

We evaluated the presence detection on the separate dataset presented in Section 3.1. In 90% of cases, the smartphone was correctly identified as being present/not present. The error occurred only in two cases. The first case was when a person left the smartphone on the table and the heuristics needed two minutes to decide that the smartphone is not present. These two minutes contributed to the error which could be eliminated by identifying the two-minute period as not present retroactively, which we decided not to do since we wanted a real-time algorithm. The second case was when the smartphone was screen-down on a non-horizontal surface on a shaking table (while someone is typing), as it is very similar to the position when smartphone is in a trousers pocket while the person is sitting.

4.2. No orientation and no location

These models are used from the time the presence of the devices changes until orientation is normalized and location detected. The results of the classifier trained with Random Forest are presented in Table 5. The accuracy for activity recognition is around 79 % for the recognition of the five activities as given in Table 4 in columns O_w and O_s . The results for kappa shows that there is high intra-class variability and that the classification is not very stable. This argues for the normalization of the orientation and detection of location of the devices. We can also observe that after feature selection the number of features decreased from 13 orientation-independent features to 8 features in the case of the wristband and 6 features in the case of the smartphone.

At this stage we are most interested in recognition of walking and the results of the per-class f-score (around 0.8 for individual device) indicate that this is feasible with specialized classifier.

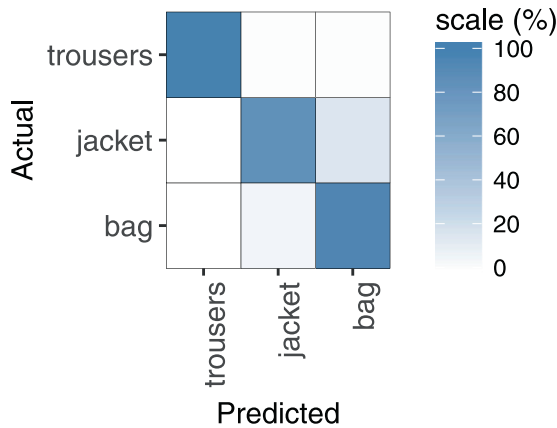
4.3. Walking detection

The walking detection is performed for both devices separately with objective to detect the gravity needed for orientation normalization of both devices. The classifier was trained on the entire dataset on orientation-independent features. The non-walking samples were re-labeled with "other", to create a binary classification problem. The results of walking detection for individual

Table 6

Results of the walking and location detection. Walking detection is performed for both devices separately and location detection only when the smartphone is present (W=wristsband, S=smartphone, # = number of features after feature selection).

Detection	Device	Accuracy	Kappa	F-score	#
Walking	W	90%	0.76	0.89	6
	S	91%	0.83	0.91	9
Location	S	91%	0.88	0.91	11

**Fig. 9.** Confusion matrix for smartphone location detection.

device are presented in Table 6. We can observe that we achieve accuracy over 90% with only 6 orientation independent features in case of a wristsband and 91% accuracy with 9 orientation independent features in case of a smartphone. One average we achieve 0.8 kappa and 0.9 f-score.

4.4. Orientation normalization

In this experiment we could evaluate only the accuracy of the orientation detection for the wristsband. The true smartphone orientation could not be clearly defined so we cannot evaluate it.

We evaluated the heuristics for wristsband orientation normalization on the entire dataset. First we waited for the ten seconds of recognized walking, which was detected using the model explained in the previous subsection. When walking was detected we used the walking data to evaluate the up-down orientation of the wristsband and to normalize x-axis. We achieved 100% accuracy on our dataset. The accuracy might decrease in such cases where the hand is raised above a head and the movements are miss-misclassified as walking. However, we did not have such events in our dataset.

4.5. Location detection

The location detection is performed for smartphone only. The classification model is trained on the normalized data classified as walking from the walking detection (Section 4.3). The feature vector is constructed from both orientation-independent and orientation-dependent features. The data was labeled with the three labels: trousers, jacket and bag which comply with the location of the smartphone from which the data was acquired. The results for location detection are presented in the bottom row of Table 6. We can observe that feature selection decreased the number of features from 90 to 11 which resulted in 91% accuracy with high kappa and f-score. Interestingly, only one orientation-independent feature remained in the final feature set, other 10 are orientation-dependent.

The confusion matrix is presented in Fig. 9 showing that the smartphone at trousers location is best recognized and that

Table 7

Results of AR per device configuration.

	Single device				Combination		
	W	T	J	B	T	J	B
Accuracy (%)	80	92	80	95	89	85	89
Kappa	0.75	0.89	0.70	.90	0.87	0.80	0.83
F-score	0.75	0.78	0.65	0.92	0.89	0.83	0.71
# activities	6	6	5	4	8	8	5

Table 8

Number of orientation independent (O-independent), orientation dependent (O-dependent), physiological (Physiological) and machine-learning (Recognized activity) for each activity recognition model after feature selection.

Model	Single device				Combination		
	W	T	J	B	T	J	B
O-independent	9	4	4	7	10	6	1
O-dependent	9	12	11	7	15	18	13
Physiological	3	0	0	0	1	0	1
Recognized activity	0	0	0	0	2	2	2
# features	21	16	15	14	28	26	17

some miss-classification occurs between jacket and bag location, probably because both locations are on the torso.

4.6. Activity recognition

Activity recognition was evaluated for each device configuration separately. The activity recognition classification models were trained on the normalized data from all scenarios in the dataset with different set of merged activities as labels (presented in Table 4 and Fig. 8). The feature selection procedure returned a set of the best performing features to be used in the feature vector. The results for each device configuration is presented in Table 7 and in the form of the confusion matrices in Figs. 10 and 11. The details on the used feature vectors are presented in Table 8.

Each model was trained to recognize a merged set of activities to find a trade-off between the acceptable accuracy and the number of recognized activities. The number of activities per device configuration is also presented in Table 7. The results show that the accuracies are mostly over 85% and that kappa and f-score also show stability for the evaluated configurations. The results of the device configurations are not directly comparable since they recognize different number of activities.

We can observe that for a single device the highest accuracy (6 activities) is achieved with the classification model for the smartphone in a trousers pocket. The confusion matrix for the same model shows that there is a slight miss-classification of chores into standing and walking, which is reflected in lower overall f-score. This miss-classification is understandable since most of the home chores are composed of these two activities and the smartphone placement makes the acceleration measurements of upper limbs difficult. The wristsband classification model achieves 80% accuracy and does miss-classifications of standing (mostly standing still data) into rest due to no movement in the wrist. The running and rest are sometimes miss-classified into movement due to wrist movement while performing these activities (gesticulating while running, gesticulating while at rest). The jacket classification model achieves 80% accuracy and does miss-classification of cycling into walking and upright activity. These errors occur because of the placement on the torso it is impossible to measure the movement of the lower limbs accurately. The classification model for bag achieves 95% accuracy since it recognizes only those activities that people do when they carry a bag (we excluded lying, eating and gardening from the training and testing data).

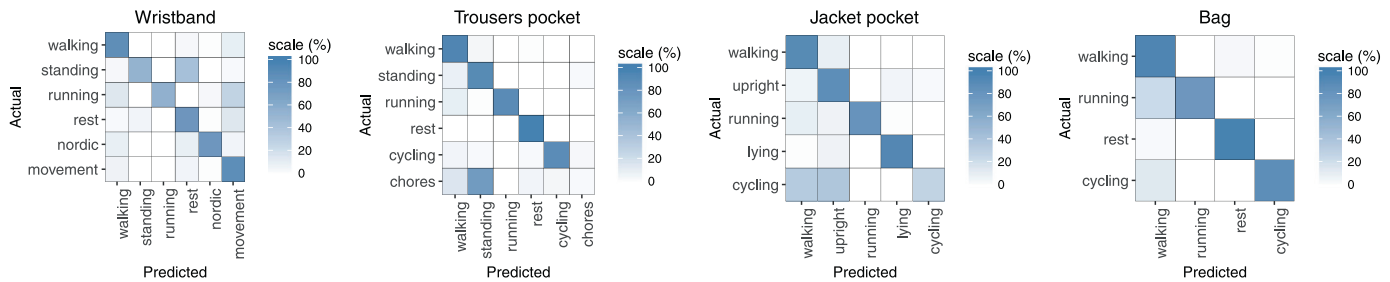


Fig. 10. Confusion matrices when activity recognition is done using single device.

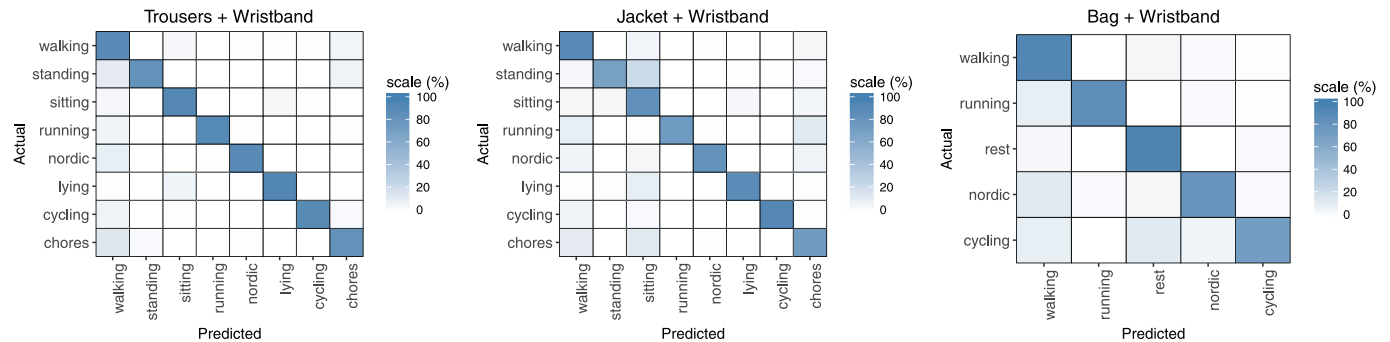


Fig. 11. Confusion matrices when activity recognition is done using two devices.

The combinations achieve accuracies towards 90% and can recognize from five to eight activities. We can see in the confusion matrices, that all models perform well and that there is no large miss-classification going on.

Additionally, the number of remaining features after feature selection do not surpass 28 (decreased from over 90 in single device and over 200 in combination). Table 8 presents the number of orientation-independent, orientation-dependent features, physiological features and in case of combination of devices also the number of features which already need machine-learning (recognized activity by each device). We can observe, that orientation-dependent features form the majority of the feature vector which is the information gained by normalizing the orientation of the device which would otherwise be lost.

4.7. Estimation of energy expenditure

Energy expenditure estimation was evaluated for each device configuration separately. The energy expenditure estimation regression model was trained on the normalized data from all scenarios in the dataset labeled with the expended energy as measured by the indirect calorimeter Oxycon mobile [53].

We evaluated and compared both approaches, the single model approach and the multiple-model approach. The results for each device configuration is presented in Table 9 and details on the feature vector structure used is presented in Table 10. We first compared the results against Bodymedia Fit armband, which is one of the most accurate energy expenditure estimation devices on the market [51]. Since smartphone only device configurations lack the presence of different sensor modalities, they do not meet the requirements of multiple-model design. In Fig. 12 we present the box-charts where multiple-model approach was applicable.

The results show that the errors decrease when device configuration combines devices and fuses different modalities from the data stream. We can observe that all device configurations outperform Bodymedia armband, probably because Bodymedia armband is specifically designed for sport exercises and not every day activities. The best estimation of energy expenditure is achieved

Table 9

Results of energy expenditure estimation errors per device configuration. MAE is expressed in MET and MAPE in %. In combination of devices the wristband is always present.

		MAE	MAPE	RMSE
Single device	No orientation no location			
	O _w	0.94	29	0.98
	O _s	1.12	34	1.21
	Single model			
	W	0.64	27	0.86
	T	0.67	26	0.92
	J	0.72	34	1.02
	B	0.75	30	1.05
	Multiple model			
	W	0.58	25	0.79
Combination	Single model			
	T	0.59	25	0.79
	J	0.71	31	0.93
	B	0.57	22	0.74
	Multiple model			
	T	0.55	23	0.76
	J	0.59	27	0.82
Bodymedia	B	0.50	18	0.70
		1.03	37	1.60

with wristband and smartphone in trousers pocket, for which we also present a chart in Fig. 13 for one person and all estimations over the duration of the entire scenario (estimations each 10 seconds). The Fig. 13/> presents the energy expended in MET as measured by the indirect calorimeter (blue line) and estimations as estimated by our multiple-model approach (pink line). We can observe that the errors occur when there is significant amount of hand movement while light-intensive activity is performed (e.g., washing hands) and at the beginning of vigorous-intensive activity, since the physiological signal need time to reach the exercise value.

The box-charts in Fig. 12 present the distribution of error for both single-model and multiple-model approach. We can observe that when multiple-model approach is used error decreases for every device configuration and that the range of error distribution narrows, which brings more stable estimations for the activities.

Table 10

Number of orientation independent (O-independent), orientation dependent (O-dependent), physiological (Physiological) and machine-learning (Recognized activity) for each energy expenditure estimation model after feature selection. The configuration which utilizes wristband data use multiple-model approach (a=ambulatory model, l=low level activity model, o=other model) and other use single model approach (s).

Model	W			T	J	B	T			J	B				
	a	l	o	s	s	s	a	l	o	a	l	o	a	l	o
O-independent	4	4	4	3	5	6	9	6	7	10	6	8	3	4	1
O-dependent	14	12	16	10	15	8	15	18	23	16	13	23	10	10	9
Physiological	3	0	4	0	0	0	3	0	2	3	0	3	2	0	4
Recognized activity	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1
# features	22	17	25	14	21	15	28	25	33	30	20	35	16	15	15

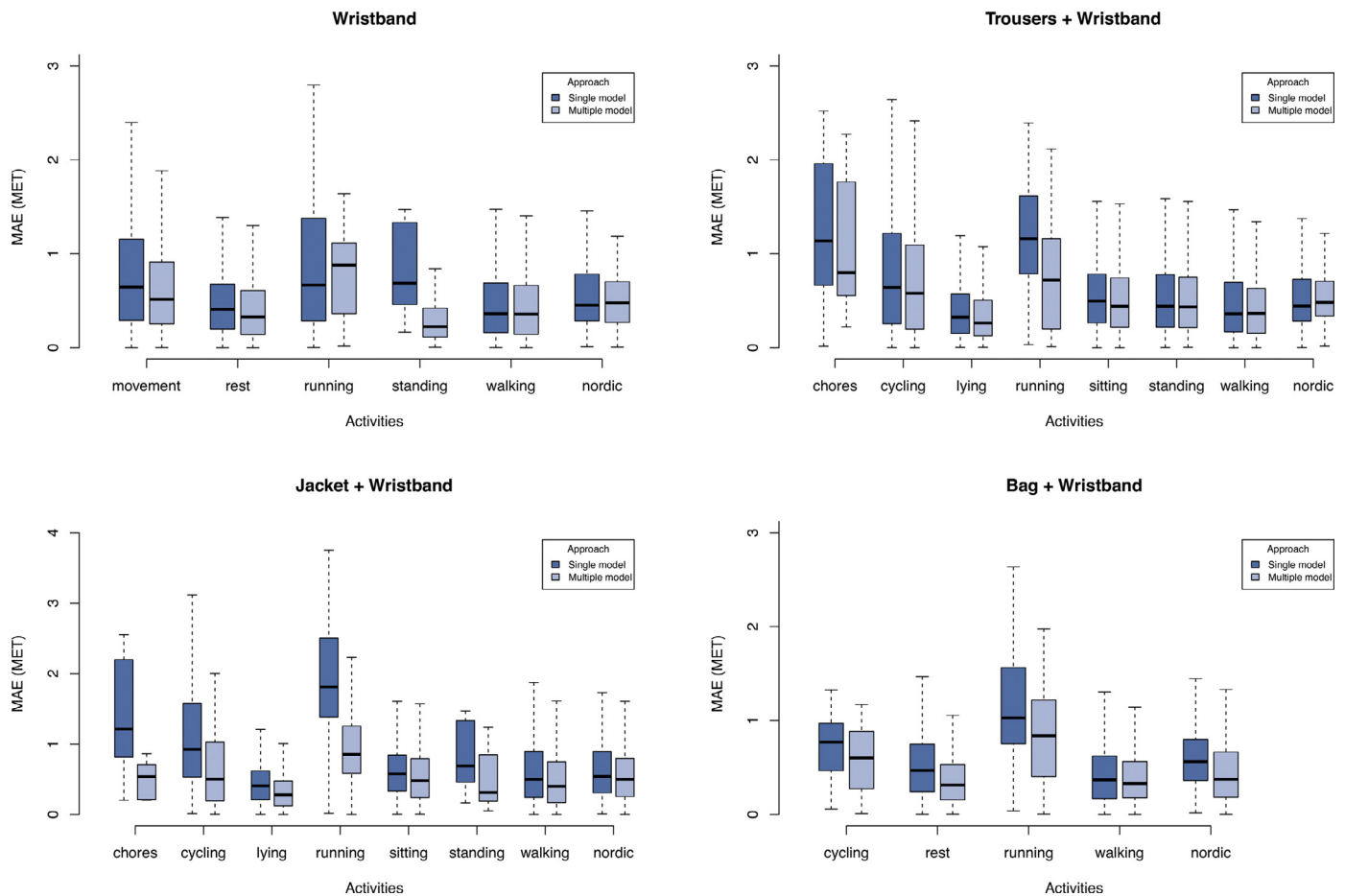


Fig. 12. Mean absolute error comparison between single model approach and multiple model approach for device configurations where multiple modalities are present per recognized activity.

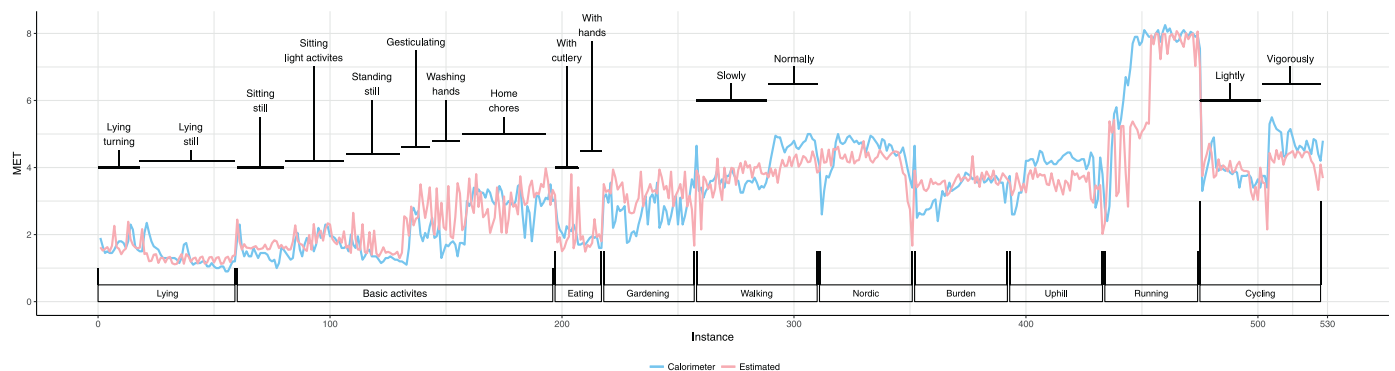


Fig. 13. The comparison between the ground truth as measured by the Oxycon mobile indirect calorimeter (blue line) and estimates from our approach (pink line) for the duration of the entire scenario. (For interpretation of the references to colour in this figure legend, the reader is referred to the web version of this article.)

Table 11

Comparison of the energy expenditure estimation results in kilo-calories. We evaluate the wristband only device configuration to be comparable to other devices. Oxycon measures expended energy and is taken as a ground truth. Bodymedia and Microsoft Band 2 are the estimates from the devices and our approach presents the estimates of our approach. The error columns present comparison against the Oxycon mobile measurements.

	Scenario										Error	
	A	B	C	D	E	F	G	H	I	J	Sum	AVG
Oxycon	13	41	7	28	34	23	27	26	35	45	/	/
Bodymedia	10	44	7	35	36	23	29	27	35	26	40	4
Microsoft band 2	5	13	2	6	10	7	6	7	11	7	202	20
Our approach	14	44	7	26	34	24	28	24	40	41	19	2

We compared our results against the Bodymedia armband and the Microsoft Band 2 in kilo-calories (kcal). For fair comparison we only compare against the results where only wristband is present. We chose to compare them in kilo-calories because Microsoft Band 2 reports the energy expenditure estimations only in this unit and the Oxycon mobile indirect calorimeter and Bodymedia armband also report kilo-calories. Since our models estimate the energy expenditure in METs, we used the Eq. (5) for the conversion of estimated METs of our approach to kilo-calories. The results are reported in Table 11 for each scenario (Section 3.1) and device. We took the Oxycon mobile measurements as the ground truth and calculated the sum of errors and an average error for the entire scenario for each device (right columns in Table 11). From the results we can conclude that our approach outperforms the compared devices and that Bodymedia approach is comparable to our according to the average error in calories. We can observe that Bodymedia perfectly estimates the burned kilo-calories while running and Nordic walking and has significantly lower error compared to Microsoft Band 2, which heavily underestimated all activities. This also proves, that the commodity devices available on the market are not very accurate and that there is a place for improvement.

$$kcal = METs \times weight(kg) \times time(hours) \quad (5)$$

Additionally, the number of remaining features after feature selection do not surpass 35 (decreased from over 90 in single device and over 200 in combination). Table 10 presents the number of orientation-independent, orientation-dependent features, physiological features and in case of combination of devices also the number of features which already need machine-learning (recognized activity by each device). The configurations in which we could not use the multiple-model approach list the number of features per single model (s=single model) and the configurations in which we use multiple-model approach list the features per model (a=ambulatory model, l= low level activity model, o=other model).

We can observe, that the same as in activity recognition, the orientation-dependent features form the majority of the feature vector. This indicates that the decrease in error was affected by the knowledge about the orientation and consequently knowledge about the location.

5. Discussion

The popularity of smartphones and wrist wearables with built-in sensors makes these devices very attractive for activity monitoring. Indeed, in recent years the popularity of activity-monitoring research in which dedicated sensor nodes are replaced with such devices has increased. This makes applications of activity monitoring accessible to a broader population. However, the available activity monitoring with such devices is either very simple and essentially counts the users steps and estimates the energy expenditure accordingly, or is limited to monitoring specific sports activities (run trackers) on the users request. The research on a broader

Table 12

Gain in number of recognized activities (#), gain in activity recognition (AR) accuracy and decrease in energy expenditure estimation error (EEE) when orientation is normalized and the location of the smartphone detected and used as a context for selection of the appropriate machine-learning model.

	#	AR	EEE		
Smartphone	+1	Accuracy	+10%	MAE	−.32
		Kappa	+.26	MAPE	−1%
		F-score	+.18	RMSE	0.1
Wristband	+1	Accuracy	+1%	MAE	−0.45
		Kappa	+.08	MAPE	−6%
		F-score	+.25	RMSE	−0.31
Combination	+3	Accuracy	+8%	MAE	−0.49
		Kappa	+.16	MAPE	−9%
		F-score	+.18	RMSE	−0.34

range of monitored activities either requires the smartphone to be worn at a specific location or mostly uses orientation-independent features which limit the recognition and estimation accuracy.

In this research we explore how to use commercial devices such as accelerometer-embedded smartphone, sensor-rich wristband or both to achieve comparable results to using dedicated sensor nodes placed in a predefined manner. Additionally, we perform feature selection to decrease the computational complexity of the algorithm and explore how to optimally define activities to exploit the location and orientation of the smartphone, wristband or both.

We start with no knowledge about the orientation and location of the devices, and use models with orientation- and location-independent features. With this approach we can recognize five activities with the accuracy of 79 %, kappa of 0.67 and f-score from 0.63 down to 0.5 in the case of the wristband. Once the algorithm is provided with the normalized orientation and the detected location, the number of recognized activities and the activity recognition accuracy increases, while the energy estimation error decreases. The results summarizing the impact of orientation normalization and location detection are presented in Table 12 and are further discussed in the rest of this section.

The activities that can be recognized depend on the locations of the available accelerometers. For example, the smartphone accelerometer has the same orientation during sitting and standing if the smartphone is placed on the torso, and during lying and sitting if the smartphone is placed in the trousers pocket. To draw maximal information from the accelerometer data, different activities need to be recognized with different sensor configurations. We therefore started with a large set of activities and merged some when they could not be distinguished with a given sensor configuration. The procedure of merging the activities is presented in Section 3.5.5 and the achieved results in Section 4.6. The average number of recognized activities gained by using orientation normalization and location detection, averaged across all the smartphone locations, is presented in the row marked with # in Table 12, and the gain in the activity recognition accuracy in the column AR. We can observe that even with an increased number of recognized activities, we achieve an increase in the accuracy of

up to 10 percentage points in the case of smartphone alone. In the case of the wristband we get a slight increase in the recognition accuracy but a significant increase in the f-score. If we fuse the data of both devices, we can recognize three more activities (up to eight) with a higher accuracy, kappa and f-score. The importance of orientation normalization is further explained by Tables 7 and 12, where one can see that the majority of features chosen by the feature selection are orientation-dependent, but one can only take full advantage of such features when the orientation is normalized.

The comparison of the energy expenditure estimation model with orientation normalization and location detection against the no-orientation and no-location model shows a large decrease in all errors (column EEE in Table 12). We start with the average MAE of 1.03 MET and decrease it by 0.32 MET in the case of the smartphone, by 0.45 MET in the case of the wristband and by 0.49 MET when both devices are present. The decrease in MAPE is from 1 to 9 percentage points and RMSE from 0.1 up to 0.34. The contributing factors to the decreased errors are not only the information about the location and normalized orientation, but also fine-grained activity recognition. Unlike in activity recognition, where the direction of the gravity acceleration (contained in the orientation-dependent features) is critical for the recognition of some activities, the magnitude of the acceleration is more important for energy expenditure estimation. The magnitude is strongly affected by the location of the smartphone (e.g., the acceleration in the trousers pocket during ambulatory activities is larger than in the jacket pocket), so location detection is critical.

We experimented only with the accelerometer sensor data from the smartphone, since our objective was to develop a power-efficient application. We believe that the use of additional sensor data such as GPS data, Wi-fi signal, etc. would enable us to recognize an even wider range of activities with a comparable accuracy. However, using additional sensors has an impact on the battery life, so the algorithm would require mechanisms to manage the power via adjusting the active sensors and their frequency, which is not a subject of this research.

6. Conclusion

We present a real-time activity monitoring algorithm for activity recognition and estimation of energy expenditure with smartphone and wristband. The design of the algorithm enables the activity monitoring with individual device as well as with the combination of both. It first detects which devices are present on the body, then it expects ten seconds of walking to detect the gravity and normalize the orientation of the devices which enables the devices to be worn in any orientation. In case the smartphone is present it detects the location of the smartphone, which can be worn freely on the body (trousers pocket, jacket pocket and bag) and uses this information for selection of activity recognition classification model. The last step utilizes the information about the location of the smartphone and the recognized activity for selection of the energy expenditure estimation regression model. The output of the activity monitoring algorithm is the performed activity and energy expenditure expresses in MET. We use indirect calorimeter Oxycon mobile to label the ground truth for estimation of energy expenditure.

We have evaluated each step in the algorithm and compared the results of the energy expenditure estimation model against two commercial devices. We first compared the resulted estimates in MET against Bodymedia armband, which is one of the most accurate EE estimation devices on the market and then we compared the estimates in kilo-calories against Bodymedia armband and Microsoft Band 2. The results have shown that our approach outperforms both devices and that even though there are many commercially available devices on the market for estimation of

energy expenditure, they might not perform as accurately as expected and have room for improvement.

Accurate activity monitoring is important in domains where further decisions about the lifestyle or person-specific recommendations rely on the user's physical activity and its intensity. In the past, we have used simpler algorithms in health domains, for example, we have used activity monitoring with dedicated wearable sensors for monitoring physical activities of schoolchildren (eGibalec project), patients with chronic heart failure and for diabetes patients, for which the quantity of the activity is highly relevant to self-manage the disease [59,60]. This paper presents the upgrade of these approaches and the presented algorithm is implemented into the prototype of the AAL project Fit4Work, pilots of which will start soon. The results of the activity monitoring will be used to provide person-specific recommendations to older workers and help maintain their physical condition in good state. Additionally, this algorithm is going to be implemented as an initial algorithm for activity monitoring in the H2020 Heartman project.

The future work first of all includes the development of the orientation normalization method for the wristband to normalize it along the remaining axis (y-axis and z-axis) and to develop a method for detecting whether the person wears the wristband on right or left wrist. We will also stay informed about wearable devices that are coming to the market and adapt the algorithm to use them. We currently own activity monitoring datasets (labeled with calorimeter) of school-children, younger adults and older workers, and we plan to enrich them with disease-specific data and with more versatile population regarding age, gender and ethnicity.

Acknowledgement

The research presented herewith has been conducted within the Fit4Work project co-financed through the AAL Program (project no. AAL-2013-6-060).

References

- [1] O. Cristian., S. Poser, R. Bernardini, R. Rinaldo, P.E.D. Prampero, Energy cost and metabolic power in elite soccer: a new match analysis approach, *Med. Sci. Sport. Exerc.* 42 (2010) 170–178, doi:10.1249/MSS.0b013e3181ae5cfd.
- [2] S. Chatterjee, A. Price, Healthy living with persuasive technologies: framework, issues, and challenges, *J. Am. Med. Inf. Assoc.* 16 (2009) 171–178, doi:10.1197/jamia.M2859.
- [3] F.J. Penedo, J.R. Dahn, Exercise and well-being: a review of mental and physical health benefits associated with physical activity, *Curr. Opin. Psychiatry.* 18 (2005) 189–193, doi:10.1097/00001504-200503000-00013.
- [4] Runkeeper, 2017. <https://runkeeper.com/>, (accessed 07.03.17).
- [5] FitBit, 2017. <https://www.fitbit.com/>, (accessed 07.03.17).
- [6] WHO, Global Health Observatory (GHO) data, 2017. <http://www.who.int/gho/>, (accessed 07.03.17).
- [7] Fit4Work, AAL project, 2017. <http://www.fit4work-aal.eu/>, (accessed 07.03.17).
- [8] F. Attal, S. Mohammed, M. Dedabirshvili, F. Chamroukhi, L. Oukhellou, Y. Amirat, Physical human activity recognition using wearable sensors, *Sensors (Basel)*. 15 (2015) 3131438, doi:10.3390/s151229858.
- [9] B. Cvetković, R. Milić, M. Luštrek, Estimating energy expenditure with multiple models using different wearable sensors, *IEEE J. Biomed. Heal. Inf.* 20 (2016) 1081–1087, doi:10.1109/JBHI.2015.2432911.
- [10] B.E. Ainsworth, W.L. Haskell, S.D. Herrmann, N. Meckes, D.R. Bassett, C. Tudor-Locke, J.L. Greer, J. Vezina, M.C. Whitt-Glover, A.S. Leon, 2011 compendium of physical activities, *Compend. Phys. Act. Track. Guid. Heal. Lifestyles Res. Center* (2011) 17, doi:10.1249/MSS.0b013e31821e2ce12. *Coll. Nurs. Heal. Innov. Arizona State Univ.*
- [11] A. Health, 2017. <http://www.apple.com/ios/health/>, (accessed 07.03.17).
- [12] R. Gravina, P. Alinia, H. Ghasemzadeh, G. Fortino, Multi-sensor fusion in body sensor networks: state-of-the-art and research challenges, *Inf. Fusion.* 35 (2017) 68–80, doi:10.1016/j.inffus.2016.09.005.
- [13] X. Lai, Q. Liu, X. Wei, W. Wang, G. Zhou, G. Han, A survey of body sensor networks, *Sensors (Basel)*. 13 (2013) 5406–5447, doi:10.3390/s130505406.
- [14] M.A. Hanson, H.C. Powell, A.T. Barth, K. Ringgenberg, B.H. Calhoun, J.H. Aylor, J. Lach, Body area sensor networks: challenges and opportunities, *Computer (Long. Beach. Calif.)*. 42 (2009) 58–65, doi:10.1109/MC.2009.5.
- [15] Y.L. Zheng, X.R. Ding, C.C.Y. Poon, B.P.L. Lo, H. Zhang, X.L. Zhou, G.Z. Yang, N. Zhao, Y.T. Zhang, Unobtrusive sensing and wearable devices for health informatics, *IEEE Trans. Biomed. Eng.* 61 (2014) 1538–1554, doi:10.1109/TBME.2014.2309951.

- [16] U. Mitra, B.A. Emken, S. Lee, M. Li, V. Rozgic, G. Thatte, H. Vathsangam, D. Zois, M. Annavaram, S. Narayanan, M. Levorato, D. Spruijt-Metz, G. Sukhatme, KNOWME: a case study in wireless body area sensor network design, *IEEE Commun. Mag.* 50 (2012) 116–125, doi:[10.1109/MCOM.2012.6194391](https://doi.org/10.1109/MCOM.2012.6194391).
- [17] H. Ghasemzadeh, P. Panuccio, S. Trovato, G. Fortino, R. Jafari, Power-aware activity monitoring using distributed wearable sensors, *IEEE Trans. Human-Mach. Syst.* 44 (2014) 537–544, doi:[10.1109/THMS.2014.2320277](https://doi.org/10.1109/THMS.2014.2320277).
- [18] G. Fortino, R. Giannantonio, R. Gravina, P. Kuryloski, R. Jafari, Enabling effective programming and flexible management of efficient body sensor network applications, *IEEE Trans. Human-Mach. Syst.* 43 (2013) 115–133, doi:[10.1109/TSMCC.2012.2215852](https://doi.org/10.1109/TSMCC.2012.2215852).
- [19] K. Kunze, P. Lukowicz, H. Junker, G. Tröster, Where am I: recognizing on-body positions of wearable sensors, *Locat. Context. Lect. Notes Comput. Sci.* Springer. (2005) 264–275, doi:[10.1007/b136418](https://doi.org/10.1007/b136418).
- [20] K. Kunze, P. Lukowicz, K. Partridge, B. Begole, Which way am i facing: Inferring horizontal device orientation from an accelerometer signal, in: *Proc. - Int. Symp. Wearable Comput. ISWC, 2009*, pp. 149–150, doi:[10.1109/ISWC.2009.33](https://doi.org/10.1109/ISWC.2009.33).
- [21] D. Mizell, Using gravity to estimate accelerometer orientation, in: *Seventh IEEE Int. Symp. Wearable Comput. 2003. Proceedings, 2003*, pp. 252–253, doi:[10.1109/ISWC.2003.1241424](https://doi.org/10.1109/ISWC.2003.1241424).
- [22] M.D. Tundo, E. Lemaire, N. Baddour, Correcting smartphone orientation for accelerometer-based analysis, in: *MeMeA 2013 - IEEE Int. Symp. Med. Meas. Appl. Proc.*, 2013, pp. 58–62, doi:[10.1109/MeMeA.2013.6549706](https://doi.org/10.1109/MeMeA.2013.6549706).
- [23] Y.E. Ustev, O.D. Incel, C. Ersoy, User, device and orientation independent human activity recognition on mobile phones, in: *ACM Conf. Pervasive Ubiquitous Comput. Adjunct. Publ.*, 2013, pp. 1427–1436, doi:[10.1145/2494091.2496039](https://doi.org/10.1145/2494091.2496039).
- [24] M. Shoaib, H. Scholten, P.J.M. Havinga, Towards physical activity recognition using smartphone sensors, in: *2013 IEEE 10th Int. Conf. Ubiquitous Intell. Comput. 2013 IEEE 10th Int. Conf. Auton. Trust. Comput.*, 2013, pp. 80–87, doi:[10.1109/UIC-ATC.2013.43](https://doi.org/10.1109/UIC-ATC.2013.43).
- [25] C.A. Ronao, S. Cho, Human activity recognition with smartphone sensors using deep learning neural networks, *Expert Syst. Appl.* 59 (2016) 235–244, doi:[10.1016/j.eswa.2016.04.032](https://doi.org/10.1016/j.eswa.2016.04.032).
- [26] M. Shoaib, S. Bosch, O. Incel, H. Scholten, P. Havinga, A survey of online activity recognition using mobile phones, *Sensors* 15 (2015) 2059–2085, doi:[10.3390/s150102059](https://doi.org/10.3390/s150102059).
- [27] H. Lu, J. Yang, Z. Liu, N.D. Lane, T. Choudhury, A.T. Campbell, The jigsaw continuous sensing engine for mobile phone applications, in: *Proc. 8th ACM Conf. Embed. Networked Sens. Syst.*, ACM, New York, NY, USA, 2010, pp. 71–84, doi:[10.1145/1869983.1869992](https://doi.org/10.1145/1869983.1869992).
- [28] S. Thiemejarus, A. Henpraserttae, S. Marukatat, A study on instance-based learning with reduced training prototypes for device-context-independent activity recognition on a mobile phone, in: *2013 IEEE Int. Conf. Body Sens. Networks*, 2013, pp. 1–6, doi:[10.1109/BSN.2013.6575462](https://doi.org/10.1109/BSN.2013.6575462).
- [29] A. Anjum, M.U. Ilyas, Activity recognition using smartphone sensors, in: *2013 IEEE 10th Consum. Commun. Netw. Conf. CCNC 2013, 2013*, pp. 914–919, doi:[10.1109/CCNC.2013.6488584](https://doi.org/10.1109/CCNC.2013.6488584).
- [30] H. Guo, L. Chen, G. Chen, M. Lv, Smartphone-based activity recognition independent of device orientation and placement, *Int. J. Commun. Syst.* 29 (2016) 2403–2415, doi:[10.1002/dac.3010](https://doi.org/10.1002/dac.3010).
- [31] H. Martín, A.M. Bernardos, J. Iglesias, J.R. Casar, Activity logging using lightweight classification techniques in mobile devices, *Pers. Ubiquitous Comput.* 17 (2013) 675–695, doi:[10.1007/s00779-012-0515-4](https://doi.org/10.1007/s00779-012-0515-4).
- [32] B. Cvetković, V. Janko, M. Luštrek, Demo abstract: activity recognition and human energy expenditure estimation with a smartphone, in: *Pervasive Comput. Commun. Work. (PerCom Work). 2015 IEEE Int. Conf.*, 2015, pp. 193–195, doi:[10.1109/PERCOMW.2015.7134019](https://doi.org/10.1109/PERCOMW.2015.7134019).
- [33] D. Biswas, A. Cranny, N. Gupta, K. Maharatna, J. Achner, J. Klemke, M. Jöbges, S. Ortmann, Recognizing upper limb movements with wrist worn inertial sensors using k-means clustering classification, *Hum. Mov. Sci.* 40 (2015) 59–76, doi:[10.1016/j.humov.2014.11.013](https://doi.org/10.1016/j.humov.2014.11.013).
- [34] P. Siirtola, P. Laurinen, E. Haapalainen, J. Rönning, H. Kinnunen, Clustering-based activity classification with a wrist-worn accelerometer using basic features, in: *2009 IEEE Symp. Comput. Intell. Data Mining, CIDM 2009 - Proc.*, 2009, pp. 95–100, doi:[10.1109/CIDM.2009.4938635](https://doi.org/10.1109/CIDM.2009.4938635).
- [35] S. Chernbumroong, A.S. Atkins, Activity classification using a single wrist-worn accelerometer, in: *2011 5th Int. Conf. Software, Knowl. Information, Ind. Manag. Appl. Proc.*, 2011, pp. 1–6, doi:[10.1109/SKIMA.2011.6089975](https://doi.org/10.1109/SKIMA.2011.6089975).
- [36] G.M. Weiss, J.L. Timko, C.M. Gallagher, K. Yoneda, A.J. Schreiber, Smartwatch-based activity recognition: a machine learning approach, in: *Proc. IEEE-EMBS Int. Conf. Biomed. Heal. Informatics*, 2016, pp. 426–429, doi:[10.1109/BHI.2016.7455925](https://doi.org/10.1109/BHI.2016.7455925).
- [37] F. Ramos, U. Federal, C. Grande, R. Rolim, U. Federal, C. Grande, H. Oliveira, A.U. Federal, C. Grande, A. Perkusich, U. Federal, C. Grande, Combining smartphone and smartwatch sensor data in activity recognition approaches: an experimental evaluation, *Int. J. Eng. Technol.* 8 (2016), doi:[10.18293/SEKE2016-040](https://doi.org/10.18293/SEKE2016-040).
- [38] J.A. Levine, Measurement of energy expenditure, *Public Health Nutr.* 8 (2005) 1123–1132, doi:[10.1079/PHN2005800](https://doi.org/10.1079/PHN2005800).
- [39] D.R. Bassett Jr., B.E. Ainsworth, A.M. Swartz, S.J. Strath, W.L. O'Brien, G.A. King, Validity of four motion sensors in measuring moderate intensity physical activity, *Med. Sci. Sport. Exerc.* 32 (2000), S471–80.
- [40] S.E. Crouter, E. Kuffel, J.D. Haas, E.A. Frongillo, D.R. Bassett, Refined two-regression model for the actigraph accelerometer, *Med. Sci. Sports Exerc.* 42 (2010) 1029–1037, doi:[10.1249/MSS.0b013e3181c37458](https://doi.org/10.1249/MSS.0b013e3181c37458).
- [41] A.G. Bonomi, G. Plasqui, A.H.C. Goris, K.R. Westerterp, Improving assessment of daily energy expenditure by identifying types of physical activity with a single accelerometer, *J. Appl. Physiol.* 107 (2009) 655–661, doi:[10.1152/japplphysiol.00150.2009](https://doi.org/10.1152/japplphysiol.00150.2009).
- [42] F. Albinali, S.S. Intille, W. Haskell, M. Rosenberger, Using wearable activity type detection to improve physical activity energy expenditure estimation, *Proc. 12th ACM Int. Conf. Ubiquitous Comput. Ubicomp* 10 (2010) 311, doi:[10.1145/1864349.1864396](https://doi.org/10.1145/1864349.1864396).
- [43] E.M. Tapia, *Using Machine Learning for Real-Time Activity Recognition and Estimation of Energy Expenditure*, MIT, 2008 Phd thesis.
- [44] A. Pande, J. Zhu, A.K. Das, Y. Zeng, P. Mohapatra, J.J. Han, Using smartphone sensors for improving energy expenditure estimation, *IEEE J. Transl. Eng. Heal. Med.* 3 (2015), doi:[10.1109/JTEHM.2015.2480082](https://doi.org/10.1109/JTEHM.2015.2480082).
- [45] M. Altini, J. Penders, R. Vullers, O. Amft, Personalizing energy expenditure estimation using physiological signals normalization during activities of daily living, *Physiol. Meas.* 35 (2014) 1797–1811, doi:[10.1088/0967-3334/35/9/1797](https://doi.org/10.1088/0967-3334/35/9/1797).
- [46] M. Altini, J. Penders, R. Vullers, O. Amft, Estimating energy expenditure using body-worn accelerometers: a comparison of methods, sensors number and positioning, *IEEE J. Biomed. Heal. Informatics* 19 (2015) 219–226, doi:[10.1109/JBHI.2014.2313039](https://doi.org/10.1109/JBHI.2014.2313039).
- [47] N. Vyas, J. Farringdon, D. Andre, J.I. Stivoric, Machine learning and sensor fusion for estimating continuous energy expenditure, *AI Mag.* 33 (2012) 55, doi:[10.1609/aimag.v33i2.2408](https://doi.org/10.1609/aimag.v33i2.2408).
- [48] M. Duclos, G. Fleury, P. Lacomme, R. Phan, L. Ren, S. Rousset, An acceleration vector variance based method for energy expenditure estimation in real-life environment with a smartphone/smartwatch integration, *Expert Syst. Appl.* 63 (2016) 435–449, doi:[10.1016/j.eswa.2016.07.021](https://doi.org/10.1016/j.eswa.2016.07.021).
- [49] Microsoft Band, 2017. <https://www.microsoft.com/microsoft-band/en-us>, (accessed 07.03.17).
- [50] B.F. Advantage, 2017. <http://www.bestfitnesstrackerreviews.com/bodymedia-fit-review.html>, (accessed 07.03.17).
- [51] J.M. Lee, Y. Kim, G.J. Welk, Validity of consumer-based physical activity monitors, *Med. Sci. Sports Exerc.* 46 (2014) 1840–1848, doi:[10.1249/MSS.0000000000000287](https://doi.org/10.1249/MSS.0000000000000287).
- [52] E. Empatica, 2017. <https://www.empatica.com/>, (accessed 07.03.17).
- [53] Oxycon, 2017. <http://www.carefusion.com/>, (accessed 07.03.17).
- [54] M.A. Akkermans, M.J.H. Sillen, E.F.M. Wouters, M.A. Spruit, Validation of the oxycon mobile metabolic system in healthy subjects, *J. Sport. Sci. Med.* 11 (2012) 182–183.
- [55] H. Rosdahl, L. Gullstrand, J. Salier-Eriksson, P. Johansson, P. Schantz, Evaluation of the oxycon mobile metabolic system against the Douglas bag method, *Eur. J. Appl. Physiol.* 109 (2010) 159–171, doi:[10.1007/s00421-009-1326-9](https://doi.org/10.1007/s00421-009-1326-9).
- [56] Y. Kim, G.J. Welk, Criterion validity of competing accelerometry-based activity monitoring devices, *Med. Sci. Sports Exerc.* 47 (2015) 2456–2463, doi:[10.1249/MSS.0000000000000691](https://doi.org/10.1249/MSS.0000000000000691).
- [57] G. Zorrilla-Revilla, A. Mateos, O. Prado-Nóvoa, M. Vidal-Cordasco, J. Rodríguez, Carrying loads: validating a portable tri-axial accelerometer during frequent and brief physical activity, *J. Sci. Med. Sport.* (2017), doi:[10.1016/j.jsams.2017.01.236](https://doi.org/10.1016/j.jsams.2017.01.236).
- [58] M. Hall, E. Frank, G. Holmes, B. Pfahringer, P. Reutemann, I. Witten, The WEKA data mining software: an update, *SIGKDD Explor.* 11 (2009) 10–18, doi:[10.1145/1656274.1656278](https://doi.org/10.1145/1656274.1656278).
- [59] N. Oldridge, D.R. Thompson, A.-D. Zwisler, K. Rees, N. Martin, R.S. Taylor, Exercise-based cardiac rehabilitation for coronary heart disease, *J. Am. Coll. Cardiol.* 67 (2016) 1–211, doi:[10.1016/j.jacc.2015.10.044](https://doi.org/10.1016/j.jacc.2015.10.044).
- [60] S.R. Colberg, R.J. Sigal, J.E. Yardley, M.C. Riddell, D.W. Dunstan, P.C. Dempsey, E.S. Horton, K. Castorino, D.F. Tate, Physical activity/exercise and diabetes: a position statement of the American diabetes association, *Diabetes Care.* 39 (2016) 2065–2079, doi:[10.2337/dc16-1728](https://doi.org/10.2337/dc16-1728).

4.3.2 Dataset

The dataset contains data of six healthy elderly volunteers. The volunteers were wearing a wristband equipped with an accelerometer and a heart-rate sensor. To measure the true value of the expended energy, they were wearing an indirect calorimeter Oxycon mobile (“Oxycon Mobile Indirect Calorimeter,” 2017). Since food impacts the metabolic rate, the volunteers were asked to refrain from eating and drinking (except for water) in the 12 hours before the data collection procedure. The wristband and Oxycon mobile data were collected while they performed activities of a predefined scenario (Table 1 in the included paper) in a controlled environment that resulted in one-and-a-half hours’ worth of data on average per person to be used for energy expenditure estimation.

For the dataset to be appropriate for the training procedure, it needs to fulfil the partitioning requirement (see Section 3.2.1). We need to be able to divide the dataset by person and by some semantic value. The division by people is straightforward, and for the semantic value we select the recognised low-level activity also presented in the included paper. The low-level activity recognition is done every 2 seconds and is able to recognise six activities (running, Nordic walk, standing, rest, walking, movement). The pipeline presented in the included paper is used for preprocessing and feature extraction for this experiment.

The amount of partitioned data per person is presented in Figure 4.1 and in Table 4.1. The left chart in Figure 4.1 presents the number of all examples per person (one example is a feature vector constructed on a 10-second window) and the right chart is the number of examples per recognised activity. The amount of examples in terms of their number and duration in minutes is presented in Table 4.1. Note that the used activities are recognised by the activity recognition module with 80% accuracy. On average, we have 516 ± 43 examples per person or 86 ± 7 minutes of data per person.

4.3.3 Training Procedure

The dataset is in line with the general requirements presented in Section 3.2.1. It is partitioned by person and by the recognised activity, which is needed for the training procedure presented in Section 3.2.3.

4.3.3.1 Data Partitioning

The data partitioning as presented in Chapter 3 is enough to train all needed models in real-life applications but for the evaluation experiments, we need to divide the data into additional datasets to ensure data for training the person-specific model, the person-

Table 4.1: Duration in minutes (t) and number of examples ($\#$) per person and recognised activity for energy expenditure estimation.

Person	Running		Nordic		Standing		Rest		Walking		Movement	
	#	t	#	t	#	t	#	t	#	t	#	t
A	31	5	0	0	6	1	132	22	138	23	221	37
B	42	7	66	11	7	1	114	19	195	32	106	18
C	24	4	35	6	22	4	98	16	132	22	125	21
D	0	0	29	5	42	7	121	20	147	25	174	29
E	28	5	40	7	0	0	200	33	143	24	155	26
F	1	0	35	6	0	0	199	33	141	24	149	25

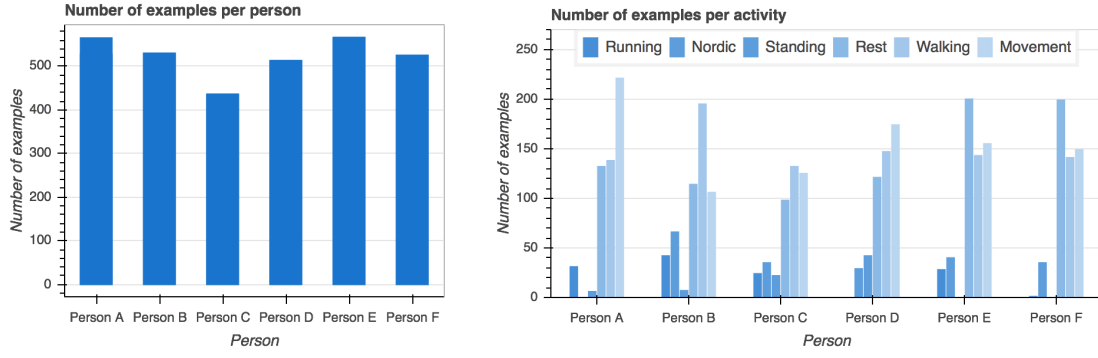


Figure 4.1: Number of examples per person and per type of recognised activity.

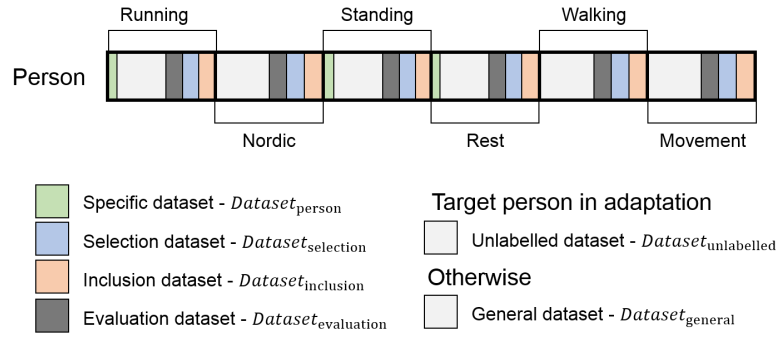


Figure 4.2: Person data divided into datasets used for training the models and evaluation of the MRAT adaptation.

alisation and the evaluation. In addition to the division of the data to $Dataset_{person}$, $Dataset_{selection}$ and $Dataset_{inclusion}$ (as seen in Chapter 3), we need to allocate data for personalisation $Dataset_{unlabelled}$ and data used for the evaluation $Dataset_{evaluation}$. $Dataset_{unlabelled}$ acts as unlabelled data that should be labelled by the method and the $Dataset_{evaluation}$ is used to always evaluate the personalised general model on the same dataset.

Before the division of the data, we have to define the data that will be included in the person-specific dataset (basic data Section 3.2.2.1). Since this should be collected on the spot with the person it should be easy for the person to perform. For this experiment, we defined that the target person should perform 50 seconds of running, 50 seconds of walking and 50 seconds resting which sums up to 90 seconds of person-specific data to be used for $Dataset_{person}$. The divided data is presented in Figure 4.2 and the three steps of the division procedure for the experiment are:

1. A small amount of data from the three predefined activities is randomly selected for $Dataset_{person}$ (green colour in Figure 4.2).
2. The half of the remaining data per activity is randomly selected for the dataset used for personalisation $Dataset_{unlabelled}$ (light gray colour in Figure 4.2).
3. The remaining half is randomly divided by thirds per activity and allocated for $Dataset_{selection}$ (blue colour in Figure 4.2), $Dataset_{inclusion}$ (red colour in Figure 4.2) and $Dataset_{evaluation}$ (dark gray colour in Figure 4.2).

Once the data are divided, we can use the folds for training and evaluation of the required machine-learning models as introduced in the next section. Please note that when the data of a person are used for training the general model, the $Dataset_{\text{unlabelled}}$ becomes $Dataset_{\text{general}}$.

4.3.3.2 Machine-Learning Model Training

We need to train the four models composing the MRAT. The general model, person-specific model, the selection model and the inclusion model. We use the divided data as presented in Section 4.3.2 for training the two regression domain models and the two decision models.

General Model

The general model training procedure is adopted from the included paper. For target person p_{target} , we utilise $Dataset_{\text{general}}$ of all remaining people to train the general model with the Random Forest regression machine-learning algorithm. The evaluation of the trained domain models is done with the LOPO approach in which we evaluate the mean absolute error (MAE) of the model on the $Dataset_{\text{evaluation}}$ of the left-out subject. MAE is calculated with Eq. 4.1 in which MET_{true} represents the MET value as measured with the indirect calorimeter and $MET_{\text{estimated}}$ the value as estimated by the trained model. The sum of differences is divided by the number of observations (n). The results of the evaluation per person are presented in Table 4.7 (row Initial). We can observe that MAE is higher for two people (Persons A and E), which increases the overall mean error, resulting in 0.66 MET.

$$MAE = \frac{\sum_{i=1}^n |MET_{\text{true}} - MET_{\text{estimated}}|}{n} \quad (4.1)$$

To evaluate the achievable MAE with the given dataset, we incrementally included correctly labelled examples from $Dataset_{\text{unlabelled}}$, then retrained and evaluated the general model. The results per person are presented in Table 4.7 (row Best) and learning curves in Figure 4.3 in which we marked the initial MAE of the general model and the final MAE of the general model, both reported in Table 4.7. The learning curves show that the initial MAE for Persons E and F is close to the error when all data are correctly labelled and included into the $Dataset_{\text{general}}$, indicating that the personalisation is not necessary. In case we perform the personalisation for the two people it should not increase the error. The overall mean MAE for the row Best is 0.52 MAE, which is 0.14 MET lower than the overall mean MAE of the initial general model. This difference corresponds to our range of possible improvement. The results of the adaptation procedure are presented in Section 4.3.4.

Person-Specific Model

The person-specific model was trained with the same feature set and regression machine-learning algorithm as the general model. The data for training the model were obtained from the specific dataset ($Dataset_{\text{person}}$) content of which was described in Section 4.3.3.1. The person-specific model was evaluated on $Dataset_{\text{evaluation}}$ for each individual person. The mean MAE of the person-specific model evaluation was high (1.13 MET) as expected since the model was trained only on energy expenditure data of three activities and evaluated on energy expenditure data of all six activities for which the personalisation needs to be performed.

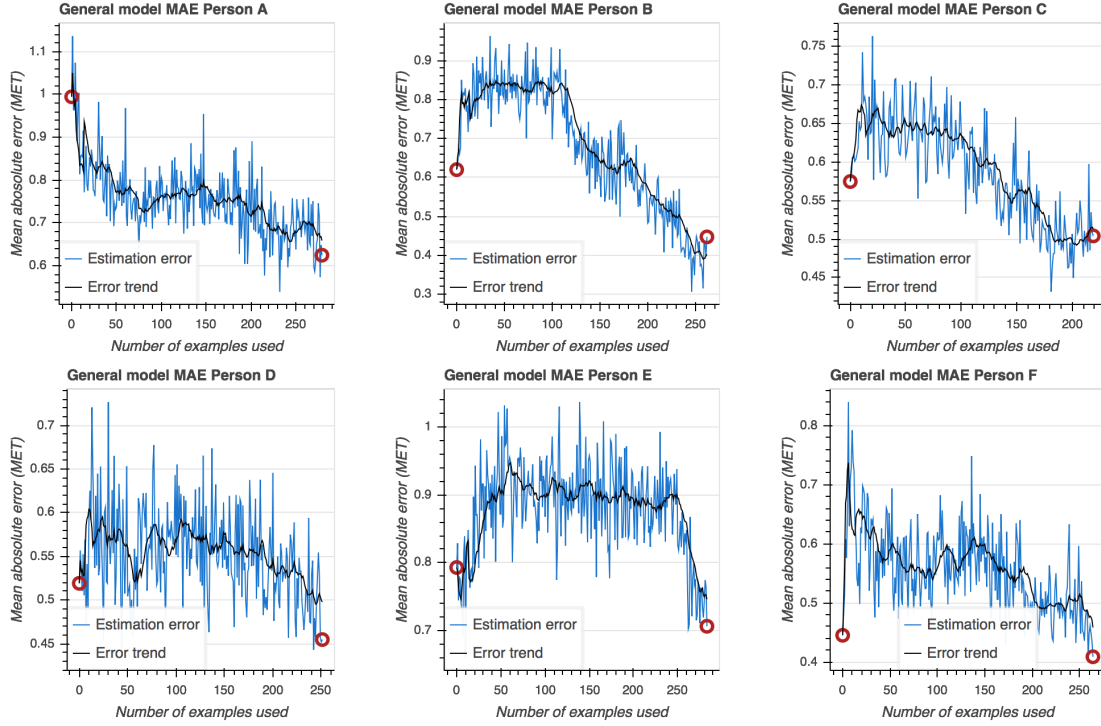


Figure 4.3: Best achieved mean absolute error with Random Forest Regression while incrementally including correctly labelled examples and retraining.

Selection Model

The selection model is trained with a classification machine-learning algorithm. The selection model is trained to decide which of the domain models is more likely to be correct and should predict the final label for the current example.

The dataset for training the selection model $Dataset_{selectionTemp}$ is generated with the modified Algorithm 3.3 using data assigned especially for that ($Dataset_{selection}$ in blue in Figure 4.2). The algorithm is modified to be used for regression domain models from line 10 to 14 in which the algorithm evaluates whether the model correctly labelled the example. We define a condition under which we consider the label correct. Instead of comparing the exact value of the labels against the true value (as in is done when the domain models solve classification task), the condition compares the MAE of the predictions. In addition to the MAE of the predictions as estimated by the domain models, we introduce a third option: a mean value of both labels as estimated by the domain models. The condition compares all three options and labels the example with the option with the lowest MAE (general, person-specific, or mean). In addition to the statistical features extracted for the feature vector, we include context features. These are the recognised activity and the level of intensity (Ainsworth et al., 2011) of the recognised activity. The feature set for the selection model is marked with a tick in the selection column (S) of Table 4.5 (see Section 3.2.2.2) and the conversion of MET value into the level of intensity is presented in Table 4.2.

The selection model can be trained with any classification machine-learning model that can handle three classes and outputs the prediction confidence. We evaluated five machine-learning classification algorithms (Random Forest, CART, SVM, Logistic Regression, and Multi-Layer Perceptron) with the Leave-One-Set-Out (LOSO) approach for training the selection model in which we train the model on the selection model training data of five

Table 4.2: Level of intensity according to MET value range.

Intensity	MET range
Low	$\text{MET} < 3$
Moderate	$3 \leq \text{MET} \leq 6$
Vigorous	$\text{MET} > 6$

Table 4.3: Evaluation of five machine-learning classification algorithms for training the selection model for energy expenditure estimation adaptation. SVM – Support Vector Machine, LogReg – Logistic Regression, MLP – Multi-Layer Perceptron, CART – Decision Tree, RF – Random Forest.

	Majority	SVM	LogReg	MLP	CART	RF
Mean Accuracy [%]	64	78	78	73	71	82
Std. dev [$\pm\%$]	1	2	1	3	1	1

people and test on the data of the remaining person. The evaluation is done for each person, and the mean value is reported in Table 4.3. We can see that the Random Forest performed the best; thus it is selected for training the final selection model.

Inclusion Model

The inclusion model is trained to decide whether a labelled example should be included in the $Dataset_{\text{general}}$ for retraining the general model or not. The output of the inclusion model is either "Yes" or "No". The dataset $Dataset_{\text{inclusionTemp}}$ used for training the inclusion model is generated with modified Algorithm 3.4 using the data especially assigned for the inclusion model ($Dataset_{\text{inclusion}}$, in red in Figure 4.2). The algorithm is modified from lines 13 to 25 in which the algorithm evaluates whether the selected model (by the selection model) labelled the example correctly and should be included in the $Dataset_{\text{general}}$. Since we must evaluate the label as provided by a regression model, we need to define the condition under which we consider the label correct and acceptable for inclusion. We evaluate the MAE of the label as estimated by the selected model against a predefined threshold, and if the MAE is lower than the threshold, it is labelled for inclusion ("Yes"); otherwise it is labelled for rejection ("No"). As in the selection model, we include contextual features into the features set. The feature set for the inclusion model is marked with a tick in the inclusion column (I) of Table 4.5 (see Section 3.2.2.2).

The inclusion model can be trained with any classification machine-learning algorithm that is able to handle the binary classification problem. We evaluated five machine-learning classification algorithms (Random Forest, CART, SVM, Logistic Regression, and Multi-Layer Perceptron) with the Leave-One-Set-Out approach for training the inclusion model.

Table 4.4: Evaluation of five machine-learning classification algorithms for training the inclusion model for energy expenditure estimation adaptation. SVM – Support Vector Machine, LogReg – Logistic Regression, MLP – Multi-Layer Perceptron, CART – Decision Tree, RF – Random Forest.

	Majority	SVM	LogReg	MLP	CART	RF
Mean Accuracy [%]	62	81	81	81	82	84
Std. dev [$\pm\%$]	1	1	2	3	2	2

Table 4.5: Feature set per decision model used for personalisation of energy expenditure estimation.

Feature	S	I	Description
$C_G(label_G)$	✓	✓	Prediction confidence of $label_G$ by G model
$C_P(label_P)$	✓	✓	Prediction confidence of $label_P$ by P model
$Activity$	✓	✓	Recognised activity
$Intensity(Activity)$	✓	✓	Intensity of recognised activity
$Basic(Activity)$	✓	✓	Is the data of recognised activity used in P model
$label_S$		✓	S model output
$C_S(label_S)$		✓	Prediction confidence of $label_S$ by S model

The results are presented in Table 4.4, from which we can see that the Random Forest performed the best; thus it is selected for training the final inclusion model.

4.3.4 Adaptation Procedure

The models trained as described in the previous section are now combined into the MRAT method to be used for personalisation of the human energy expenditure estimation model. We evaluate the procedure in two modes:

- Batch adaptation.
- Incremental adaptation.

In the batch mode, the experiment is set to process all $Dataset_{unlabelled}$ examples before retraining the general model. The examples that are selected for inclusion are removed from $Dataset_{unlabelled}$ and the examples that are not selected for inclusion remain in $Dataset_{unlabelled}$ to be used in the next iteration. In the incremental mode, the retraining is performed every time a new example from $Dataset_{unlabelled}$ is selected for inclusion. The adaptation stopping criterion in batch mode is set as a definite number of iteration runs, and in the incremental mode, it is set as the percentage of discarded examples labelled in an iteration.

We start with the evaluation of the adaptation procedure in the batch mode in which we first evaluate how the error value chosen as a condition for training the inclusion model impacts the personalisation procedure. We have already seen in related work and in Section 4.1 (columns ST and TB in Table 5 of the included paper in which the confidence threshold is set to 100%) that if only the examples with the highest confidence are included in the $Dataset_{general}$, it may occur that the general model's accuracy slightly increases or, in the worst case, stays the same, since retraining with the most confident examples does not provide new information to the general model. We started with a conservative range

Table 4.6: Evaluation of the condition value selected for training the inclusion model.

Condition error	mean MAE	Std. dev.
< 0.1 MAE	0.63	0.20
< 0.2 MAE	0.63	0.20
< 0.3 MAE	0.60	0.14
< 0.4 MAE	0.62	0.19
< 0.5 MAE	0.62	0.20

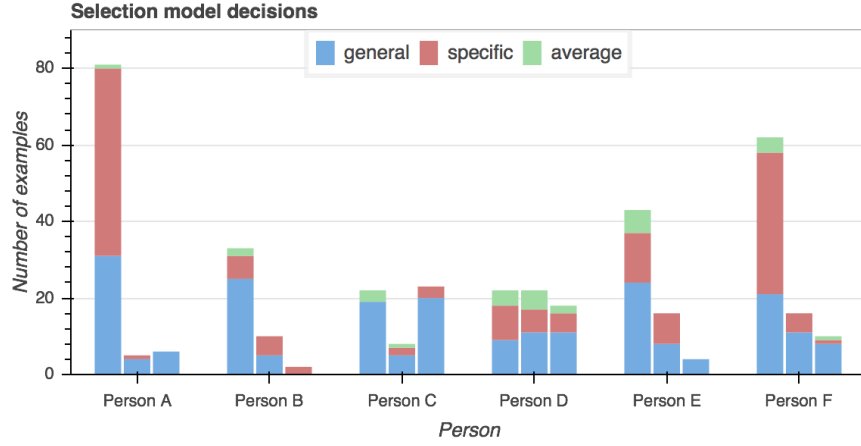


Figure 4.4: Selection model decisions per person and iteration for adaptation of human energy expenditure estimation in the batch setting.

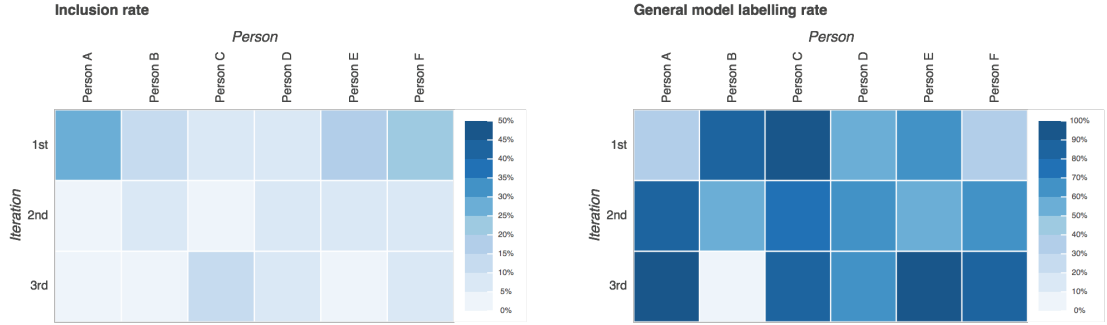


Figure 4.5: The inclusion rate and general model labelling rate per iteration for adaptation of the human energy expenditure estimation model in batch mode.

where the inclusion model is trained to include an example only in case the error is lower than 0.1 MET; in the next experiments, we gradually increased the error up to 0.5 MET and evaluated the performance. The results of these experiments are presented in Table 4.6, where we report the overall mean MAE after 3rd iteration of the procedure. We can see that the most conservative choice of the condition (0.1 and 0.2 MET) slightly decreased the overall error by 0.03 MET (from 0.66 MET row Initial in Table 4.7), whereas the condition set to 0.3 MET results in the overall error decreasing to 0.60 MET. The conditions of 0.4 and 0.5 MET results in a lower overall error than the most conservative approach. We can conclude that, similar to classification tasks, if the condition for inclusion is too conservative, the decrease in error may be lower. The best result in the batch mode was obtained with the condition set to < 0.3 MAE, so we report the results of all three iterations in column B (batch) of Table 4.7, and use the same condition for training the inclusion model in the incremental mode.

The adaptation in the batch mode decreased the overall MAE error to 0.60 MET in the first iteration. In the second and third iteration, the adaptation does not change the overall MAE; thus, it stays 0.60 MET. The decisions of the selection and inclusion model are presented in Figure 4.4 and in Figure 4.5. Figure 4.4 shows that the selection model somewhat equally favours both domain models for making the final prediction and that mean value (green colour in Figure 4.4) is not selected very often. Figure 4.5 shows

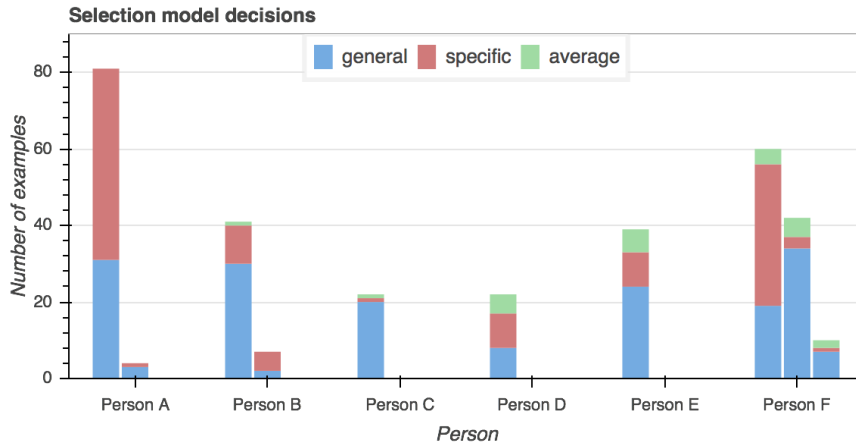


Figure 4.6: Selection model decisions per person and iteration for adaptation of human energy expenditure estimation in the incremental mode.

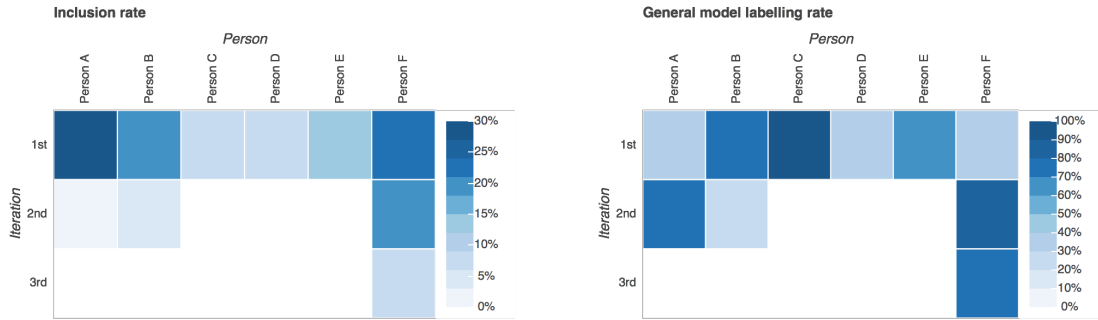


Figure 4.7: The inclusion rate and general model labelling rate per iteration for adaptation of the human energy expenditure estimation model in incremental mode.

that the number of included examples per iteration is low; more specifically, in the first iteration only 17% of examples overall are included in the $Dataset_{general}$, while 6% of the remaining examples are included in second iteration, and 5% in the third iteration. The personalisation is successful for all people except for Person F for whom the personalisation procedure slightly increased the MAE. The reason for this result could be that the initial error of the general model was already low and that personalisation was not even needed. The best result was obtained for Person A, whose initial error of the general model was the highest. The decisions of the selection and inclusion model show that the main portion of the examples for Person A are included in the first iteration in which the selection model mostly prefers the person-specific model for labelling the included examples. In the next two iterations, we can see that the general model mostly labels the included examples, which indicates that it was personalised. We can also see that for Person E, who has mild heart failure and is on beta blocker drugs that maintain low heart rate, the adaptation does not affect the general model performance, probably because the amount of included examples is too low for such specific case.

In the incremental mode, the error is decreased to 0.62 MET (rows I in Table 4.7) in the first iteration. The decisions of the selection model per iteration are presented in Figure 4.6 which shows that the selection model favours somewhat equality both domain models for the final prediction and that average is not selected often. The decisions of inclusion model

Table 4.7: Results of the initial general model, selection model and human energy expenditure estimation adaptation in batch and incremental mode per iteration. Labelled – when all examples are labelled correctly and included for retraining, G – initial general model, P – person-specific model, S – selection model, A – Average, Batch – batch mode, Incremental – incremental mode.

		Person						Mean	Std. dev
		A	B	C	D	E	F		
Labelled (Figure 4.3)		0.62	0.45	0.50	0.45	0.71	0.41	0.52	0.12
G		0.99	0.62	0.58	0.52	0.79	0.45	0.66	0.20
P		1.58	0.83	1.27	0.88	0.97	1.27	1.13	0.29
S		0.97	0.60	0.59	0.55	0.69	0.50	0.65	0.17
A		1.49	0.84	1.29	0.86	0.94	1.22	1.11	0.26
B	1 st iteration	0.80	0.48	0.55	0.51	0.79	0.50	0.60	0.15
	2 nd iteration	0.77	0.50	0.55	0.49	0.81	0.47	0.60	0.15
	3 rd iteration	0.74	0.47	0.55	0.48	0.79	0.56	0.60	0.14
I	1 st iteration	0.95	0.59	0.48	0.55	0.77	0.44	0.62	0.20
	2 nd iteration	0.75	0.48	-	-	-	0.45	0.57	0.15
	3 rd iteration	-	-	-	-	-	0.44	0.57	0.15

and the general model labelling rate are presented in Figure 4.7. The overall number of included examples in the first iteration is low, more specifically 17% of examples, which is the same as in the first iteration in batch mode. In the incremental mode, we use the stopping criterion. The three people that include more than 15% of examples proceeded with the second iteration in which the MAE is decreased to 0.57 MET. The third iteration is performed only for Person F, which does not change overall MAE. The error results in Table 4.7 show that we adapt the general model successfully for all people and that for person C, we achieve a lower error than in the adaptation when all examples from $Dataset_{unlabelled}$ are correctly labelled and included into the $Dataset_{general}$.

Comparing the achieved adaptation results against the initial MAE shows that the overall error is decreased in both modes. In the batch mode, we adapt the general model for all people except for Person F whose initial error was already low. In the incremental mode, we achieve lower error for all people; for Person C, we even obtain lower error than in the case when all examples from $Dataset_{unlabelled}$ are labelled correctly and included into the $Dataset_{general}$ (row Labelled in Table 4.7). As this was also observed in the classification task, it indicates that with the MRAT inclusion model, we exclude examples that do not contribute to decreasing the error. We also include the results of the selection model and results of choosing the correct label using a heuristic such as average label (row A in Table 4.7). We can conclude from the results that stacking the domain models instead of using a heuristic for selection yields better results.

Since the room for improvement in the human energy expenditure estimation domain is very narrow, the test of significance showed non-significant results. This calls for further research of MRAT on other regression domains that can result in modifications of the procedure (e.g., the selection model trained as a regression model, different measure for inclusion criterion, etc.).

4.3.5 Summary

We evaluated and analysed the MRAT on a human energy expenditure domain. We describe the training procedure and the selection of the machine-learning algorithm for

each composing model. The adaptation is done in two modes: the batch mode, in which we retrain after iteration, and the incremental mode, in which the retraining occurs every time an example is included in the general model training set.

The domain was very challenging since the initial general model already performed with low error. We identified one person from the set for whom the estimations using the initial model stood out in terms of error and who was a good candidate for personalisation (Person A) of the model using MRAT. We performed the adaptation on all people in the set and saw that, compared to the classification tasks, the number of included examples is lower, but the overall result shows that the MRAT decreased the overall error. In case of the batch mode, we improved the estimations for the person whose error stood out, but we also increased the error of the person whose initial general model error was the lowest. In the incremental mode in which we used the stopping criterion, we managed to decrease the error even further. The overall error decreased, and so did the error per each person.

The MRAT is a good candidate to be analysed and modified in future work. The low improvement might come from the fact that in this domain we had little room for improvement. MRAT needs a thorough analysis of the possibility to use the regression model instead of the classification model for the selection model and to include the dynamic stopping criterion in the batch mode.

4.4 Use Case 4: Identification Task – Classification

Gait analysis using acceleration data with the goal to identify a person is a research in the biometrics domain (Nickel, Brandt, & Busch, 2011; Choi, Youn, LeMay, Burns, & Youn, 2014; Johnston & Weiss, 2015). The research is based on the hypothesis that people differ according to their walking signature, hence if analysed correctly it can be used for identification purposes. Moreover, using acceleration data for this purpose is less intrusive than using video cameras and is as such more appealing.

The identification by analysing the walking signature can be used to grant access to authorised people, either to the device itself or to the secured environments. For example, at the entrance to a secured facility, a person takes the device, puts it in the pocket, and walks in. If the identification and authorisation are successful, the access is granted. Once the user leaves the facility, the device is left there to be used by other people. We assume that the device is generating acceleration data while in use. These data can be used for personalisation of the model which will improve identification.

The research showed that identification from the gait analysis is reasonably accurate if the training dataset contains various types of walking for each person to be identified. As in other similar domains, the lack of person-specific data makes it difficult to personalise and adapt the models in a supervised way. The adaptive approaches to improve the identification focus on normalisation of the segmented data into step-sized windows to achieve pace-independent patterns and use them for identification (Juefei-Xu, Bhagavatula, Jaech, Prasad, & Savvides, 2012; J. H. Wang, Ding, Chen, & Chen, 2012). These methods take into account only the step size and speed during the regular walk and not during various types of walking (e.g., walking up the hill, walking while carrying a burden).

In this task, we would like to train an identification model, that will identify a person regardless of the type of walking. To achieve that we employ MCAT method. The MCAT is gradually fed six types of walking for the current person. The MCAT performs the identification using general and person-specific model, then uses the selection model to decide on the final identification label, and finally, the inclusion model is used to evaluate the reliability of the prediction to be used for retraining.

There are two types of problems addressed in this task:

- Introduction of a new label. We add a new person as a new label for which we perform the personalisation. This is analysed when a single new label is introduced and when two new labels are introduced.
- Adapt to the a new walking type per person. Personalisation of the identification model to accurately identify a person regardless of the type of walking performed (nature of sudden and gradual concept drift Section 1.1).

This task is a special case since in contrast to previous use cases we start with the initial general model that has low classification accuracy since it does not contain many (or any) examples that are labelled with the new person identifier we are introducing to the system. We chose to present this use case to show how MCAT can be used in such cases. In this use case, we show how MCAT is used when we focus on adapting to a single label that has many intra-label variations (person is a label that needs to be identified regardless of the walking type). We extend this to adapting to two labels since this requires the modification of the data selection for the training procedure. Moreover, we experiment with the case in which the person-specific model is trained with the machine-learning model that can handle single class.

The section is structured as follows. We begin with the introduction of the dataset used in the experiment, then we present training and evaluation of the models composing the MCAT utilising the training procedure (Section 3.2.3) and finally, we present the evaluation and analysis of the model personalisation using MCAT.

4.4.1 Dataset

The identification dataset contains smartphone tri-axial acceleration data of six elderly people (three female and three male). The volunteers carried a smartphone in their trousers pocket and performed a predefined scenario presented in the included journal paper in Section 4.3. The scenario was composed of various activities from which we extracted six types of walking segments and labelled them with a person identifier. The walking types present the variations inside a single label, the person identifier. The walking types are:

- Casual walk (Casual) – people were instructed to walk casually around the room at their most comfortable pace
- Slow walk (Slow) – people were on a treadmill walking at 4 km/h
- Normal walk (Normal) – people were on a treadmill walking at 6 km/h
- Walking uphill (Uphill) – people were on a treadmill walking at 3 km/h with 4% inclination
- Walking while carrying a burden (Burden) – people were on a treadmill walking at 4 km/h and carrying a burden (4 kg or 6 kg, gender-dependent)
- Nordic walk (Nordic) – people were instructed to perform Nordic walk at 6 km/h

The original dataset presented in the included paper contained 10 people, but due to loss of walking data and due to some peoples' health constraints (not being comfortable to perform intenser walking types for a longer period of time), we selected the six people that had a comparable amount of walking data per type. We denoted the set of people with P_{all} (Eq. 4.2). The type and duration of the walking segments in minutes are presented in Table 4.8. Walking segments are composed of examples. One example is a feature vector consisting of features extracted from the accelerometer data in a two-second window and

Table 4.8: Duration in minutes (t) and number of examples ($\#$) per person and type of walking.

Person	Casual		Slow		Normal		Uphill		Burden		Nordic	
	#	t	#	t	#	t	#	t	#	t	#	t
A	10	0.3	87	2.9	88	2.9	0	0	111	3.7	0	0
B	43	1.6	88	2.9	91	3.0	150	5.0	162	5.4	137	4.6
C	0	0	105	3.5	120	4.0	163	5.4	157	5.2	120	4.0
D	12	0.6	92	3.1	103	3.4	157	5.2	155	5.2	141	4.7
E	36	1.2	105	3.5	9.8	3.3	161	5.4	167	5.6	148	4.9
F	43	1.4	99	3.3	103	3.4	0	0	173	5.8	149	5.0

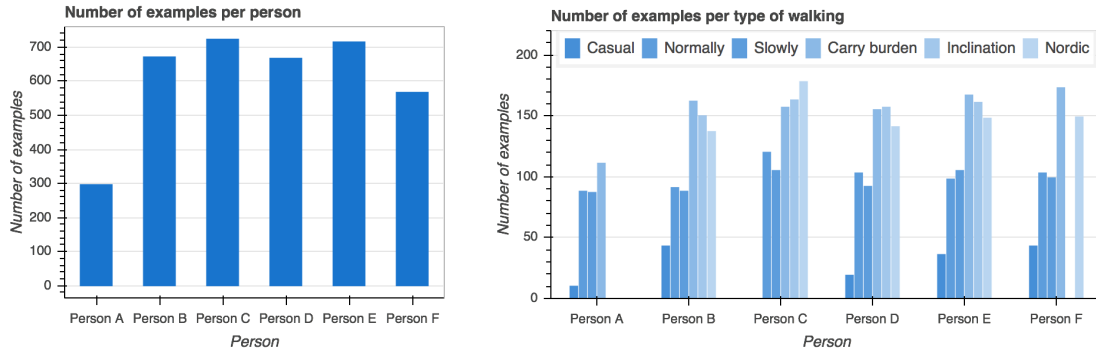


Figure 4.8: Number of examples in the dataset per person and per type of walking.

labelled with the person identifier. The number of examples per person and walking type is presented in Table 4.8 and in Figure 4.8. Please note that the difference in the amount of data compared to the data used in Section 4.3 occurs because the data is collected in the different stage of the pipeline presented in the included paper. We observed that the number of examples varies among the people and that, in some cases, there are no examples for certain types of walking (due to health constraints or loss of data). On average, we have 606 ± 162 examples per person or 20.2 ± 5.4 minutes of data per person.

$$P_{\text{all}} = \{A, B, C, D, E, F\} \quad (4.2)$$

4.4.2 Training procedure

The dataset is in line with the general requirements presented in Section 3.2.1. It can be partitioned by person and by type of walking, which is needed for the training procedure presented in Section 3.2.3.

4.4.2.1 Data partitioning

The data partitioning as presented in Chapter 3 is enough to train all needed models in real-life applications. For evaluation experiments, we need to divide the data into additional datasets to ensure data for training the person-specific model, the personalisation and the evaluation since we are simulating the target person. In addition to the division of the data to $Dataset_{\text{person}}$, $Dataset_{\text{selection}}$ and $Dataset_{\text{inclusion}}$ (as seen in Chapter 3), we need to allocate data for personalisation $Dataset_{\text{unlabelled}}$ and data used for the evaluation $Dataset_{\text{evaluation}}$. $Dataset_{\text{unlabelled}}$ acts as unlabelled data that should be labelled by the

method and the $Dataset_{evaluation}$ is used to always evaluate the personalised general model on the same dataset.

Before the division of the data, we have to define the data that will be included in the person-specific dataset. Since this should be collected on the spot with the person it should be something easy to perform. For this experiment, we defined that the target person should perform 10 seconds of casual walk, 10 seconds of normal walk and 10 seconds walking up the hill which sums up to 30 seconds of person-specific data to be used for $Dataset_{person}$. This represents the first difference compared to use cases presented in previous sections. We select different variations of the same class to be included in the $Dataset_{person}$ and not data of different classes as seen previously. The divided data for a single person are presented in Figure 4.9. The division procedure is done in three steps:

1. Randomly select examples per defined type for $Dataset_{person}$ (green colour in Figure 4.9).
2. Randomly select half of the remaining examples for $Dataset_{unlabelled}$ (light gray colour in Figure 4.9).
3. Randomly select one-third of the remaining examples for $Dataset_{selection}$ (blue colour in Figure 4.9), third for $Dataset_{inclusion}$ (red colour in Figure 4.9) and third for $Dataset_{evaluation}$ (dark gray colour in Figure 4.9).

Once the data are divided, we can use the datasets as folds for training and evaluation of the required machine-learning models as introduced in the next section. Please note that when the data of a person are used for training the general model, $Dataset_{unlabelled}$ is used as $Dataset_{general}$.

4.4.2.2 Machine-Learning Model Training

We need to train four models for the MCAT method: the general model, the person-specific model, the selection model, and the inclusion model. We use the partitioned datasets presented in the previous section for training and evaluation of the individual models. We adopt the preprocessing module with the feature extraction for the domain models from the included paper in Section 4.3 (Cvetković et al., 2017). The feature vector is composed of features extracted for accelerometer, the label (class) of the feature vector is the respective person identifier.

We evaluate the MCAT for personalisation of the identification model in two separate tasks: the adaptation of the model to identify one new person (single person adaptation),

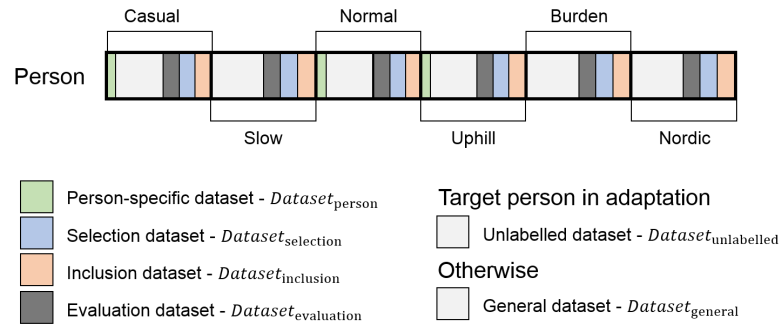


Figure 4.9: Person data divided into datasets used for training the models and evaluation of the adaptation.

Table 4.9: The evaluation of five machine-learning classification algorithms for training the general model in a simulation of adaptation using correctly labelled examples. SVM – Support Vector Machine, LogReg – Logistic Regression, MLP – Multi-Layer Perceptron, CART – Decision Tree, RF – Random Forest.

	SVM	LogReg	MLP	CART	RF
Mean Accuracy (25% examples)	43	32	7	52	64
Mean Accuracy (50% of examples)	62	58	14	74	76
Mean Accuracy (75% of examples)	73	68	17	80	81
Mean Accuracy (100% of examples)	84	73	30	85	89

followed by adaptation of the model to identify two new people (two-people adaptation). We choose to present both experiments because of the important difference in the number of classes in the person-specific model. Training of each model is composed of the selection and evaluation of the machine-learning algorithm.

General Model

The aim is to use the MCAT method to gradually include self-labelled examples into the $Dataset_{\text{general}}$ and retrain the general model, thus introduce person-specific examples to the model and increase its identification accuracy.

To train an efficient initial general model, we need to define the feature set and select the best performing machine-learning model for the task. The feature set is adopted from our previous research (Section 4.3 (Cvetković et al., 2017)). As mentioned before, the feature vector is composed of features extracted from the accelerometer, and the label is the respective person identifier. The selection of the machine-learning model is slightly difficult in this task. Since we are about to introduce a new label (person identifier) to the model, the $Dataset_{\text{general}}$ does not contain this label, so the common Leave-one-person-out (LOPO) approach to evaluate machine-learning models is not appropriate. For example, the general dataset is able to identify five people, but we would like to test the model in identifying the sixth person (LOPO), this will always result in misclassification. We choose to evaluate the machine-learning algorithms for training the general model in a simulated personalisation procedure in which we start with a general model trained on $Dataset_{\text{general}}$ ($P_{\text{general}} = P_{\text{all}} \setminus P_{\text{target}}$) that has no examples of the target person (no examples labelled with the target person identifier) and incrementally include correctly labelled examples from $Dataset_{\text{unlabelled}}$ for target person and evaluate the general model on $Dataset_{\text{evaluation}}$ of the target person. We evaluate five classification machine-learning algorithms as implemented in the Scikit-learn library (Random Forest, CART, SVM, Logistic Regression, Multi-Layer Perceptron) for Python (Pedregosa et al., 2011). The results are presented in Table 4.9. We present the mean accuracy achieved by each algorithm when 25%, 50%, 75%, and 100% of the examples from the $Dataset_{\text{unlabelled}}$ are used for retraining. We can observe that CART and Random Forest achieve comparable final results and that in each section Random Forest outperforms CART by a few percentage points. Most importantly, Random Forest achieves good results even with 25% of data. We choose the Random Forest algorithm for all general models we train for the identification task. The accuracy while incrementally including correctly labelled examples in the $Dataset_{\text{general}}$ and retraining the general model is presented in Figure 4.10 for every person. The charts also include marks (red circles) that indicate when the new walking type was introduced for retraining.

In **single-person adaptation**, we introduce a new label to the model and as presented

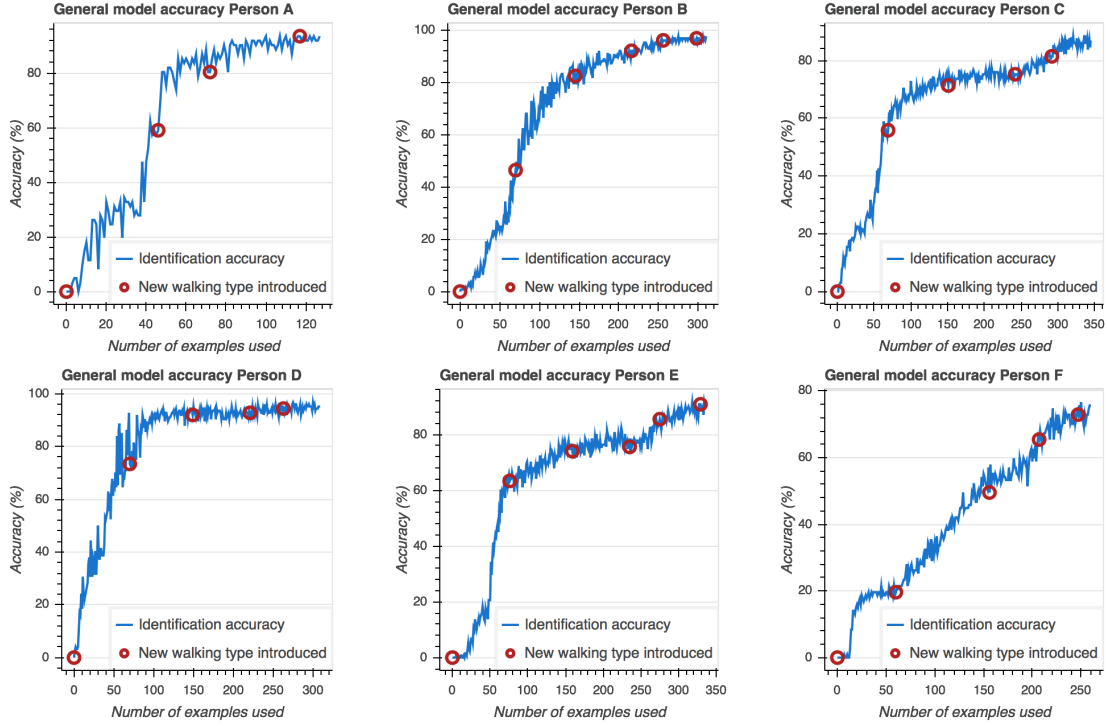


Figure 4.10: Accuracy achieved while incrementally including correctly labelled examples and retraining with Random Forest.

in Chapter 3 (Section 3.2.2.1), in this case we merge the $Dataset_{\text{general}}$ of the five remaining people (Eq. 4.3) with target person's (p_{target}) $Dataset_{\text{person}}$ for training the initial general model (Eq. 4.4). This extends the number of labels in the initial general model with the Random Forest algorithm which yields initial mean accuracy of $8\% \pm 6\%$. The initial accuracies per person will be elaborated in Section 4.4.3. In **two-people adaptation**, the initial $Dataset_{\text{general}}$ is composed of the data for four remaining people P_{general} (Eq. 4.5 and Eq. 4.6) target persons $Dataset_{\text{person}}$ for training the initial general model (Eq. 4.7) with the Random Forest algorithm. This introduces the two new labels to the initial general model. The initial mean accuracy of the general model is $14\% \pm 8\%$. The initial accuracies per person will be elaborated in Section 4.4.3.

$$P_{\text{general}} = P_{\text{all}} \setminus \{p_i\} \quad (4.3)$$

$$Dataset_{\text{general}} = \bigcup_{p_k \in P_{\text{general}}} Data_{\text{general}}(p_k) \cup Data_{\text{person}}(p_i) \quad (4.4)$$

$$P_{\text{target}} = \{p_i, p_j\} \quad (4.5)$$

$$P_{\text{general}} = P_{\text{all}} \setminus P_{\text{target}} \quad (4.6)$$

$$Dataset_{\text{general}} = \bigcup_{p_k \in P_{\text{general}}} Data_{\text{general}}(p_k) \cup \bigcup_{p_l \in P_{\text{target}}} Data_{\text{person}}(p_l) \quad (4.7)$$

Table 4.10: Evaluation of machine-learning algorithms to be used for person-specific model.

	Random	LOF	OneClassSVM
Mean Accuracy [%]	16.7	54.3	75.0
Std. deviation [$\pm\%$]	4.0	8.3	2.8

Table 4.11: Evaluation of five machine-learning classification algorithms for training the person-specific model for adaptation of two people. SVM – Support Vector Machine, LogReg – Logistic Regression, MLP – Multi-Layer Perceptron, CART – Decision Tree, RF – Random Forest.

	Majority	SVM	LogReg	MLP	CART	RF
Mean Accuracy [%]	44	56	83	52	80	84
Std. dev [$\pm\%$]	9	13	19	10	26	11

Person-Specific Model

For person-specific models, we take the $Dataset_{\text{person}}$ for the target person. The content of $Dataset_{\text{person}}$ was introduced in Section 4.4.2.1. The training of the person-specific model is done with the best performing machine-learning classification algorithm for the task, which is evaluated on $Dataset_{\text{evaluation}}$.

In **single-person adaptation**, the specific dataset contains data of only one person, therefore it is able to identify only one person. The person-specific model needs to be trained with a machine-learning algorithm that can handle one-class classification and outputs the prediction confidence (label probability). We evaluated the Local Outlier Factor (LOF) algorithm and One Class Support Vector Machine (OneClassSVM) (Schölkopf, Williamson, Smola, Shawe-Taylor, & Platt, 1999) algorithms as implemented in the Scikit-learn library for Python, and compared the results against a random output. The evaluation of the accuracy of the person-specific model is done on the $Dataset_{\text{evaluation}}$ for the currently evaluated person. The results are presented in Table 4.10 from which we see that OneClassSVM returned the highest accuracy for the task, making it the final algorithm for training the person-specific model for single-person adaptation.

In **two-people adaptation**, the $Dataset_{\text{person}}$ contains data of the two people ($\{p_{\text{target}_i}, p_{\text{target}_j}\}$) for which the adaptation is performed. There are 15 unique combinations of two people for which we train the person-specific model. The person-specific model can be trained with any machine-learning algorithm that can handle binary classification and outputs the prediction confidence. We evaluated five machine-learning classification algorithms: Random Forest, CART, SVM, Logistic Regression, and Multi-Layer Perceptron. The evaluation was done on $Dataset_{\text{evaluation}}$ for the $\{p_{\text{target}_i}, p_{\text{target}_j}\}$. The results are presented in Table 4.11. The Random Forest, CART, and Logistic Regression return a comparable accuracy. We choose the Random Forest algorithm, which scored 84% mean accuracy.

Selection Model

The selection model is trained to decide which of the domain models is more likely to be correct and should provide the final class prediction for the current example. The dataset for the selection model ($Dataset_{\text{selectionTemp}}$) is generated with Algorithm 3.3 using data assigned especially for the selection model $Dataset_{\text{selection}}$ (blue in Figure 4.9). The features used in the feature set of the selection model are presented in column S of Table 4.14 (also

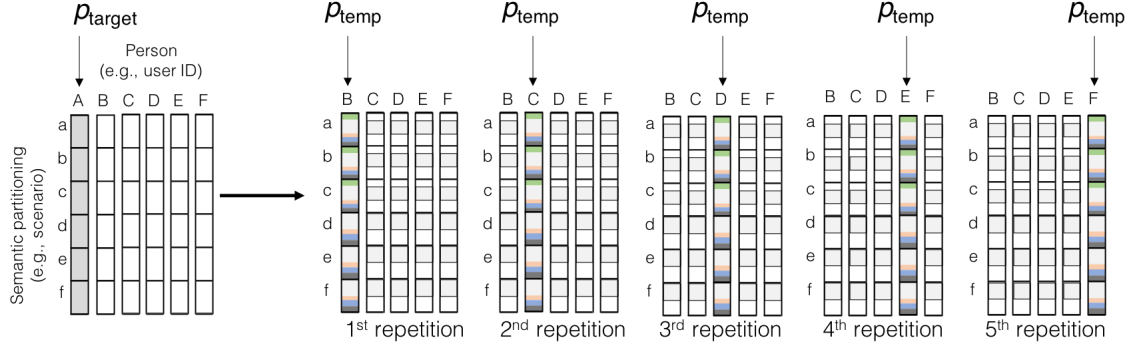


Figure 4.11: Selection and inclusion model training data division for single-person adaptation.

Table 4.12: Evaluation of five machine-learning classification algorithms for training the selection model for single-person adaptation (S) and for two-people adaptation (T). SVM – Support Vector Machine, LogReg – Logistic Regression, MLP – Multi-Layer Perceptron, CART – Decision Tree, RF – Random Forest.

Task	Measure	Majority	SVM	LogReg	MLP	CART	RF
S	Mean Accuracy [%]	50	50	89	50	95	87
	Std. dev [$\pm\%$]	1	2	17	4	1	3
T	Mean Accuracy [%]	61	82	91	88	92	91
	Std. dev [$\pm\%$]	1	2	1	2	1	2

see Section 3.2.2.2). The model can be trained with any classification machine-learning algorithm that can handle binary classification and outputs the prediction confidence.

The difference between generation of the $Dataset_{\text{person}}$ in **single-person adaptation** and **two-people adaptation** is in the exclusion of person data from the Algorithm 3.3. In the single-person adaptation, we exclude the single target person and run the procedure generating the $Dataset_{\text{person}}$ and training the selection model on the remaining data as presented in Section 3.2.3. In the two-people adaptation, we exclude the pair of target people and run the Algorithm 3.3 on the remaining data, in each repetition taking two people as a temporary target pair. The training procedure for single-person adaptation is presented in Figure 4.11 and the training procedure for two-people adaptation in Figure 4.12.

The output of a repetition is a $Dataset_{\text{selection}}$ for the p_{temp} in case of single-person adaptation and for the $\{p_{\text{temp}_i}, p_{\text{temp}_j}\}$ in case of two-people adaptation. The outputs of all repetitions are merged into the $Dataset_{\text{selectionTemp}}$, which is then used for training the final selection model for the target person p_{target} . For the evaluation we perform the Leave-One-Set-Out (LOSO) approach in which we train the selection model on outputs of all repetitions excluding one, which we use for evaluation. This procedure is repeated for each p_{temp} and $\{p_{\text{temp}_i}, p_{\text{temp}_j}\}$. We evaluated five machine-learning classification algorithms (Random Forest, CART, SVM, Logistic Regression, and Multi-Layer Perceptron) on $Data_{\text{evaluation}}$ for the p_{temp} in single-person adaptation and for $\{p_{\text{temp}_i}, p_{\text{temp}_j}\}$ in two-people adaptation in each repetition. The results are presented in Table 4.12. The results for single-person adaptation are in first two rows marked with S and the results for two-people adaptation in bottom two rows marked with T. We chose CART algorithm in both tasks.

The analysis of the trees trained with CART in single-person adaptation show that the

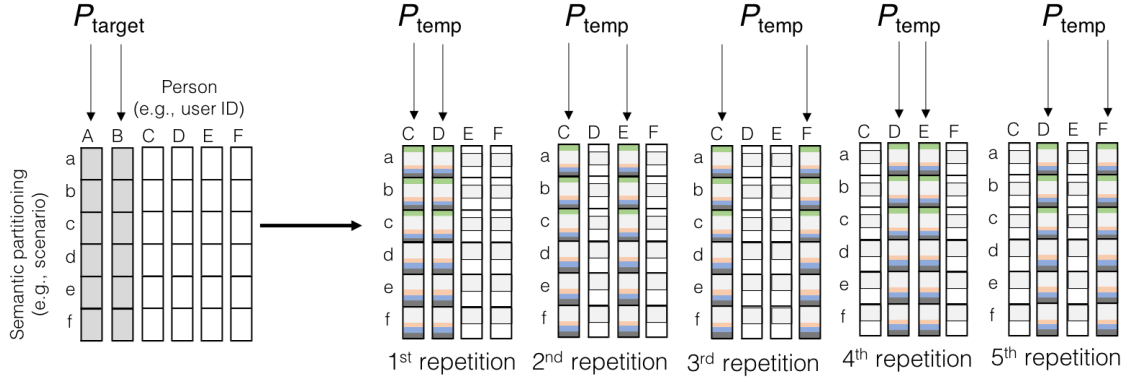


Figure 4.12: Selection and inclusion model training data division for two-people adaptation.

most important feature is the person-specific model's prediction confidence in its own label $C_P(\text{label}_P)$; if it is too low, the general model should be selected for the final labelling. The next most important feature is the $\text{Basic}(\text{label}_G)$, which tells the specific model whether the label predicted by the general model is the one that can be predicted by the person-specific model (see Section 3.2.2.2). If it is, then the person-specific model is selected for the final labelling; otherwise, the general model should label the example. The analysis of the trees trained with CART in two-people adaptation show that the most important feature is the person-specific model's prediction confidence in label as predicted by the general model $C_G(\text{label}_G)$ (see Section 3.2.2.2); if the confidence is low, then the general model is selected for final labelling. The next most important feature is the person-specific model's confidence in its own label $C_P(\text{label}_P)$. If the confidence is high, then the person-specific model predicts the final label; otherwise, the general model does.

Inclusion Model

The inclusion model is trained to decide whether the currently labelled example should be included in the general dataset and used for retraining the general model or not. The output of the inclusion model is either “Yes” or “No”. The dataset for the inclusion model $\text{Dataset}_{\text{inclusionTemp}}$ is generated with Algorithm 3.4 using data assigned especially for the inclusion model $\text{Dataset}_{\text{inclusion}}$ (red in Figure 4.9). The features used in the feature set of the inclusion model are presented in column I of Table 4.14 (see Section 3.2.2.2). The inclusion model can be trained with any classification machine-learning algorithm that can handle binary classification.

The same as in training the selection model, the difference between generation of the $\text{Dataset}_{\text{inclusion}}$ in **single-person adaptation** and **two-people adaptation** is in the exclusion of person data from the Algorithm 3.4. In the single-person adaptation, we exclude the single target person and run the procedure generating the $\text{Dataset}_{\text{inclusion}}$ and training the inclusion model on the remaining data as presented in Section 3.2.3. In the two-people adaptation, we exclude the pair of target people and run the Algorithm 3.4 on the remaining data, in each repetition taking two people as a temporary target pair. The training procedure presented in Figure 4.11 and Figure 4.12 also applies for training the inclusion model.

The output of a repetition is a $\text{Dataset}_{\text{inclusion}}$ for the p_{temp} in case of single-person adaptation and for the $\{p_{\text{temp}_i}, p_{\text{temp}_j}\}$ in case of two-people adaptation. The outputs of all repetitions are merged into the $\text{Dataset}_{\text{inclusionTemp}}$, which is then used for training the final inclusion model for the target person p_{target} . For the evaluation we perform the

Table 4.13: Evaluation of five machine-learning classification algorithms for training the inclusion model for the single-person adaptation (S) and for two-people adaptation (T). SVM – Support Vector Machine, LogReg – Logistic Regression, MLP – Multi-Layer Perceptron, CART – Decision Tree, RF – Random Forest.

Task	Measure	Majority	SVM	LogReg	MLP	CART	RF
S	Mean Accuracy [%]	65	89	93	34	98	97
	Std. dev [$\pm\%$]	1	4	1	1	0.1	1
T	Mean Accuracy [%]	73	85	88	90	98	98
	Std. dev [$\pm\%$]	1	2	1	2	1	1

Table 4.14: Feature set per decision model used for personalisation of identification task.

Feature	S	I	Description
$C_G(label_G)$	✓	✓	Probability of $label_G$ by G model
$C_P(label_P)$	✓	✓	Probability of $label_P$ by P model
$C_G(label_P)$	✓	✓	Probability of $label_P$ by G model
$C_P(label_G)$	✓	✓	Probability of $label_G$ by P model
$Basic(label_G)$	✓	✓	Can be $label_G$ recognised by P model
$Equal(label_G, label_P)$	✓	✓	Are $label_G$ and $label_P$ equal
$label_S$		✓	S model output
$C_S(label_S)$		✓	Probability of $label_S$ by S model

Leave-One-Set-Out (LOSO) approach in which we train the inclusion model on outputs of all repetitions excluding one, which we use for evaluation. This procedure is repeated for each p_{temp} and $\{p_{temp_i}, p_{temp_j}\}$. We evaluated the same five machine-learning classification algorithms in the same way as in the selection model training experiments. The results are presented in Table 4.13. The results for single-person adaptation are in first two rows marked with S and the results for two-people adaptation in bottom two rows marked with T. We chose CART algorithm in both tasks. We can observe that in single-person adaptation, CART, Random Forest, and Logistic Regression return comparable results. But since CART returned slightly better results than the Random Forest, we chose it for training the final inclusion model in single-person adaptation. Similar results are observed in two-people adaptation, we chose CART as a final machine-learning algorithm since it was also selected in all previous decision models.

The tree analysis of the inclusion models per target user (single-model adaptation) shows that the most important feature is the prediction confidence in the label as predicted by the person-specific model ($C_P(label_P)$). If the confidence is too low, then the example is not included in the $Dataset_{general}$; if it surpasses a threshold, then the tree examines whether the general model’s label is basic ($Basic(label_G)$). If it is *Basic*, then the example is not included in the $Dataset_{general}$. If it is not *Basic*, the label is evaluated for $C_G(label_G)$, and if it surpasses a threshold, it is selected for inclusion and not otherwise. In two-people adaptation, the tree analysis shows that the most important feature is the person-specific model’s prediction confidence in its own label ($C_P(label_P)$). If the confidence is high enough, then the only reason for not including the example in the $Dataset_{general}$ occurs if the selection model decided that the general model should give the final label of the example (= General). If the probability is lower than a threshold, the example is included in case the selection model selects the general model for the final label and the general model’s confidence in the label predicted by the person-specific model is low ($C_G(label_P)$).

4.4.3 Adaptation Procedure

The trained models as presented in Section 4.4.2.2 are now used in the MCAT procedure for adaptation and personalisation. We evaluate the MCAT for adaptation of the general model to a single person and to two people.

The adaptation is evaluated for each p_{target} in single-person adaptation task and for each pair $\{p_{\text{target}_i}, p_{\text{target}_j}\}$ in two-people adaptation. We analyse the adaptation procedure in terms of inclusion rate, the number of instances included in the $Dataset_{\text{general}}$ and in terms of the retraining accuracy.

For each target person p_{target} or a pair of target people $\{p_{\text{target}_i}, p_{\text{target}_j}\}$ we train an initial general model, person-specific model, selection and inclusion model as introduced in Section 4.4.2.2. The input to the MCAT is the $Dataset_{\text{unlabelled}}$ (Figure 4.9) of current target people. We experiment in two modes to compare the speed and the success of the adaptation. The modes are:

- Batch adaptation.
- Incremental adaptation.

In batch mode, the experiment the MCAT evaluates all examples in $Dataset_{\text{unlabelled}}$ before retraining. The examples selected for inclusion are removed from the $Dataset_{\text{unlabelled}}$, while the examples that are not selected remain in the $Dataset_{\text{unlabelled}}$ to be used in the next iteration. In the incremental mode, the retraining is performed every time a new example from $Dataset_{\text{unlabelled}}$ is selected for inclusion. The adaptation-stopping criterion in the batch mode is set as a definite number of iterations; in an incremental mode, it can be set as a percentage of examples labelled by the general model compared to a percentage of examples labelled by the specific model or as a percentage between included and discarded examples in one iteration. We set the stopping criterion to 85% of discarded examples in one iteration. The percentage was chosen experimentally. A lower value can make the adaptation stop too quickly, whereas a higher value can make the adaptation loop indefinitely.

4.4.3.1 Single-Person Adaptation

We start with training the initial general model to be used in both modes. The initial accuracy of the general model is, therefore, the same in both modes and is presented in Table 4.15 for each person (row Initial). Next, we evaluate the impact on accuracy when the selection model is used without adaptation (row Selection mode). The results show an increase compared to the initial accuracy of the general model, which is an important starting point for successful adaptation and personalisation of the general model. The rest of the results are divided into groups per mode in which the adaptation is performed. The first group of results are from the batch mode adaptation (rows B) and the second group from the incremental mode (rows I).

The adaptation in the batch mode is performed in three iterations. In the first iteration, the method processed the entire $Dataset_{\text{unlabelled}}$ for the current target person and selected the examples which are included in the $Dataset_{\text{general}}$; the discarded examples are input into the second iteration. The decisions of the selection model per iteration are presented in Figure 4.13. The Figure shows how many times the selection model chose each domain model for final prediction of the label for the included examples. Additionally, we show the inclusion rate and the general model labelling rate per iteration in Figure 4.14. The inclusion rate shows the percent of included examples per iteration from the entire unlabelled set in the 1st iteration and from the remaining set of unlabelled examples in the 2nd and 3rd iteration. The general labelling rate presents the percentage of included



Figure 4.13: Selection model decisions per person and iteration in single-person adaptation in batch mode.

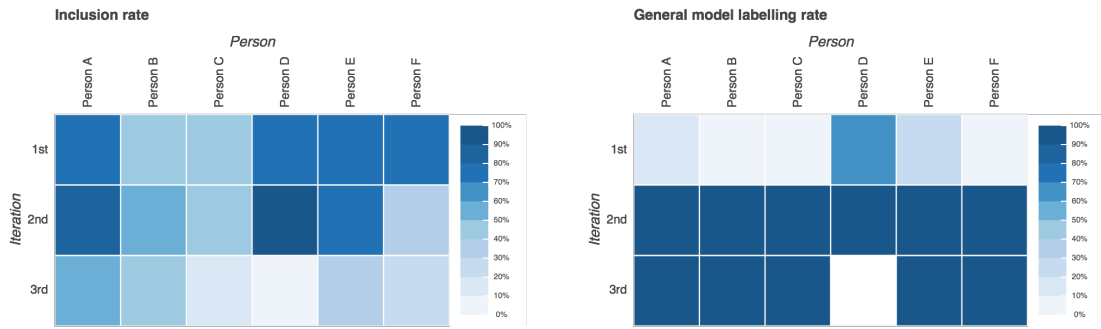


Figure 4.14: Inclusion and general model labelling rate per iteration for single-model adaptation in the batch mode.

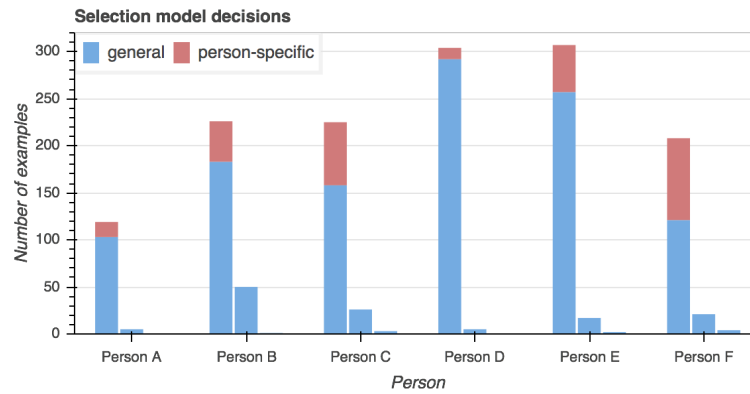


Figure 4.15: Selection model decisions in single-person adaptation per person and iteration in incremental mode.

examples that were labelled by the general model. The accuracy results per iteration are presented in row B in Table 4.15.

The first iteration includes 65% of the examples overall in the $Dataset_{\text{general}}$, which is presented in the first rows in heat maps in Figure 4.14. Figure 4.13 and the second heat map in Figure 4.14 shows that the person-specific model is selected for labelling the included

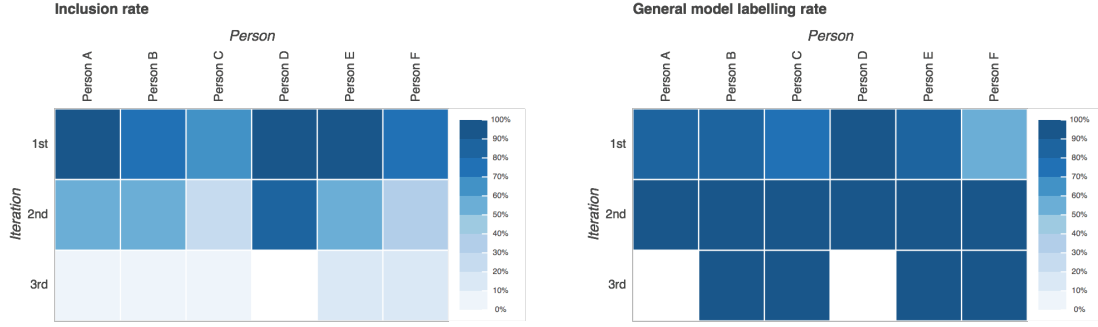


Figure 4.16: Inclusion and general model labelling rate per iteration for single-person adaptation in the incremental mode.

examples more often than the general model (20% of included examples are labelled by the general model). The evaluation results after the first iteration in batch mode (Table 4.15) show that the accuracy of the retrained general model surpasses the accuracy of using only the selection model without adaptation for all target people except for Person F. The heat map shows that the inclusion rate for Persons B and C is lower than for the rest of the people. The analysis of these two events shows that the reason is in the lower accuracy of the selection model (Selection model row in Table 4.15) and good judgement of the inclusion model. The general model labelling rate heat map shows that, in the case of Person D, the general model is selected for labelling the included example more often than the person-specific model, which can be explained by the accuracy achieved by the initial general model (Initial general model row in Table 4.15) due to this person having a significant difference in the way they walked due to a hip disease.

In the second iteration (second bars per person in Figure 4.13, the second row of heat maps in Figure 4.14 and row B 2nd iteration in Table 4.15), as well as the number of included examples, remain high (65%), but the examples that are included are labelled only by the general model, which indicates that the retrained general model is already able to perform the domain task on its own. The accuracy increases for each target person. The examples that are not included in the second iteration are used as the input into the third iteration (third bars per person in Figure 4.13, the third row of heat maps in Figure 4.14 and row B 3rd iteration in Table 4.15). In the third iteration, only 30% of the remaining examples are included in the $Dataset_{general}$ and the general model labels all the examples. The accuracy of the general model slightly increases, achieving the mean accuracy of 86.6%, which is two percentage points lower compared to the achievable accuracy when all data are correctly labelled and used for retraining (row Labelled Table 4.15).

The adaptation in the incremental mode retrains the general model every time an example is included in the $Dataset_{general}$, and the retrained general model is immediately used as a domain model for the task. The stopping criterion in the incremental mode is set according to the ratio between the included and discarded examples in one iteration. The adaptation is stopped once the number of discarded examples represents more than 85% of all examples in the iteration. The adaptation for all people meets this criterion after the third iteration. The accuracy results per iteration are presented in row I in Table 4.15. The decisions of the selection model per iteration are presented in Figure 4.15 in terms of which domain model labelled the included example. Heat maps in Figure 4.16 show the inclusion and general model labelling rate per iteration.

The input to the first iteration is $Dataset_{unlabelled}$ for the target person, while the discarded examples are used in the next iteration. Compared to the batch mode, we can

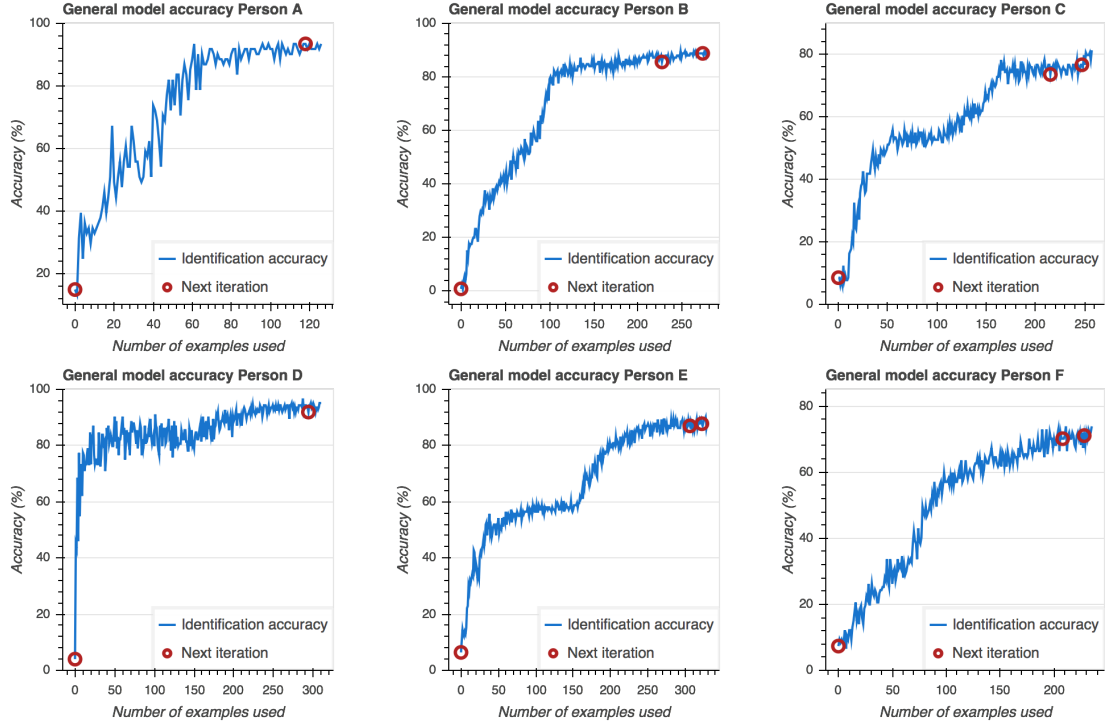


Figure 4.17: The accuracy of the general model in incremental mode. The accuracy was evaluated after each example was included into the $Dataset_{\text{general}}$ and the general model retrained.

Table 4.15: Results of the initial general model, selection model, and single-person adaptation in batch and incremental mode per iteration. The last row presents the achievable accuracy if all examples are labelled correctly and included into the $Dataset_{\text{general}}$.

	Person						Mean	Std. dev
	A	B	C	D	E	F		
Initial general model	18.0	0.8	8.5	34.0	6.9	7.5	12.6	11.8
Person-specific model	32.2	32.8	13.2	14.5	41.8	38.3	28.8	12.1
Selection model	70.5	53.6	48.8	68.5	70.2	77.6	64.9	11.1
B 1 st iteration	80.3	76.0	60.5	92.7	81.7	66.4	76.3	11.5
B 2 nd iteration	91.8	88.8	75.2	95.2	87.0	72.0	85.0	9.3
B 3 rd iteration	93.4	90.0	79.1	-	89.3	72.9	86.6	8.8
I 1 st iteration	91.8	85.6	73.6	92.7	87.0	70.1	83.5	8.6
I 2 nd iteration	93.4	88.8	76.7	95.2	87.8	71.0	85.5	8.7
I 3 rd iteration	-	89.6	81.4	-	89.3	73.8	87.1	7.4
Labelled (Figure 4.10)	91.8	96.0	85.2	95.1	90.1	73.8	88.7	8.3

observe that the number of included examples in the first iteration increases to 83% and that the general model is more often selected for the final labelling (79% of examples), which indicates faster adaptation. The mean accuracy of the general model increases up to 83.5% after the first iteration. In the second iteration, 53% of remaining examples are selected for inclusion, and the included examples are labelled by the general model only. The third iteration is final since the inclusion rate discards more than 85% of the remaining examples for all people (see third rows of heat maps in Figure 4.16). Interestingly, even

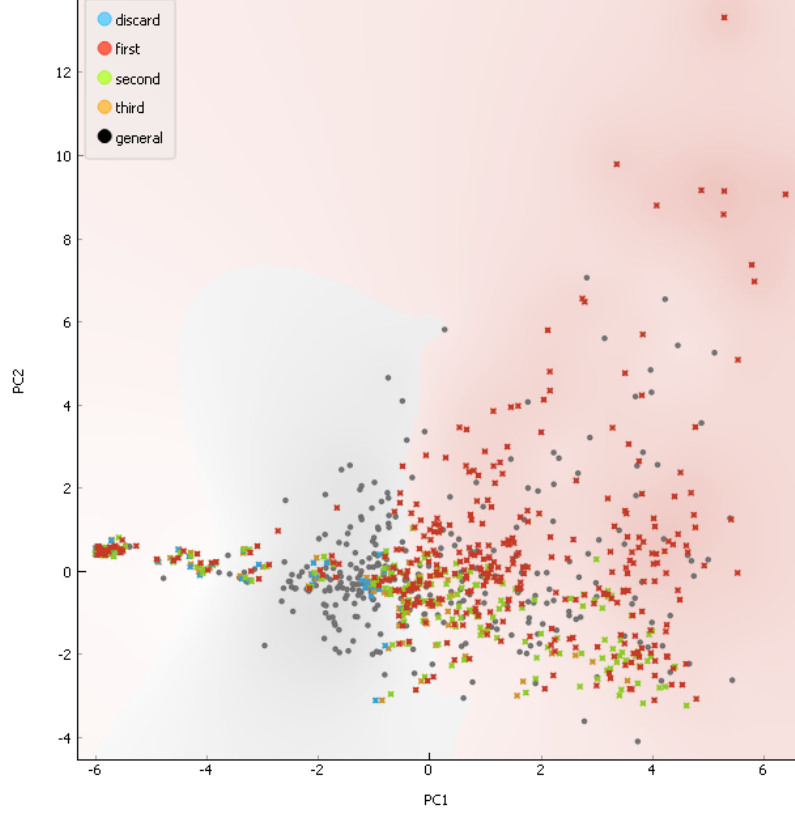


Figure 4.18: The first two components of the Principal Component Analysis for included and discarded examples.

though some examples are not included in the $Dataset_{general}$, the final accuracy for Persons E and F is comparable to the accuracy when all data are correctly labelled and included in the $Dataset_{general}$. In case of Person A, we can see that the achieved accuracy surpasses this. The reason may be that the MCAT discards the noisy examples that do not contribute to the accuracy. We also present the change in accuracy over the duration of the entire procedure in the incremental mode for each target person in Figure 4.17. The charts present the evaluated accuracy of the general model; every time a new example is added into the $Dataset_{general}$, the circles present the beginning of the iteration.

The analysis of the included examples per iteration, as well as discarded examples, are presented in Figure 4.18. Examples from the general dataset are presented with grey marks, while those included in the first iteration have red marks, those in the second iteration have green marks, those in the third iteration have orange marks, and discarded examples have blue marks. We can see that, in the first iteration, the method includes examples from the entire decision space, including those that are not covered with the general model and were correctly classified by the person-specific model. In the second and third iterations, the method focuses on including the examples that overlap with those from the general model. This subset is perceived as examples that are hard to label. The general model needs to be adapted to some extent (first iteration), to get a positive decision from the inclusion model.

We also evaluated the statistical significance of each step against the procedure for both modes. The evaluation was performed with one-way ANOVA method (Montgomery & Runger, 2003), and the results are presented in Table 4.16. The statistically significant pairs are marked with ✓ (the p -value is lower than 0.05), and statistically insignificant pairs

are marked with $\times$ (the p -value is higher than 0.05). We can observe that the accuracies of all steps are statistically significantly higher compared to the accuracy of the initial general model. The increase in accuracy compared to the selection model achieves statistical significance in the second iteration in the batch mode and first iteration in the incremental mode. The increase in accuracy between iteration steps is not statistically significant.

Table 4.16: The evaluation of the statistical significance between the initial general model and the accuracies achieved when using the selection model without adaptation or after each iteration of the adaptation with MCAT. B – batch mode, I – incremental mode.

		Initial G	S model	1 st iteration	2 nd iteration	3 rd iteration
B	Initial G	-	-	-	-	-
	S model	✓	-	-	-	-
	1 st iteration	✓	$\times$	-	-	-
	2 nd iteration	✓	✓	$\times$	-	-
	3 rd iteration	✓	✓	$\times$	$\times$	-
I	Initial G	-	-	-	-	-
	S model	✓	-	-	-	-
	1 st iteration	✓	✓	-	-	-
	2 nd iteration	✓	✓	$\times$	-	-
	3 rd iteration	✓	✓	$\times$	$\times$	-

4.4.3.2 Two-People Adaptation

The methodology is the same as in single-person adaptation. We start with training all models as presented in Section 4.4.2.2 for two-people adaptation and then we proceed with the adaptation in batch and incremental mode. The accuracy of the initial general model is the same in both modes and is presented in column G in Table 4.17. Next, we evaluate the contribution of the selection model to the accuracy (column S in Table 4.17). We can see that the selection model already improves the labelling accuracy compared to the initial general model, which is fundamental for the successful adaptation and personalisation of the general model. As in single-person adaptation, we also evaluated the increase in accuracy if all examples from the $Dataset_{unlabelled}$ for the pair of target people were correctly labelled and included into the $Dataset_{general}$ used for retraining. The results of this evaluation are presented in the Labelled column in Table 4.17. The rest of the results are divided into groups per mode: the results in the Batch columns are from the batch mode adaptation and those in the Incremental columns are from the adaptation in the incremental mode.

The adaptation in the batch modes is performed in three iterations. In the first iteration, the method processed the entire $Dataset_{unlabelled}$ for the current pair of target persons and chose the examples that are included in the $Dataset_{general}$, while the discarded examples are input into the second iteration. The results of the selection model decisions on the final labelling per iteration are presented in Figure 4.19. The inclusion rate and general model labelling rate per iteration are presented in heat maps in Figure 4.20. The accuracy results are presented in column Batch 1st in Table 4.17. The results after the first iteration show that the adaptation improves the overall accuracy compared to the selection model results for five percentage points. From Figure 4.19 and Figure 4.20, we see that 87% of examples overall are selected for inclusion in the first iteration, and that the majority of these examples are labelled with the person-specific model. The discarded examples are fed into the second iteration. The results of the second iteration show an increase of the

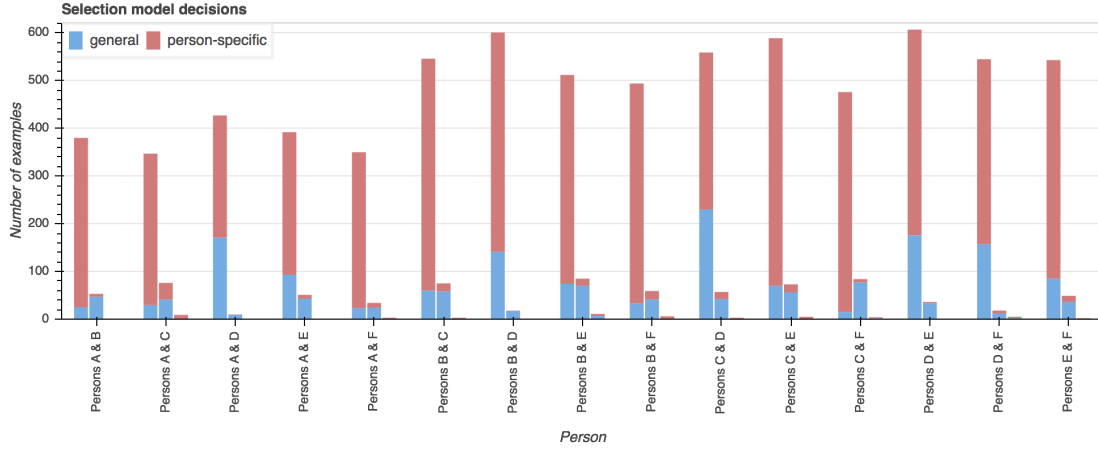


Figure 4.19: Selection model decisions per iteration and per people pair in two-people adaptation in batch mode.

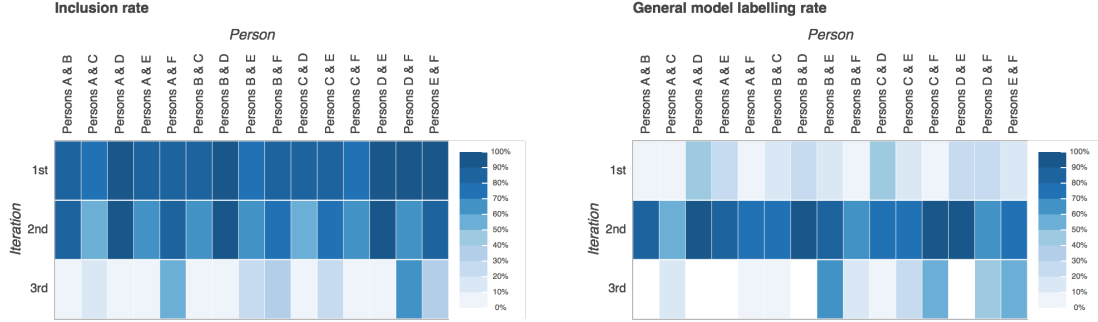


Figure 4.20: The inclusion and general model learning rate for two-people adaptation in the batch mode.

overall mean accuracy for two percentage points compared to the first iteration, but the accuracy for two pairs (AB and EF) experienced a minor decrease (less than one percentage point). The percentage of included examples remains high (76%), but the decision of the selection model shows that it mostly favoured the general model for labelling the included examples. The discarded examples are processed in the third iteration in which only 18% of the remaining examples are included, and most are labelled by the person-specific model, which indicates that only hard examples are left in the $Dataset_{unlabelled}$. We achieve a minor increase in the overall mean accuracy, but also a minor decrease in accuracy of seven target people pairs (column Batch 3rd) compared to the second iteration and of two target people pairs compared to the first iteration (AD and BE). Compared to the Labelled column, we achieve higher or the same accuracy for six target people pairs. These minor decreases and overall increase indicate that we have exploited the given $Dataset_{unlabelled}$ to the point where the accuracy starts converging and that only hard examples are left in the $Dataset_{unlabelled}$. A new set of unlabelled data is needed for further improvement.

The adaptation in the incremental mode starts with the same general and selection model as the batch mode adaptation. In the first iteration, the method is fed with all examples from $Dataset_{unlabelled}$ for the target people pair. Every time the example is selected for inclusion in the $Dataset_{general}$, the general model is retrained. The discarded examples from one iteration serve as an input into the next iteration until the stopping

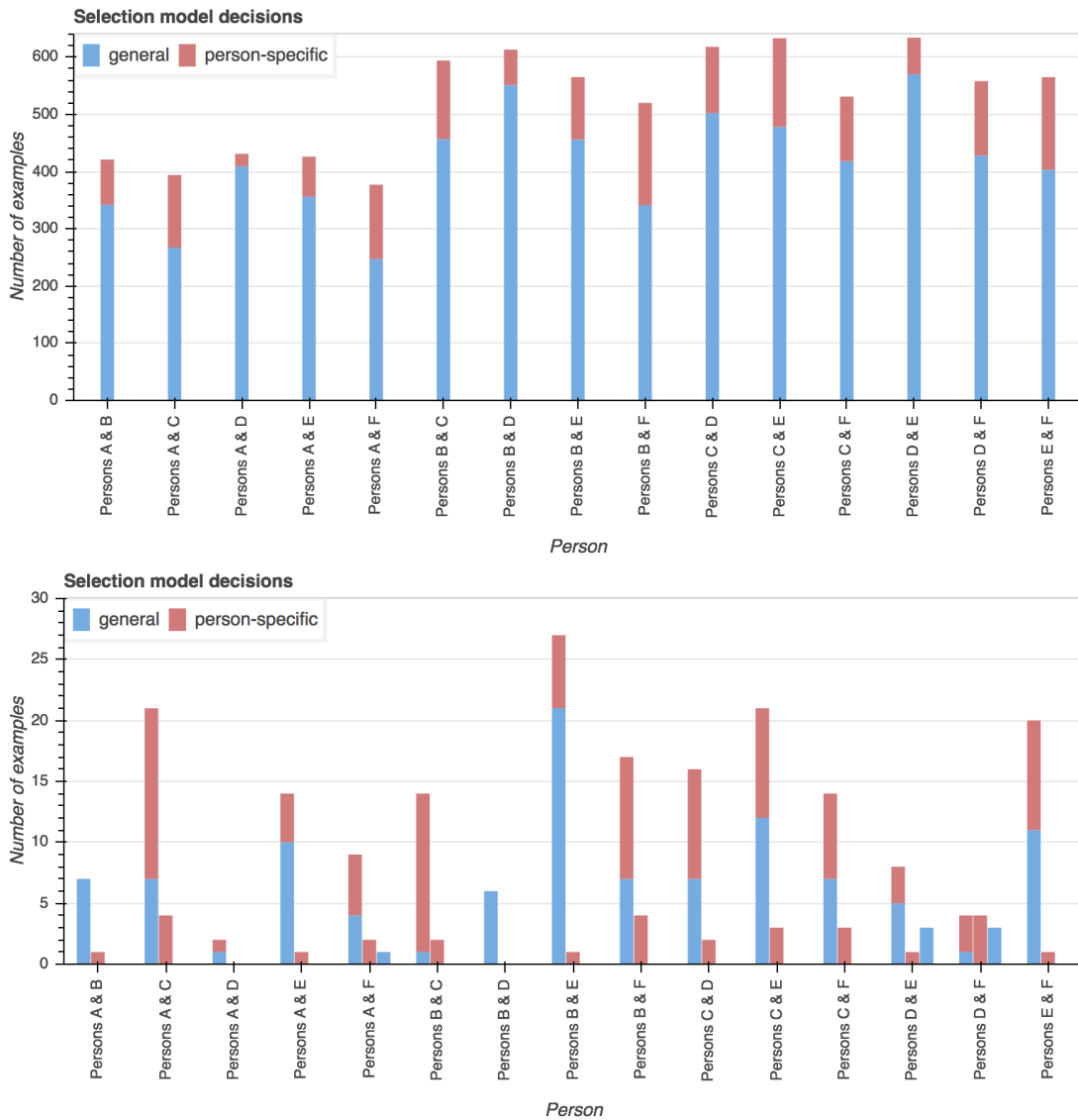


Figure 4.21: The selection model decisions in the incremental mode. First iteration upper chart and the rest of the iterations as applicable in the lower chart.

criterion is met. The criterion is the same as in single-person adaptation; when 85% of examples are discarded in one iteration the adaptation for the target people pair stops. The classification accuracy results are presented in the Incremental column in Table 4.17 and the decisions of the selection model in charts in Figure 4.21 and the inclusion rates and the general model learning rates per iteration in heat maps in Figure 4.22. The upper chart in Figure 4.21 presents the selection model decision for the first iteration and the lower chart for the second, third, and fourth where applicable.

First iteration results in a 5.8 percentage point increase in overall mean accuracy compared to the selection model. From the first chart in Figure 4.21 and the first rows in heat maps in Figure 4.22, we can see that the majority of examples (93%) are included in the first iteration and that most (77%) are labelled by the general model, which indicates faster adaptation compared to the batch mode. In the second iteration, the overall mean accuracy slightly increases, and the number of included examples falls compared to the first iteration (42% of examples are included). The decision of the selection model is

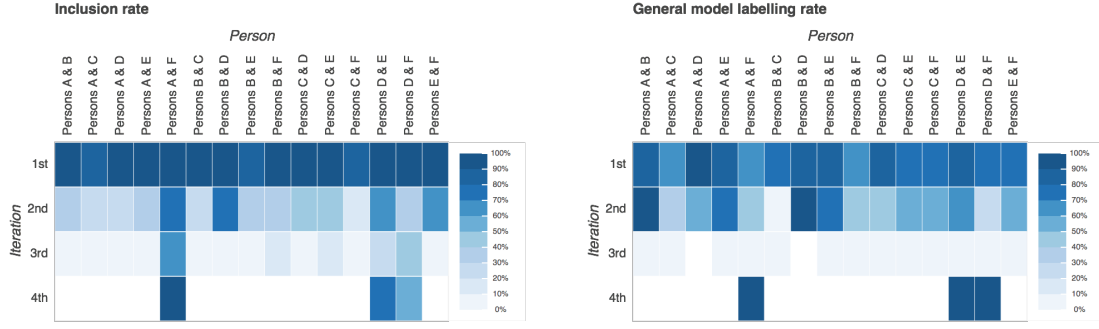


Figure 4.22: The inclusion and general model learning rate for two-person adaptation in the incremental mode.

Table 4.17: Results of the two-person adaptation per mode and per iteration. Labelled – when all examples are labelled correctly and included for retraining, G – initial general model, P – person-specific model, S – selection model, Batch – batch mode, Incremental – incremental mode.

Person	Labelled	Models			Batch			Incremental			
		G	P	S	1 st	2 nd	3 rd	1 st	2 nd	3 rd	4 th
AB	95.7	12.9	56.3	85.0	95.7	95.2	95.7	95.7	95.7	96.7	-
AC	88.9	10.0	45.8	74.1	82.6	85.8	84.7	82.1	84.2	85.2	-
AD	94.1	34.1	43.8	89.3	95.1	95.1	93.5	94.1	94.1	95.1	-
AE	89.6	12.0	50.5	81.6	89.6	89.6	89.6	90.1	90.6	90.1	-
AF	81.0	12.5	52.6	85.5	79.2	82.1	82.7	80.4	81.0	80.8	81.0
BC	92.1	5.9	41.7	79.8	86.6	89.8	89.0	88.6	89.0	89.6	-
BD	95.2	24.1	52.7	89.6	95.6	97.2	96.0	95.6	95.6	96.6	-
BE	94.9	5.1	46.8	77.9	88.7	89.5	88.3	88.7	88.3	89.3	-
BF	85.3	11.6	60.3	76.5	80.2	81.0	82.3	81.0	81.5	82.0	-
CD	91.7	17.4	51.7	88.8	88.9	91.3	90.9	90.9	92.1	91.5	-
CE	87.7	10.8	66.9	77.4	78.1	81.9	82.3	82.7	82.3	82.5	-
CF	86.4	9.3	55.3	68.3	75.4	76.3	78.0	77.1	77.5	77.3	-
DE	93.3	20.4	58.4	84.2	92.5	94.5	94.1	93.3	94.5	93.9	94.5
DF	86.1	26.0	52.7	85.5	84.0	84.0	86.6	83.5	83.5	85.4	85.8
EF	84.9	13.9	61.3	72.3	76.9	76.1	79.8	78.6	79.0	80.0	-
Mean	89.8	15.1	53.1	81.0	85.9	87.3	87.6	86.8	87.3	87.7	87.8

evenly distributed between the general and person-specific model, the general model labels 54% of included examples. The discarded examples are processed in the third iteration in which the overall mean accuracy slightly increases. From the charts in Figure 4.21 and Figure 4.22, we see that only 13% of examples overall are selected for inclusion and that the person-specific model labels the majority of included examples. In the third iteration, the 12 target people pairs met the stopping criterion (the inclusion rate is below 15%); for the three people, we ran the fourth iteration with the few remaining examples. We can observe that the accuracy for the three people increased, which contributed to the overall mean accuracy. The included examples are labelled only by the general model.

The evaluation of statistical significance is presented in Table 4.18 of each step of the procedure. The results show the same results as in single-person adaptation, and the accuracy of each adaptation iteration is statistically significant compared to the accuracy

Table 4.18: The evaluation of the two-people adaptation statistical significance between initial general model and the accuracies achieved when using the selection model without adaptation or after each iteration of the adaptation with MCAT. B – batch mode, I – incremental mode.

		Initial G	S model	1 st iter.	2 nd iter.	3 rd iteration	4 th iter.
B	Initial G	-	-	-	-	-	-
	S model	✓	-	-	-	-	-
	1 st iter.	✓	✗	-	-	-	-
	2 nd iter.	✓	✓	✗	-	-	-
	3 rd iter.	✓	✓	✗	✗	-	-
I	Initial G	-	-	-	-	-	-
	S model	✓	-	-	-	-	-
	1 st iter.	✓	✓	-	-	-	-
	2 nd iter.	✓	✓	✗	-	-	-
	3 rd iter.	✓	✓	✗	✗	-	-
	4 th iter.	✓	✓	✗	✗	✗	-

of the initial general model. The increase in accuracy compared to the selection model achieves statistical significance in the second iteration in the batch mode and in the first iteration in the incremental mode. The increase in accuracy between iteration steps is not statistically significant.

4.4.4 Summary

Section 4.4 introduces and evaluates the MCAT method for the adaptation of the general model trained to identify people from walking. In the identification task we have a case in which we introduce new label to the model and aim at adapting the model to improve identification accuracy for the new person. The MCAT models were trained and evaluated in a single-person adaptation task in which one new label is introduced, and in a two-people adaptation task in which two new labels are introduced. In addition to the new label, the model is presented with data of different types of walking per person, which may be considered a concept drift. For each task, we described the training procedure and the adaptation procedure with evaluation in batch and incremental mode.

We first evaluate the accuracy if all data are correctly labelled and included into the *Dataset_{general}* used for retraining the general model. This result represents the goal of the adaptation. The results of the single-person adaptation are presented in Table 4.15 and for two-people adaptation in Table 4.17. We observed that both modes' final accuracies are comparable, yet the incremental adaptation outperforms the batch adaptation in terms of the adaptation speed of the general model. This was observed in the analysis of the selection and inclusion model decisions per mode. The decisions for single-person adaptation are presented in Figure 4.13 for the batch mode, and in Figure 4.15 for the incremental mode. The decisions for two-people adaptation are presented in Figure 4.19 for the batch mode, and in Figure 4.21 for the incremental mode. In both modes, we achieve comparable results to those achieved if all data were correctly labelled and included in the *Dataset_{general}* for retraining.

Chapter 5

Conclusions and Further Work

In this dissertation, we proposed a novel SSL method called Multi-Model Adaptive Training for personalisation of machine-learning models using data from wearable sensors. The method is designed in such way that it can be used for personalisation in real time, exploiting the unlabelled data generated by wearable devices while in use.

In this chapter, we summarise the hypotheses of the thesis along with the scientific contributions, discuss the strengths and limitations of the proposed method, and conclude with ideas for further work.

5.1 Scientific Contributions

This dissertation has three scientific contributions, each summarised along with the hypotheses in an individual subsection.

5.1.1 Design and Implementation of a Novel SSL Method Called Multi-Model Adaptive Training

In Section 1.4 we first hypothesised that SSL can efficiently exploit unlabelled data generated by wearable sensors for personalisation of machine-learning models. To analyse this hypothesis we designed and implement a novel SSL method called Multi-Model Adaptive Training. The method is designed to receive unlabelled sensor data in real time, automatically label the data and decide whether the automatically labelled data should be used for personalisation of the domain machine-learning models. This scientific contribution was achieved by reaching Goal 1 of the dissertation set in Section 1.3.

The content of this scientific contribution is presented in Chapter 3 and was published in the paper included in Section 4.1 in which we established the architecture of the method, proposed the design of the machine-learning models after evaluating various datasets and feature sets, and proposed the training-procedure algorithm.

5.1.2 Analysis of the Multi-Model Adaptive Training

This scientific contribution applies to the second and the third hypothesis of the dissertation introduced in Section 1.4. This scientific contribution was achieved by reaching the Goal 2 of the dissertation set in Section 1.3.

The second hypothesis of the dissertation was that employing an additional model to aggregate the outputs of the introduced domain models for the selection of the final label improves the performance, as opposed to the use of a simple heuristic. First, we explored several approaches to efficiently combine the two datasets (the general dataset and person-specific dataset) to train single or multiple domain models for final label prediction. We

showed that the best results are obtained when two domain models are trained, one per dataset, and predictions of both models combined. Second, we analysed and evaluated methods for combining the prediction of the two domain models and showed that stacking an ensemble of the domain models with the dedicated selection model yields superior performance in both classification and regression tasks compared to combining the domain model using heuristic.

The third hypothesis of the dissertation was that employing an additional model for the decision on the inclusion of self-labelled examples enables inclusion of more diverse examples into the dataset used for retraining the general domain model. The most common inclusion approach in Self-training methods is the inclusion of those examples that were labelled with high prediction confidence. These examples already overlap with the examples contained in the general model, so they may not contribute to the performance of the model. We analysed the Multi-Model Adaptive Training inclusion procedure per iteration in terms of principal component analysis (PCA). We showed that the first iteration includes examples from the entire decision space, which can be interpreted as the inclusion of diverse examples. The second and the third iteration includes examples which have an overlap with the examples from the general model, which can be interpreted as the inclusion of non-diverse examples since the model has already adapted.

The content of this scientific contribution is presented in published papers and in the Section 4. More specifically, the analysis of the domain models, selection model and the decision model, as well as the comparison of the selection model and decision model against heuristics was published in papers included in Section 4.1 and Section 4.2. The analysis of the outputs of the selection and decision model in successive iterations of adaptation and the PCA analysis is presented in Section 4.3 and Section 4.4.

5.1.3 Demonstration and Evaluation of the Applicability

This scientific contribution applies to the first and the third hypothesis of the dissertation introduced in Section 1.4. This scientific contribution was achieved by reaching the Goals 3 and 4 of the dissertation set in Section 1.3.

The first hypothesis of the dissertation was that SSL can efficiently exploit unlabelled data generated by wearable sensors for personalisation of machine-learning models. We empirically evaluated the Multi-Model Adaptive Training on four tasks that needed personalisation. The Multi-Classifer Adaptive Training (MCAT – Multi-Model Adaptive Training for personalisation of classification models) was evaluated on three classification tasks: the personalisation of the low-level activity recognition model, the personalisation of the context-recognition model and the personalisation of the identification model. We developed the best performing general classification model trained in a supervised manner for the target task and used the MCAT method for the personalisation. We showed that MCAT successfully personalised the domain model in all three tasks. The Multi-Regressor Adaptive Training (MRAT – Multi-Model Adaptive Training for personalisation of regression models) was evaluated on the human energy expenditure estimation task. We developed the best performing general regression model trained in a supervised manner for the task and used the MRAT method for the personalisation. We showed that MRAT successfully decreased the overall mean absolute error of the estimation model.

The third hypothesis was that employing an additional model for the decision on the inclusion of self-labelled examples enables inclusion of more diverse examples which enables efficient adaptation and personalisation in comparison to the heuristic approach that includes only the examples with high labelling confidence. The diversity of the inclusion was covered as a part of Contribution 2 (Section 5.1.2). For evaluation of the efficiency of the adaptation using the included examples, we compared the MCAT method against

the following heuristic based SSL methods: the Self-training, a version of Democratic Co-learning, and threshold-based MCAT in the low-level activity recognition task. We showed that the MCAT outperformed the comparing methods from 4 to 10 percentage points. Additionally, we compared the MCAT against the common personalisation approaches in the context recognition domain and showed that MCAT achieves superior personalisation results. In case of MRAT we evaluated how the criterion value affects the improvement of the model. The results confirm that inclusion of only highly confident predictions does not contribute to the improvement while selecting those with appropriately less confident predictions contributed to the improvement the most.

The content of this scientific contribution is presented in published papers and in the Section 4. More specifically, the low-level classification task is published in the paper included in Section 4.1, the context-recognition task is published in the paper included in Section 4.2 and the human energy expenditure estimation task is published in the paper included in 4.3. In Section 4.4 we present the evaluation and the analysis of the identification task and in Section 4.3 the extension of the human energy expenditure task with the analysis of the inclusion criterion.

5.2 Discussion

5.2.1 Critical Aspects of The Method

Successful evaluation of the Multi-Model Adaptive Training on the presented domains confirms the method design for successful personalisation of the models used in person-centric tasks. Despite the good personalisation results, we can need to emphasise several aspects that may need revisiting for the method to be ready for real-time applicability and list the conditions under which the method operates as limitations.

The critical aspects of the method are:

- The computational complexity of retraining the models is a critical aspect of the method. The training time depends on the chosen machine-learning algorithm as well as the number of examples and the number of features in the dataset. We showed that incremental mode is faster in personalising the model, which means that every time a new example is included in the dataset, the retraining is performed. With each included example the size of the dataset increases and so does the retraining time which makes the application using the model slow and after a while potentially unusable. To overcome this problem, we can employ a machine-learning algorithm that can retrain incrementally to accelerate the retraining or introduce an amending mechanism that can detect and remove outdated examples and control the size of the dataset. We did not experiment with either of the solutions at this time but have left them as ideas for the future work.
- We evaluated the method on an unbalanced dataset in the papers included in Section 4.1 and Section 4.2 and showed that the method can also adapt to the classes that occur less often. What we see as a critical aspect of a current design of the method is personalisation using data in which certain labels (classes) do not occur for a longer period. We assume that this would reflect in overfitting the model to other classes and would decrease the probability of such class being picked by the model if it occurs. This can be solved by including an amending mechanism which will evaluate the structure of the dataset and remove or amend the contained examples accordingly.

- Evaluation of the personalised model in a real-life environment is challenging since no labelled data can be dedicated for this purpose. In the dissertation, we dedicated a portion of the labelled dataset only for assessment of the personalisation procedure. At this point, one can use the person-specific dataset, but the evaluation will show the achieved accuracy for the labels included in the person-specific model and not for all possible labels per task.
- We proved that retraining the model using examples labelled with the highest label confidence might not contribute to personalisation of the model as much as if examples were chosen more liberally. The inclusion of examples with lower confidence were the ones that personalised the model better, which was achieved by using the selection and inclusion model instead of heuristic. However, in the current design of the method, we use the measure of accuracy in classification and error in the regression for inclusion and not the potential novel information carried by the example. This may have discarded too many examples as seen in the regression task and needs further research.

5.2.2 Conditions Under Which the Method Operates

The analysis of the method was also focused on finding the conditions under which the method operates and can be used to personalise the classification or regression model successfully. The method is applicable in domains that use sensor data and need personalisation. In addition to the domains evaluated in this dissertation, we are sure that it can be applied to other person-centric tasks that use sensor data streams, such as blood pressure estimation or sleep analysis, and in tasks that use images, such as person identification or handwriting recognition. The method needs to translate into similar problem form as presented in the thesis. The conditions that need to be fulfilled to use the method for personalisation are the following:

- The general dataset needs to be dividable per person and per some semantic value to fulfil the requirements for the training procedure.
- The general dataset needs to be labelled correctly and should not have noisy data.
- The evaluation of the initial general model trained in a supervised manner using the general dataset needs to yield acceptable results with the stratified cross-validation approach in case of an introduction of a new label and with the LOPO approach in case of an adaptation to the changed statistical parameters of a label.
- At least one of the domain models needs to perform with moderate accuracy.
- The domain models and the selection model must be trained with a machine-learning algorithm able to report probability or any other measure that can be interpreted as prediction confidence.

5.3 Further Work

The proposed method was evaluated as effective for personalisation of machine-learning models utilising data from wearable sensors. Further work is defined along five lines:

- Research in employing an ensemble of several domain models. The framework presented in Chapter 3 permits more domain models, the number of which could be defined by the number of available labelled general datasets. The current design will

have to be evaluated and adjusted accordingly. In case multiple selection models are required, the training procedure will need to be upgraded to work with additional levels of stacking.

- Improve the selection procedure by engineering additional features that will present measures to evaluate the reliability of the predicted label by each domain model. Current selection and inclusion feature vector contains features which are mostly domain independent, except if we use context features. We would need to better exploit the feature vector used by the domain models and create features that represent more complex correlation to the class.
- Research in finding a better measure for identifying good examples for inclusion. It would be beneficial for the personalisation if a quality measure or a feature is developed that would improve the decision on the suitability of a newly labelled example to be included in the general model training set. Improve the selection procedure by engineering additional features that will present measures to evaluate the reliability of the predicted label by each domain model. Current selection and inclusion feature vector contains features which are mostly domain independent, except if we use context features. To create features that represent the value of the class they belong to, more thorough exploitation of the feature vector information used in the domain models would be necessary.
- Research in a modification of the method for regression tasks. MRAT evaluation yielded acceptable results; however, in the scope of this dissertation we did not investigate different types of selection models (e.g., selection model solving a regression task instead of classification), and we did not evaluate different types of errors to be used for training the selection and inclusion model. The current architecture of MRAT does not include as many examples as MCAT does, so it would be beneficial to research in developing the selection and inclusion model that will increase the number of included examples.
- Development of an amending mechanism that removes outdated examples. Even though it is commonly accepted that more data means better models, it may not be so when it comes to models that are person-specific. For example, the physical characteristics, as well as the person's habits, change over time, and each time we adapt the model new examples describing the current statistical properties of the task are included in the dataset, while the examples that contain old statistical properties remain and can be outdated. Future development includes amending mechanism for evaluation and exclusion of the outdated examples.
- Development of an amending mechanism that evaluates the structure of the dataset used for retraining, and ensures that none of the classes prevail due to personalisation. It can happen that data of a certain class is not available. For example, the recognition of falling in activity recognition: people mostly do not fall very often, so MCAT can not pick up the 'activity' of falling and use it for personalisation, on the other hand, it can pick up other events, which will then prevail over it.
- Further evaluation of the proposed method is needed in a real-life setting on both classification and regression domains.
- Exploration of a possibility to use the proposed method structure as a deep learning architecture, for example using deep learning models instead of regular machine-learning algorithms for domain and decision models and to connect the entire structure into a single deep neural network.

References

- Ainsworth, B. E., Haskell, W. L., Herrmann, S. D., Meckes, N., Bassett, D. R., Tudor-Locke, C., . . . Leon, A. S. (2011). 2011 compendium of physical activities: A second update of codes and met values. *Medicine and Science in Sports and Exercise*, 43(8), 1575–1581.
- Altini, M., Casale, P., Penders, J. F., & Amft, O. (2015). Personalization of energy expenditure estimation in free living using topic models. *IEEE Journal of Biomedical and Health Informatics*, 19(5), 1577–1586.
- Belkin, M., Matveeva, I., & Niyogi, P. (2004). Regularization and semi-supervised learning on large graphs. In J. Shawe-Taylor & Y. Singer (Eds.), *Learning theory* (pp. 624–638). Berlin, Heidelberg: Springer Berlin Heidelberg.
- Belkin, M. & Niyogi, P. (2002). Using manifold structure for partially labelled classification. In *Proceedings of the 15th International Conference on Neural Information Processing Systems, NIPS'02* (pp. 953–960). Cambridge, MA, USA: MIT Press.
- Bennett, K. P. & Demiriz, A. (1999). Semi-supervised support vector machines. In *Proceedings of the 1998 Conference on Advances in Neural Information Processing Systems II* (pp. 368–374). Cambridge, MA, USA: MIT Press.
- Bleser, G., Steffen, D., Reiss, A., Weber, M., Hendeb, G., & Fradet, L. (2015). Personalized physical activity monitoring using wearable sensors. In A. Holzinger, C. Röcker, & M. Ziefle (Eds.), *Smart Health: Open Problems and Future Challenges* (pp. 99–124). Springer International Publishing.
- Blum, A. & Mitchell, T. (1998). Combining labeled and unlabeled data with co-training. In *Proceedings of the Eleventh Annual Conference on Computational Learning Theory, COLT' 98* (pp. 92–100). Madison, Wisconsin, USA: ACM.
- Breiman, L. (1996). Bagging predictors. *Machine Learning*, 24(2), 123–140.
- Breiman, L. (2001). Random forests. *Machine Learning*, 45(1), 5–32.
- Chapelle, O., Schölkopf, B., & Zien, A. (2006). *Semi-supervised learning*. Cambridge, MA, USA: The MIT Press.
- Chapelle, O. & Zien, A. (2005). Semi-supervised classification by low density separation. In *Proceedings of the tenth international workshop on artificial intelligence and statistics (aistats 2005)* (pp. 57–64). Max-Planck-Gesellschaft.
- Choi, S., Youn, I. H., LeMay, R., Burns, S., & Youn, J. H. (2014). Biometric gait recognition based on wireless acceleration sensor using k-nearest neighbor classification. In *2014 International Conference on Computing, Networking and Communications (ICNC)* (pp. 1091–1095).
- Cortes, C. & Vapnik, V. (1995). Support-vector networks. *Machine Learning*, 20(3), 273–297.
- Crouter, S. E., Churilla, J. R., & Bassett, D. R. (2006). Estimating energy expenditure using accelerometers. *European Journal of Applied Physiology*, 98(6), 601–612.
- Cvetković, B., Janko, V., & Luštrek, M. (2015). Demo abstract: Activity recognition and human energy expenditure estimation with a smartphone. In *13th IEEE International*

- Conference on Pervasive Computing and Communication, PerCom Workshops 2015* (pp. 193–195).
- Cvetković, B., Janko, V., Romero, A. E., Kafali, O., Stathis, K., & Luštrek, M. (2016). Activity recognition for diabetic patients using a smartphone. *Journal of Medical Systems*, 40(12), 256.
- Cvetković, B., Kaluža, B., Gams, M., & Luštrek, M. (2015). Adapting activity recognition to a person with Multi-Classifer Adaptive Training. *Journal of Ambient Intelligence and Smart Environments*, 7(2), 171–185.
- Cvetković, B., Kaluža, B., Luštrek, M., & Gams, M. (2012). Multi-Classifer Adaptive Training: Specialising an activity recognition classifier using semi-supervised learning. In F. Paternò, B. de Ruyter, P. Markopoulos, C. Santoro, E. van Loenen, & K. Luyten (Eds.), *Ambient intelligence* (pp. 193–207). Springer Berlin Heidelberg.
- Cvetković, B. & Luštrek, M. (2014). Analiza možnosti zaznavanja podobnosti med uporabniki. In *Intelligent systems : Proceedings of the 17th International Multiconference Information Society - IS 2014*. Ljubljana, Slovenia: Jožef Stefan Insitute.
- Cvetković, B., Luštrek, M., Kaluža, B., & Gams, M. (2011). Semi-supervised learning for adaptation of human activity recognition classifier to the user. In *IJCAI Workshop on Space, Time and Ambient Intelligence*.
- Cvetković, B., Milić, R., & Luštrek, M. (2016). Estimating energy expenditure with multiple models using different wearable sensors. *IEEE Journal of Biomedical and Health Informatics*, 20(4), 1081–1087.
- Cvetković, B., Mirchevska, V., Janko, V., & Luštrek, M. (2015). Recognition of high-level activities with a smartphone. In *Adjunct Proceedings of the 2015 ACM International Joint Conference on Pervasive and Ubiquitous Computing and Proceedings of the 2015 ACM International Symposium on Wearable Computers, UbiComp/ISWC'15 Adjunct* (pp. 1453–1461). Osaka, Japan: ACM.
- Cvetković, B., Szeklicki, R., Janko, V., Lutomski, P., & Luštrek, M. (2017). Real-time activity monitoring with a wristband and a smartphone. *Information Fusion*. In press. doi:10.1016/j.inffus.2017.05.004
- Delalleau, O., Bengio, Y., & Le Roux, N. (2005). Efficient non-parametric function induction in semi-supervised learning. In *Proceedings of the Tenth International Workshop on Artificial Intelligence and Statistics* (pp. 96–103). Society for Artificial Intelligence and Statistics.
- Demiriz, A. & Bennett, K. P. (2001). Optimization approaches to semi-supervised learning. In M. C. Ferris, O. L. Mangasarian, & J.-S. Pang (Eds.), *Complementarity: Applications, algorithms and extensions* (pp. 121–141). Boston, MA: Springer US.
- Dempster, A. P., Laird, N. M., & Rubin, D. B. (1977). Maximum likelihood from incomplete data via the em algorithm. *Journal of the Royal Statistical Society. Series B*, 39(1), 1–38.
- Fayyad, U., Piatetsky-Shapiro, G., & Smyth, P. (1996). From Data Mining to Knowledge Discovery in Databases. *AI Magazine*, 17(3), 37. doi:10.1609/aimag.v17i3.1230
- Gama, J. [João]. (2010). *Knowledge discovery from data streams* (1st). Chapman & Hall/CRC.
- Gama, J. [João], Žliobaitė, I., Bifet, A., Pechenizkiy, M., & Bouchachia, A. (2014). A survey on concept drift adaptation. *ACM Computing Surveys*, 46(4), 1–37.
- Grandvalet, Y. & Bengio, Y. (2006). Entropy regularization. In *Semi-Supervised Learning* (pp. 151–168). Cambridge, MA, USA: The MIT Press.
- Haque, A., Khan, L., Baron, M., Thuraishingham, B., & Aggarwal, C. (2016). Efficient handling of concept drift and concept evolution over Stream Data. In *2016 IEEE 32nd international conference on data engineering, icde 2016* (pp. 481–492). doi:10.1109/ICDE.2016.7498264

- Harris, J. W. & Stocker, H. (1999). Handbook of mathematics and computational science. *Computers & Mathematics with Applications*, 37, 133.
- Hosmer, D. W. (1973). A comparison of iterative maximum likelihood estimates of the parameters of a mixture of two normal distributions under three different types of sample. *Biometrics*, 29(4), 761–770.
- Hu, T., Fan, J., Wu, Q., & Zhou, D.-X. (2013). Learning theory approach to minimum error entropy criterion. *Journal of Machine Learning Research*, 14(1), 377–397.
- Huynh, T. & Schiele, B. (2006). Towards less supervision in activity recognition from wearable sensors. In *2006 10th IEEE International Symposium on Wearable Computers* (pp. 3–10).
- Jänicke, M., Sick, B., Lukowicz, P., & Bannach, D. (2014). Self-adapting multi-sensor systems: A concept for self-improvement and self-healing techniques. In *Proceedings - 2014 IEEE 8th International Conference on Self-Adaptive and Self-Organizing Systems Workshops, SASOW 2014* (pp. 128–136).
- Joachims, T. (2002). *Learning to classify text using support vector machines: Methods, theory and algorithms*. Norwell, MA, USA: Kluwer Academic Publishers.
- Johnston, A. H. & Weiss, G. M. (2015). Smartwatch-based biometric gait recognition. In *2015 IEEE 7th International Conference on Biometrics Theory, Applications and Systems, BTAS 2015* (pp. 1–6).
- Juefei-Xu, F., Bhagavatula, C., Jaech, A., Prasad, U., & Savvides, M. (2012). Gait-id on the move: Pace independent human identification using cell phone accelerometer dynamics. In *2012 IEEE 5th International Conference on Biometrics: Theory, Applications and Systems, BTAS 2012* (pp. 8–15).
- Kaluža, B., Cvetković, B., Dovgan, E., Gjoreski, H., Gams, M., Luštrek, M., & Mirchevska, V. (2014). A multi-agent care system to support independent living. *International Journal on Artificial Intelligence Tools*, 23(01), 1–30.
- Koren, Y. & Bell, R. (2015). Advances in collaborative filtering. In F. Ricci, L. Rokach, & B. Shapira (Eds.), *Recommender systems handbook* (pp. 77–118). Boston, MA: Springer US.
- Levatić, J., Ceci, M., Kocev, D., & Džeroski, S. (2015). Semi-supervised learning for multi-target regression. In A. Appice, M. Ceci, C. Loglisci, G. Manco, E. Masciari, & Z. W. Ras (Eds.), *New frontiers in mining complex patterns* (pp. 3–18). Cham: Springer International Publishing.
- Levine, J. A. (2005). Measurement of energy expenditure. *Public Health Nutrition*, 8(7a), 1123–1132.
- Li, M. & Zhou, Z. H. (2007). Improve computer-aided diagnosis with machine learning techniques using undiagnosed samples. *IEEE Transactions on Systems, Man, and Cybernetics Part A: Systems and Humans*, 37(6), 1088–1098.
- Lops, P., de Gemmis, M., & Semeraro, G. (2011). Content-based recommender systems: State of the art and trends. In F. Ricci, L. Rokach, B. Shapira, & P. B. Kantor (Eds.), *Recommender systems handbook* (pp. 73–105). Boston, MA: Springer US.
- Maekawa, T. & Watanabe, S. (2011). Unsupervised activity recognition with user’s physical characteristics data. In *Proceedings - International Symposium on Wearable Computers, ISWC* (pp. 89–96).
- Marin, J.-M., Mengersen, K. L., & Robert, C. (2005). Bayesian modelling and inference on mixtures of distributions. In D. Dey & C. Rao (Eds.), *Handbook of statistics: Volume 25*. Elsevier.
- Medrano, C., Plaza, I., Igual, R., Sánchez, Á., & Castro, M. (2016). The effect of personalization on smartphone-based fall detectors. *Sensors*, 16(1).

- Mills, M. T. & Bourbakis, N. G. (2014). Graph-based methods for natural language processing and understanding — a survey and analysis. *IEEE Transactions on Systems, Man, and Cybernetics: Systems*, 44(1), 59–71.
- Montgomery, D. C. & Runger, G. C. (2003). *Applied statistics and probability for engineers*. John Wiley and Sons.
- Neil D., L. & Michael I., J. (2006). Gaussian processes and the null-category noise model. In *Semi-Supervised Learning* (pp. 136–150). Cambridge, MA, USA: The MIT Press.
- Nickel, C., Brandt, H., & Busch, C. (2011). Classification of acceleration data for biometric gait recognition on mobile devices. In A. Brömme & C. Busch (Eds.), *In Proceedings of the Special Interest Group on Biometrics and Electronic Signatures* (Vol. 191, pp. 57–66).
- Oxycon Mobile Indirect Calorimeter. (2017). Retrieved March 6, 2018, from <https://www.vyair.com/>
- Parkka, J., Cluitmans, L., & Ermes, M. (2010). Personalization algorithm for real-time activity recognition using pda, wireless motion bands, and binary decision tree. *IEEE Transactions on Information Technology in Biomedicine*, 14(5), 1211–1215.
- Pedregosa, F., Varoquaux, G., Gramfort, A., Michel, V., Thirion, B., Grisel, O., . . . Duchesnay, E. (2011). Scikit-learn: Machine learning in Python. *Journal of Machine Learning Research*, 12, 2825–2830.
- Ramírez-Gallego, S., Krawczyk, B., García, S., Woźniak, M., & Herrera, F. (2017). A survey on data preprocessing for data stream mining: Current status and future directions. *Neurocomputing*, 239, 39–57. doi:<https://doi.org/10.1016/j.neucom.2017.01.078>
- Reiss, A. & Stricker, D. (2013). Personalized mobile physical activity recognition. In *Proceedings of the 2013 International Symposium on Wearable Computers* (pp. 25–28). Zurich, Switzerland: ACM.
- Schölkopf, B., Williamson, R., Smola, A., Shawe-Taylor, J., & Platt, J. (1999). Support vector method for novelty detection. In *Proceedings of the 12th International Conference on Neural Information Processing Systems, NIPS'99* (pp. 582–588). Denver, CO: MIT Press.
- Scudder, H. (1965). Probability of error of some adaptive pattern-recognition machines. *IEEE Transactions on Information Theory*, 11(3), 363–371.
- Settles, B. (2012). Active learning. *Synthesis Lectures on Artificial Intelligence and Machine Learning*, 6(1), 1–114.
- Szummer, M. & Jaakkola, T. (2002a). Information regularization with partially labeled data. In *Proceedings of the 15th International Conference on Neural Information Processing Systems, NIPS'02* (pp. 1049–1056). Cambridge, MA, USA: MIT Press.
- Szummer, M. & Jaakkola, T. (2002b). Partially labeled classification with markov random walks. In T. G. Dietterich, S. Becker, & Z. Ghahramani (Eds.), *Advances in Neural Information Processing Systems 14* (pp. 945–952). MIT Press.
- Triguero, I., García, S., & Herrera, F. (2015). Self-labeled techniques for semi-supervised learning: Taxonomy, software and empirical study. *Knowledge and Information Systems*, 42(2), 245–284.
- Truong, Q. D., Truong, Q. B., & Dkaki, T. (2016). Graph methods for social network analysis. In P. C. Vinh & L. Barolli (Eds.), *Nature of Computation and Communication: Second International Conference, ICTCC 2016* (pp. 276–286). Springer International Publishing.
- Vapnik, V. (1998). *Statistical learning theory*. John Wiley & Sons.
- Vapnik, V. & Chervonenkis, A. (1974). *Theory of pattern recognition [in russian]*. Moscow: Nauka.

- Vapnik, V. & Sterin, A. (1977). On structural risk minimization or overall risk in a problem of pattern recognition. *Automation and Remote Control*, 10(3), 1495–1503.
- Wang, J. H., Ding, J. J., Chen, Y., & Chen, H. H. (2012). Real time accelerometer-based gait recognition using adaptive windowed wavelet transforms. In *2012 IEEE Asia Pacific Conference on Circuits and Systems* (pp. 591–594).
- Wang, J., Luo, S. W., & Zeng, X. H. (2008). A random subspace method for co-training. In *2008 IEEE International Joint Conference on Neural Networks (IEEE World Congress on Computational Intelligence)* (pp. 195–200).
- Weiss, G. & Lockhart, J. (2012). The impact of personalization on smartphone-based activity recognition. In *AAAI Workshops*.
- Wolpert, D. H. (1992). Stacked generalization. *Neural Networks*, 5(2), 241–259. doi:10.1016/S0893-6080(05)80023-1
- Wright, S. J. (2015). Coordinate descent algorithms. *Mathematical Programming*, 151(1), 3–34.
- Yan, K. K., Zhao, H., & Pang, H. (2017). A comparison of graph- and kernel-based – omics data integration algorithms for classifying complex traits. *BMC Bioinformatics*, 18(1).
- Yarowsky, D. (1995). Unsupervised word sense disambiguation rivaling supervised methods. In *Proceedings of the 33rd Annual Meeting on Association for Computational Linguistics* (pp. 189–196). Cambridge, Massachusetts: Association for Computational Linguistics.
- Yaslan, Y. & Cataltepe, Z. (2010). Co-training with relevant random subspaces. *Neurocomputing*, 73(10), 1652–1661.
- Zhou, D., Bousquet, O., Lal, T. N., Weston, J., & Schölkopf, B. (2003). Learning with local and global consistency. In *Proceedings of the 16th International Conference on Neural Information Processing Systems, NIPS’03* (pp. 321–328). Whistler, British Columbia, Canada: MIT Press.
- Zhou, Y. & Goldman, S. (2004). Democratic co-learning. In *Proceedings - International Conference on Tools with Artificial Intelligence, ICTAI* (pp. 594–602).
- Zhou, Z. H. & Li, M. (2005). Tri-training: Exploiting unlabeled data using three classifiers. *IEEE Transactions on Knowledge and Data Engineering*, 17(11), 1529–1541.
- Zhou, Z.-H. (2012). *Ensemble methods: Foundations and algorithms*. Chapman and Hall.
- Zhu, X. & Ghahramani, Z. (2002). *Learning from labeled and unlabeled data with label propagation* (Technical report No. CMU-CALD-02-107). School Comput Sci Carnegie Mellon Univ.
- Zhu, X., Ghahramani, Z., & Lafferty, J. (2003). Semi-supervised learning using gaussian fields and harmonic functions. In *Proceedings of the Twentieth International Conference on International Conference on Machine Learning, ICML’03* (pp. 912–919). Washington, DC, USA: AAAI Press.
- Zhu, X. & Goldberg, A. (2009). *Introduction to semi-supervised learning*. Morgan and Claypool.
- Zinnen, A., Blanke, U., & Schiele, B. (2009). An analysis of sensor-oriented vs. model-based activity recognition. In *2009 International Symposium on Wearable Computers* (pp. 93–100).

Bibliography

Publications Related to the Dissertation

Journal Articles

- Cvetković, B., Szeklicki, R., Janko, V., Lutomski, P., & Luštrek, M. (2017). Real-time activity monitoring with a wristband and a smartphone. *Information Fusion*. In press. doi:10.1016/j.inffus.2017.05.004
- Cvetković, B., Janko, V., Romero, A. E., Kafali, O., Stathis, K., & Luštrek, M. (2016). Activity recognition for diabetic patients using a smartphone. *Journal of Medical Systems*, 40(12), 256.
- Cvetković, B., Milić, R., & Luštrek, M. (2016). Estimating energy expenditure with multiple models using different wearable sensors. *IEEE Journal of Biomedical and Health Informatics*, 20(4), 1081–1087.
- Cvetković, B., Kaluža, B., Gams, M., & Luštrek, M. (2015). Adapting activity recognition to a person with Multi-Classifer Adaptive Training. *Journal of Ambient Intelligence and Smart Environments*, 7(2), 171–185.
- Kaluža, B., Cvetković, B., Dovgan, E., Gjoreski, H., Gams, M., Luštrek, M., & Mirchevska, V. (2014). A multi-agent care system to support independent living. *International Journal on Artificial Intelligence Tools*, 23(01), 1–30.

Conference Papers

- Cvetković, B., Janko, V., & Luštrek, M. (2015). Demo abstract: Activity recognition and human energy expenditure estimation with a smartphone. In *13th IEEE International Conference on Pervasive Computing and Communication, PerCom Workshops 2015* (pp. 193–195).
- Cvetković, B., Mirchevska, V., Janko, V., & Luštrek, M. (2015a). Recognition of high-level activities with a smartphone. In *Adjunct Proceedings of the 2015 ACM International Joint Conference on Pervasive and Ubiquitous Computing and Proceedings of the 2015 ACM International Symposium on Wearable Computers, UbiComp/ISWC'15 Adjunct* (pp. 1453–1461). Osaka, Japan: ACM.
- Cvetković, B. & Luštrek, M. (2014). Analiza možnosti zaznavanja podobnosti med uporabniki. In *Intelligent systems : Proceedings of the 17th International Multiconference Information Society - IS 2014*. Ljubljana, Slovenia: Jožef Stefan Institute.
- Cvetković, B., Kaluža, B., Luštrek, M., & Gams, M. (2012). Multi-Classifer Adaptive Training: Specialising an activity recognition classifier using semi-supervised learning. In F. Paternò, B. de Ruyter, P. Markopoulos, C. Santoro, E. van Loenen, & K. Luyten (Eds.), *Ambient intelligence* (pp. 193–207). Springer Berlin Heidelberg.
- Cvetković, B., Luštrek, M., Kaluža, B., & Gams, M. (2011). Semi-supervised learning for adaptation of human activity recognition classifier to the user. In *IJCAI Workshop on Space, Time and Ambient Intelligence*.

Other Publications

Journal Articles

- Janko, V., Cvetković, B., Gradisek, A., Luštrek, M., Strumbelj, B., & Kajtna, T. (2017). E-gibalec: Mobile application to monitor and encourage physical activity in schoolchildren. *JAISE*, 9(5), 595–609.
- Cvetković, B., Gjoreski, H., Janko, V., Kaluža, B., Gradisek, A., Jurinčič, I., . . . Luštrek, M. (2016). E-turist: An intelligent personalised trip guide. *Informatica*, 40(4), 447–455.
- Luštrek, M., Gjoreski, H., Vega, N. G., Kozina, S., Cvetković, B., Mirchevska, V., & Gams, M. (2015). Fall detection using location sensors and accelerometers. *IEEE Pervasive Computing*, 14(4), 72–79.
- Piltaver, R., Cvetković, B., & Kaluža, B. (2015). Denoising human-motion trajectories captured with ultra-wideband real-time location system. *Informatica*, 39(3), 311–322.
- Solar, H., Fernández, E., Tartarisco, G., Pioggia, G., Cvetković, B., Kozina, S., . . . Lampe, J. (2013). A non invasive, wearable sensor platform for multi-parametric remote monitoring in chf patients. *Health and Technology*, 3(2), 99–109.

Book Chapters

- Mazomenos, E. B., Rodríguez, J. M., Barca, C. C., Tartarisco, G., Pioggia, G., Cvetković, B., . . . Maharatna, K. (2014). Case studies. In K. Maharatna & S. Bonfiglio (Eds.), *Systems design for remote healthcare* (pp. 277–332). New York, NY: Springer New York. doi:10.1007/978-1-4614-8842-2_10

Conference Papers

- Cvetković, B., Drobnič, V., & Luštrek, M. (2017). Recognizing hand-specific activities with a smartwatch placed on dominant or non-dominant wrist. In *Slovenian Conference on Artificial Intelligence : proceedings of the 20th International Multiconference Information Society - IS 2017*. Ljubljana, Slovenia: Jožef Stefan Institute.
- Cvetković, B., Gjoreski, M., Šorn, J., Frešer, M., Bogdański, M., Jackowska, K., . . . Luštrek, M. (2017). Management of physical, mental and environmental stress at the workplace. In *2017 International Conference on Intelligent Environments (IE)* (pp. 76–83).
- Cvetković, B., Gjoreski, M., Šorn, J., Maslov, P., Kosiedowski, M., Michałand Bogdański, Stroiński, A., & Luštrek, M. (2017). Real-time physical activity and mental stress management with a wristband and a smartphone. In *Proceedings of the 2017 ACM International Joint Conference on Pervasive and Ubiquitous Computing and Proceedings of the 2017 ACM International Symposium on Wearable Computers UbiComp '17* (pp. 225–228). Maui, Hawaii: ACM. doi:10.1145/3123024.3123184
- Cvetković, B., Gjoreski, M., Šorn, J., Maslov, P., & Luštrek, M. (2017). Monitoring physical activity and mental stress using wrist-worn device and a smartphone. In Y. Altun, K. Das, T. Mielikäinen, D. Malerba, J. Stefanowski, J. Read, . . . S. Džeroski (Eds.), *Machine Learning and Knowledge Discovery in Databases* (pp. 414–418). Springer International Publishing.
- Cvetković, B., Gjoreski, M., Frešer, M., Kosiedowski, M., & Luštrek, M. (2016). Monitoring and management of physical, mental and environmental stress at work. In *Slovenian Conference on Artificial Intelligence : proceedings of the 19th International Multiconference Information Society - IS 2016*. Ljubljana, Slovenia: Jožef Stefan Institute.

- Cvetković, B., Janko, V., Gradišek, A., Luštrek, M., Kajtna, T., & Strumbelj, B. (2016). Mobile application to stimulate physical activity in schoolchildren. In *IE 2016, The 12th International Conference on Intelligent Environments*.
- Cvetković, B., Pangerc, U., Gradišek, A., & Luštrek, M. (2016). Monitoring patients with diabetes using wearable sensors : Predicting glycaemias using ecg and respiration rate. In *1st ECAI Workshop on Artificial Intelligence for Diabetes, AID, at the 22nd European Conference on Artificial Intelligence (ECAI 2016)*.
- Cvetković, B., Vodopija, A., Rudel Balorda, Z., & Luštrek, M. (2016). Hospitalisation prediction from telemonitoring data in congestive heart failure patients. In *Workshop on Knowledge Discovery in Healthcare Data [in conjunction with] 25th International joint Conference on Artificial Intelligence (IJCAI)*. New York, NY, USA.
- Frešer, M., Cvetković, B., Gradišek, A., & Luštrek, M. (2016). Anticipatory system for T–H–C dynamics in room with real and virtual sensors. In *Proceedings of the 2016 ACM International Joint Conference on Pervasive and Ubiquitous Computing: Adjunct, UbiComp '16* (pp. 1267–1274). Heidelberg, Germany: ACM.
- Frešer, M., Gradišek, A., Cvetković, B., & Luštrek, M. (2016). An intelligent system to improve T–H–C parameters at the workplace. In *Proceedings of the 2016 ACM International Joint Conference on Pervasive and Ubiquitous Computing: Adjunct, UbiComp '16* (pp. 61–64). Heidelberg, Germany: ACM.
- Gjoreski, M., Janko, V., Gjoreski, H., Cvetković, B., Luštrek, M., & Gams, M. (2016). Activity and stress monitoring using smartphone and wrist device. In *Jožef Stefan International Postgraduate School Students' Conference*. Ljubljana, Slovenia: Jožef Stefan International Postgraduate School.
- Kajtna, T., Cvetković, B., Janko, V., Štrumbelj, B., Štihec, J., Luštrek, M., & Gradišek, A. (2016). Effectiveness of a mobile application with respect to its personalization and use of motivational elements. In *Youth sport : proceedings of the 8th Conference for Youth Sport*. Faculty of Sport, Ljubljana.
- Cvetković, B., Mirchevska, V., Janko, V., & Luštrek, M. (2015b). Recognition of high-level activities with a smartphone. In *Proceedings of the 2015 ACM International Joint Conference on Pervasive and Ubiquitous Computing and Proceedings of the 2015 ACM International Symposium on Wearable Computers - UbiComp '15* (pp. 1453–1461).
- Cvetković, B., Pangerc, U., Gradišek, A., & Luštrek, M. (2015). Prepoznavanje in napovedovanje hiperglikemij in hipoglikemij na neinvaziven način. In *Intelligent systems : proceedings of the 18th International Multiconference Information Society - IS 2015*. Ljubljana, Slovenia: Jožef Stefan Institute.
- Cvetković, B., Piltaver, R., Gams, M., & Luštrek, M. (2015). Evolution of activity monitoring through various projects. In *ICT Innovations 2015*.
- Luštrek, M. & Cvetković, B. (2015). Zdravje na daljavo je več kot le zajem, prenos in prikaz podatkov. In *Intelligent systems : proceedings of the 18th International Multiconference Information Society - IS 2015*. Ljubljana, Slovenia: Jožef Stefan Institute.
- Luštrek, M., Cvetković, B., Janko, V., Štrumbelj, B., Štihec, J., & Kajtna, T. (2015). Mobile application encouraging physical activity of schoolchildren for more effective physical education. In *Intelligent systems : proceedings of the 18th International Multiconference Information Society - IS 2015*. Ljubljana, Slovenia: Jožef Stefan Institute.
- Starič, K. D., Cvetković, B., Levicnik, A.-U., & Starič, J. (2015). One health concept of measuring and monitoring wellbeing. In *ICT Innovations 2015*.
- Barca, C., Rodriguez, J., Puddu, P., Luštrek, M., Cvetković, B., Bordone, M., ... Tamburini, E. (2014). Advanced medical expert support tool (a-mest): Ehr-based inte-

- gration of multiple risk assessment solutions for congestive heart failure patients. In *IFMBE Proceedings*.
- Luštrek, M., Cvetković, B., & Janko, V. (2014). Monitoring patients' lifestyle with a smart-phone and other devices placed freely on the body. In *Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)* (Vol. 8850, pp. 96–99).
- Zupančič, D., Cvetković, B., & Gams, M. (2014). Smart-home energy management system: A trade-off between energy consumption and thermal comfort experience according to occupant's activity. In *6th Jožef Stefan International Postgraduate School Students' Conference*. Ljubljana, Slovenia: Jožef Stefan International Postgraduate School.
- Cvetković, B., Kaluža, B., Gjoreski, H., & Luštrek, M. (2013). Hybrid recommender system for personalized poi selection. In *Proceedings of the 16th International Multiconference Information Society - IS 2013*. Ljubljana, Slovenia: Jožef Stefan Institute.
- Cvetković, B. & Luštrek, M. (2013). Estimation of human energy expenditure using inertial sensors and heart rate sensor. In *5th Jožef Stefan International Postgraduate School Students Conference*. Ljubljana, Slovenia: Jožef Stefan International Postgraduate School.
- Gjoreski, H., Cvetković, B., Kaluža, B., & Luštrek, M. (2013). Sightseeing route planning. In *Proceedings of the 16th International Multiconference Information Society - IS 2013*. Ljubljana, Slovenia: Jožef Stefan Institute.
- Luštrek, M., Kaluža, B., Cvetković, B., & Gjoreski, H. (2013). E-turist : Inteligentni elektronski turistični vodnik. In *Proceedings of the 16th International Multiconference Information Society - IS 2013*. Ljubljana, Slovenia: Jožef Stefan Institute.
- Cvetković, B. & Luštrek, M. (2012a). Algoritem lof kot metoda v sistemu za podporo odločanja. In *Proceedings of the 15th International Multiconference Information Society - IS 2012*, Ljubljana, Slovenia: Jožef Stefan Institute.
- Cvetković, B. & Luštrek, M. (2012b). Risk assessment using local outlier factor algorithm. In *4th Jožef Stefan International Postgraduate School Students Conference*. Ljubljana, Slovenia: Jožef Stefan International Postgraduate School.
- Luštrek, M., Cvetković, B., & Kozina, S. (2012). Energy expenditure estimation with wearable accelerometers. In *Iscas 2012 - 2012 ieee international symposium on circuits and systems* (pp. 5–8).
- Luštrek, M., Kaluža, B., Cvetković, B., Dovgan, E., Gjoreski, H., Mirchevska, V., & Gams, M. (2012). Confidence: Ubiquitous care system to support independent living. In *Frontiers in Artificial Intelligence and Applications* (pp. 1013–1014).
- Luštrek, M., Gjoreski, H., Kozina, S., Cvetković, B., Mirchevska, V., & Gams, M. (2011). Detecting falls with location sensors and accelerometers. In *Proceedings of the Twenty-Fifth AAAI Conference on Artificial Intelligence and the Twenty-Third Innovative Applications of Artificial Intelligence* (pp. 1662–1667).

Biography

Božidara Cvetković was born on December 3, 1983 in Slovenj Gradec, Slovenia. She finished her primary and secondary education in Velenje, Slovenia. In 2002, she started her studies at the Faculty of Computer and Information Science, University of Ljubljana, and, in 2008, she obtained a BSc by defending her BSc thesis titled "Feasibility analysis of realization of primitive computer structures on DNA assembly elements" under the supervision of Prof. Dr. Miha Mraz and co-supervision of Prof. Dr. Roman Jerala.

In the fall of 2011, she enrolled in the PhD program entitled "Information and Communication Technologies" at the Jožef Stefan International Postgraduate School, Ljubljana, Slovenia, and started working as a research assistant at the Department of Intelligent Systems, Jožef Stefan Institute, Ljubljana, Slovenia.

As a research assistant, she collaborated on EU-funded projects Confidence (Ubiquitous Care System to Support Independent Living), CHIRON (Cyclic and person-centric Health management: Integrated appRoach for hOme, mobile and clinical eNvironments), Commodity12 (COntinuous Multi-parametric and Multi-layered analysis Of DIabetes TYpe 1 & 2), Fit4Work, and HeartMan (Personal Decision Support System for Heart Failure Management), and on national projects eTurist, eVratar, and eGibalec. The PhD topic was obtained from the Confidence project in which the personalisation was a major issue due to physical differences of older and sometimes sick people involved, for which we needed to train and tailor machine-learning models.

During her work at the Jožef Stefan Institute, she was active in the Slovenian Artificial Intelligence Society (SLAIS), where she was a member of the executive board and a secretary and treasurer from 2014 to 2018. She was also active in the student board of the Jožef Stefan International Postgraduate School, where she was a representative of the ICT study programme in 2013 and 2014.

Her field of research is in the area of machine learning; more specifically, modelling human behaviour using machine-learning techniques on data from wearable sensors for intelligent systems in the scope of ambient intelligence. The research covers supervised and semi-supervised machine-learning for activity monitoring, user modelling, user profiling, anomalous behaviour detection, and personalisation. She is also an experienced software developer and has developed several applications that embed machine-learning models for human activity monitoring or provide recommendations based on the user profile.

