WIJEKOON, A. 2021. *Personalised exercise recognition towards improved self-management of musculoskeletal disorders*. Robert Gordon University, PhD thesis. Hosted on OpenAIR [online]. Available from: https://doi.org/10.48526/rgu-wt-1358224

Personalised exercise recognition towards improved self-management of musculoskeletal disorders.

WIJEKOON, A.

2021

The author of this thesis retains the right to be identified as such on any occasion in which content from this thesis is referenced or re-used. The licence under which this thesis is distributed applies to the text and any original images only – re-use of any third-party content must still be cleared with the original copyright holder.

This document was downloaded from https://openair.rgu.ac.uk

Personalised Exercise Recognition Towards Improved Self-management of Musculoskeletal Disorders

Anjana Wijekoon

A thesis submitted in partial fulfilment
of the requirements of
Robert Gordon University
for the degree of Doctor of Philosophy

February 2021

Abstract

Musculoskeletal Disorders (MSD) have been the primary contributor to the global disease burden with increased years lived with disability. Such chronic conditions require self-management, typically in the form of maintaining an active lifestyle while adhering to prescribed exercises. Today, exercise monitoring in fitness applications wholly relied on user input. Effective digital intervention for self-managing MSD should be capable of monitoring, recognising and assessing performance quality of exercises in real-time. Exercise Recognition (ExRec) is the Machine Learning problem that investigates the automation of exercise monitoring. Multiple challenges arise when implementing high performing ExRec algorithms for a wide range of exercises performed by people from different demographics. And in this thesis, we explore three personalisation challenges.

Different sensor combinations can be used to capture exercises to improve usability and deployability in restricted settings. Accordingly, a recognition algorithm should be adaptable to different sensor combinations. To address this challenge, we investigate the best feature learners for individual sensors and effective fusion methods that minimise the need for data and very deep architectures. We implement a modular hybrid attention fusion architecture that emphasises significant features and understates noisy features from multiple sensors for each exercise.

Persons perform exercises differently when not supervised; they incorporate personal rhythms and nuances. Accordingly, a recognition algorithm should be able to adapt to different persons. To address personalised recognition challenge, we investigate how to adapt learned models to new, unseen persons. Key to achieving effective personalisation is the ability to personalise with few data instances. Accordingly, we bring together personalisation methods and advances in meta-learning to introduce personalised meta-learning methodology. Resulting personalised meta-learners are learning to adapt to new end-users with only few data instances.

It is infeasible to design algorithms to recognise all expected exercises a physiotherapist would prescribe. Accordingly, the ability to integrate new exercises after deployment is another challenge in ExRec. The challenge of adapting to unseen exercises is known as open-ended recognition. We extend the personalised meta-learning methodology to the open-ended domain such that an end-user can introduce a new exercise to the model with only few data instances.

Finally, we address the lack of publicly available data and collaborate with health science researchers to curate a heterogeneous multi-modal physiotherapy exercise dataset, MEx. We conduct comprehensive evaluations of the proposed methods using MEx to demonstrate that our methods successfully address the three ExRec challenges. We also show that our contributions are not restricted to the domain of ExRec but applicable in a wide range of activity recognition tasks by extending the evaluation to other HAR domains.

Keywords: Exercise Recognition, Self-management, Human Activity Recognition, Multi-modal Fusion, Personalised HAR, Open-ended HAR

Declaration of Authorship

I declare that I am the sole author of this thesis and that all verbatim extracts contained in the thesis have been identified as such and all sources of information have been specifically acknowledged in the bibliography. Parts of the work presented in this thesis have appeared in the following publications.

- Wijekoon, A., Wiratunga, N., Cooper, K., & Bach, K. (2020, May). Learning to Recognise Exercises in the Self-Management of Low Back Pain. In The Thirty-Third International Flairs Conference. AAAI Press.
(**Chapters 4, 7**)

- Wijekoon, A., Wiratunga, N., Cooper, K. (2020, July). Heterogeneous Multi-Modal Sensor Fusion with Hybrid Attention for Exercise Recognition. In International Joint Conference on Neural Networks (IJCNN). IEEE.
(**Chapters 4, 7**)

- Wijekoon, A., & Wiratunga, N. (2020, June). Evaluating the Transferability of Personalised Exercise Recognition Models. In International Conference on Engineering Applications of Neural Networks. Springer, Cham.
(**Chapters 3, 5, 7**)

- Wijekoon, A., & Wiratunga, N. (2020, December). Personalised Meta-Learning for Human Activity Recognition with Few-data. In International Conference on Innovative Techniques and Applications of Artificial Intelligence. Springer, Cham.
(**Chapters 5, 7**)

- Wijekoon, A., Wiratunga, N., Sani, S., & Cooper, K. (2020). A knowledge-light approach to personalised and open-ended human activity recognition. Knowledge-Based Systems, 192, 105651.
(**Chapters 3, 6, 7**)

- Wijekoon, A., Wiratunga, N., & Sani, S. (2018, July). Zero-shot learning with matching networks for open-ended human activity recognition. In Proceedings of the SICSA Workshop on Reasoning, Learning and Explainability (ReaLX 2018), CEUR Workshop Proceedings.
(**Chapters 6, 7**)

Acknowledgements

I would like to express my most profound appreciation to my supervisory team, Prof. Nirmalie Wiratunga, Prof. Kay Cooper, Dr. Stewart Massie and Dr. Sadiq Sani from Robert Gordon University (RGU). You always offered me the best support and advice whenever I needed them. I am also grateful for the critical and robust supervision you have provided for my research - thank you very much.

I am immensely grateful to Prof. Nirmalie Wiratunga for providing me with numerous opportunities to not only grow as a researcher but also to be a confident writer and a speaker. This research benefited from generous funding from EU Horizon 2020 project selfBACK and RGU. selfBACK and RGU offered the training and infrastructure to conduct research, and travel opportunities to participate in conferences and gather research experience. My sincere gratitude goes to the selfBACK team and School of Computing, at RGU for this wonderful opportunity.

I am thankful for the past and present members of the AI research group for the fond memories and insightful conversations that were immensely helpful. I also appreciate the administrative team and the IT team at the School of Computing for their support. Finally, I am thankful to my friends and family who supported and encouraged me during the past few years.

Contents

List of Tables

List of Figures

Chapter 1

Introduction

In 2017, Global Health Metrics reported that Musculoskeletal Disorders (MSD) are the primary contributor to the global disease burden with increased years lived with disability. Prevalent cases of MSD have consistently increased from 1990 to 2017 by 38.4% and the affected demographic ranges from ages 15 to 95 (James et al., 2018). Low back pain, neck pain and arthritis are several chronic conditions that falls under MSD (Figure 1.1), which are affecting the joints, bones, muscles, spine and in some cases, multiple body areas or systems. MSDs require self-management, typically in the form of maintaining an active lifestyle while adhering to exercises prescribed by a healthcare professional or rehabilitation specialist. Therefore it is vital to raise awareness about the importance of adherence and provide necessary support mechanisms to encourage active lifestyles.

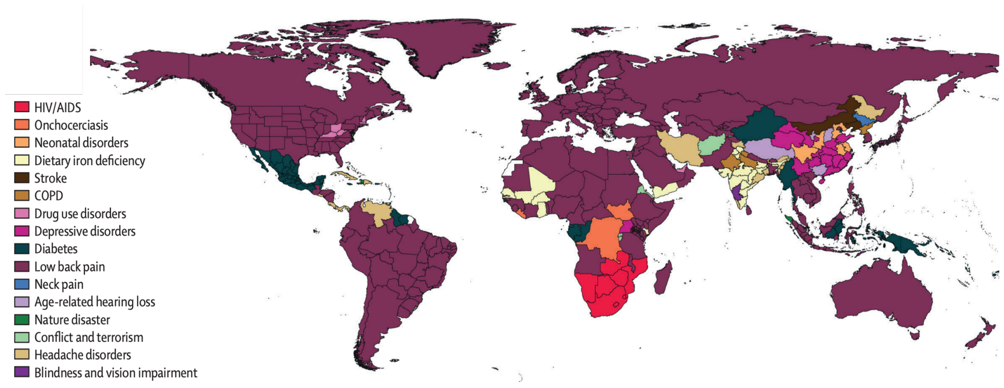


Figure 1.1: Leading causes of years lived with disability rate by country, 2017 (from James et al. (2018), image licensed under CC-BY-4.0)

Finding technological solutions for the self-management of MSD has been a research area that emerged over the last few years. Smart devices collect data that help to keep track of fitness activities such as walking or jogging with high precision using accelerometers. In the recent past, fitness applications gained popularity by gamification of physical activity monitoring and made a positive impact on general health and well-being. However, automated exercise monitoring is yet to be implemented in fitness applications due to the inherent challenges of complex movement monitoring and reasoning with sensors. Today, exercises monitored in fitness applications are wholly reliant on user input, resulting in low accuracy and reliability.

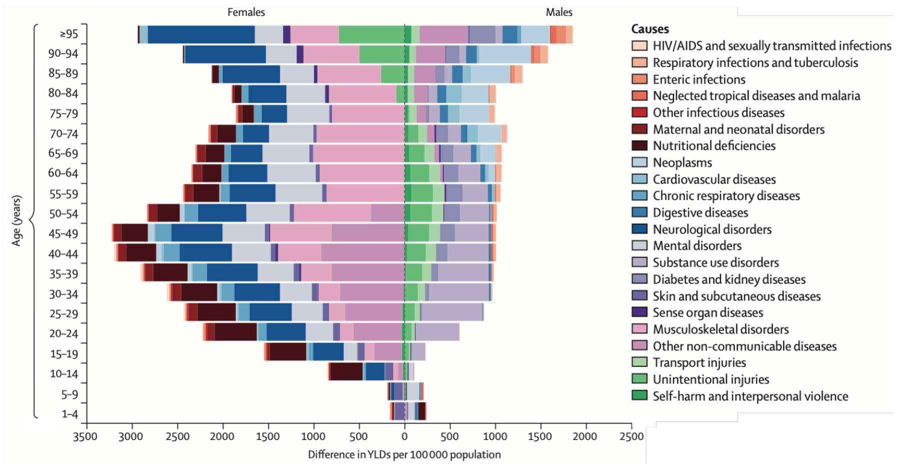


Figure 1.2: Years lived with disability rate by demographic groups, 2017 (from James et al. (2018), image licensed under CC-BY-4.0)

Effective digital intervention for self-managing MSD should be composed of three main components: monitoring exercises in real-time; recognising exercise being performed; and evaluating the quality of the performance. In this thesis, we focus on the first steps, monitoring and recognition of physiotherapy exercises. Exercise Recognition (ExRec) is the Machine Learning problem that investigates automating exercise monitoring which is an expert domain of Human Activity Recognition (HAR). Exercises are complex movements and need to be captured with multiple, strategically placed sensors. In HAR literature, a sensor (wearable or ambient) with a specific placement is referred to as a sensor modality (or modality in short). Sensor modalities used for ExRec are often generate different types of data (i.e. heterogeneous). This is in contrast to a single or multiple homogeneous sensor modalities used to recognise daily fitness activities such as running or walking (Ordóñez and Roggen, 2016; Radu et al., 2016; Yao et al., 2017). In addition, as shown in Figure 1.2, both male and female in a wide range of ages are affected by MSD (in pink).

Numerous challenges arise when implementing ExRec algorithms that can recognise a wide range of exercises that perform with high precision in many demographics.

Specifically, we recognise three challenges from previous work in the domain of HAR that need to be addressed to accomplish end-to-end recognition: heterogeneous multi-modal recognition, personalised recognition and open-ended recognition. Each challenge addresses a different usability aspect of ExRec: the ability to choose sensor modalities preferred by the end-user with multi-modal recognition; the ability to adapt to varying demographics with personalisation; and the ability to add preferred exercises with open-ended recognition.

In this research, we focus on improving usability and robustness of exercise recognition by implementing deep learning algorithms. Throughout, our methods are inspired by the need to reduce the demand for data and knowledge. We affirm that our contributions are not restricted to the domain of ExRec and applicable in a wide range of HAR tasks. Importantly, this work also collaborates with health science researchers to curate a heterogeneous multi-modal physiotherapy exercise dataset which is shared publicly to promote open-research in the HAR research domain.

1.1 Human Activity Recognition

Human Activity Recognition (HAR) is an extensively researched area in Machine Learning. HAR, as a supervised learning problem, is modelled as a multi-class classification task that requires labelled sensor data, preferably stratified across all activity classes. Gómez and Rojas (2016) said that there exists no one reasoning model or methodology that fits all tasks i.e. *no free lunch*. Accordingly, a broad spectrum of machine learning algorithms are used in literature to implement recognition models.

1.1.1 Early Research

In early research, classification with sensor data for HAR consists of three main steps: data collection, pre-processing and classification (Figure 1.3). The pre-processing pipeline is formed of two components: segmentation to create data instances from incoming sensor data streams; and feature extraction. Windowing is the commonly used segmentation method where a constant time window splits the sensor data stream in to data instances. Then each data instance is transformed to features, using a pre-defined set of feature extraction methods that extract time-domain, frequency-domain and spatial features. Time domain features include mean, standard deviation and min/max for the specific interval. Frequency-domain features include mean frequency, energy and entropy (Sani et al., 2017a). Methods such as Local Binary Pattern and Local Phase Quantisation (Nanni et al., 2017) are commonly used to extract spatial features.

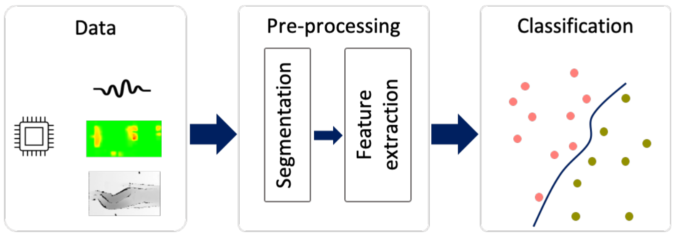


Figure 1.3: Classification with hand-crafted features

These features are used by classification algorithms such as k-nearest neighbour (Miu et al., 2015), Naive Bayes (Gomes et al., 2012), Support Vector Machines (Sani et al., 2016) and ensemble methods (Ravi et al., 2005). In early research, feature extraction and classification are disjoint steps except when feature selection strategies are used to optimise feature selection (Dash and Liu, 1997; Su et al., 2014). Manual creation and selection of features is found to be burdensome, yet, these methods have achieved high performance with limited data for bespoke HAR applications.

1.1.2 Deep Learning Methods

More recently, Deep Learning (DL) algorithms consolidated feature extraction and classification steps where the feature learning is conditioned by iterative optimisation. Similar to early methods, windowing is applied to obtain data instances and optionally feature transformation methods are applied. These feature transformation methods include frequency domain transformation (Sani et al., 2017a; Yao et al., 2017) for time-series data and skeleton extraction for depth camera data (Khaire et al., 2018; Rahmani and Bennamoun, 2017).

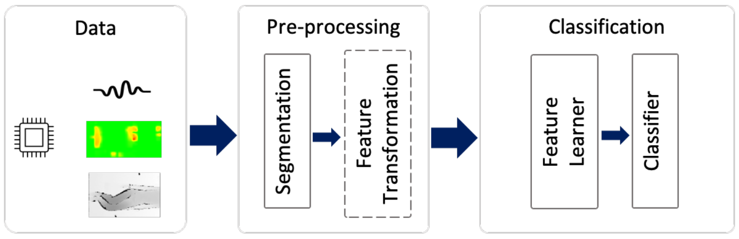


Figure 1.4: Classification with feature learners

During the classification step, convolutional and recurrent neural constructs are used to create feature learning architectures. Convolutional networks are selected to extract spatial features, and recurrent networks extract temporal features (Ordóñez and Roggen, 2016; Yao et al., 2017). The output of a deep feature learning architecture is connected to a neural network classifier. The network is then trained end-to-end, optimising feature learner weights to learn features that correctly predict class labels. As a result, DL methods eliminate the need for curating feature extraction techniques specific to each sensor modality. Recent literature shows that DL methods consistently outperform early methods in several comparative studies (Ordóñez and Roggen, 2016; Radu et al., 2018; Yao et al., 2017).

1.2 Research Motivation

HAR research, in general, has embraced the recent advances in DL similar to other application domains such as computer vision and machine translation. Although, ExRec which is a challenging HAR problem that involves multiple modalities and a wide range of complex activities, is yet to exploit these advances. ExRec is a high impact application area, especially for the self-management of MSDs and physiotherapy rehabilitation. We identify three challenges faced when implementing a high-impact, high-performance yet a user-friendly solution for physiotherapy exercise monitoring and recognition: the ability to support different sensor modality preferences; the ability to adapt to personal characteristics; and the ability to integrate personal activity preferences. We illustrate the three challenges in Figure 1.5 and identify them all under the umbrella term *personalisation*.

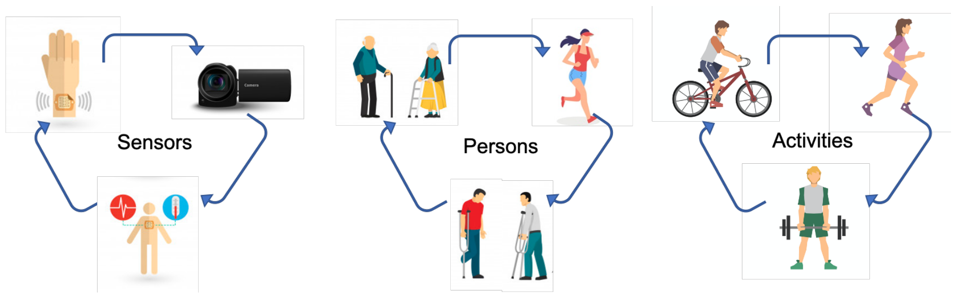


Figure 1.5: Facets of personalisation for human activity recognition

Preference of sensor modalities varies between different persons and constraints in deployment environments can call for restricted modality configurations. For example, a person at the physiotherapy clinic may perform exercises under a sensor-rich environment, and at home, they may only have access to a minimal setup. A recognition algorithm that is strictly bounded to a specific modality combination impedes usability. The heterogeneity of modality data suggests that there exists no one solution that fits all modalities and modality combinations. Accordingly, we find it is challenging to learn an effective feature representation for ExRec from any given modality combination. An algorithm should adapt to modality combinations and learn to identify effective modality and feature combinations while learning to discard the noise.

Sensors capture personal nuances that are more pronounced in exercises. Exercises can be modified to suit individual preferences, performed in unique rhythms, or the person may unintentionally incorporate personal idiosyncrasies. Accordingly, personalisation has been identified as a transferability challenge in ExRec. Creating personalised models specifically for the end-user is infeasible when reaching a broad audience because each end-user has to provide a large quantity of data. However, literature highlights that the best way to learn personal nuances is to learn from end-user data. Existing active learning methods rely on periodical end-user involvement which can be obtrusive in practice (Gomes et al., 2012; Longstaff et al., 2010; Losing et al., 2019). Accordingly, an effective personalised algorithm should adapt to personal nuances given only few data instances, limiting the end-user involvement which improves the user-experience.

A similar transferability challenge with high impact is the ability to recognise exercise classes not seen during model development, or open-ended recognition. For instance, physiotherapists prescribe different exercises plans for different patients considering their physiology, pain levels and many other factors. Current methods rely on the assumption that it is possible to know about all expected activity descriptions encoded using expert knowledge prior to deployment. Open-ended ExRec is an explicitly challenging domain, where an exercise is better presented by a data instance, rather than a description. This is our inspiration when designing methods to address the challenge of open-ended ExRec. Similarly to personalised ExRec, open-ended ExRec algorithms should also consider the limited availability of end-user data and unobtrusive deployment.

This thesis addresses the above challenges towards implementing an effective ExRec solution to self-manage MSDs. Accordingly, we investigate the following research questions (RQ1-3):

1. How to recognise exercises, given a set of sensor modalities, by learning modality and feature combinations and learning to discard noise?

2. How personalise exercise recognition to end-users with limited data and minimal

end-user interaction?

3. How to extend exercise recognition to recognise new unseen exercises with limited data and minimal end-user interaction?

1.3 Research Objectives

To address the challenges raised in exercise recognition and personalisation, we identify five key objectives for this thesis as follows:

1. *Multi-modal Recognition: Develop a fusion algorithm to recognise exercises using a combination of heterogeneous sensor modalities.*

   This objective addresses RQ 1. The emphasis is on introducing a fusion architecture that learns to highlight significant features and understate noisy features for each exercise class. We are also keen to minimise the depth of the feature learning architectures with the use of attention methods and to create an architecture that is customisable to many modality combinations. Realising this objective will improve overall recognition accuracy, and also implement architectures that are adaptable to end-user modality preferences.

2. *Personalised Recognition: Develop Exercise Recognition algorithms that are adaptable to unseen persons or person groups.*

   This objective addresses RQ 2. Key to achieving effective personalisation is the ability to personalise with few data instances. Accordingly, we investigate learning with few-data and meta-learners for ExRec. The resulting algorithm should be able to personalise a model using only few data instance obtained from the end-user. Consequently, a rehabilitating patient should be able to customise an activity recognition application to their physiological needs and personal nuances.

3. *Open-ended Recognition: Develop algorithms to recognise exercises not seen during model implementation.*

   This objective addresses RQ 3. Physiotherapists prescribe different exercise plans to various patients, and it is infeasible to design algorithms to recognise all expected exercise classes. Key to addressing this objective is to find the best representation of a new and unseen class that is *knowledge-light*. We explore few-shot learning and meta-learning methods to address this objective. The resulting algorithms should be able to integrate new unseen classes to the model with only few data instances obtained from the end-user.

4. *Create an open, sensor-rich dataset for physiotherapy exercise recognition.*

   We identify the lack of publicly available datasets to implement and evaluate exercise recognition algorithms. Accordingly, we investigate the sensors, exercises and data collection protocols that are used to create a comprehensive physiotherapy exercises dataset. The resulting dataset should be sensor-rich to identify the most effective and user-friendly sensor combinations. It should also include exercises that represent a wide range of physiotherapy exercises performed by persons from different demographics.

5. *Conduct a comprehensive evaluation of all developed methods for physiotherapy exercise recognition*

   Using the dataset created in Objective 4, a comprehensive evaluation of methods introduced in Objectives 1, 2 and 3 is performed. The objective thoroughly evaluates our methods against methods in recent literature and appropriate baselines. We extend the evaluation when possible to provide visual evidence to support our findings. The evaluation outcomes should demonstrate that our methods successfully address each exercise recognition challenge.

1.4 Contributions

We describe the contributions in this thesis that address the ExRec research challenges. In Figure 1.6, we show the current research landscape of HAR in the three areas of interest. We identify the demand for knowledge and data as one of the main drawbacks of existing methods and look to minimise data and knowledge requirements. Accordingly, we highlight where our contributions fall within the landscape of Neural Machine Learning and Human Activity Recognition against the demand for data. We also highlight how objectives and contributions are aligned and which chapters investigate each contribution.

1.4.1 Multi-modal Recognition

The first contribution of this thesis is the Multi-modal Hybrid Attention Fusion architecture, mHAF for ExRec presented in Chapter 4. In mHAF, first individual feature

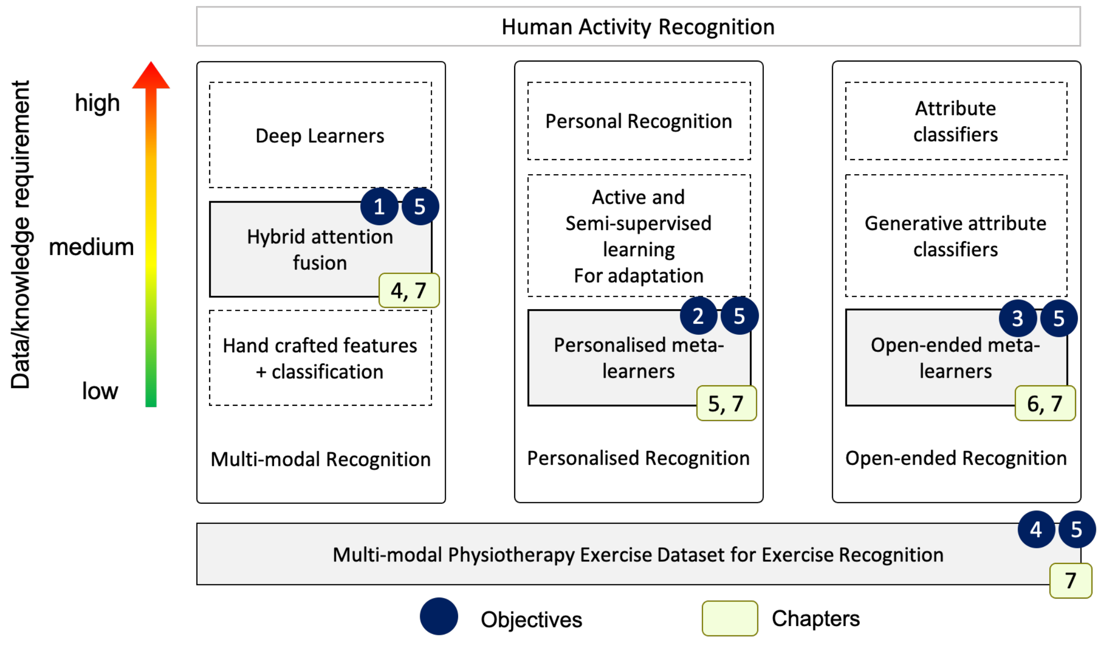

Human Activity Recognition
Data/knowledge requirement
high
medium
low
Deep Learners
1
5
Hybrid attention fusion
4, 7
Hand crafted features + classification
Multi-modal Recognition
Personal Recognition
Active and Semi-supervised learning For adaptation
2
5
Personalised meta-learners
5, 7
Personalised Recognition
Attribute classifiers
Generative attribute classifiers
3
5
Open-ended meta-learners
6, 7
Open-ended Recognition
Multi-modal Physiotherapy Exercise Dataset for Exercise Recognition
4
5
7
Objectives
Chapters

Figure 1.6: Contributions in the HAR research landscape

representations are learned independently for each individual heterogeneous modality. Then the resulting feature vectors are concatenated using a Hybrid Attention Fusion module (HAF). HAF module learns to highlight significant features and understate noisy features using attention which results in a shallow neural network architecture. HAF consists of two attention modules: soft attention and hard attention. The soft attention module in HAF is designed to highlight multiple informative features and the hard attention module highlights the few most informative features from multiple sensor modalities. Accordingly, HAF learn to highlight discriminatory features that otherwise may only have been learned using a very deep architecture trained on a more extensive training dataset. We stress the necessity of learning the feature importance for each exercise as different exercises are best captured with different sensor modality combinations. A comparative evaluation shows how our method outperforms fusion methods in literature and several baselines. We further confirm our results using confusion matrices and visualisation of attention weights where we show how mHAF architecture learns sensor modality and feature combinations for each exercise class. We verify the contribution of each module in the mHAF using an ablation study. And we perform an empirical evaluation to identify the best minimal sensor modality combinations to accommodate personal preferences or to deploy in restricted environments.

1.4.2 Personalised Recognition

The second contribution of this thesis is the personalised meta-learning methodology for ExRec presented in Chapter 5. In the ExRec domain, we find it is intuitive to treat a *person-activity* pair as the class label when sensors capture pronounced personal traits and nuances. Accordingly, we formalise the personalised meta-learning methodology, where we consider each person as an independent multi-class classification task to recognise *person-activity* classes. Once a person is considered an individual task, there is only a limited amount of data instances per *person-activity* class, which resembles a few-shot learning setting. Accordingly, we explore two meta-learning algorithms to implement our personalisation methodology: adaptation-optimised MAML and similarity-optimised Relation Networks. We further affirm the robustness of the personalisation methodology by improving an existing personalised few-shot learner. With a use case, we describe how to implement personalisation in the real-world. Our personalisation approach requires a one-time interaction with the end-user to obtain few seconds (max 15 seconds) of calibration data. In an empirical evaluation, we find the most optimal feature learners and the most optimal few-shot setting. A comparative evaluation demonstrates that personalised meta-learners outperform the deep learning methods and conventional meta-learning methods. We visualise how our personalisation methodology learns a generic meta-model that successfully adapts to any person given only few seconds of labelled data.

1.4.3 Open-ended Recognition

The third contribution of this thesis is the open-ended meta-learning methodology for ExRec presented in Chapter 6. Instead of the existing *knowledge-intensive* methods that require expert knowledge, we introduce a *knowledge-light* approach to perform open-ended ExRec. Our open-ended recognition algorithm is inspired by the research in few-shot learning and zero-shot learning. We investigate how conventional zero-shot meta-learners fail to dynamically integrate new activities without disposing previously known activities. We adapt the personalised meta-learning from the previous contribution to the open-ended setting to present the open-ended meta-learning methodology. Unlike conventional zero-shot learners, an open-ended meta-learner can increasingly add activity classes to the reasoning model, without having to remove original classes. In our evaluation, we compare three open-ended meta-learners to find the most robust algorithm across different activity classes and with an increasing number of classes. These are essential properties of an open-ended recognition algorithm to maintain performance across a wide range of user preferences. Our comparative evaluation shows that our algorithm outperforms recent open-ended algorithms from literature and a baseline lazy-learner algorithm. With a use case, we present how this methodology will be implemented in the real-world. An end-user is required to provide few seconds of data to introduce a new unseen activity that is integrated with zero model re-training.

1.4.4 MEx Dataset

The final contribution of this thesis is the *Heterogeneous **M**ulti-modal Physiotherapy **Ex**ercises Dataset*, ***MEx***. As identified in this thesis, the gaps in ExRec research begins with the lack of publicly available datasets and standardised evaluation practices. Accordingly, we present the MEx dataset; a sensor-rich dataset with 4 sensor modalities, recorded for 7 exercises with 30 participants. The exercises are selected by an expert to include that are recommended for physiotherapy rehabilitation. Sensors and sensor setup is selected to capture human movement expected to observed in theses exercises. 30 participants for the data collection are selected via convenient sampling. We present the comprehensive methodology of collecting, refining an publishing MEx in Chapter 7. This dataset is used to fine-tune and evaluate the first three contributions, both in single sensor modality settings and multi-modal settings. Accordingly, we affirm that the methods introduced in the first three contributions are effective in a range of sensor modalities and multi-modal environments. The long term goal is to establish MEx as a standardised benchmark dataset in the HAR domain.

1.5 Thesis Outline

An outline of the rest of this thesis is as follows.

Chapter 2 reviews relevant works in literature. We start by investigating the research in Exercise Recognition, algorithms and datasets available. Next, we review the related literature in the three branches of HAR challenges investigated in this thesis: heterogeneous multi-modal fusion, personalised HAR and open-ended HAR. We explore the research landscape of each challenge by discussing different approaches, strengths and limitations. Finally, we review two research areas with a view to addressing aforementioned challenges while minimising the demand for data and knowledge: attention fusion and few-shot learning.

Chapter 3 presents the background in neural networks, data and evaluation strategies.

We start by discussing the basics of multi-class classification with Neural Networks and the deep feature representation learning methods. Next, we introduce the four HAR datasets used for evaluation and the pre-processing steps applied on each modality. We also introduce the data formalisation, evaluation methodology, performance evaluation metrics and significance testing methods applied in this research.

Chapter 4 presents the first technical contribution of this thesis, Multi-modal Hybrid Attention Fusion architecture, mHAF. We formalise the fusion problem and discuss the main design considerations when designing a fusion architecture for heterogeneous multi-modal fusion. First, we explore how we can represent each modality and then the variants of fusion levels suitable for ExRec with heterogeneous multi-modal data. Next, we present our hybrid attention fusion module that is designed to learn feature importance for each exercise. Lastly, we bring the modules together to present the mHAF architecture, trained end-to-end for ExRec. In our empirical evaluation, we find the best feature learner for individual modalities, explore different modality combinations to identify the best minimal modality combinations and perform an ablation evaluation of our architecture using three HAR datasets.

Chapter 5 presents the second contribution, personalised meta-learning methodology for personalised exercise recognition. First, we formalised personalised meta-learning and introduced a use case where we discuss the practical implications of our methods. We present three meta-learners implementing the personalisation methodology, including the train and test conditions under the personalisation approach. Next, we perform two empirical experiments to fine-tune the three personalised meta-learners. We find the most effective feature learners for each algorithm suitable for a wide range of modalities. We also explore a range of few-shot settings to find the most optimal, considering the user-friendly deployment and performance.

In Chapter 6 we take the ideas of few-shot learning and meta-learning to address the challenge of open-ended HAR. We present a use case to highlight that our method only requires a one-time micro-interaction with the end-user to obtain few seconds of labelled data to introduce a new exercise class. We present an investigation on how zero-shot compatible meta-learners fail in an open-ended setting. Then, we introduce the open-ended meta-learning methodology and implement with three similarity-optimised meta-learners that alleviate recognised challenges. Our empirical evaluation finds the best open-ended meta-learner with two properties: firstly it is robust across a wide range of unseen activity classes; and secondly, it is robust when incrementally adding new, unseen activity classes.

Chapter 7 presents the MEx dataset and a comprehensive evaluation of our methods using the MEx dataset for ExRec. We start by detailing the sensor and exercise selection, data collection process and summarise the data collection results. We then present a comprehensive evaluation of the three contributions using the MEx dataset. First, we present a comparative evaluation of mHAF against several baselines and recent methods from literature. We show how mHAF learned modality and feature combinations for each exercise class using confusion matrices and visualisation of attention weights. Next, we present a comparative evaluation of personalised meta-learners against conventional meta-learners and deep learners. We further verify that personalisation improves performance by comparing the conventional and personalised meta-learner training and testing processes. Finally, we evaluate our open-ended meta-learner approach against a baseline and a few-shot learner to demonstrate the robustness of our method. We conclude the Chapter with a comprehensive evaluation of the same methods using three other HAR datasets. Here we further verify the applicability of our methods in different domains of HAR such as general fitness, activities of daily living and pose recognition.

We conclude in Chapter 8 revisiting our objectives with a review of the extent to which we met our research objectives. We also outline the limitations and implications of the work presented in this thesis and considerations for future extensions.

Chapter 2

Literature Review

There are many open research challenges in the Human Activity Recognition (HAR) domain such as, unobtrusive deployment, energy consumption on edge devices and adaptability to user preferences (Abdallah et al., 2018; Lara and Labrador, 2012). We identified three personalisation challenges that need addressed to implement Exercise Recognition (ExRec) solution as part of an effective digital intervention for self-managing Musculoskeletal Disorders.

- **Multi-modal recognition** to support modality preferences
- **Personalised recognition** to adapt to personal characteristics
- **Open-ended recognition** to adapt to activity preferences

This chapter explores HAR research landscape to identify related literature and research gaps to address these challenges. The investigations are grounded in classification methods using neural networks. We first explore research in ExRec to identify the need for public data and open research to implement methods addressing above challenges. It is followed by a comprehensive review of literature on the three areas of interest within the HAR research domain. We explore the recent advances in Machine Learning to address these personalisation challenges while alleviating the demand for data or knowledge. Accordingly, we investigate two areas in literature with the view to limit the demand for data and knowledge: attention fusion and few-shot learning.

2.1 Exercise Recognition

Exercises is a category of human activities with sequences of complex movements, repeated in short intervals. Cardio exercises, flexibility exercises, resistance exercises and rehabilitation exercises are some subcategories. ExRec is the task of learning reasoning models to recognise exercises from streams of sensor data that capture human movements. ExRec can be seen as an expert domain of HAR and is highly impactful in several application areas such as personal fitness, physiotherapy rehabilitation, recreation and self-management of chronic pain.

2.1.1 Sensor Modalities and Datasets

A wide range of sensors is used to capture exercise movements, that are either wearable or embedded in the environment (i.e. ambient sensors). Accelerometers are the most common wearable sensor used for ExRec that is either embedded in a smartwatch or a smartphone (Burns et al., 2018; Guo et al., 2018; Mendiola et al., 2019; Morris et al., 2014; Velloso et al., 2013). At each time-stamp, tri-axial accelerometers capture proper linear acceleration, i.e. acceleration not caused by gravity, along three orthogonal axes, x, y and z. Gyroscope and Magnetometers are two sensors commonly bundled with an accelerometer, referred to as an Inertial Measurement Unit (IMU) (Chavarriaga et al., 2013; Reiss and Stricker, 2012; Young et al., 2010, 2011). A gyroscope measures angular velocity and a magnetometer measures the strength of the magnetic field along three orthogonal axes. Abdallah et al. (2018) showed that a single IMU device did not recognise activities with sufficient accuracy and highlighted the need for additional sensors to capture complex activities. Accordingly, in literature we find alternative wearables for activity monitoring, such as pressure-sensitive fabric and heart-rate monitors. Pressure-sensitive fabric, commonly designed into clothing or designed as a band is used as a body-worn sensor. The pressure sensors in the fabric measure the pressure variances produced by a movement generating a stream of time-series data (Zhou et al., 2016). Heart-rate monitors measure the pace of the blood-flow (Reiss and Stricker, 2012) that is analogous to the intensity of an activity.

Many ambient sensors embedded in the environment, such as pressure sensors and cameras are also used to capture human movements. Ambient sensors are effectively used for ExRec in group settings like a gym or a smart home (Antón et al., 2015; Khurana et al., 2018). In a gym setting, multiple persons are present and also the activities involve complex movements that cannot be successfully captured by a single wearable sensor. Depth cameras and RGBD cameras are commonly used in such settings. A depth camera captures a series of grey-scale images of the view where the values are directly proportional to the distance between the object and the camera. Pressure sensitive fabric can be used as ambient sensors where the fabric is integrated into a target surface (Sundholm et al., 2014). A pressure mat is an ambient sensing device that consists of multiple pressure sensors typically arranged in a matrix. Over time, a pressure mat generates a sequence of heat-maps that closely resembles a low-resolution video.

A sensor modality is identified by the sensor type and its placement. Table 2.1 presents a comprehensive list of datasets selected from ExRec research and the sensor modalities used. We include literature since 2012, that present datasets of exercises, monitored using wearable or ambient sensors that were collected to develop machine learning algorithms. Following notations are used to identify sensors modalities: PM: Pressure mat, IMU: Inertial Measurement Unit, A: Accelerometer, G: Gyroscope, CSI: Channel State Information, ECG: Electrocardiogram, H: Heart rate monitor and PB: Pressure sensor band. There is a wide range of sensors modalities and modality combinations used in bespoke application areas. Notably, all except Vakanski et al. (2018) (marked with an asterisk) are not publicly available for open research to the best of our knowledge.

2.1.2 Recognition Algorithms

Given sensor data streams, ExRec is often viewed as a multi-class classification task. The two-stage approach found in early HAR research is commonly used in the ExRec domain. Labelled data instances are used to model a multi-class classifier where the feature vectors are hand-crafted using a pipeline of pre-processing steps. In literature, classification algorithm such as k-nearest neighbour (Sundholm et al., 2014; Xiao et al., 2018), SVM (Morris et al., 2014), Decision Trees (Zhou et al., 2016), Random Forest (Mendiola et al., 2019; Velloso et al., 2013) and HMM (Qi et al., 2018) are used as to model labelled data. While Deep Learning methods are the state-of-the-art in general HAR, only recently, such methods are implemented for ExRec. Burns et al. (2018) uses convolutional and recurrent components in their ExRec architecture to recognise shoulder rehabilitation exercises. Burns et al. (2018) also demonstrated that traditional machine learning algorithm that uses hand-crafted features such as k-NN, RF and SVM fail to outperform their deep learning architecture.

ExRec models are evaluated using different methodologies. Guo et al. (2018); Nguyen et al. (2015a); Zhou et al. (2016) and Burns et al. (2018) build ExRec models for personal

Table 2.1: Exercise Recognition Datasets

Dataset	Activity Classes	Sensor Modalities
Velloso et al. (2013)	5 Weight Lifting Exercises	IMU:wrist, arm, back and on equipment
Cheng et al. (2013b)	10 Calisthenic and Weight Exercises	A&G:wrist, arm and hip
Koskimäki and Siirtola (2014)	30 Calisthenic and Weight Exercises	A:wrist ans torso
Sundholm et al. (2014)	10 Calisthenic and Weights Exercises	PM:floor
Morris et al. (2014)	26 callisthenics and weight training exercises	A:arm, G:arm
Ar and Akgul (2014)	8 rehabilitation exercises	Kinect
Antón et al. (2015)	6 Upper Body Exercises	Kinect
Lin et al. (2015)	6 exercises for frozen shoulder rehabilitation	IMU:wrist and arm
Pernek et al. (2015)	6 upper body strength training exercises	A:left and right wrists, left and right arms and chest
Nguyen et al. (2015a)	8 Basketball Actions	IMU:back, left and right feet and left and right legs
Bleser et al. (2015)	13 upper body and lower body strength exercises	IMU:wrist, elbow and chest
Pernek et al. (2015)	6 weight exercises	5:wrists, arms and torso
Zhou et al. (2016)	4 Gym Leg Exercises	PB:thigh
Parmar and Morris (2016)	5 large amplitude movement exercises	Kinect
Rybarczyk et al. (2017)	4 rehabilitation exercises	Kinect
Vox and Wallhoff (2017)	20 rehabilitation exercises	Kinect
Das et al. (2017)	7 Calisthenic and Weight Exercises	A:wrist, H:chest
Crema et al. (2017)	9 weight exercises	IMU:wrist
Xiao et al. (2018)	4 Calisthenic and Weight Exercises	CSI
Burns et al. (2018)	7 Shoulder Exercises performed bilaterally	A:wrist, G:wrist
Qi et al. (2018)	12 Weight Exercises	ECG:wrist and chest, A:wrist and chest
Guo et al. (2018)	7 Ambulatory activities + Rope Jumping, Cycling, Gymnastics	IMU:wrist, arm, waist, thigh, ankle
Chapron et al. (2018)	4 exercises for neuromuscular rehabilitation	IMU:wrist
Triantafyllidis et al. (2018)	Aerobic exercises in a group setting	Kinect, H:wrist
*Vakanski et al. (2018)	10 rehabilitation exercises	Optical tracker and Kinect
Mendiola et al. (2019)	4 Calisthenic and Weight Exercises	A:wrist
Nardi (2019)	Upper Body Exercises	A:wrist, G:wrist and arm
Zhu et al. (2019)	4 rehabilitation exercises	IMU:shoulders, back, elbows, and forehead

fitness applications. The classification algorithm in a personal fitness application is likely to be used by one person. Accordingly, leave-one-person-out evaluation is performed where the data of the test person is not seen during training. Khurana et al. (2018) presents an ExRec model for a gym environment with groups of people. They apply leave-one-day-out evaluation methodology to learn from a set of previous days to predict exercises on a future day. In a gym environment, it is expected to encounter the same person in multiple days, but different days have different schedules. Accordingly, the leave-one-day-out methodology correctly evaluates the transferability of their models to different days.

Mendiola et al. (2019) and Qi et al. (2018) are two personal ExRec models that apply cross-fold evaluation methodology. Cross-fold is seen as a person-agnostic methodology where train and test data is split disregard of the person or day. Such a method share persons across training and test data, thus exhibits exaggerated performance improvements. It also fails to emulate how the model will perform when used by persons not encountered during training. Accordingly, we find person agnostic evaluation methods are not suitable for personal ExRec model evaluation.

Unavailability of public datasets and inconsistent evaluation methodologies are two main challenges when advancing ExRec research. Few other challenges identified in the literature are listed as follows. The complexity of exercises calls for multiple sensor modalities working in sync to improve performance (Abdallah et al., 2018). The variability in performances between persons which calls for personalisation (Morris et al., 2014). Exercises can be confused with non-activity time if not captured with the correct set of sensor modalities. In addition, compared to ambulatory activities, there is a broader range of exercise classes resulting in complex multi-class classifiers.

2.2 Challenges in Human Activity Recognition

This thesis explore three branches of activity recognition research: multi-modal recognition, personalised recognition and open-ended recognition of activities. In this section, a review recent literature in these three areas of interest is presented.

2.2.1 Multi-modal Recognition

Advances in sensing technologies encourage the use of multiple sensor modalities to improve activity recognition performance (Abdallah et al., 2018). Also, the wide variety of HAR applications have different sensory requirements and often require more than one sensor modality to capture activities with high precision. Research in HAR has highlighted the challenge of reasoning with a variety of sensor modalities and modality combinations which calls for multi-modal fusion (Radu et al., 2018).

HAR Sensor Modalities

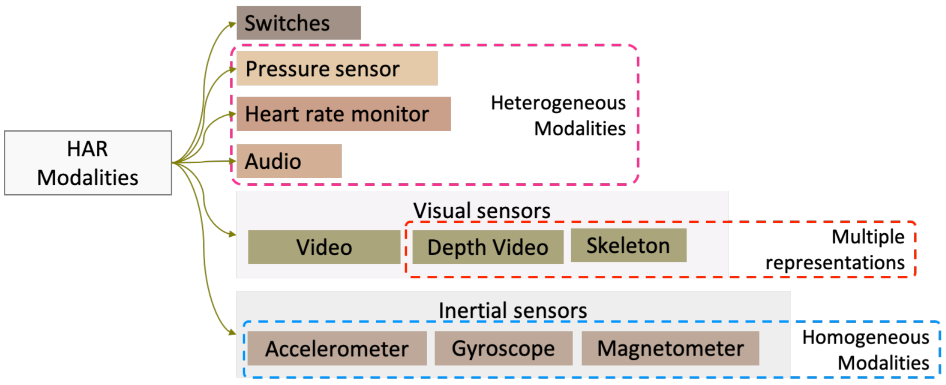


Figure 2.1: Sensor modalities used in human activity recognition

Figure 2.1 categorises the types of sensor modalities seen in HAR literature. There are two types of modality combinations: homogeneous modalities where every sensor generate the same type of data such as a set of inertial sensors placed on different parts of the body; and heterogeneous modalities where sensors generate different types of data. Research in HAR broadly cover the fusion of homogeneous sensor modalities (Ordóñez and Roggen, 2016; Radu et al., 2018; Yao et al., 2017). In contrast, domains such as video caption generation (Xu et al., 2017a), speech recognition (Ngiam et al., 2011) and emotion analysis (Chen and Jin, 2016; Kahou et al., 2016) often explore the fusion of heterogeneous modalities. In these domains, modalities considered are audio, video, motion, image, and text. Accordingly, this section investigates multi-modal fusion architectures from both HAR and other domains to identify implications and limitations that can be informative when designing a fusion architecture for ExRec.

While many use a single feature representation of the sensor modality for fusion, using more than one representation of the modality is found advantageous in literature. These feature representations are either derived from raw data (i.e. such as depth video and skeleton) (Das et al., 2019; Escalera et al., 2013; Ghosal et al., 2018; Gu et al., 2018) or obtained from multiple feature representation learners (Chen and Jin, 2016; Ghosal et al., 2018; Jiang et al., 2019). Das et al. (2019) presented a hybrid architecture for HAR that uses multiple modalities derived from an RGB-D camera data stream. RBG image data, skeleton data and video data are obtained as three independent modalities by applying data sampling and data transformation methods. Language classification architecture by Gu et al. (2018) is an example of applying multiple raw feature transformations methods on raw audio data to create multiple modalities.

Extracting multiple feature representations from deep feature learners is used in many application domains. Jiang et al. (2019) extract feature presentations at different levels of a multi-layer convolution architecture, as different representations of the depth camera modality. These representations are later aggregated to perform motion tracking. Ghosal et al. (2018) uses two variations of each modality, audio, video and text in the final modality fusion for sentiment classification. For each modality, first, a feature presentation is learned, and it is transformed using attention methods (more details in Section 2.3.1). Both feature representation and attended feature representation are used as independent modalities for fusion.

Multi-modal Fusion

The landscape of multi-modal fusion research can be categories into three design consideration: feature learning architectures, fusion methods and fusion levels. These categories are illustrated in Figure 2.2.

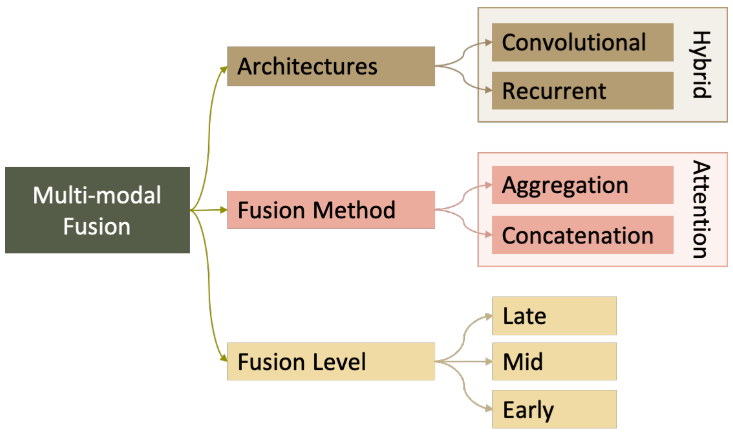
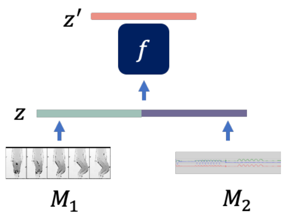


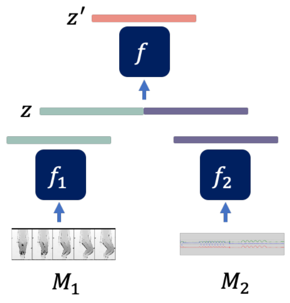
Figure 2.2: Design considerations of multi-modal fusion

The most common multi-modal fusion architectures are hybrids of convolution and recurrent constructs (Ordóñez and Roggen, 2016; Yao et al., 2017). Resulting architectures are deep and have high parametric complexity to capture both spatial and temporal patterns from sensor modalities. Convolutional neural networks have also been applied successfully to learn fusion feature presentations (Münzner et al., 2017; Sani et al., 2017a) when there are no significant temporal dependencies.

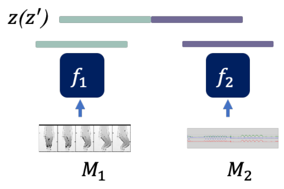
Fusion level determines at which stage the modality features are unified. We categorise levels found in literature as late, mid and early. Late fusion allows a model to learn modality-specific feature representations, thus preferred by heterogeneous modality combinations (Chen and Jin, 2016). In contrast, early and mid fusion are preferred by homogeneous modality combinations (Ordóñez and Roggen, 2016; Yao et al., 2017). We illustrate the three levels in Figures 2.3a, 2.3b and 2.3c using DC (M_1) and ACT (M_2) modalities in the MEx dataset.



(a) Early fusion



(b) Mid fusion



(c) Late fusion

Figure 2.3: Multi-modal fusion levels

Fusion methods are used to create a unified feature vector from multiple modalities, and concatenation is the most commonly used method. As illustrated in Figure 2.4, an important fusion method design consideration is along which axis the fusion is applied: temporal axis or feature axis. Applying fusion at the temporal axis is advantageous for early fusion methods where modalities are still independently presented to the feature learner (Ordóñez and Roggen, 2016). DeepConvLSTM (Ordóñez and Roggen, 2016) is a homogeneous multi-modal fusion architecture that applies early fusion at the temporal axis to recognise activities using inertial sensor data. Raw data instances are concatenated along the temporal axis as multiple convolutional channels to form the early fusion feature vector.

Applying early fusion on the feature axis can be detrimental where modality properties may get lost in detail. For instance, when using a heterogeneous modality combination, concatenating features from two modalities can hinder the features of one modality. Instead, fusion on the feature axis is found to be advantageous at a mid fusion level where modalities are already transformed in to feature representations (Yao et al., 2017). DeepSense (Yao et al., 2017) is a homogeneous multi-modal fusion architecture that applies mid-level fusion along the feature axis. First, independent, but identical convolutional feature learners are applied on each modality which allows the network to learn modality features independently. Resulting features at a given timestamp are concatenated to create the fusion feature vector.

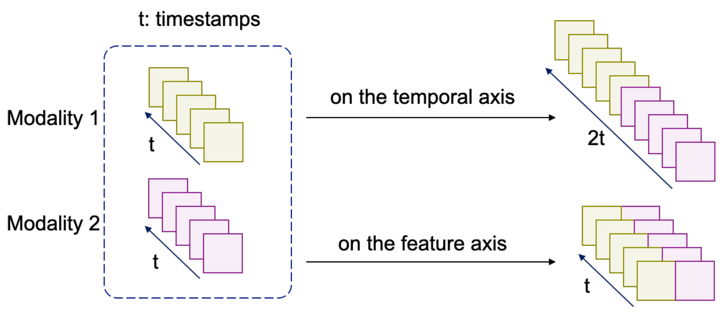


Figure 2.4: Modality fusion on the temporal axis vs. feature axis

Recent literature in multi-modal fusion applied attention to elevate modality fusion methods. Attention can be seen as a weighted fusion, and we discuss attention in detail on Section 2.3.1. QualityDeepSense (Yao et al., 2018), DeepFusion (Xue et al., 2019) and AttenSense (Ma et al., 2019) are predecessors of the DeepSense (Yao et al., 2017) architecture that exploit attention to improve fusion.

An alternative method of fusion is presented by Münzner et al. (2017) for homogeneous modalities. Instead of applying early fusion to create a unified representation of homogeneous modalities (like in DeepConvLSTM (Ordóñez and Roggen, 2016)), modalities are kept independent but applied the same convolutional filters. This method is similar to a metric learning architecture where multiple data instances are applied the same feature learner to create comparable representations (Bromley et al., 1994; Hoffer and Ailon, 2015). The feature representations obtained for each modality are concatenated at a late level resulting in a shallow architecture. This architecture benefits from using a shared convolutional feature learner to outperform conventional early and late fusion architectures, but the convolutional filter sharing method can only be applied to homogeneous modalities.

Temporal fusion is used in an architecture with recurrence at the last stage to learn temporal dependencies (Ma et al., 2019; Yao et al., 2017, 2018). Instead of using the output at the last timestamp T, temporal fusion applies an aggregation of all hidden states h_t, to obtain the recurrence output. Here, we view a hidden state, h_t, as a temporal modality, thus referred to as temporal fusion.

2.2.2 Personalised Recognition

In general, a HAR algorithm that is trained on a large population can be considered generalised to many users that may encounter after deployment (Longstaff et al., 2010). Although, it is intuitive that different users have their interpretation of activities and only the end-user can provide the ground truth of their activities (Miu et al., 2015). Accordingly, personalising an activity recognition algorithm using end-user data to personal characteristics such as gait patterns and physiology can improve recognition performance. Personalised HAR research has explored how reasoning models can obtain end-user data and how reasoning models can integrate end-user data.

We find that most early machine learning algorithms opt to only train with end-user data (i.e. person-dependent) to achieve personalisation. Berchtold et al. (2010); Tapia et al. (2007) and Wahl and Amft (2014) reported performance improvements of 39.3%, 19.0% and $20 \sim 25\%$ respectively with classification algorithms trained with end-user data over the same algorithms trained with person-independent data. Person-dependent evaluation demonstrates the capacity of end-user data to improve performance but infeasible to implement in practice due to the significant amount of labelled data required by today's state-of-the-art recognition models.

In more recent literature, active learning and online learning methods are used to obtain a limited amount of labelled data from the end-user data for personalisation (Gomes et al., 2012; Losing et al., 2019). Longstaff et al. (2010); Losing et al. (2019) and Gomes et al. (2012) integrate an active learning loop to obtain the labelled data and personalise a pre-trained recognition model. Longstaff et al. (2010) rebuilds a decision tree classifier using end-user data in conjunction with the training data to achieve significant performance improvements compared to the generic classifier. Gomes et al. (2012) uses the end-user data to personalise and to maintain personalisation over time by learning personal changes of the end-user.

An alternative approach is to have access to a set of classifiers (i.e. experts) trained for each user seen during training and access to their data. Online Multi-task Learning method by Sun et al. (2012) treats each train-user as a task. A classifier is learned for each user (i.e. an expert) in a multi-task setting, thus allowing to learn from other train-users. Alternatively, experts can be trained in a personal setting (Reiss and Stricker, 2013). When an end-user provides a set of labelled data instance for each activity class, it is used to determine the similarity (i.e. weight) of the end-user to train-users. This similarity knowledge is used to either aggregate experts and obtain a personalised classifier (Sun et al., 2012) or to obtain a prediction using an ensemble of experts using majority voting Reiss and Stricker (2013).

If we only have access to the data from train-users, the labelled data instances provided by the end-user are used to find similar train-users. Sani et al. (2017b) uses the labelled data provided by the end-user in conjunction with the labelled data from the most similar train-users to train a personalised convolutional classifier. Miu et al. (2015) avoids the similarity calculation, and simply add labelled end-user data to the training data set against which they perform k-nearest neighbour retrieval for class prediction. Accordingly, there is a mix of data from the general population and ones own data in the dataset to achieve a lazy personalisation.

Most recently, few-shot learning is adopted as an approach to personalisation by personalised Matching Networks (Sani et al., 2018; Vinyals et al., 2016). Matching Networks (MN) was introduced for few-shot classification by Vinyals et al. (2016). Architecturally it is a predecessor of Relation Networks (details in Section 2.4.2) where a static similarity function with non-parametric attention calculates the similarity distribution instead of a parametric regression model. Sani et al. (2018) introduced personalised Matching Networks (MN^p) to perform personalised HAR. MN^p learns a reasoning model, that is optimised to find the best match for a query instance when given a set of labelled data instances from the same person. At deployment, the network transfers the learning to new end-user given only few labelled data instances for matching. This method does not require access to train-persons data which is privacy-preserving compared to previous methods. It also avoids post-deployment model re-training which less burdensome on an edge device. MN^p training task design is similar to Sun et al. (2012) where a person is considered a task. Instead of a person creating a single task, many tasks are created for each person in the training data set, presenting many variants of the same person to the reasoning algorithm. A set of training tasks (500 in Sani et al. (2018)) are created per person, where each training task is comprised of a randomly selected support set and a single query instance. Accordingly, MN^p is trained as a conventional classifier with mini-batching and categorical cross-entropy loss.

Personalisation is explored in application domains other than HAR to improve user experience. Some examples are facial expression recognition for personalised gaming (Blom et al., 2014), pain level recognition from video (Thiam et al., 2017), emotion recognition from audio and video (Ramya and Bhatt, 2019) and personalised dialogue generation (Madotto et al., 2019).

Challenges in Personalised Recognition

Seemingly, all personalised HAR algorithms rely on having access to many or few labelled data instances of the end-user. Active learning and online learning methods have shown significant performance improvements by integrating few instances of end-user data after deployment. However, we identify few existing challenges highlighted in the literature.

One of the main challenges is the access to train-user data. Sani et al. (2017b); Sun et al. (2012) and Reiss and Stricker (2013) are methods that rely on identifying the similarity of the end-user to train-users using data. Algorithms that require access to train-users data can raise privacy concerns as well as storage limitations on a mobile platform. Recent advances of privacy preserving machine learning methods (McMahan et al., 2017) have highlighted the importance of distributed or federated machine learning instead of conventional machine learning to minimises the sharing of sensitive data.

Personalisation using pre-trained classifiers (Sani et al., 2018) eliminate the privacy concerns by sharing only the trained models instead of data. However, MN^p by Sani et al. (2018) followed a training methodology which results in a standard classifier instead of a meta-learner as intended in original MN. There is high variability within the tasks created from the same user, and the model has less opportunity to identify and isolate personal nuances to capture commonalities between users. In addition, this setting does not emulate the real-life environment where a person records a set of calibration data to use as the support set for all query instances. Improving MN^p is further investigated in Chapter 5.

The online active learning method used by Miu et al. (2015) requires complete model retraining every time labelled data is obtained from the end-user. This is in contrast to incremental learning algorithms that seamlessly learn from new data. In an active learning setting, the system asks the end-user for labelled data periodically, which results in retraining the reasoning model frequently. Computational and resource limitations of a mobile platforms makes it less desirable to re-train a model from scratch. In addition, it is also desirable to limit the interaction with the end-user to one or few instances to improve non-invasiveness.

Berchtold et al. (2010) highlighted the importance of obtaining end-user data in similar conditions to which it was seen during training, such as the orientation of the sensor. Their results report 20% decrease in performance when end-user data is obtained in different orientations. It is noteworthy that Berchtold et al. (2010) considered personalisation with accelerometer data where the orientation of the sensor can change over time. Issues in sensor orientation are less applicable with ambient modalities like pressure mat or depth camera where positioning of the sensors is fixed. But obtaining end-user data from an ambient sensor is more challenging compare to a wearable.

While many report significant performance improvements with personalisation, Longstaff et al. (2010) highlight that such improvements are only significant against weak baselines. Accordingly, it is recommended to evaluate algorithms not only against non-personalised counterparts but against the best deep learning algorithms for recognition.

2.2.3 Open-ended Recognition

Open-ended Human Activity Recognition (HAR) aims to create models that can recognise new activities encountered after deployment in addition to activities seen during training (Xian et al., 2018). Existing methods reported in literature fall under unsupervised and supervised approaches. Unsupervised methods such as clustering, by nature, do not rely on labelling, thus suited for open-ended recognition. Incremental updates to the clusters allow integration of new classes as instances are introduced after deployment. Often, heuristics such as cluster size and temporal thresholds are introduced to condition cluster creation and retirement in an online setting (Gjoreski and Roggen, 2017).

Open-ended recognition as a supervised learning problem often relies on semantic features to describe new, unseen classes (Lampert et al., 2014; Liu et al., 2011; Xu et al., 2017b). While there are no labelled data instances available during training for unseen classes, semantic features help to position unseen classes among seen classes during the training process. Such open-ended recognition algorithm is comprised of two modules: first, the input data is mapped to the semantic features; and secondly, the semantic features are mapped to activity labels (seen and unseen). Accordingly, the semantic features act as a bridge between the input data and the activity labels (Liu et al., 2011).

Hand-crafted intermediary features and their mapping to class labels (also known as attributes and attribute mapping) provided by an expert is the main form of intermediary semantic features for open-ended HAR (Cheng et al., 2013a,b; Liu et al., 2011). The mapping between the raw input data and the intermediary features can be a set of reasoning models, one for each intermediate feature, learned using the training data. Binary classifiers (Cheng et al., 2013b; Liu et al., 2011), multi-class classifiers (Ohashi et al., 2018) and regression models (Ohashi et al., 2018) are commonly used to learn the mappings. Once a set of intermediary features are predicted, the nearest neighbour algorithm is used to find the activity class that best matches the predicted features in the intermediary feature space (Cheng et al., 2013b; Ohashi et al., 2018).

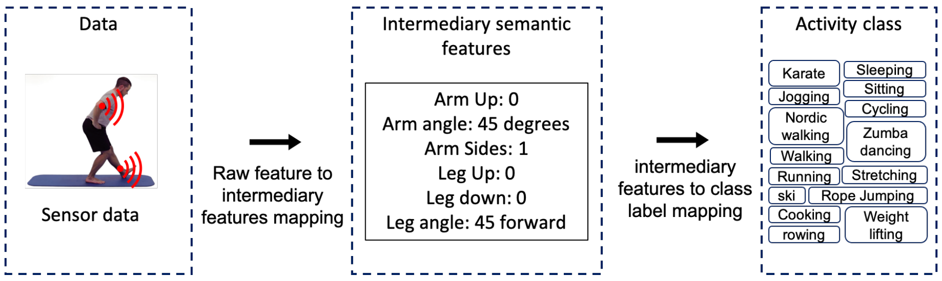


Figure 2.5: Open-ended activity recognition with an intermediary semantic features space

An example of open-ended HAR using semantic features is illustrated in Figure 2.5. Sensor data is transformed into six intermediary attributes, using 4 binary classifiers and two regression models. The resulting set of features are matched against the pre-defined attribute mapping to find the most similar combination, and the class label is obtained.

An attribute mapping for open-ended HAR encodes domain expert knowledge in which an activity class is described by a set of intermediate-level actions. For instance, Cheng et al. (2013b) describes chest-press exercise as a sequence of known action primitives, such as arms side, arms curl and arms forward. This method is further improved by Cheng et al. (2013a) to incorporate temporal aspects of activity sequences. Attribute importance is a method that has further enhanced the attribute mapping. Ohashi et al. (2018) applied weighted nearest neighbour retrieval to amplify relevant attributes for each activity class. A key advantage of attribute-based classification is the interpretability of the predictions. For instance, an activity label prediction can be explained by the set of intermediary attributes predicted using the raw sensor data.

An alternative method to hand-crafted features is the automated discovery of intermediary features through word-vector embedding. In Xu et al. (2017b) and Al Machot et al. (2020) all class labels (seen and unseen) are embedded using a pre-trained word embedding neural model to obtain the intermediary features. Accordingly, the intermediary features are an encoding of the class label. The mapping between the input data and the intermediary features is learned using the training data from seen classes as a regression task. For instance, a shallow neural network is used by Al Machot et al. (2020) to predict the word representation (Word2Vec) of the class label from sensor data. Moreover, similar to hand-crafted feature approach, the nearest neighbour algorithm is used to find the class that best matches the predicted class in the intermediary feature space.

An alternative approach to open-ended recognition is generative models such as Gradient Matching Generative Networks used in open-ended image classification. With recent advances in conditional generative models learn to generate high-quality synthetic data for unseen classes using the semantic features (Mirza and Osindero, 2014; Sariyildiz and Cinbis, 2019). Synthetic data instances in conjunction with training data are used in training a conventional classifier that performs conventional close-set classification.

Challenges in Open-ended Recognition

We identify a few key challenges with regards to semantic-feature based open-ended recognition methods. Curating a complete attribute mapping is a challenging task, and in practice, frequent manual updates by an expert are required to maintain completeness (Cheng et al., 2013b; Ohashi et al., 2018). Accordingly, we view this method as *knowledge-intensive*. In addition, a set of attributes should be able to represent a wide range of activities. Capturing temporal dependencies with attributes (Cheng et al., 2013a) and attribute importance (Ohashi et al., 2018) are methods that contribute towards improving this intermediary representations.

Although the semantic attribute approach provides a generalised solution to open-ended recognition, there are aspects in activity performance that is challenging to encode into an intermediary feature space. For instance, personal characteristics and intricate movements in complex activity classes are intuitively challenging to decompose into attributes. These limitations cause poor performance with complex activity types where individual variations are more prominent compared to ambulatory activities. Active learning is used as a method of personalisation for open-ended HAR (Cheng et al., 2013b). The end-user is asked to verify model predictions with low certainty and verified data instances are used to re-train both *input to intermediary feature* mapping and *intermediary features to activity class* mapping.

An automated approach to acquiring the intermediary features is comparatively less burdensome compared to hand-crafted features. The main drawback of this method is the black-box nature of intermediary features. Moreover, the input to intermediary feature mappings is learned by one or several reasoning models using training data from seen classes. Often, these reasoning models over-fit to seen classes and perform poorly on unseen classes. Geng et al. (2020) proposed to recognise seen and unseen classes using two separate classifiers to mitigate over-fitting. First, the model determines if a test data instance is part of the seen classes or unseen classes using a heuristic, then forward the data to the respective classifier. The success of this method relies on the accuracy of the heuristic.

We find generative models can also alleviate over-fitting to seen classes as synthetic data can be generated for both seen and unseen classes using semantic features. Although, the generated synthetic data are used to train a conventional close-set classifier which is restricting such that new classes cannot be added after model deployment.

Open-set recognition (Bendale and Boult, 2016) is a more generalised form of open-ended recognition where semantic features on unseen classes are not provided during model training. However for HAR applications, we argue that we have access some knowledge or data on unseen classes, during or after model development.

2.3 Attention Fusion

Attention is learning to highlight or *attend to* a subset of features towards rapid performance improvement. In this section, we explore the literature on Attention and Attention Fusion with a view to address the heterogeneous multi-modal fusion challenge.

2.3.1 Attention

Attention was first introduced in the domain of neural machine translation (NMT) (Bahdanau et al., 2014; Luong et al., 2015). Given an RNN encoder-decoder model for machine translation, attention mechanism was used to learn the alignment between words when a sentence is given in two languages.

Figure 2.6 shows a simple RNN encoder-decoder architecture (Cho et al., 2014b) that learns to translate a sentence from a source language to a target language. The RNN encoder hidden state, h^e_t, is derived using the previous hidden state and the input as in Equation 2.1 where f^e is a parametric model or a non-parametric activation function. The last encoder hidden state, h^e_T, is the summary, c, which encompasses the encoding of the source sentence, $\{x_1, x_2, ..., x_T\}$.

$$h^e_t = f^e(h^e_{t-1}, x_t) \tag{2.1}$$

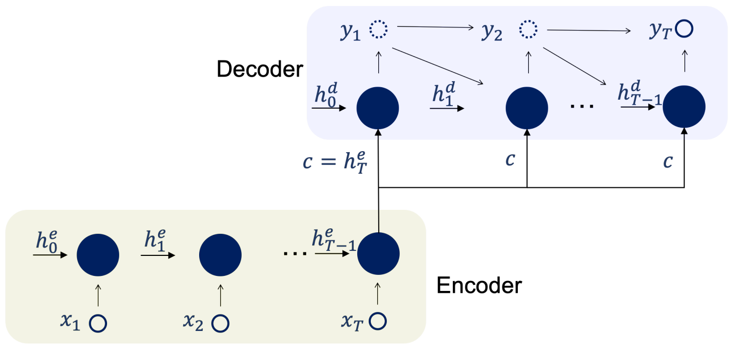


Figure 2.6: RNN encoder-decoder

The RNN decoder receives the summary, c at every timestamp, and used to derive the decoder hidden state and the decoder output as in Equation 2.2. Here f^d is similar to f^e and g is a parametric model with softmax activation for classification.

$$\begin{aligned} h_t^d &= f^d(h_{t-1}^d, y_{t-1}, c) \\ y_t &= g(h_t^d, y_{t-1}, c) \end{aligned} \tag{2.2}$$

Instead of using the last hidden state of the encoder, h_T^e, as the encoder summary, c, Bahdanau et al. (2014) proposed to learn a weighted aggregation of all encoder hidden states. Here, attention weights, α_{tj} are learned to indicate how much each encoder hidden state h_j^e should contributes to the summary c_t, at decoder timestamp t (Equation 2.3).

$$\begin{aligned} c_t &= \sum_{j=1}^{T} \alpha_{tj} h_j^e \\ y_t &= g(h_t^d, y_{t-1}, c_t) \end{aligned} \tag{2.3}$$

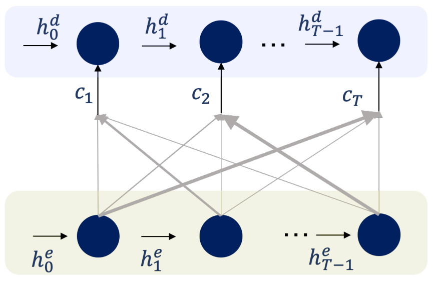


Figure 2.7: RNN encoder-decoder augmented with attention

Consequently, the attention mechanism learned how a target word (y_t) aligns to each word in the source sentence. In Figure 2.7 we augment the RNN encoder-decoder from Figure 2.7 with attention. After training, the state of the attention vector (α_{tj}) at each decoder output is illustrated with arrows of different widths to indicate the attention weight. For instance, the second output is most influenced by the last input and the last output is most influenced by the first input. Attention mechanism has further inspired self-attention, where, given a sentence, each word learns the alignment against other words in the same sentence (Vaswani et al., 2017). For instance through self-attention a model learns how a noun is referenced by a pronoun in the same sentence.

2.3.2 Attention Fusion

Multi-modal fusion research in many domains adapted the concept of weighted aggregation to learn effectively from multiple modalities. Attention mechanism for multi-modal fusion learns an effective selection of significant features to highlight and noisy features to discard. Two forms of attention are found in multi-modal fusion literature: learning modality significance given a set of modalities is referred to as modality attention fusion; and learning feature significance of a modality is referred to as feature attention.

Modality Attention Fusion

Most commonly, a modality is considered analogous to a word in the source sentence, and one hot encoded classification output to the target sentence. Accordingly, given a set of feature vectors, $\{x_i\}$, for m number of modalities, attention fusion feature vector, z', is derived as in Equation 2.4. Here α_i are the attention weights that indicate the significance of each modality. Modality fusion in Zhang et al. (2018), AttenSense (Ma et al., 2019) and Gu et al. (2018) are recent examples of applying modality attention fusion. Importantly, all the features of a modality are applied the same weight.

$$z' = \{\alpha_i x_i \mid 0 < i < m\} \tag{2.4}$$

Attention weights can be derived in a parametric manner or a non-parametric manner. AttenSense (Ma et al., 2019) and DeepFusion (Xue et al., 2019) are using a single dense layer parametric model to learn the attention weights from the original modality features. QualityDeepSense (Yao et al., 2018) and Jiang et al. (2019) instead exploit a non-parametric method where original modality features are normalised to obtain attention weights. A parametric model stochastically transforms the input features to attention weights using the parameters of the network. Alternatively, a non-parametric method, applies an algebraic transformation to derive attention weights from modality features.

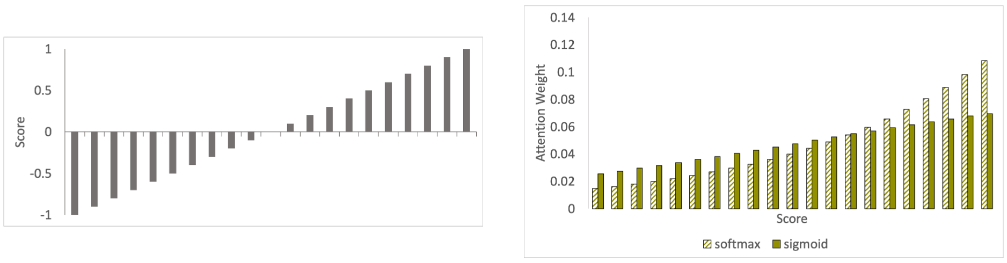
As in Equation 2.4, attention weights are used to augment modality feature vectors, $\{x_i\}$ and unified to obtain the fusion feature vector, z'. Fusion feature vector can be one of two forms: aggregated or concatenated. Aggregation as in Equation 2.5 results in a feature vector z', where $|z'| = |x_i|$. Concatenation results in a feature vector, z', where $|z'| = \sum_i^m |x_i|$. AttenSense (Ma et al., 2019) and the weighted-combination module in DeepFusion (Xue et al., 2019) use aggregation to form the fusion feature vector. Late fusion on the DeepFusion (Xue et al., 2019) architecture, Zhang et al. (2018) and Gu et al. (2018) are examples where concatenation forms the fusion feature vector (Equation 2.6).

$$z' = \frac{1}{m} \sum_i^m \alpha_i x_i \tag{2.5}$$

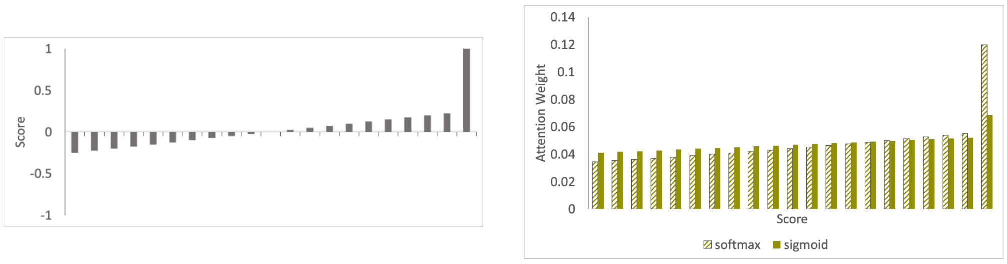
$$z' = concat(\alpha_1 x_1, \alpha_2 x_2, ..., \alpha_m x_m) \tag{2.6}$$

Aggregation requires each modality to produce a feature vector of the same length and aggregates features from multiple modalities by their feature vector index. This method is sound when using a shared feature learner (Münzner et al., 2017) or in the homogeneous multi-modal setting. However, it is detrimental in a heterogeneous multi-modal setting. Different modalities use modality-specific feature representation methods where indices may not co-relate between feature vectors. It can be argued even in an acceptable setting, the essential features may get lost in aggregation. Concatenation does not introduce a feature-length constraint, also allows the flexibility to use different feature representation learners. Although, resulting feature vector can be excessively large in a setting with many modalities.

The set of attention weights α are often normalised using either softmax or sigmoid functions. Softmax normalisation method used in AttenSense (Ma et al., 2019), Zhang et al. (2018) and Gu et al. (2018) skew the attention weights to significantly highlight one modality over the others. Accordingly, it is referred to as Hard Attention (HA). Notably, $softmax$ is a stochastic transformation, where each weight is dependent on the other weights. The weighted-combination module in DeepFusion (Xue et al., 2019) instead uses sigmoid normalisation where the attention weights are less skewed, highlighting



(a) Normally distributed score vector normalised using softmax and sigmoid functions



(b) Skewed score vector normalised using softmax and sigmoid functions

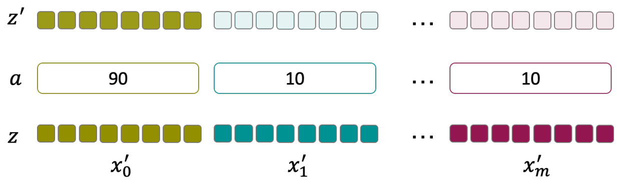
Figure 2.8: Score vector normalisation using softmax and sigmoid functions

more than one modality in the final fusion feature vector. Accordingly, it is also referred to as Soft Attention (SA). Notably, *sigmoid* is a deterministic transformation where each weight is independent of other weights.

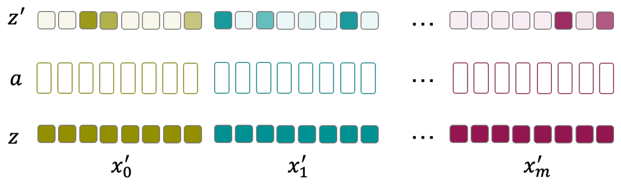
In modality attention fusion, skewed normalisation is beneficial when learning to select one modality from a set for a classification task. In contrast, non-skewed normalisation learns patterns of modality combinations for classification. In Figure 2.8 we plot examples of the two normalisation functions normalising attention weights. Imagine a vector of 21 values dispersed between -1 and 1. In Figure 2.8a the values are dispersed evenly, and both sigmoid and softmax transform original values to a 0-1 range reflecting a similar distribution to the original values. In Figure 2.8b, the original values are not dispersed evenly. Instead, there is a weight that is a significant outlier. While sigmoid normalisation transforms with a normal distribution, softmax transformation is skewed such that the positive discriminatory feature is further highlighted. As a result, HA accentuates positive values significantly such that only one or few features are highlighted.

An alternative attention method introduced by Hori et al. (2018) learn feature importance instead of modality importance for modality fusion. Given a set of feature vectors, $\{x_i\}$ each of length k, from m modalities, a set of parametric attention weights, α_j are learned for each feature index, j $(0 < j < k)$. This is in contrast to learning an attention weight for each modality. α^j contains m attention weights, one for each corresponding feature (j^{th} feature) of a modality. Accordingly, the size of α is $m \times k$. Hori et al. (2018) also exploits any correlation that may exist between modalities when aligned by feature index.

Modality importance and feature importance can be viewed as the granularity at which the attention applied. We illustrate the two methods in Figures 2.9b and 2.9a. Modality level granularity calculates an attention weight for each modality. In contrast, feature level granularity calculates an attention weight for each feature of each modality.



(a) Modality level



(b) Feature level

Figure 2.9: Attention at feature level vs. modality level granularity

Feature Attention

Feature attention learns an enhanced feature representation of an individual modality, either in a multi-modal or a single modal setting. In a multi-modal setting, learning feature attention can be either independent or against other modalities. Language classifier presented by Gu et al. (2018) details learning feature attention using a parametric model, for text and audio modalities. If the modality feature vector is x, a set of attention weights α is learned where each feature is assigned an attention weight (i.e. $\mid \alpha \mid = \mid x \mid$). Resulting feature vector x' is augmented to highlight features that are significant and hinder features that a noisy for the classification task.

Learning an attended feature representation in a multi-modal setting with respect to one or a few other modalities is referred to as intra-attention in literature. The cross-sensor module in DeepFusion (Xue et al., 2019), UHAN architecture (Zhang et al., 2018) and Jiang et al. (2019) are examples of intra-attention in a multi-modal setting. In DeepFusion (Xue et al., 2019), feature attention weights for each modality are learned using the correlations to other modalities. The aim is to situate a modality with respect to other modalities, as such self-correlation is omitted from the attention learning. It is beneficial to learn intra-attention for a modality pair that is in natural alignment with each other (Ghosal et al., 2018). Sentiment analysis architecture by Ghosal et al. (2018) use audio, video and text modalities where bi-modal feature attention is learned for modality pairs: (Audio, Video); (Audio, Text); and (Video, Text). However, in HAR, reasoning with modalities that naturally align with each other can be seen as a redundancy, given the goal is to minimise the number of modalities to improve usability.

$$y_T = \sum_{t}^{T} \alpha_t h_t \tag{2.7}$$

The idea of weighted aggregation can be applied to temporal fusion where a recurrent model is used for class prediction. Here, each output (from each timestamp) is considered a modality and as in Equation 2.7, α_t are the attention weights that aggregate temporal outputs. AttenSense (Ma et al., 2019) use temporal attention fusion to aggregate the hidden states of their last GRU layer using a parametric model. In contrast, QualityDeepSense architecture (Yao et al., 2018) use a non-parametric approach to derive temporal attention weights that encompass intra-dependencies between timestamps.

2.4 Few-shot Learning

Few-shot learning is a multi-class classification setting where there is a large number of classes with only few labelled examples for each class. The goal of a few-shot classifier is to successfully learn class boundaries for any given subset of classes using the few-data available. A few-shot classifier is commonly described as k-shot n-way where k indicates the number of labelled data available per class and n indicate the size of the subset of classes.

Creating a decision layer with many classes is feasible when there is an abundance of labelled data to discriminate one from all other classes in the feature space. Instead, in a few-shot setting, it is more effective to create smaller decision boundaries. The $k \times n$ number of labelled data from the n number of classes are mapped to a feature space where the boundaries are separated. Often in practice, only a subset of classes can

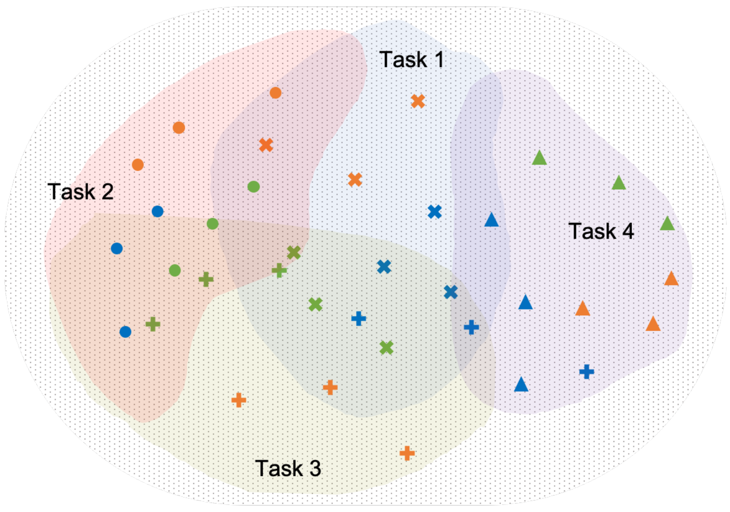


Figure 2.10: Few-shot learning setting

occur at a given time, thus may not need to learn a classifier for all classes. Unarguably, in a few-shot setting, there are similarities between these subsets of classes, that can be advantageous when learned together, rather than apart as individual classifiers. A subset of classes that occur together naturally is referred to as a task, and the task distributions can be illustrated as in Figure 2.10. Similar to a conventional classification task, a few-shot task has training and test data sets where the training set consist of $k \times n$ labelled data instances. The state-of-the-art approach to implement a few-shot classifier is meta-learning (Oreshkin et al., 2018).

2.4.1 Meta-Learning

Meta-Learning is the learning of a model that is generalised across many tasks and is rapidly adaptable to any new task, thus referred to as *learning-to-learn.* Given a dataset D, in the conventional setting, *learning* is referred to optimising a parametric model θ using training data such that θ perform well on test data. *Learning to learn* instead consider the meta-dataset, $\mathcal{D}$, where each instance is a dataset, D, representing a few-shot task. A meta-model θ is optimised on a set of training tasks to perform well on a set of test tasks.

Historically, there are three types of meta-learners, considering the type of information used from prior tasks to adapt to a new task (Vanschoren, 2019). Early methods exploit the availability of data from prior tasks to learn a new model. Given access to data from prior tasks (i.e. meta-data), a model for a new task is learned by aggregating data from prior tasks and the new task. In a setting where meta-features are available instead of data of prior tasks, the similarity between prior tasks and the new task is calculated using meta-features. The similarities are used to select the most similar prior tasks and create a model for the new task. More recently, with the advances in learning parametric models, a generic meta-model learned for a set of prior tasks is used to transfer to a new task (i.e. exploit meta-model instead of meta-data or meta-features). Essentially, meta-learning with neural networks is learning a feature space from prior tasks that is rapidly adaptable to a new task. Here, the feature space is parameterised by a neural network model and followed by a decision layer. In this section, we explore the scope of meta-learners for few-shot classification using neural networks.

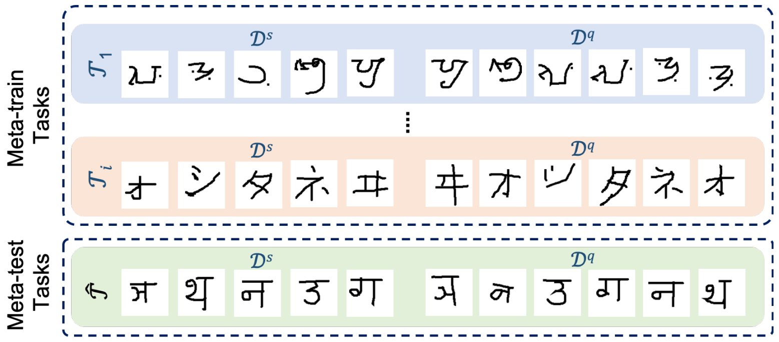


Figure 2.11: Meta-learning task design

To reiterate, meta-learning for few-shot classification is the learning of a meta-model that is generalised over many few-shot classification tasks, and rapidly adaptable to any new few-shot classification task. More formally, a few-shot classification task, $\mathcal{T}$ has a set of train data instances and a set of test data instances, referred to as the *support set*, $\mathcal{D}^s$, and the *query set*, $\mathcal{D}^q$. Here, the number of instances in a support set is $k^s \times n$. For example, for the character recognition task in Figure 2.11, a language forms a task, and the support set contains distinct characters, each of which is a representative of a class (i.e. $k^s = 1$). The query set, $\mathcal{D}^q$, is the set of test data of $\mathcal{T}$, which has no overlap with the support set, $\mathcal{D}^s$. This is similar to a train/test split in a classification task.

Implementation of a meta-learner for few-shot classification is viewed using the universal machine learning principle, *test and train conditions must match* (Vinyals et al., 2016). The goal of meta-learner is to learn a model that solves a few-shot classification task not seen during training. Accordingly, the test condition is identified as classifying any query instance, $\hat{x}_i^q$ of the test task $\hat{\mathcal{T}}$ utilising its support set, $\hat{\mathcal{D}}^s$. To match the test condition, the training data or meta-train tasks are designed as in Figure 2.11 where each training data instance is a few-shot classification task. During meta-model training, a parametric model learns to solve many meta-train tasks. That is, given a task, learn to classify a query data instance, x_i^q utilising its support set, $\mathcal{D}^s$. In comparison to a conventional classifier, a train instance is a few-shot classification task, and a test instance is a few-shot classification task not see during training.

We view the modern landscape of meta-learners for few-shot classification in Figure 2.12. Mainly there are three approaches, based on how the meta-learner is optimised to learn from many tasks: model-optimised; similarity-optimised; and adaptation-optimised.

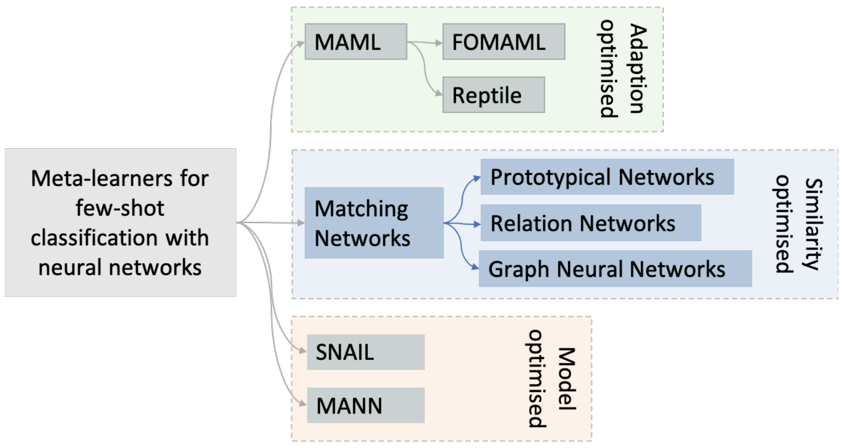


Figure 2.12: Meta-learners for few-shot classification with neural networks

Model-optimised Meta-learners

Model optimised meta-learners such as SNAIL (Mishra et al., 2018) and MANN (Santoro et al., 2016) create unique neural architectures to learn from prior tasks. Given a query instance and the support set, both methods sequentially arrange the support set instances as part of the input of size $(k^s \times n) + 1$ where the last timestamp is reserved for the query instance. The model is optimised to predict the class of the query instance with respect to the support set instances, and their class labels. An abstract representation of a model-optimised meta-learner is depicted in Figure 2.13 where we emphasise the sequential formation of the input. Accordingly, a model-optimised meta-learner learn a new task by using the support set in conjunction with the query instance in the input. Furthermore, the model is generalised over many tasks using iterative optimisation.

SNAIL (Mishra et al., 2018) uses a temporal convolution architecture (Bai et al., 2018) to encode the support set and attend accordingly using self-attention to predict the class of

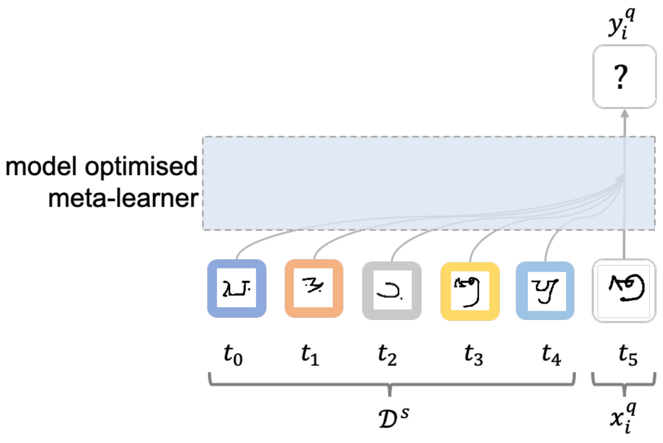


Figure 2.13: Model-optimised meta-learner

the query instance. MANN (Santoro et al., 2016) instead creates an external memory in which it remembers the support set encodings. Mishra et al. (2018) has highlighted the necessity of deep feature learners for SNAIL, that would otherwise over-fit to few data in a generic deep learning setting. Accordingly, SNAIL is better suited for few-shot tasks with complex data types such as image and text that benefit from deep feature learners. In the HAR domain, model-optimised meta-learners are used for human motion prediction in a few-shot setting. MoPredNet by Zang et al. (2020) uses a temporal convolution architecture similar to SNAIL for motion prediction from 3D skeletal data. Given the complexity of 3D skeletal data, MoPredNet benefits from learning feature representations from deep feature learners.

Similarity-optimised Meta-learners

Matching Networks (MN) (Vinyals et al., 2016) and its predecessors Prototypical Networks (PN) (Snell et al., 2017) and Relation Networks (RN) (Sung et al., 2018) are examples of meta-learners optimised for similarity. A similarity-optimised meta-learner learns to predict a class label for a query instance based on its similarity to each support set instance. Given a query instance (from the query set) and the support set, first, each element is applied a feature transformation using a feature learner. Then, pairs are created by pairing the query instance feature vector with each support set element feature vectors. The objective is to learn the feature learner parameters such that the similarity between a matched pair (i.e. a query instance from class A and a support set instance from class A) is maximised, and similarity between any unmatched pair (i.e. a query instance from class A and a support set instance from class B) is minimised. Accordingly, after iterative optimisation, a similarity-optimised meta-learner learns a feature learner that produce feature pairings for any few-shot learning task to predict class labels. Figure 2.14 illustrates an abstract representation of a similarity-optimised meta-learner. We highlight the paired formation of the input compared to the sequential formation seen in model-optimised meta-learners.

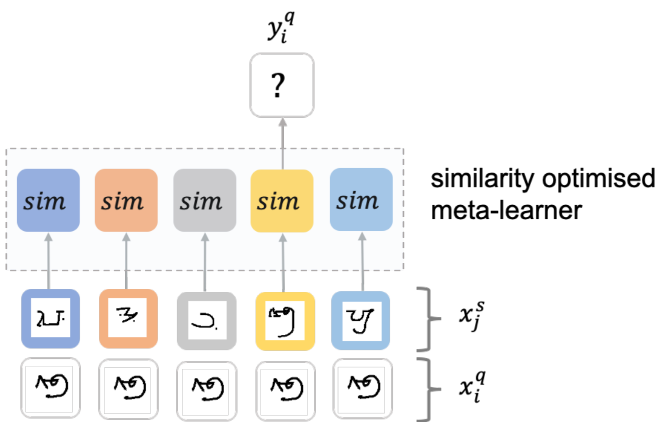


Figure 2.14: Similarity optimised meta-learner

With MN and PN, the similarity between a feature representation pair is simply calculated using a similarity metric like Euclidean distance or cosine similarity. Graph Neural Networks (GNN) (Satorras and Estrach, 2018) and RN (Sung et al., 2018) further parameterise the similarity calculation by using parametric models. Accordingly, optimisation of MN or PN, training only update network parameters of the feature learner. With GNN or RN, back-propagation is applied end-to-end to optimise the feature learner and the similarity learner. In comparison to model-optimised meta-learners (also known as black-box meta-learners (Hospedales et al., 2020)), similarity-optimised meta-learners is considered to be more interpretable. They can be also viewed as parametric k-nearest neighbour algorithms.

In the HAR domain, MN is adapted for personalised HAR by Sani et al. (2018). However, their training approach resembles a conventional classifier training than a meta-learner training which we investigated in Section 2.2.2. Multi-scale RN is an adaptation of RN for motion tracking by Ding et al. (2019). They extract features from different levels of a multi-layer convolutional feature learner to form the multi-scale input for similarity learning.

Adaptation-optimised Meta-learners

The goal of an adaptation optimised meta-learner is to learn a model that is most adaptable to any new unseen task. LSTM meta-learner (Ravi and Larochelle, 2017), MAML (Finn et al., 2017) and its predecessors FOMAML (Finn et al., 2017) and Reptile (Nichol et al., 2018) are example adaptation-optimised meta-learners from the literature.

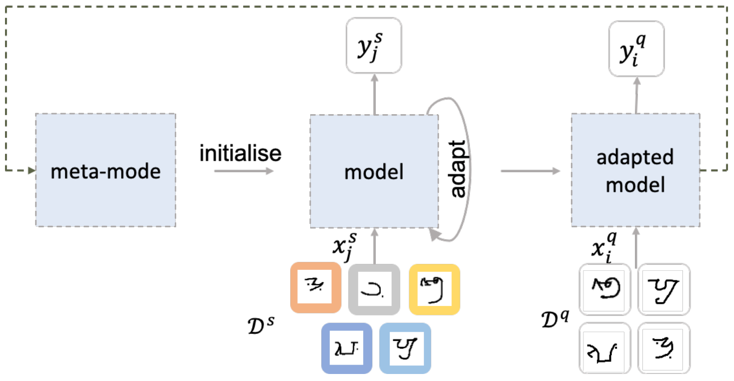


Figure 2.15: Adaptation optimised meta-learner

The goal of training the meta-model is to encapsulate the learning experiences from many training tasks such that it is the best initial model for any test task. Accordingly, at a given iteration, a set of task-specific models are created and initialised by the meta-model. They are then independently optimised using the task support set (similar to test condition described above). After adaptation, the learning experience of each adapted model is measured and aggregated to update the meta-model such that it represents a generalised view of many tasks. Importantly, the meta-model update is a gradient descent optimisation, using the aggregated loss from adapted models. This is instead of using a dedicated dataset for meta-model training. This training process ensures that the aggregated learning experiences of the selected optimised task-models do not disconcert the meta-model. We visualise an abstract view of a adaptation optimised meta-learner task adaption in Figure 2.15. The dotted line indicates a training task, where the learning experience (loss in MAML) of the task is fed into meta-model optimisation.

LSTM meta-learner (Ravi and Larochelle, 2017) is an early adaptation-optimised meta-learner where the meta-learning is inspired by the LSTM architecture. LSTM meta-learner aggregates the learning experiences of task-specific models by learning to adapt the learning rate using a parametric learner (i.e. meta-model). The parameters of the learning rate are updated considering past learning rate and past performance of the model. MAML aggregate the learning experiences of task-specific models by calculating the loss against their query sets. The collective losses are used to perform gradient descent on the meta-model. In comparison to LSTM meta-learner, where the meta-model only parameterises the learning rate, MAML maintains a meta-view of task-specific model weights. Reptile (Nichol et al., 2018) algorithm further simplify MAML and does not optimise the meta-model using GD. Instead the meta-model is an aggregation of optimised task-specific model weights and compared to MAML, Reptile achieves comparable performance. Reptile can be seen as the non weighted equivalent of the FedAvg algorithm introduced for privacy-preserving federated learning (McMahan et al., 2017). In the HAR domain, similar to other meta-learner approaches, human motion prediction is attempted with a variant of MAML in Gui et al. (2018). Here the MAML meta-model learns to predict future activities based on past activities using only few data instances.

2.4.2 Meta-learning Algorithms

This section investigates two meta-learning algorithms in detail: adaptation-optimised meta-learner MAML and similarity-optimised meta-learner RN.

MAML

MAML (Finn et al., 2017) is an adaptation-optimised meta-learner applicable to any parametric model optimised with Gradient Descent (GD). At a high level, MAML iteratively learns a meta-model generalised over many tasks such that the meta-model is rapidly adaptable to new unseen tasks (Figure 2.16).

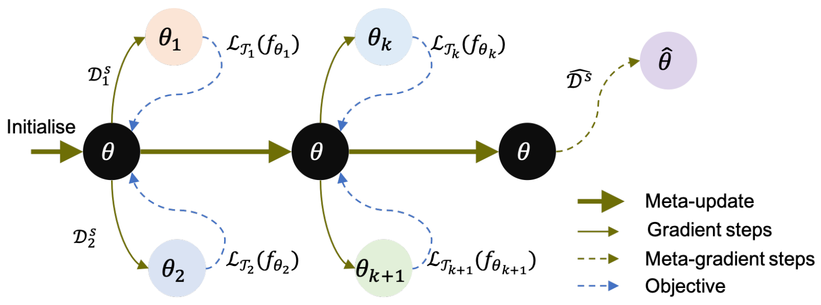

Initialise
θ
θ_1
θ_2
θ_k
θ_{k+1}
$\hat{\theta}$
$\mathcal{D}_1^s$
$\mathcal{D}_2^s$
$\widehat{\mathcal{D}}^s$
$\mathcal{L}_{\mathcal{T}_1}(f_{\theta_1})$
$\mathcal{L}_{\mathcal{T}_2}(f_{\theta_2})$
$\mathcal{L}_{\mathcal{T}_k}(f_{\theta_k})$
$\mathcal{L}_{\mathcal{T}_{k+1}}(f_{\theta_{k+1}})$
Meta-update
Gradient steps
Meta-gradient steps
Objective

Figure 2.16: Model Agnostic Meta Learner

First, the meta-model θ, is randomly initialised. At each iteration, a set of tasks are sampled from the meta-train set. For each task, $\mathcal{T}_i$, a support set and a query set is selected according to the meta-learning task design discussed in Section 2.4.1. The support set is used to train a task model, θ_i, initialised by the meta-model, θ. θ_i training is performed over gs number of training epochs referred to as the *gradient steps* using a learning rate of α. A gradient step includes computing the loss of θ_i against the support set and using gradient descent to update θ_i parameters. For multi-class classification (such as HAR), categorical cross-entropy is the preferred loss metric and is calculated as in Equation 2.8 against $\mathcal{D}^s$. At the end of the gradient steps, θ_i is now adapted for $\mathcal{T}_i$ using its $\mathcal{D}^s$.

$$\mathcal{L}_{\mathcal{T}_i}(\theta_i) = \sum_{x^s, y^s \sim \mathcal{D}^s} y^s \log \theta_i(x^s) + (1 - y^s) \log(1 - \theta_i(x^s)) \tag{2.8}$$

A task query set, $\mathcal{D}^q$, is formed as the test set to evaluate the classification task learned by θ_i which is often disjoint from $\mathcal{D}^s$. The loss of a task model, θ_i, is calculated using $\mathcal{D}^q$, which we view as the learning experience of θ_i. The collective loss from the set of selected tasks is used as the loss on which the meta-model is trained. Similar to training, categorical cross-entropy is the preferred loss and is calculated as in Equation 2.8, now against $\mathcal{D}^q$. A learning rate of β is used to update the meta-model parameters θ in a single step referred to as *meta-update*. This process of *meta-update* is iterated over many meta-train task samplings towards minimising the collective loss from tasks.

Meta-model optimisation for an unseen meta-test task $\hat{\mathcal{T}}$ is similar to creating a task model during training. A model, $\hat{\theta}$ is initialised from the current meta-model and is trained using the support set, $\hat{\mathcal{D}}^s$ of $\hat{\mathcal{T}}$. The training is performed in few iterations, referred to as the *meta-gradient steps*. Once $\hat{\theta}$ is optimised, it is used to predict class labels of the query set, $\hat{\mathcal{D}}^q$. Parameter updates during gradient steps, meta-update and meta-gradient steps are performed using gradient descent. Intuitively, mini-batching is not used in the few-shot learning task setting, where there are only few data instances for training and in practice implemented using an optimiser such as Adam or Adagrad.

Relation Networks

Relation Network (RN) (Sung et al., 2018) is a similarity-optimised meta-learning algorithm that *learns-to-match*. RN has a similar goal to other meta-learners, of learning a generalised model over many tasks and as a similarity-optimised meta-learner, RN is learning to find the best match for a query from a set of anchors (i.e. support set). The network learns to predict the similarity of a query against each anchor and the pair with the highest similarity is selected for class prediction. Instead of learning the probability distribution across possible class labels using a conventional classifier, RN learns a similarity distribution across anchors.

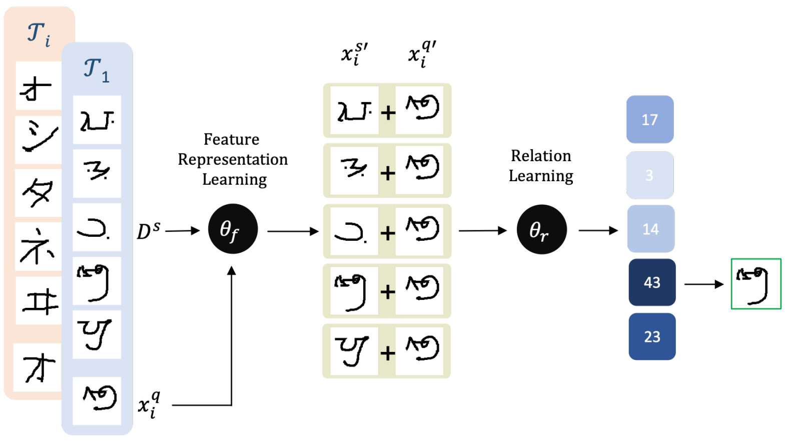


Figure 2.17: Relation Networks

There are two parametric modules to RN as shown in Figure 2.17: first to learn feature representations, θ_f; and second to predict the similarity between a pair of data instances, θ_r. First, both networks are randomly initialised. A task, $\mathcal{T}_i$ is formed with a support set, $\mathcal{D}^s$ and a query set, $\mathcal{D}^q$ as in detailed in Section 2.4.1. Next, a set of training data instances are created from $\mathcal{T}_i$, by pairing each query instance, x_i^q, with the support set. Given a training data instance, $(x_i^q, \mathcal{D}^s)$, each data instance in the training instances is encoded with θ_f. Suppose the support set has more than one representative from an activity class (i.e. $k^s > 1$). In that case, prototypical representatives for each class is created by calculating the mean of all x_j^s that belongs to the class (similar to prototypical networks by Snell et al. (2017)). The size of the resulting feature transformed support set is $|\mathcal{C}|$. A $|\mathcal{C}|$ number of pairs are created by pairing each support set instance, x_j^s with the query instance, x_i^q. Given a concatenated pair is (x_i^q, x_j^s), θ_r predicts the relation score (i.e a scalar value) as in Equation 2.9.

$$\text{Relation Score: } r_{ij}^{qs} = \theta_r(x_i^q, x_j^s) \ : \ r_{ij}^{qs} \in \mathbb{R}^1 \tag{2.9}$$

A $|\mathcal{C}|$ number of relation scores are obtained for a single training instance, and the goal of the network is to learn the parameters of θ_f and θ_r such that the highest relation score belongs to the matching pair (x_i^q, x_j^s) (Equation 2.10).

$$y_i^{q'} = \arg\max_{|\mathcal{C}|} \ r_{ij}^{qs} \tag{2.10}$$

The expected relation scores are 1 for the matching pair and 0 for non-matching pairs, and the objective is to minimise the difference between the predicted score and the expected score. Originally, it is calculated using the mean squared error as in Equation 2.11 (Sung et al., 2018).

$$\mathcal{L}_{\mathcal{T}_i}(\theta_f, \theta_r) = \sum_{x^q, y^q \sim \mathcal{D}^q} \parallel y^{q'} - y^q \parallel_2^2 \tag{2.11}$$

In summary, both MAML and RN iteratively optimise meta-models to capture commonalities between many task such that the resulting meta-model is the best starter model for an unknown task. In Chapter 5 we exploit this property of meta-learning algorithms. The optimisation is viewed as a personalisation where the resulting model is personalised to a new and unseen person.

2.4.3 Zero-shot Learning with Meta-learners

Zero-shot learning (ZSL) can be seen as a natural extension to few-shot learning where a model at deployment can successfully predict previously unseen class. Similar to open-ended recognition formalised in Section 2.2.3, zero-shot setting also rely on intermediary semantic knowledge to be aware of unseen classes. At test time, the model is tested for either only unseen classes (ZSL) or both seen and unseen classes (generalised ZSL). Similar to few-shot learning, zero-shot learning focuses on learning to discriminate between any given subset of classes. Accordingly, zero-shot learning is described as 0-shot, n-way. Accordingly, the goal of a zero-shot learner is to predict the class for a query instance from a subset of classes of size n, that it has not seen during training.

Looking at the meta-learner landscape, Prototypical Networks and Relation Networks have explored the integration of intermediary semantic knowledge to implement the zero-shot setting (Snell et al., 2017; Sung et al., 2018). Both similarity-optimised meta-learners use intermediary features to represent a prototypical instance of each seen and unseen class during training. Accordingly, the feature learners transform the input data into intermediary features, and the similarity score is predicted for a pair of intermediary features. Similar to the generative approach for open-ended recognition (Mirza and Osindero, 2014; Sariyildiz and Cinbis, 2019), an alternate approach to zero-shot learning is generative meta-learners. In Verma et al. (2020), data instances are synthesised for unseen classes using a conditional generative model and used in learning a meta-learner that perform few-shot learning.

It is challenging to extend adaptation-optimised and model-optimised meta-learners from zero-shot to open-ended recognition due to the use of conventional classifier layer with a fixed class length. For instance, $MAML$ is restricted to performing multi-class classification with a conventional fixed-size soft-max layer. Open-ended recognition requires dynamic expansion of the decision layer as new unseen classes are added in addition to the seen classes. Similarity-optimised meta-learners such as MN and RN instead have a similarity distribution at the decision layer with the potential to extend to open-ended setting. We will investigate this further in Chapter 6.

2.5 Conclusions from the Literature

We summarise the findings from the literature, highlighting the main research items that inspire and influence our methods in Chapters 4, 5, 6 and 7.

During the exploration of literature for modality fusion, we highlighted the importance of selecting the suitable fusion levels and fusion axis to learn from modality combinations. The use of early and mid fusion in the literature (Ordóñez and Roggen, 2016; Yao et al., 2017) raised the question of its suitability for the fusion of heterogeneous modalities. Ordóñez and Roggen (2016) explored an early fusion along the temporal axis that may preserve unique modality features of heterogeneous modalities. Instead we explore a late fusion setting similar to Münzner et al. (2017) in Chapter 4. We exploit the advances of deep feature learners to learn from each modality independently and create a modular architecture that is loosely coupled such that it is adaptable to many modality combinations. Finally, we exploit the advances of attention fusion from Section 2.3.2 to reduce the parametric complexity of the fusion architecture. Notably, we draw inspirations from Hori et al. (2018) to learn feature level attention instead of the more common modality level attention to learn modality and feature combinations unique to each activity.

Recent literature on personalised recognition highlight two main approaches: active learning (Bleser et al., 2015; Losing et al., 2019); and multi-task learning Sani et al. (2018); Sun et al. (2012). We take forward the multi-task approach, where each person is considered an independent task. Here the algorithms learn to create a personalised model for each person while learning abstract activity characteristics from multiple persons. Compared to an active learning approach, this approach minimises the need for model re-training and end-user interactions. Furthermore, Sani et al. (2018) minimised the data requirements by using a few-shot learning approach to learn from multiple persons. In Chapter 5, we exploit the most recent advances of meta-learning for few-shot learning to further improve methods introduced by Sani et al. (2018). The characteristics of the three main meta-learning approaches in literature are investigated in Section 2.4.1 to evaluate their suitability in the personalised HAR domain. In Chapter 5, we select adaptation-optimised meta-learner MAML (Finn et al., 2017) and similarity-optimised meta-learner RN (Sung et al., 2018) over model-optimised meta-learners to improve multi-task approach to personalisation with few data.

The most common approach to open-ended recognition is to use an intermediary semantic feature representation of seen and unseen classes (Lampert et al., 2014). Such knowledge is either hand-crafted by an expert (Lampert et al., 2014; Ohashi et al., 2018) or learned using a large knowledge corpus (Al Machot et al., 2020; Xu et al., 2017b). In Section 2.2.3, we describe these methods as *knowledge-intensive*. Semantic features can be incomplete and fails to encode personalised features. We instead investigate the use of meta-learners from in Section 2.4.3 for open-ended recognition. We identify the opportunity and challenge of extending similarity-optimised meta-learners (Snell et al., 2017; Sung et al., 2018; Vinyals et al., 2016) into open-ended recognition while incorporating personalised characteristic. The ability to use few data instances from the end-user to introduce unseen activity classes eliminates the need for experts, thus, we identify our approach as *knowledge-light*.

2.6 Chapter Summary

In this chapter, we reviewed the literature on four areas of interest; Exercise Recognition, HAR personalisation challenges, Attention Fusion and Meta-learning. We first outlined the state-of-the-art research in the ExRec domain. We investigated existing data and algorithms to highlight the need for open-access data and the need to exploit the advances in Deep Learning research.

Next, we discussed the literature related to the three HAR challenges addressed in this thesis. We discussed the multi-modal configurations found in HAR literature to improve recognition accuracy and the key design aspects of existing fusion algorithms; feature learning architectures, fusion methods and fusion levels. Then we explored the research in personalised recognition. We found the challenges of early approaches and highlight advances in recent methods that explore learning from few-data. Thirdly, we explored the literature in open-ended recognition. We highlight that existing methods rely on the completeness of an intermediary feature space curated by experts.

Next we investigate two areas of research to addresses aforementioned challenge: Attention Fusion and Meta-learning. We start by discussing the background in attention and explore attention fusion approaches used for multi-modal fusion. Each approach is critically reviewed concerning the implication in a heterogeneous multi-modal setting. Thereafter, we explored the state-of-the-art meta-learning research for implementing few-shot classification. We categorised meta-learning algorithms considering how they *learn-to-learn* from few-data and discuss their implications in an open-ended setting. We finish this chapter with the conclusions drawn from the literature, highlighting the most influential research items for our contributions.

Chapter 3

Background, Data and Evaluation Methodology

In this Chapter, we aim to set the background for our contributions by describing the data and methods used in this thesis from Machine Learning and Human Activity Recognition research. We start by introducing the theoretical background of neural networks and deep neural constructs. Next, we formalise HAR as a multi-class classification task using a conceptual representation of a HAR dataset. Finally, we present evaluation datasets, pre-processing steps, evaluation methodologies and performance metrics applied in this research.

3.1 Background in Neural Networks

A Neural Network (NN) is comprised of layers of multiple neurons, each layer connected to the previous layer. This forms a network where every edge is a parameter that is learned during training. We illustrate a simplified version of a neuron in Figure 3.1. A neuron calculates the weighted sum of previous layer activations and normalise using an activation function, φ (Equation 3.1). Commonly used activations functions are Sigmoid, Tanh and Relu. Here, all w_i and b are trainable parameters, l indicates the layer, and n is the number of neurons in the previous layer.

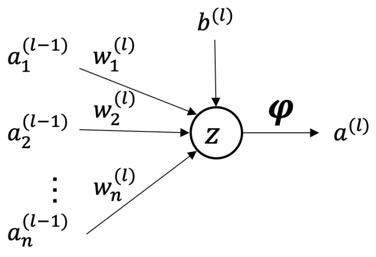


Figure 3.1: Neuron

$$
\begin{gathered}
z^{(l)} = \sum_{i}^{n} w_i^{(l)} a_i^{(l-1)} + b^{(l)} \\
\text{Neuron Activation: } a^{(l)} = \varphi\left(z^{(l)}\right)
\end{gathered}
\tag{3.1}
$$

3.1.1 Neural Network Classifier

A Neural Network (NN) classifier has a $\mathcal{C}$ number of neurons at the output layer, one for each class. Softmax activation at the output layer produces a skewed probability distribution where the neuron activation with the highest probability is selected as the predicted class (Equation 3.2).

$$
\begin{gathered}
\text{Softmax Activation: } \varphi(z_i) = \frac{e^{z_i}}{\sum_j^{\mathcal{C}} e^{z_j}} \\
y' = \arg\max_i \varphi(z_i)
\end{gathered}
\tag{3.2}
$$

An NN classifier is trained using the gradient descent optimisation algorithm, which minimises the error of class prediction of training data. An input data instance, x, is propagated through the layers of neurons, and the NN produce an output, y', which is also the activations, $a^{(o)}$, at the output layer, o. The difference between the expected class, y, and the predicted class y' is calculated using categorical cross-entropy (Equation 3.3). Here y_i and y'_i are elements of one-hot encoding representation of y and y'.

$$
\text{Loss: } \mathcal{L} = -\sum_{i}^{\mathcal{C}} y_i \; log(y'_i) + (1 - y_i)\log(1 - y'_i) \text{ where } y'_i = a_i^{(o)} \tag{3.3}
$$

Gradient descent iteratively refines the parameters towards minimising the loss, $\mathcal{L}$; or the parameter values that bring the gradient of $\mathcal{L}$, $\nabla\mathcal{L}$, to zero (in Equation 3.4). With the chain rule, we write Equations in 3.5, the partial derivative for each parameter w_i and b, of the the neuron in Figure 3.1. A single neuron augmented by the back-propagating partial derivatives is illustrated in Figure 3.2. Here each partial derivative is calculated using Equations 3.1 and 3.3. Once we calculate the gradients vector, $\nabla\mathcal{L}$, the parameters are updated using the learning rate α (Equation 3.6).

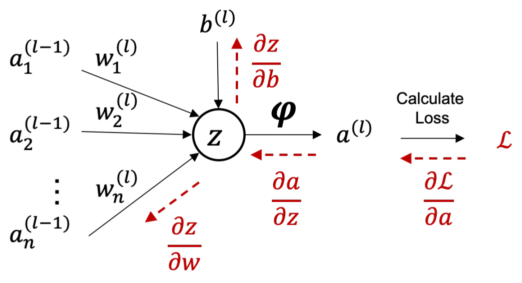

$a_1^{(l-1)}$
$a_2^{(l-1)}$
$a_n^{(l-1)}$
$w_1^{(l)}$
$w_2^{(l)}$
$w_n^{(l)}$
$b^{(l)}$
$\frac{\partial z}{\partial b}$
z
φ
$a^{(l)}$
Calculate Loss
$\mathcal{L}$
$\frac{\partial a}{\partial z}$
$\frac{\partial \mathcal{L}}{\partial a}$
$\frac{\partial z}{\partial w}$

Figure 3.2: Neuron augmented with partial derivatives for back-propagation

$$\text{Gradient: } \nabla\mathcal{L} = \left[\frac{\partial\mathcal{L}}{\partial w_1^{(l)}}, \frac{\partial\mathcal{L}}{\partial w_2^{(l)}}, \ldots, \frac{\partial\mathcal{L}}{\partial w_n^{(l)}}, \frac{\partial\mathcal{L}}{\partial b^{(l)}}\right] \tag{3.4}$$

$$\begin{aligned}\text{Partial derivative of } \mathcal{L} \text{ w.r.t. } w_i^{(l)}: \ \frac{\partial\mathcal{L}}{\partial w_i^{(l)}} &= \frac{\partial\mathcal{L}}{\partial a^{(l)}}\frac{\partial a^{(l)}}{\partial z^{(l)}}\frac{\partial z^{(l)}}{\partial w_i^{(l)}} \\ \text{Partial derivative of } \mathcal{L} \text{ w.r.t. } b^{(l)}: \ \frac{\partial\mathcal{L}}{\partial b^{(l)}} &= \frac{\partial\mathcal{L}}{\partial a^{(l)}}\frac{\partial a^{(l)}}{\partial z^{(l)}}\frac{\partial z^{(l)}}{\partial b^{(l)}}\end{aligned} \tag{3.5}$$

$$\text{Parameter update: } w_i' = w_i + \alpha\,\frac{\partial\mathcal{L}}{\partial w_i^{(l)}} \text{ and } b' = b + \alpha\,\frac{\partial\mathcal{L}}{\partial b^{(l)}} \tag{3.6}$$

It is noteworthy that a NN classifier learns a mapping, between the input data and the output class, parameterised by a finite number of neurons. As such, a NN can only learn an approximate mapping, limited by training examples and the number of neurons. Thus, a NN classifier can not achieve 100% performance, which is also described as the Bayes error or irreducible error (Antos et al., 1999). NN classifier can be used in conjunction with many neural feature extraction architectures, such as Deep Neural Networks (DNN), Convolutional Neural Networks (CNN) and Recurrent Neural Networks (RNN). The basics of each architecture is explored next, with emphasis on the underlying neural constructs.

3.1.2 Deep Neural Networks

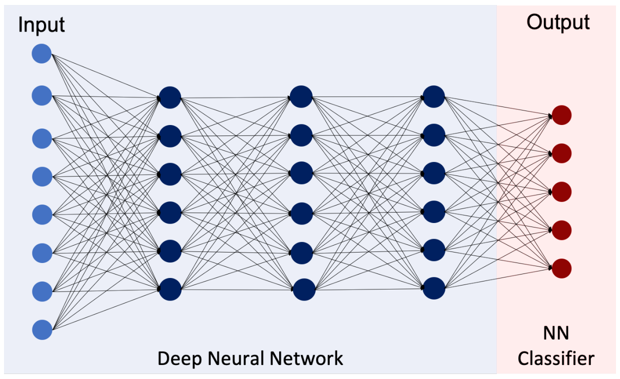


Figure 3.3: A deep neural network

Deep Neural Network (DNN) is a network of many hidden layers of neurons as illustrated in Figure 3.3. The DNN is connected to a NN classifier, with softmax activation to predict classes (5 class classifier in this example) which is a standard configuration for all DNN, CNN or RNN architectures. A DNN has many layers of neurons that are densely connected, with Relu activation. Using many layers increases the parametric complexity allowing the network to learn complex relationships. A DNN layer with Relu activation is denoted by $dense(n)$ where n refers to the number of neurons in the rest of this thesis.

Having a large number of parameters also increases the abundance of trainable parameters to memorise all training data that is known as over-fitting. In this research, we used Batch Normalisation (BN) (Ioffe and Szegedy, 2015) as a regularisation method to prevent over-fitting. A BN layer is formalised in Equation 3.7 where μ and ν are trainable parameters and $\mu \times \hat{x}$ is an element-wise multiplication. Input, x, is normalised to obtain $\hat{x}$, where $E(x)$ is the expected value(i.e. mean) and $Var(x)$ is the variance, both w.r.t a subset of training data (mini-batch) (Equation 3.8).

$$x' = \mu \times \hat{x} + \nu \tag{3.7}$$

$$\hat{x} = \frac{x - E(x)}{\sqrt{Var(x)}} \tag{3.8}$$

BN introduces a non-deterministic augmentation to the input, $\hat{x}$, using a parametric model and is preferred over other regularisation methods such as Dropout or early stopping. Dropout and early stopping are methods that try to avoid over-fitting on training data, by either partially removing random neuron activations or by evaluating the training network for over-fitting on an independent data set (i.e. validation set). BN is applied after each dense or convolutional layer in every architecture of this thesis and refer to a BN layer as *bn*.

3.1.3 Convolutional Neural Networks

Convolutional Neural Networks (CNN) (Lawrence et al., 1997) have demonstrated ability to learn spatial features and are notably successful in image classification (Krizhevsky et al., 2012). A typical CNN architecture consists of an array of convolution and pooling functions followed by one or few dense layers. Convolution function extracts latent spatial dependencies in the input and organises in to feature maps. Deeper layers enable the discovery of features that are increasingly conceptual. The pooling function manipulates the spatial dimensions, by either compressing or dispersing the feature space.

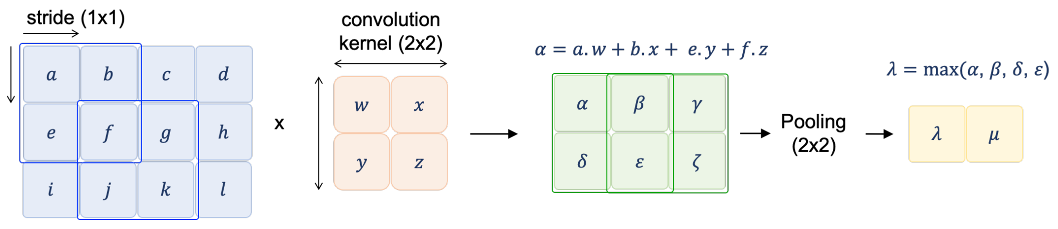


Figure 3.4: A pipeline of convolutional and pooling operations, adapted from Goodfellow et al. (2016)

A simple example is depicted in Figure 3.4, where a convolution kernel of size (2×2) is applied on an input of size (3×4) with 1 channel. Each convolution operation maps an area on the image to a single value, resulting in a feature map of size (2×3). Here w, x, y, z are trainable kernel parameters. In practice, multiple kernels are applied on the same input to create multiple projections of the input. A convolutional layer is denoted by $conv(n)k$ where n refers to the kernel size, and k is the number of kernels. Pooling is applied on the resulting feature map to alter dimensions, most commonly, max-pooling selects the maximum value from the select area on the feature map. Pooling is a non-parametric operation with no trainable parameters. Over training epochs, kernels learn latent spacial patterns while max-pooling amplifies the features by reducing the dimensionality of feature maps. A max-pooling layer is denoted by $maxpool(n)$ where n refers to the pooling size. A convolutional block is formed by a convolutional layer, a pooling layer and a BN layer.

3.1.4 Recurrent Neural Networks

Recurrent Neural Networks (RNN) learn temporal dependencies in data streams. This is in contrast to convolutions that learn spatial dependencies. In literature, RNN and its variants such as LSTM (Hochreiter and Schmidhuber, 1997) and GRU (Cho et al., 2014a) networks have been applied successfully for time-series predictions, speech recognition and language translation. We illustrate a simplified RNN un-rolled over time in Figure 3.5. At a given time-stamp t, the hidden state of the previous time-stamp h_{t-1} and the input x_t is encoded using parametric models, and the output y_t is derived as in Equations 3.9. Accordingly, at timestamp t, the network intuitively absorb temporal patterns learned over time by integrating the hidden state from the previous timestamp.

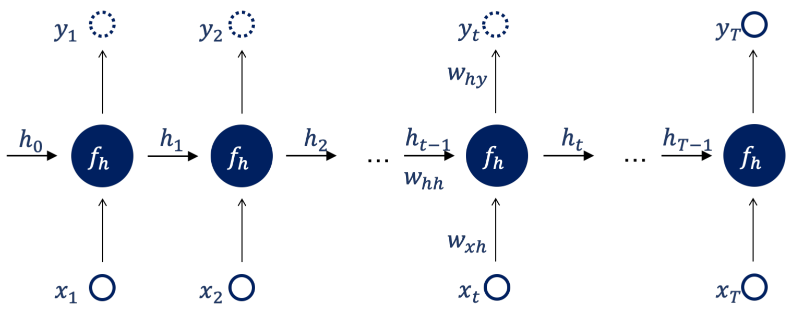


Figure 3.5: A recurrent network unrolled over time

$$\begin{aligned} h_t &= tanh(w_{hh}h_{t-1} + w_{xh}x_t) \\ y_t &= \sigma(w_{ht}h_t) \end{aligned} \tag{3.9}$$

For a classification task, we only consider the final output at timestamp T. Note that Figure 3.5 is simplified representation of RNN, compared to variants like LSTM or GRU that have multiple encoding modules that increase parametric complexity. Commonly, one or few RNN blocks are used in conjunction with convolutional and dense layers in an RNN architecture. LSTM is used as the preferred recurrent architecture in this research and is denoted by $lstm(k)$ where k refers to the number of hidden encoding units.

3.2 HAR Problem Definition

A HAR dataset D, consists of one or multiple streams of sensor data, annotated at each timestamp with the corresponding activity label. Given D, activity recognition, like any supervised learning problem, is the task of learning the mapping θ, between given data instances, x, and activity classes, y. Thereafter, θ can predict the activity class $\hat{y}$ for a test data instance $\hat{x}$. Here, $y, \hat{y}$ are from the set of activity classes, $\mathcal{C}$ (Equation 3.10). We consider θ to be a parametric neural network model that learns an approximation of the mapping between x and y, using a finite number of train-able parameters.

$$\hat{y} = \theta(\hat{x}) \text{ where } \hat{y} \in \mathcal{C} \tag{3.10}$$

In comparison to image or text classification, with HAR, each data instance in D belongs to a person, p. Accordingly, D is the collection of data instances from the population $\mathcal{P}$ (Equation 3.11) where D^p is the set of data instances obtained from person p. As before, each data instance in D^p will belong to a class in $\mathcal{C}$, except in the open-ended setting where $\mathcal{C}$ is not fully specified at train time.

$$D = \{D^p \mid p \in \mathcal{P}\} \text{ where } D^p = \{(x, y) \mid y \in \mathcal{C}\} \tag{3.11}$$

A dataset may consist of multiple sensor modalities. A sensor modality is identified by the sensor type and its placement. For instance, selfBACK dataset consists of two homogeneous sensor modalities (i.e. two accelerometers on the wrist and the thigh) and MEx dataset consists of four heterogeneous sensor modalities (i.e. two accelerometers, a pressure mat and a depth camera). A comprehensive list of sensors and sensor placements (i.e. modalities) seen in previous ExRec research is presented in Table 2.1.

Let M represent the set of sensor modalities of size m in D (Equation 3.12). For instance, $m = 2$ for selfBACK.

$$M = \{M_i\} \text{ where } 0 < i < m \tag{3.12}$$

$$M_i^t = [\mu_1, \mu_2, ..., \mu_n] \tag{3.13}$$

Each sensor modality is recorded for a period of time, and at each timestamp, t, modality M_i has an array of raw features, μ of size n (Equation 3.13). For instance, at timestamp t, an accelerometer modality has an array of raw features $[x, y, z]$ where $n = 3$. We will refer to the above formalisation throughout this thesis.

3.3 HAR Datasets and Pre-processing

The three HAR datasets MEx, selfBACK and PAMAP2 are used throughout this research where each of them represents a unique activity category. The goal is to validate our method not only on physiotherapy exercise recognition but on a wide range of HAR tasks. It also allows comparing our methods against the methods in the literature.

MEx is the physiotherapy exercise datasets collected with 30 individuals. The dataset is recorded with four sensors, two accelerometers worn on the wrist (MEx_{ACW}) and the thigh (MEx_{ACT}), a pressure mat (MEx_{PM}) and a depth camera (MEx_{DC}). MEx consists of 7 physiotherapy exercises (details on Appendix B) where each person performed the 7 exercise, each for approximately 60 seconds. The dataset is publicly available in the UCI Machine Learning Data Repository [1].

selfBACK dataset is compiled with two tri-axial accelerometer data streams, performed by 33 individuals. The dataset includes 9 activity classes, where 6 are ambulatory activities, and the rest are sedentary activities. The list of activities is jogging, walking in a slow, medium, fast pace, walking upstairs, walking downstairs, standing, lying and sitting. Each activity is performed for approximately 3 minutes with accelerometers mounted on the right wrist (SB_W) and the thigh (SB_T) of a participant. The dataset is publicly available in the UCI Machine Learning Data Repository [2].

PAMAP2 is a physical activity monitoring dataset recorded with 3 IMU sensors (Reiss and Stricker, 2012). The sensors are located on the wrist (PAMAP2_H) and ankle (PAMAP2_A) are on the dominant side and on the chest (PAMAP2_C). Data is recorded with 9 participants for 18 activity classes by following a pre-defined protocol. Activities include that are ambulatory, sedentary and daily living. One user and 10 activities were filtered out of this dataset due to inconsistencies in data collection. The refined dataset contained 8 users and 8 activity classes. 3 ambulatory activities: walking, walking upstairs and walking downstairs; 3 sedentary activities: sitting, standing and lying; and 2 activities of daily living: vacuum cleaning and ironing. The dataset is publicly available in the UCI Machine Learning Data Repository [3].

[1] https://archive.ics.uci.edu/ml/datasets/MEx
[2] https://archive.ics.uci.edu/ml/datasets/selfBACK
[3] http://archive.ics.uci.edu/ml/datasets/pamap2+physical+activity+monitoring

HDPoseDS dataset contains 22 activity classes (poses and sedentary activities), recorded with 9 participants (Ohashi et al., 2018). The dataset is available to download from the project website [4]. Data is recorded while wearing 31 IMU sensors. Sensor placements are 1 on the head, 2 on shoulders, 2 on upper arms, 2 on lower arms, 2 on hands, 14 on fingers, 1 on the spine, 1 onhe t hip, 2 on upper legs, 2 on lower legs and 2 on feet. HDPoseDS dataset is used to evaluate open-ended recognition methods introduced in Chapter 6. HDPoseDS a sensor-rich dataset which can be obtrusive in real-world applications. Therefore our methods are evaluated against more restricted sensor configurations derived from this dataset.

3.3.1 Pre-processing

Given a dataset D, to learn the model θ, we create labelled data instances, (x, y), using the sliding window method. It is the standard approach to create labelled data instances from a stream of time-series data (Keogh et al., 2001). Figure 3.6 illustrates the sliding window method where window size, w, and overlap, o, are hyper-parameters selected to create data instances. A data instance, x_i, of modality M_i, is a data frame sequence of w timestamps, starting at timestamp t (Equation 3.14).

$$x_i = [M_i^t : M_i^{t+w}] \tag{3.14}$$

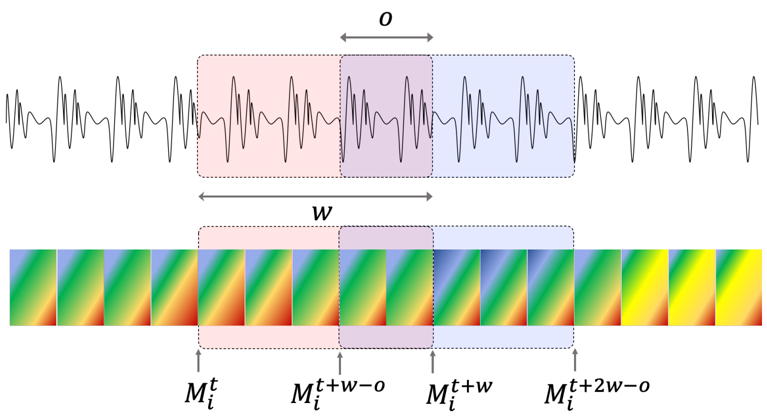


Figure 3.6: Sliding window method applied synchronously on two sensor data streams

Each modality in the dataset uses the same w and o values and starts applying the sliding window from the same timestamp, t_0. Resulting data instance, x, is the set of

[4]http://projects.dfki.uni-kl.de/zsl/data/

data instances from each modality in M starting at timestamp t (Equation 3.15). The same w and o values are used at test time, where a test data instance is created at every $w - o$ timestamp (i.e. *increment*). We summarise the resulting properties of each dataset in Table 3.1. Here $*$ indicates the original number of modalities in HDPoseDS and PAMAP2 datasets.

$$x = \{x_i\} \text{ where } 0 < i < m \tag{3.15}$$

Table 3.1: Dataset properties for MEx, selfBACK, PAMAP2 and HDPoseDS

Property	MEx	selfBACK	PAMAP2	HDPoseDS
m	4	2	9*	31*
$\mid \mathcal{C} \mid$	7	9	18(8)	22
$\mid \mathcal{P} \mid$	30	33	9(8)	9
w (seconds)	5	5	5	1
o (seconds)	3	3	2.5	0
Instances: $\mid D \mid$	6326	22493	5660	5356
Instances per person:$\mid D^p \mid$	210	681	707	595
Instances per person-activity	30	75	88	27

Once the data instances are created using the sliding window method, modality-specific pre-processing steps are applied to further prepare the data for model learning. Our aim is not to hand-craft features, but to prepare the data instances, such that it is complementary to a feature representation learner. We use three pre-processing pipelines for the three sensor data types accelerometer, depth camera and pressure mat.

An accelerometer data instance consists of three raw data sequences, (x, y, z) of length w. Recent literature has shown that applying frequency domain feature transformation is advantageous in comparison to raw data. The most common methods seen in HAR literature are Discrete Fourier Transformation (DFT) (Yao et al., 2017) and Discrete Cosine Transformation (DCT) (He and Jin, 2009; Sani et al., 2017a). A comparative study showed that DCT outperforms DFT (Sani et al., 2017a), and another showed evidence that DFT did not yield any performance improvement over raw data (Yang et al., 2015). Accordingly, the DCT method is selected which decomposes a signal into a set of constituent cosine waves at different frequencies, that collectively approximates the original signal. DCT is applied to each one-second segment of each axis of accelerometer data, and the most significant cosine frequency coefficients are selected (i.e.truncated). The final feature vector is formed by appending the resulting coefficients for the w number of one-second segments and 3 axes x, y and z. It is noteworthy that we refer to DCT as a feature transformation instead of a pre-processing step.

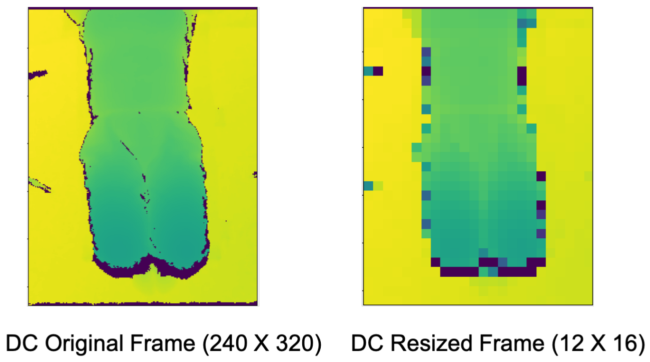


Figure 3.7: DC modality pre-processing frame resize, left: original frame (240x320), right: resized frame (12x16)

A depth camera produces a time series of images, where each pixel value in the image corresponds to the distance between the camera and the object. We apply three pre-processing steps to a DC data instance. Exercises selected in the MEx dataset, and physiotherapy exercises in general do not have quick repetitive movements where a higher frame rate is required. Therefore we reduce the frame rate to 5 frames per second where we select the frame at each $1/5$ second increment (i.e. approximately every 3rd frame given 15Hz original frame rate). Next we reduce the frame size from 240×320 to 12×16 (Figure 3.7) and finally we normalise the pixel values to fit $0-1$ range.

Similar to a depth camera, a pressure mat also produces a time series of heat-maps. A heat-map is a 2-dimensional matrix of pressure sensor points (32×16 for PM data from the MEx dataset) and each point is recording the pressure applied. We apply similar pre-processing steps to the depth camera data: reduce the frame rate to 5 frames per second (from 15Hz original frame rate); reduce frame size to 16×16 (from 32×16); and normalise the frame data to fit $0-1$ range.

Pre-processing steps for depth camera and pressure mat are selected following an exploratory study presented in Appendix B. Radu et al. (2018) pointed out that the same sensor with different configurations (such as different frame rates) is detrimental to reasoning algorithm performance in the real-world. Creating reasoning models using the minimal frame rates and frame sizes without affecting the performance partly alleviate temporary sensor malfunctions or delays. It also reduce the memory requirements and computational capacities demanded on an edge device. Table 3.2 summarises the resulting data instance features properties of each sensor modality.

Table 3.2: Data instance properties for each modality, before and after pre-processing

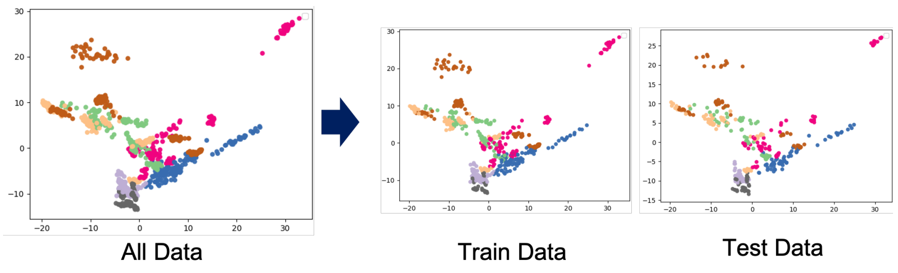
Dataset	Sensor	Original $(window \times fps \times features)$	Pre-processed $(window \times features)$
MEx	AC	$(5 \times 100 \times 3)$	$(5 \times 60 \times 3)$
	DC	$(5 \times 15 \times 240 \times 320)$	$(5 \times 5 \times 12 \times 16)$
	PM	$(5 \times 15 \times 32 \times 16)$	$(5 \times 5 \times 16 \times 16)$
selfBACK	AC	$(5 \times 100 \times 3)$	$(5 \times 60 \times 3)$
PAMAP2	AC	$(5 \times 100 \times 3)$	$(5 \times 60 \times 3)$
HDPoseDS	AC	$(1 \times 60 \times 3)$	$(1 \times 30 \times 3)$

3.4 Evaluation Methodology

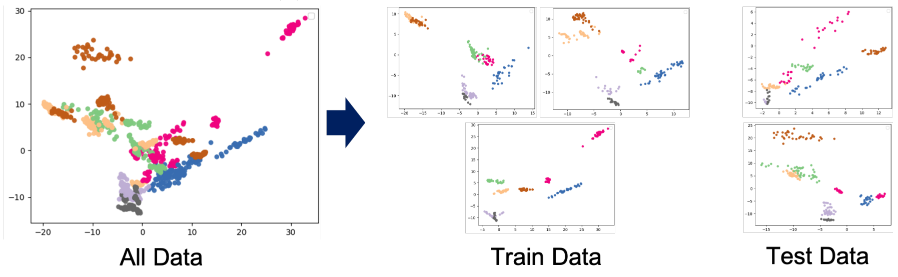
In literature, activity recognition evaluation methodologies adopt one of three approaches: *person-dependent* where an algorithm is trained and tested with one user; *person-agnostic* where an algorithm is trained and tested with the a user group; and *person-aware*, where an algorithm is trained and tested with different user groups. They consistently maintain disjointed sets of data instances in train and test, but the person-aware methodology also preserves disjoint persons by maintaining the person-to-data relationship.

In early literature, the person-dependent evaluation methodology is commonly used where an algorithm is learned for a specific end-user (Gomes et al., 2012; Zhou et al., 2016). This method yield high performances, yet infeasible to implement in practice, especially with the large data requirements of DL algorithms. Person agnostic methods such as repeated hold-out (R-HO) and cross-fold (CF) are also used in HAR evaluation (Inoue et al., 2018; Mendiola et al., 2019). Person agnostic methods discard the person identifier of data when creating hold-out sets, or folds. Accordingly, the resulting train and test sets share the same population, $\mathcal{P}$, hence the same data distributions (see Figure 3.8a). Accordingly, these methodologies are not designed to evaluate the robustness of an algorithm on a different population following deployment. However, they provide the *upper-bound* performance for an algorithm.

A person-aware evaluation can be performed using Repeated Persons Hold-Out (R-PHO) or Leave-One-Person-Out (LOPO) methodologies. With R-PHO, a percentage (typically 1/3), of the user population is selected as the test user set, rest forming the train user set, and this is repeated for multiple iterations. Accordingly, the algorithm is trained



(a) Person-agnostic evaluation



(b) Person-aware evaluation

Figure 3.8: Example train and test data splits of person-agnostic and person-aware evaluation methodologies

and tested on different data distributions as in Figure 3.8b. With LOPO methodology, a single user is put aside, to form a singleton test user group, and the rest of the users create the training user group. LOPO ensures that each user in the population $\mathcal{P}$ is used as the test set in one of the trials (analogous to person-agnostic CF method). A person-aware methodology creates a challenging setting compared to person-agnostic methods, hence it provides the *lower-bound* performance for an algorithm.

We identify that performance of an algorithm should be evaluated using a person-aware method to observer the expected performance in the real-world. Accordingly, recognition algorithms discussed in Chapters 4 and 5 are evaluated using the person-aware methodology LOPO. Each algorithm create $|\mathcal{P}|$ number of experiments, by leaving out one person at a time.

3.4.1 Evaluating Open-ended Recognition

Open-ended recognition evaluation settings can be identified by the number of unseen classes and the number of classes in the test setting. In HAR, such evaluation setting also has to follow a person-aware evaluation methodology. An evaluation setting in an open-ended recognition experiment is often described as Leave-N-class-out (LNCO).

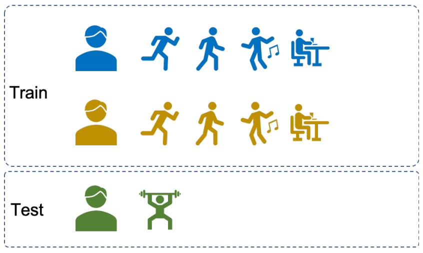
For instance, in a 5 class dataset (i.e. $|\mathcal{C}^*| = 5$), LNCO evaluation can create four experiments: L1CO, L2CO, L3CO and L4CO, where any 1, 2, 3 or 4 activity classes are selected as unseen classes, $\hat{\mathcal{C}}$ and there are 4, 3, 2 and 1 seen classes, $\mathcal{C}$.

L1CO (i.e N=1) provides the opportunity to observe how a model perform when different activity classes are introduced as the unseen class. The performance may vary for activity classes depending on how well the selected modalities capture them and on how different they are from seen classes. For instance, if the discriminatory features of an activity class are not captured by appropriate sensor modalities, the open-ended recognition algorithm may fail to recognise the new activity apart from seen activities. Accordingly, in the L1CO setting we create $|\mathcal{C}^*|$ number of experiments where each class is once considered as the $\hat{\mathcal{C}}$.

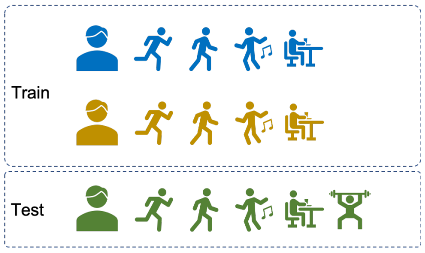
It is increasingly challenging to perform classification when a subset of classes were not seen during training. Accordingly, $N > 1$ settings demonstrate the robustness of open-ended recognition with an increasing number of unseen activity classes. When creating $N > 1$ experiments, we follow a repeated classes hold-out methodology to avoid a combinatorial explosion. For instance, for L2CO, with a dataset with 10 activity classes, there are 45 experiments with unique 2 class combinations. Instead, we repeat an experiment for 20 times each with a randomly selected N number of unseen classes. For instance, for L2CO experiments, we repeat the experiment 20 times, each time, 2 classes are randomly selected from $\mathcal{C}^*$ as the $\hat{\mathcal{C}}$ and the rest are considered as $\mathcal{C}$.

While LNCO setting defines the number of unseen classes, the test can be performed for two sets of test classes ($\mathcal{C}_{te}$): for only unseen classes; or seen and unseen classes. More formally $\mathcal{C}_{te} = \hat{\mathcal{C}}$ or $\mathcal{C}_{te} = \mathcal{C} \cup \hat{\mathcal{C}}$. In the literature, when $\mathcal{C}_{te} = \hat{\mathcal{C}}$ it is referred to as the conventional open-ended setting. It is assumed that the open-ended model only encounters unseen classes at test time. Thus, the performance is measured only for unseen classes. In contrast, $\mathcal{C}_{te} = \mathcal{C} \cup \hat{\mathcal{C}}$ presents a more real-world setting, hence known as the generalised open-ended setting. Here, the open-ended model is evaluated for both seen, and unseen classes and the evaluation captures how the recognition of seen classes is affected by the introduction of and unseen classes after deployment.

As discussed in Section 3.4, for HAR, using a person-aware evaluation methodology is essential. Accordingly, any evaluation setting described above needs to conform to a person-aware evaluation methodology. To avoid a combinatorial explosion, we select a Repeated Persons Hold-out (R-PHO) methodology. At a given iteration of an experiment, the train and test persons are split randomly. 2/3 of the persons are selected as the train set and the remaining 1/3 persons form the test set. Each experiment is repeated for 20 times with a random train-test split to account for non-deterministic features in the algorithms. The final result of each experiment is the mean over the 20 iterations. Accordingly, a L1CO experiment creates a total number of $20 \times |\mathcal{C}^*|$ repetitions; an L2CO experiment (any $N > 1$) creates a total number of 20×20 repetitions.



(a) Conventional open-ended setting



(b) Generalised open-ended setting

Figure 3.9: Open-ended recognition evaluation settings

Figure 3.9 illustrates an experiment in the evaluation setting we propose for open-ended recognition. Here we consider an example dataset with 3 person, 5 activity classes ($\mathcal{C}^*$) where $N = 1$. In the conventional setting (Figure 3.9a), the open-ended model is only tested for the unseen classes ($\mathcal{C}_{te} = \hat{\mathcal{C}}$). In the generalised setting (Figure 3.9b), the algorithm is tested for both seen and unseen classes ($\mathcal{C}_{te} = \mathcal{C} \cup \hat{\mathcal{C}}$). At each iteration of the experiment, 2 persons are randomly selected for the train set and the remaining person for the test set. Accordingly, in the conventional setting, neither the person nor the weight-lifting class is seen during training. In the generalised setting, neither the person nor the weight-lifting class is seen during training, but other 4 classes are seen from train-persons.

3.4.2 Performance Measures

In this section, we choose a set of performance metrics to evaluate algorithms and perform comparative studies in this thesis. We use the F1-score as the performance measure for multi-modal recognition experiments (Equations 3.16, 3.17, 3.18). To mitigate class imbalance that exists in PAMAP2 and selfBACK datasets (more details on Appendix C), we use weighted averaging where F1-score is first calculated for each class label and is weighted by the fraction of data instances per each class. Equation 3.19 shows the weighted averaging of F1-score, where the weight for class c, w_c, is calculated as the fraction of data instances for class c, n_c, over the total number of data instances.

$$\text{Precision} = \frac{TP}{TP + FP} \tag{3.16}$$

$$\text{Recall} = \frac{TP}{TP + FN} \tag{3.17}$$

$$F_1 \text{ score} = 2 \times \frac{precision \times recall}{precision + recall} \tag{3.18}$$

$$\begin{aligned} \text{Weighted } F_1 \text{ score} &= \sum_{c}^{\mathcal{C}} w^c \times F_1^c \\ w^c &= \frac{n^c}{\sum_c^{\mathcal{C}} n_c} \end{aligned} \tag{3.19}$$

For personalised and open-ended recognition experiments, we explicitly maintain the class balance in train and test sets with the meta-task creation process. Accordingly, we report accuracy as the performance measure (Equations 3.20). Both accuracy and F1-score are averaged over many folds introduced by the evaluation methodology used. For examples, a recognition algorithm that uses LOPO on a 30 person dataset, reports the final F1-score which is the mean F1-score over 30 folds. An open-ended recognition algorithm uses R-PHO and repeats an experiment for 20 times to report the final mean accuracy.

$$\text{Accuracy} = \frac{TP + TN}{TP + TN + FP + FN} \tag{3.20}$$

3.4.3 Statistical Significance Testing

In general, the use of t-tests for statistical significance testing assumes that the test-statistic is normally distributed. Person-aware evaluation methodologies produce results that are not normally distributed. For examples, when we test a model using a person who has a distinctly different data distribution to training data, the model may perform poorly. However, when tested using a person who has a similar data distribution to the training data, the model may perform significantly well. Thus the performance is varied and does not represent a normal distribution. The performances may start to approximate to a normal distribution if the population is sufficiently large. Given the number of persons in the HAR datasets considered in this thesis range between 8 and 33, we opt for a non-parametric statistical significance test. We select Wilcoxon signed-rank test for paired samples to evaluate the statistical significance at 95% confidence interval. The test is performed using the SciPy Python libraries and in evaluations, significantly performances are highlighted in bold text.

3.5 Chapter Summary

In this chapter, we presented the theoretical background of neural networks, datasets and evaluation methods. This chapter aimed to set the background for our technical contributions and formalise evaluation methodologies followed this thesis. First, we presented the basics of neural networks, the neural network classifier and neural network training. We also discussed the foundations of deep feature learners, specifically, deep neural networks, batch normalisation, convolutions and recurrence. Next, we formalised HAR as a multi-class classification problem and detailed the datasets used to evaluate the methods introduced in this thesis. We detailed the pre-processing steps, including the windowing method and feature transformations used by each modality. Finally, we presented the evaluation methodologies, performance measures and statistical significance testing methodology used in this thesis.

Chapter 4

Multi-modal Recognition with Hybrid Attention Fusion

Human Activity Recognition (HAR) algorithms are implemented to recognise activities monitored using different sensor modalities. The type of activities, the environment and personal preferences influence the selection of modalities to capture activities with high precision. Accordingly, recognition algorithms should be adaptable to different modalities and perform an effective fusion of heterogeneous sensor data streams.

Our goal is to learn the most effective modality combinations for improved recognition accuracy while minimising parametric complexity. To this end, we present the Multi-modal Hybrid Attention Fusion architecture, MHAF. MHAF is a multi-class classifier with deep feature representation learner. The modularised architecture enables easy adaptation to suit different modalities and modality combinations. Reduced parametric complexity of the architecture will allow training a model with a limited amount of data.

4.1 Use Case

Figure 4.1 presents three use cases of multi-modal fusion for exercise recognition in the real-world. Imagine person A, who has low-back pain, performs exercises at the physiotherapy clinic. It is a speciality sensor-rich setting where there are multiple heterogeneous modalities such as wearable sensors, heart-rate monitors, cameras and pressure sensors to monitor exercise performance. Person B is a healthy older adult who regularly performs aerobic exercises in the outdoors. They prefer to have a single wearable device on the

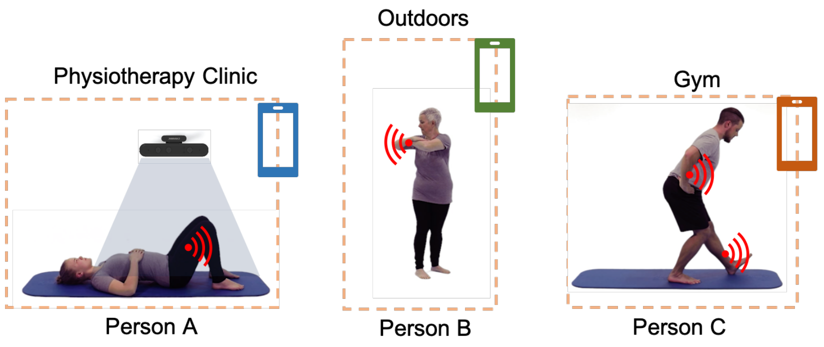


Figure 4.1: Activity recognition with different modality combinations

wrist such that it is not intrusive in the outdoor setting. This wearable device includes multiple homogeneous modalities such as accelerometer, gyroscope and magnetometer. Person C is a healthy young adult who performs intensive workout routines at a gym. To track performance, they use multiple wearables and ambient modalities. These three use cases highlight different sensor modalities or modality combinations selected based on the activities, environment and user preference.

Given a set of modalities, the main goal of a fusion architecture is to learn the most effective modality and feature combinations for each activity class. From a usability perspective, a modular architecture can train multiple models that are fitted for such modality combinations. For instance, for person B, a fusion model is created for their modality preference, instead of treating other modalities as missing. The bespoke architecture for person B should only causes minimal changes to the primary architectural constructs.

Design of the modular fusion architecture involves three main design considerations: 1) how to represent individual sensor modalities; 2) when to aggregate modalities and create shared representations; and 3) how to attend to features to highlight the most desirable features. Furthermore, as with any deep neural architecture, the amount of train-able parameters in θ is constrained by the amount of training data available to avoid over-fitting.

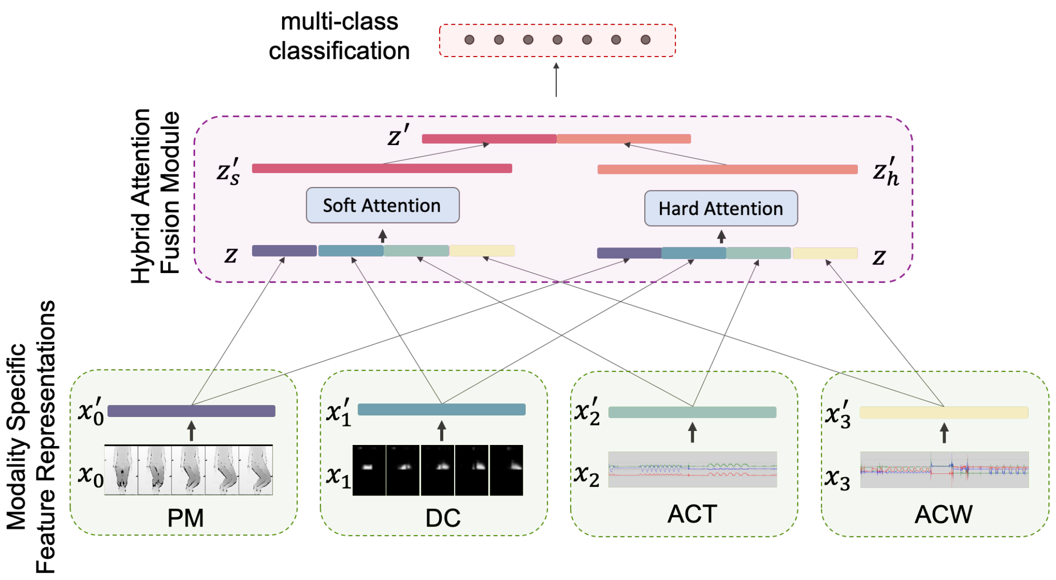


Figure 4.2: A modular view of the multi-modal hybrid attention fusion architecture (mHAF) with the four MEx modalities

4.2 mHAF Architecture

mHAF has three key modules: modality specific feature learners; hybrid attention fusion learner; and the classification layer. In Figure 4.2 we illustrate the mHAF architecture for MEx four modality combination. Firstly, each modality-specific feature learner transforms raw input data into a feature representation using the most optimal method identified using an empirical study we detail in Section 4.3. Next, the Hybrid Attention Fusion (HAF) module learns a shared feature representation by exploiting two attention approaches, Hard Attention and Soft Attention. Lastly, a softmax classifier predicts the class label. Given the output of the HAF module is z', a dense layer with softmax activation perform multi-class classification as in Equation 4.1. $\mathcal{C}$ refers to the set of activity classes, w and b are parameters where $w \in \mathbb{R}^{|z'| \times |\mathcal{C}|}$ and $b \in \mathbb{R}^{1 \times |\mathcal{C}|}$.

$$y = \underset{c \,\in\, \mathcal{C}}{\operatorname{argmax}} \left(softmax(wz' + b)\right) \tag{4.1}$$

mHAF architecture is trained end to end using the cross-entropy loss, which minimises the prediction error on training data. Given the mHAF architecture parameters are θ, and the training dataset is $\mathcal{D}$, the loss is calculated as in Equation 4.2. Importantly, we should select the number of trainable parameters in the architecture to encourage convergence during model training (i.e. avoid under-fitting) and avoid over-fitting, given a relatively small quantity of training data.

$$\mathcal{L}_{\mathcal{D}}(\theta) = \sum_{(x,y)\sim\mathcal{D}} y \log \theta(x) + (1-y)\log(1-\theta(x)) \tag{4.2}$$

4.3 Modality Specific Feature Representations

The heterogeneity of sensor modalities calls for feature representations that are modality specific instead of modality agnostic. Finding the best representation for each sensor modality instead of a generic feature set maximises the utility to improve the recognition task. For instance, a feature extraction method for accelerometer data with numeric time-series data is intuitively not optimal for depth camera data with visual time-series data. With the advances in DL, deep parametric feature extraction methods such as Deep Neural Networks (DNN), Convolutional Neural Networks (CNN) and Recurrent Neural Networks (RNN) are preferred over compiling a manual feature extraction pipeline. DL Networks are commonly trained end-to-end for classification where the feature representation is optimised for the best classification accuracy. In contrast, manual methods are disjoint from the classification task and require iterative refinements to the hand-crafted feature extraction pipeline.

A comprehensive collection of deep learning feature representation learners are compared in the single modality recognition task to identify the best learner for each modality. We note that the input is the data instances segmented using a sliding window and pre-processed/feature transformed as described in Section 3.3.1.

Deep Neural Networks: We select two NN architectures for feature extraction, a shallow NN and a deep NN. The shallow NN consists of one hidden layer densely connected with 100 units (1st row on Table 4.1), and the deep NN consists of five hidden layers with an output of 32 neurons (2nd row on Table 4.1). For AC modalities, the input is the set of DCT features as described in Section 3.3.1. For pre-processed DC and PM modalities, the frames within the window are flattened and concatenated as the input to match the 1-dimensional input expected by the DNN architectures.

Convolutional Neural Networks: We explore two variations of CNN to suit different sensor modalities. In general, there are two convolution blocks, followed by two dense blocks that output a feature vector of size 100 as follows:

- 1D-CNN: Comprised of 1-dimensional convolutional layers with kernel size 5 (3rd row on Table 4.1). For AC modalities, the input is a DCT feature vector with 1 channel. For pre-processed PM and DC modalities, the frames within the window are flattened and concatenated to create the input with 1 channel.

- 2D-CNN: For PM and DC modalities, 2-dimensional convolutional layers with kernel size 3×3. The frames within the window are concatenated to create the input with 1 channel (4th row on Table 4.1).

Recurrent Neural Networks: We explore recurrent architectures where the recurrent module is an LSTM. Two variants of convolution blocks are considered to learn a low-level feature vector which forms the input to the LSTM. The convolution block is shared among the data frames within a window, to encode input data, thus referred to as the *time-distributed* conv block. Time-distributed model is selected over two alternatives: 1) independent conv blocks for each timestamp of the input where the parameters are not shared; 2) creating a concatenated input, merging all timestamps, learning a feature vector with a larger conv block and finally splitting the output to original timestamps. The first approach is not desirable as each input is of the same modality. Learning multiple parametric models can create inconsistent inputs to the LSTM layer also introduce unnecessary parametric complexity. The second approach learns a parametric model by applying an encoding on the full window. Therefore once the outputs are split, they are not an exact representation of raw data frame from the respective timestamp. With the time-distributed conv block, each timestamp is now represented by a feature vector and is the input to the LSTM layer, to learn the temporal dependencies within the time window. Given the differences between our modalities, we explore two LSTM variations as follows.

- 1D-CNN: Time distributed 1D convolutional architecture as the conv block, followed by an LSTM block, and two dense layers with an output feature vector of size 100 (5th row on Table 4.1). For AC modalities, the input is DCT transformation applied to each 1 second data segment with 1 channel. For pre-processed PM and DC modalities, the frames selected for each timestamp is flattened to create the input, with 1 channel.

- 2D-CNN: Time distributed 2D convolution architecture as the conv block, followed by an LSTM block, and two dense layers with an output feature vector of size 100 (6th row on Table 4.1). For PM and DC modalities, 2-dimensional convolutions (kernel size 3×3) are applied. The frames selected for each timestamp is appended to create the input, with 1 channel.

Table 4.1: Modality specific feature learner architecture details

Model	Architecture
Shallow NN	$dense(100) \rightarrow bn$
Deep NN	$dense(128) \rightarrow bn \rightarrow dense(96) \rightarrow bn \rightarrow dense(64) \rightarrow bn \rightarrow dense(48) \rightarrow bn \rightarrow dense(32) \rightarrow bn$
1D-CNN	$conv(5)32 \rightarrow maxpool(2) \rightarrow bn \rightarrow conv(5)64 \rightarrow maxpool(2) \rightarrow bn \rightarrow dense(600) \rightarrow bn \rightarrow dense(100) \rightarrow bn$
2D-CNN	$conv(3 \times 3)32 \rightarrow maxpool(2 \times 2) \rightarrow bn \rightarrow conv(3 \times 3)64 \rightarrow maxpool(2 \times 2) \rightarrow bn \rightarrow dense(600) \rightarrow bn \rightarrow dense(100) \rightarrow bn$
1D-CNN-LSTM	$td[conv(5)32 \rightarrow maxpool(2) \rightarrow bn \rightarrow conv(5)64 \rightarrow maxpool(2) \rightarrow bn] \rightarrow lstm(1200) \rightarrow bn \rightarrow dense(600) \rightarrow bn \rightarrow dense(100) \rightarrow bn$
2D-CNN-LSTM	$td[conv(3 \times 3)32 \rightarrow maxpool(2 \times 2) \rightarrow bn \rightarrow conv(3 \times 3)64 \rightarrow maxpool(2 \times 2) \rightarrow bn] \rightarrow lstm(1200) \rightarrow bn \rightarrow dense(600) \rightarrow bn \rightarrow dense(100) \rightarrow bn$

Feature learners are evaluated for classification by adding a NN classifier layer with $\mathcal{C}$ number of neurons and softmax activation to predict activity classes. If the parameters of the feature learner are f, class label prediction using data from modality M_i, is formalised as in Equation 4.3. $\mathcal{C}$ refers to the set of class labels and output layer parameters are w_o and b_o such that, $w_o \in \mathbb{R}^{|x'_i| \times |\mathcal{C}|}$ and $b_o \in \mathbb{R}^{1 \times |\mathcal{C}|}$.

$$\begin{gathered} x'_i = f(x_i) \\ y = \underset{c \in \mathcal{C}}{\operatorname{argmax}} \left(softmax(w_o x'_i + b_o)\right) \end{gathered} \tag{4.3}$$

A comparative evaluation of the above methods is performed to identify the most optimal method for each sensor modality and are used to construct the mHAF architecture.

4.4 Multi-modal Fusion

In its simplest form, fusion can be viewed as the fusion of feature vectors, x_i, from m number of sensor modalities, using a fusion method, g, to create the feature vector

z (Equation 4.4).

$$z = g(x_0, x_1, ..., x_m) \tag{4.4}$$

In mHAF we use concatenation as the fusion method where feature vectors, x_i from modalities M_i, are appended to form z. Fusion, g, can be performed in three fusion levels. With early fusion, raw input feature vectors, x_i, are concatenated to form the fusion feature vector z. A feature representation learner f is designed for learning a feature representation from z, to output, z'.

$$\begin{gathered} z = concat(x_1, x_2, ..., x_m) \\ z' = f(z) \end{gathered} \tag{4.5}$$

In a mid fusion setting, each modality learns a feature vector x'_i with an independent learner, f_i, before fusion. Then, a shared feature vector is learned using a shared feature learner. Feature vectors x'_i are concatenated to create a single fusion feature vector z as the input to the shared feature learner f. In addition, the feature vector size of x'_i is kept consistent across all modalities with the f_i design, such that each modality is represented equally when learning with f.

$$\begin{gathered} x'_i = f_i(x_i) \\ z = concat(x'_1, x'_2, ..., x'_m) \\ z' = f(z) \end{gathered} \tag{4.6}$$

Late fusion, is similar to mid fusion but with the exclusion of the shared feature representation learner f. Here we note that $z = z'$ and similar to mid fusion, the feature vector size of x'_i is kept consistent across all modalities with the f_i design for equal modality representation.

$$\begin{gathered} x'_i = f_i(x_i) \\ z = concat(x'_1, x'_2, ..., x'_m) \end{gathered} \tag{4.7}$$

Literature showed that with heterogeneous sensor modalities, early attention can be detrimental since the learner f not optimised to support different types of input data. Creating an early fusion feature vector z, is masking the differences in dimensionality and the modality specificity in feature values. For example, once normalised to 0-1 range, a feature value from AC modality and a feature value from PM modality may be equal, however, they do not carry the same semantics. In addition, once flattened, PM modality creates a comparably larger feature vector than the AC modality, which may cause the network to pay more attention to the PM modality. In other words, f is unaware of the composition of the z vector. Thus, there is less opportunity to learn from individual modalities. We suggest this method is indifferent when modalities are homogeneous, such as with a combination of AC modalities.

Late and mid fusion levels alleviate disadvantages of early fusion on heterogeneous modality combinations. Mid fusion setting is suitable for both homogeneous and heterogeneous modality combinations, where there is opportunity to learn modality-specific properties. Admittedly, in comparison to the early fusion setting, the parametric model components are dramatically increased with multiple modality-specific models, f_i, and the shared model f in the mid fusion setting. Late fusion setting further improves the opportunity for capturing the heterogeneity by only using modality-specific learners. Late fusion also reduces the parametric model components with the exclusion of the shared representation learner. Accordingly, we adapt the late fusion setting for MHAF.

4.5 Attention Fusion

Attention is learning the importance of features or *learning to attend* to features towards improved classification performance. It is particularly beneficial for achieving comparable performance using a shallower parametric model trained on a smaller dataset, compared to a very deep architecture trained on a large training dataset.

As detailed in Section 4.4, z is the unified fusion feature vector where $\mid z \mid = m \times \mid x'_i \mid$ and z is the input to the NN classifier that predicts class label. Size of z depends on the number of modalities, and a larger z can be detrimental to classification performance, because informative features may get overwhelmed and marginalised within a large feature vector. Attention can be used to alleviate this issue by learning feature importance to highlight advantageous features and to hinder noisy features. Accordingly, in this section, we explore Attention Fusion (AF) to enhance the fusion of multi-modal data.

In its simplest form, AF is a parametric module with trainable parameters learning a weighted feature vector, where each weight corresponds to the significance of the feature. Accordingly, we learn attention at feature level disregard of the modality which is a contrasting approach to modality level attention (Ma et al., 2019; Yao et al., 2018) and modality feature attention (Hori et al., 2018) seen in literature. There is a corresponding attention weight (i.e. α) for each feature (i.e. $\mid \alpha \mid = m \times \mid x'_i \mid$), and the attention fusion feature vector, z', is z weighted by α. At a high level, our attention fusion method consists of four steps:

1. Create the fusion feature vector z by concatenating feature representations, x'_i, from m modalities.
$$z = concat(x'_1, x'_2, ..., x'_m) \tag{4.8}$$

2. Learn the score vector with a parametric model, θ, where the input is the feature vector, z, and the output is the vector, *score*. The length of, *score*, is equal to the length of, z, such that each feature of, z, has its corresponding score.
$$score = \theta(z) \tag{4.9}$$

3. Normalise the score using a normalisation function, φ, resulting in an output vector of attention weights, α, where the length of, α, is equal to, $|z|$.
$$\alpha = \varphi(score) \tag{4.10}$$

4. Finally the feature vector, z, is multiplied (element-wise) by the attention weights, α, resulting in the context vector, z', to replace, z, as the new attended fusion feature vector.
$$z' = \alpha \times z \tag{4.11}$$

Next, we explore alternatives approaches for implementing each step and the intuition behind our design choices that contributed to the mHAF architecture.

4.5.1 Learning Attention Weights

There are two main methods to obtaining attention weights: non-parametric attention; and parametric attention. Parametric attention learns the score vector using a dense parametric model θ, which is later transformed to attention weights using a normalisation function (Equations 4.9 and 4.10). Instead of learning the score vector, non-parametric attention apply the normalisation on the feature vector z, to obtain the context vector (i.e. $z' = \varphi(z)$). While there exist attention architectures where a non-parametric approach is used, we argue that parametric approach is more advantageous in a multi-modal setting.

With modality specific feature learners, feature vectors produced by learners are not intuitively correlated, accordingly using them directly as attention weights without a stochastic transformation can be detrimental to overall performance.

There are two granularity levels in which attention can be applied: modality level and feature level. At modality-level, only m attention weights are learned such that all features from one modality is assigned the same weight. This is in contrast to the feature-level where each feature is assigned a weight regardless of the modality. We argue that feature level attention is better suited for MHAF for two reasons. Firstly, all features from a single modality are not equally contributing towards improved classification. Secondly, more than one sensor modality does contain features that contribute towards improved classification. To achieve feature-level granularity, we select θ as $score = tanh(w.z + b)$. w and b refers to trainable parameters and $w \in \mathbb{R}^{|z| \times |z|}$ and $b \in \mathbb{R}^{1 \times |z|}$ such that $|score| = |z|$.

4.5.2 Normalisation

Attention can be complementary to fusion in two ways. Firstly, attention can help boost multiple features that in conjunction, contribute towards improving recognition. Secondly, attention can boost one or few features that are significantly contributing to improved recognition while discarding the majority of the features as noise. Given a modality combination, different activity classes may prefer either the first, the second or a combined approach. For example, consider two exercises monitored using two modalities. Intuitively, exercise 1 is recognised using only few features from modality 1 and exercise 2 is recognised using a combination of features from modalities 1 and 2. With exercise 1, many features from modality 1 and all features from modality 2 are considered noise, and the recognition of exercise 2 require features from both modalities.

The two approaches are discriminated by the type of normalisation function used to obtain the attention weights, α. To achieve the former, we consider a Soft Attention (SA) module. SA uses the *sigmoid* function as φ where the soft attention weights, α^s, are normalised such that the resulting attention weights are normally distributed. To achieve the latter, we consider a Hard Attention (HA) module. HA uses the softmax as φ function where the attention weights, α^h, are skewed to attend to only one or few features.

4.5.3 Hybrid Attention Fusion

We examine the varying needs of activity recognition and select a hybrid approach where both hard and soft attention are harnessed in a single attention module. Figure 4.3 illustrates our architecture where we concatenate output of hard attention fusion($z^{h'}$) and soft attention fusion($z^{s'}$) to form the final feature vector z' as in Equation 4.12. This concatenation is applied at late fusion level, and attention weights are calculated at a feature level granularity. This module is referred to as the Hybrid Attention Fusion module, HAF.

$$z' = concat(z^{h'}, z^{s'}) \tag{4.12}$$

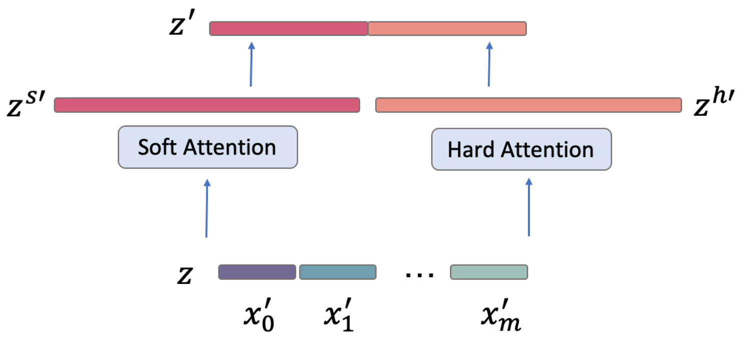


Figure 4.3: Hybrid attention fusion module

4.6 Evaluation

In this section, we perform a set of empirical evaluations to demonstrate the fine-tuning of MHAF using MEx, PAMAP2 and SELFBACK datasets. First, we find the best modality-specific feature learners for each modality (Section 4.6.1). Next, we evaluate multiple modality combinations with MHAF to find the minimal modality combinations (Section 4.6.2). Finally, an ablation study is performed to demonstrate the contribution and significance of each module in the MHAF modular architecture (Section 4.6.3).

4.6.1 Comparison of Modality Specific Feature Representations

We evaluate modality-specific feature representation learners detailed in Section 4.3 to empirically find the best learner for each modality. All NN models were implemented using Keras (Ketkar, 2017) and TensorFlow (Abadi et al., 2016) libraries for Python.

NN models are trained for 50 epochs, minimising the categorical cross-entropy loss. We use the AdaDelta optimiser with default parameters. Two conventional machine learning algorithms kNN (k=1,3) and SVM are selected as baselines against the NN models. They are implemented using the Scikit-learn Python library where the input is the pre-processed/transformed features (detailed in Section 3.3.1). Each experiment follows the LOPO evaluation methodology and presents the mean F1-score over person folds as the performance measure (see Section 3.4 for details).

MEx Modalities

Table 4.2: Modality specific feature learner performance comparison with MEx modalities

Classifier	Embedding		MEx_{ACT}	MEx_{ACW}	MEx_{DC}	MEx_{PM}
1-NN	DCT/Raw		0.7605	0.4521	0.6824	0.5691
3-NN	DCT/Raw		0.7631	0.4648	0.6742	0.5654
SVM	DCT/Raw		0.8472	0.4771	0.7325	0.3830
NN	ANN	Shallow	0.8673	0.5649	0.6183	0.6386
		Deep	0.8437	0.5412	0.6675	0.6710
	CNN	1D-CNN	0.8785	0.5605	0.8242	0.7060
		2D-CNN	-	-	**0.8720**	0.6943
	LSTM	1D-CNN-LSTM	**0.9015**	**0.6335**	0.8363	**0.7408**
		2D-CNN-LSTM	-	-	0.7821	0.7079

Table 4.2 present the results obtained with the MEx modalities. Compared to k-NN and SVM, NN classifiers achieved the best performance with each modality. Overall best performances with ACT, ACW and PM modalities were obtained using the 1D-CNN-LSTM architecture, and with DC modality using the 2D-CNN architecture (highlighted in bold text). Accordingly, these architectures will be used to encode MEx modalities in the mHAF fusion architecture.

When comparing deep and shallow NN classifiers, deep feature learners performed comparatively better than the shallow feature learners with visual data (DC and PM). In contrast, DCT transformed inertial data preferred the shallow feature learners to the deep feature learners. ACT and ACW modalities were best represented by the 1D-CNN-LSTM architecture to achieve F1-scores 0.9015 and 0.6335, respectively. Accordingly, we find that learning temporal dependencies with a recurrent architecture has resulted in better feature representation for accelerometer data. Performance of ACT and ACW suggests that ACT is less noisy when capturing exercises, which is intuitive given the low freedom of movement of thigh compared to wrist. In contrast, there are multiple exercises with no wrist movement and high movement of freedom makes ACW noisy. Accordingly, ACW benefits the most from feature learning.

Best performances for DC and PM data were 0.8720 and 0.7408, achieved using 2D-CNN and 1D-CNN-LSTM architectures, respectively. k-NN and SVM performed significantly poorly with DC and PM modalities highlighting the importance of learning feature representations for visual data. When comparing Deep and Shallow ANN, the DC and PM results affirm that visual sensor data are best learned with deep architectures. DC, which is predominantly a visual sensor benefited from learning feature representations that extract spatial dependencies using the 2D-CNN architecture compared to the 1D-CNN or LSTM architectures. Visually, PM data closely aligned with the DC modality rather than ACT and ACW modalities. However, PM preferred the 1D-CNN-LSTM architecture. These results suggest that PM data consists of prominent temporal dependencies within a given time window compared to the DC sensor. Moreover, the spatial dependencies within the pressure sensor matrix are trivial.

selfBACK Modalities

Table 4.3: Modality specific feature learner performance comparison with selfBACK modalities

Classifier	Embedding		SB_W	SB_T
1-NN	DCT		0.6113	0.6982
3-NN	DCT		0.6073	0.7021
SVM	DCT		0.6994	0.7313
NN	ANN	Shallow	0.6741	0.7562
		Deep	0.6686	0.7266
	CNN	1D-CNN	0.6675	**0.7753**
	LSTM	1D-CNN-LSTM	**0.6781**	0.7676

Table 4.3 presents the results obtained with the selfBACK modalities. Considering the NN architectures, SB_W achieves the best performance of 0.6781 with the 1D-CNN architecture and SB_T achieves 0.7753 performance with the 1D-CNN-LSTM architecture. It is noteworthy that SB_W and SB_T modalities achieve comparable performances with 1D-CNN and 1D-CNN-LSTM architectures, respectively. Accordingly, we select 1D-CNN-LSTM architecture to encode both SB_W and SB_T in the mHAF fusion architecture.

Similar to MEx we find that the wrist modality captures more noise compared to the thigh modality when performing ambulatory and sedentary activities. Moreover, deep architecture fails to outperform shallow architecture. Notably, the SVM baseline is efficiently learning against the noise compared to NN architectures to achieve the best overall performance with SB_W data.

PAMAP2 Modalities

Table 4.4: Modality specific feature learner performance comparison with PAMAP2 modalities

Classifier		Embedding	$PAMAP2_H$	$PAMAP2_C$	$PAMAP2_A$
1-NN		DCT	0.5619	0.6523	0.6561
3-NN		DCT	0.5785	0.6633	0.6577
SVM		DCT	0.2362	0.4313	0.3253
NN	ANN	Shallow	0.7160	0.7470	0.7900
		Deep	0.6690	0.7505	0.7485
	CNN	1D-CNN	0.6970	0.7565	0.7763
	LSTM	1D-CNN-LSTM	**0.7383**	**0.7782**	**0.8034**

Table 4.4 presents the results for PAMAP2 modalities for performing ambulatory, sedentary and daily living activities. All three modalities achieve the best performance with the 1D-CNN-LSTM architecture. Ankle sensor achieves the best performance while the hand is the noisiest modality. These observations are in par with the MEx and SELFBACK accelerometer performances. In contrast to SELFBACK modalities, SVM and k-NN methods fail to achieve comparable performances against NN architectures.

In summary, we find the most optimal feature representation learners for a wide range of sensor modalities in three different activity domains: exercise recognition, general fitness and activities of daily living. Importantly the results with the MEx dataset showed that heterogeneous modalities call for bespoke feature learners to highlight their inherent characteristics that may otherwise get overlooked. Moreover, inertial sensors preferred the same feature learner for all modalities, which was the 1D-CNN-LSTM architecture.

4.6.2 Comparison of Modality Combinations

Lara and Labrador (2012) highlighted the importance of identifying minimal modality combinations to implement user-friendly and energy-friendly recognition algorithms. Both Lara and Labrador (2012) and Anjum and Ilyas (2013) observed that some sensor-rich datasets do not necessarily benefit from a large number of sensors. In the ExRec domain, using the required minimum number of sensor modalities will contribute towards developing an economical solution with improved usability and deployability in restricted home settings. Accordingly, we aim to empirically identify the sets of minimal sensor modality combination. We consider MEx and PAMAP2 datasets and exempt SELFBACK because there is only one multi-modal combination.

Best performing modality-specific feature learners from the previous section are used to create a bespoke MHAF architecture for each modality combination. All MHAF architectures were implemented using Keras and TensorFlow libraries for Python and were trained end-to-end for 100 epochs using AdaDelta optimiser to minimise the categorical cross-entropy loss. Each experiment follows the LOPO evaluation methodology and report the mean F1-score averaged over person folds.

MEx Modality Combinations

Modalities in the MEx dataset create 11 modality combinations by considering 2 modality combinations (6 combinations) and 3 modality combinations (4 combinations) and the default 4 modality combination. In this evaluation we create their respective MHAF architectures to identify the best 2 modality combination and the best 3 modality combination for unobtrusive deployment. Given the heterogeneous modalities in MEx, in this evaluation, the unobtrusiveness is only considering the number of sensor modalities not the type of modalities.

Table 4.5: MHAF performance comparison with MEx modality combinations

m	Modality Combination	F1-measure
2	ACW and PM	0.8466
	ACT and ACW	0.8688
	PM and DC	0.9041
	ACW and DC	0.9125
	ACT and DC	0.9276
	ACT and PM	**0.9354**
3	ACW, DC and PM	0.9147
	ACT, ACW and PM	0.9485
	ACT, ACW and DC	0.9421
	ACT, DC and PM	**0.9624**
4	**ACT, ACW, DC and PM**	**0.9584**

Table 4.5 presents the performances of MHAF against modality combinations, grouped by the number of modalities. Importantly, the 3M combination ACT, PM and DC

achieves the best performance, outperforming the default 4 modality combination. We refer to Table 4.2, which indicates that removing the noisiest modality (i.e. least performing) enhanced the overall performance. Closer examination of 2M combinations confirms that sensors that are more prone to noise such as the on-body wrist accelerometer is detrimental to fusion in a modality rich environment. For instance, the wrist and thigh accelerometer combination perform at 0.8688, which is lower than the best thigh accelerometer performance, which was 0.9015 (Table 4.2). But the best performing 2M combination MEx$_{ACT,PM}$ outperform best single modality performance of ACT by 3.39%. Best performing 2 modality combination is ACT and PM, but ACT and DC is a close second. It is noteworthy that although ACT and DC are the best performing modalities individually, the modality combination ACT and PM has learned a more effective multi-modal fusion. Accordingly, we find that ACT and PM are a complementary modality combination which provides a wider sensory coverage more suitable for the selected exercise classes.

PAMAP2 Modality Combinations

Table 4.6: mHAF performance comparison with PAMAP2 modality combinations

m	Modality Combination	F1-measure
	PAMAP2$_{HC}$	0.8462
2	**PAMAP2$_{HA}$**	**0.9004**
	PAMAP2$_{CA}$	0.8453
3	**PAMAP2$_{HCA}$**	**0.9070**

Table 4.6 presents the results for modality combinations derived from PAMAP2 modalities. There are three 2 modality combinations in addition to the default 3 modality combination. Best 2 modality combination is the hand and ankle accelerometer combination performing at 0.9004. Although chest and ankle are the best performing modalities individually, hand and ankle combination has learned a more effective multi-modal fusion. This observation indicates that the wider sensory coverage created by the hand and ankle combination is suitable for PAMAP2 activities. Notably, the best 2 modality combination performance is comparable to the default three modality combination. Each modality combination has out-performed the best performing single sensor, which is A (0.8034). The addition of H modality has improved the best performing single modality performance (0.8034) by 9.7%. However, the addition of the C modality has failed to significantly improve the best 2M performance.

It can be argued that these modality combinations can be learnt by a deep architecture where the input is all available modalities. Such deep architecture needs to be *very deep* with many trainable parameters, which would typically require a large quantity of training dataset. As stated in Section 4.1, one of the consideration when designing mHAF is to mitigate the demand for data, using attention fusion. Accordingly, we empirically explore the modality combinations to reduce the burden on the fusion architecture. As a result, mHAF is a minimal and shallower architecture that is trainable with a comparably smaller quantity of training data.

4.6.3 Ablation Study

An ablation study is performed to evaluate the contribution of each module of the mHAF architecture. We take forward the best modality specific feature learners from Section 4.6.1 and best modality combinations from Section 4.6.2 in these experiments. Ablation study is commonly used to verify the contribution and necessity of each architecture component to any performance improvements achieved in addition to adding parametric complexity. Accordingly, we decompose our mHAF architecture as in Figure 4.4 to create the five variants listed below.

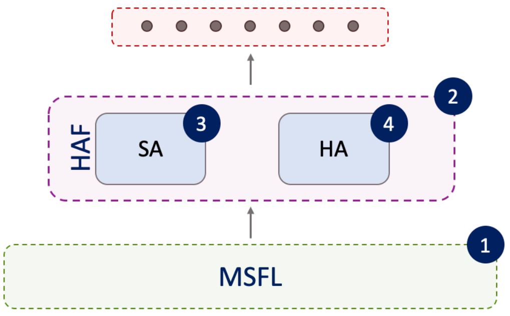


Figure 4.4: An abstract modular view of the mHAF architecture

mHAF-noMSFL-noHAF: mHAF without the HAF module (2) and modality-specific feature learners (1). A 1D-CNN-LSTM feature learner is used by each individual modality and concatenated at a late-fusion level.

mHAF-noMSFL: mHAF without modality-specific feature learners (1). A 1D-CNN-LSTM feature learner is used by each individual modality before applying hybrid attention fusion.

mHAF-noHAF: mHAF without the HAF module (2). Modality specific feature representations are concatenated at a late-level.

mHAF-noSA: mHAF without the SA module (3) in HAF, only the HA module

mHAF-noHA: mHAF without the HA module (4) in HAF, only the SA module

The first two variants evaluate the significance of modality-specific feature representations, and the last three variants evaluate the significance of attention module and its components in the mHAF architecture. The 1D-CNN-LSTM architecture used in the first two variants is from Table 4.1 and we adjust the input and layer sizes to suit the input modality. For implementation and evaluation details we refer to Section 4.6.2.

Table 4.7: mHAF performance compared against ablated variants with MEx

Algorithm	2M	3M	4M
mHAF-noMSFL-noHAF	-	0.9228	0.9370
mHAF-noMSFL	-	0.9417	0.9394
mHAF-noHAF	0.9057	0.9378	0.9465
mHAF-noSA	0.9075	0.9495	0.9525
mHAF-noHA	0.9036	0.9345	0.9425
mHAF	**0.9354**	**0.9624**	**0.9584**

Table 4.7 detail the results for MEx modality combinations where 2M, 3M and 4M refers to MEx$_{ACT,PM}$, MEx$_{ACT,PM,DC}$ and MEx$_{ALL}$ modality combinations. We note that the first two variants do not apply to 2M as both modalities preferred the same feature learner (Section 4.6.1). Overall, mHAF has significantly outperformed all other variants with all three modality combinations.

Removing the HAF attention module has decreased the performance significantly: 2.97%, 2.46% and 1.19% for 2M, 3M and 4M configurations respectively. Removing the modality-specific feature learners has also significantly decreased overall performance: 2.07% and 1.90% for 3M and 4M configurations.

HA and SA contributions are comparatively similar for the MEx$_{ACT,PM}$ modality combination but more distinct in the 3M and 4M settings. HA module contribution is measured with the mHAF-noSA variant, and it is significantly larger compared to the SA module. When the HA module is removed (mHAF-noHA), performance is dropped by 1.5% and 1% for 3M and 4M. In comparison, the removal of the SA only drop performance by 1.29% and 0.59% for 3M and 4M. Notably, mHAF-noSA vs mHAF difference is only 0.59% with 4M configuration. This observation suggests that removing SA module which promote many features, is not significantly detrimental to the performance when noisy modalities are present.

Overall, we observe performance improvements when both SA and HA modules are present in the MHAF architecture. These results suggest that MHAF learns an effective fusion of more than one modality in a heterogeneous multi-modal setting.

Table 4.8: MHAF performance compared against ablated variants with SELFBACK and PAMAP2

Algorithm	PAMAP2$_{HA}$	PAMAP2$_{HCA}$	SB$_{WT}$
MHAF-noHAF	0.8850	0.8975	**0.7989**
MHAF-noSA	**0.9003**	**0.9091**	**0.8027**
MHAF-noHA	**0.8992**	**0.9037**	0.7921
MHAF	**0.9004**	**0.9070**	0.7919

Table 4.8 presents the ablation study results obtained for the two PAMAP2 modality combinations, PAMAP2$_{HA}$ and PAMAP2$_{HCA}$ and the SELFBACK modality combination SB$_{WT}$. We note that the first two variants are not applicable for PAMAP2 and SELFBACK modality combinations that preferred the same feature learner (Section 4.6.1). PAMAP2 modality combinations show similar results to MEx modality combinations when the HAF module is removed. There is a significant performance decline, 1.54% and 0.95% respectively for PAMAP2$_{HA}$ and PAMAP2$_{HCA}$ modality combinations. We fail to observe any significant performance differences between MHAF-noSA, MHAF-noHA and MHAF architectures with the PAMAP2 modality combinations. In contrast, SELFBACK modality combination of wrist and thigh accelerometers fail to improve performance using the HAF module. Instead, the best performance is observed with the MHAF-noSA module architecture. These results suggest that, with soft attention (SA module), many noisy features from wrist accelerometer are promoted that is detrimental to the classification performance.

In summary, we find that MHAF architecture is best tailored for heterogeneous multi-modal settings. With homogeneous multi-modal combinations, MHAF perform comparatively similar to ablated variants. The results also highlight that the inclusion or exclusion of the SA module is an important design consideration when working with modalities that are noisy. To further validate this hypothesis, we compare ablation study results for MEx$_{ACW,ACT}$ combination where ACT is the least noisy modality, and ACW is the noisiest modality. We report F1-scores of 0.7871, 0.8786 and 0.8021 for MHAF-noHAF, MHAF-noSA and MHAF-noHA variants while MHAF architecture achieve 0.8688. As expected, none of the fusion architectures outperforms the ACT performance of 0.9015 and MHAF-noSA (only HA) architecture achieves the best fusion performance.

4.7 Chapter Summary

In this chapter, we introduced the heterogeneous multi-modal attention fusion architecture, mHAF. Our goal was to overcome the limitations of existing multi-modal fusion architectures such as lack of support for multi-modal heterogeneity and parametric complexity. We followed a modularised approach where each module has a distinct purpose. First, modality-specific feature learners are identified for each modality to provide the best opportunity to learn the most effective feature representations. Secondly, an attention fusion module was introduced to create a fusion feature vector efficiently avoiding the need for a very deep architecture. We apply attention at the feature-level granularity, providing each modality feature equal opportunity to contribute towards improved performance.

Our empirical evaluation was designed to identify the best design components of the mHAF architecture bespoke to several modality combinations. First, we identified the best feature learners for modalities to form bespoke mHAF architectures for different modality combinations. Using all modalities can be obtrusive, economically infeasible and discouraging to users. Accordingly, we identified the best performing minimal modality combinations in addition to the default modality combination suitable for deployment in restricted settings. The modularised architecture of mHAF allowed effortless amendments of modules to suit different modality combinations. We further verified our fusion architecture with an ablation study, which highlighted the significance and the necessity of each component of the mHAF architecture for heterogeneous multi-modal fusion.

Chapter 5

Personalised Recognition with Meta-learners

Activities performance patterns differ from person to person. Personal characteristics such as gait pattern, preferences, physiology and nuances are few factors that affect these unique activity patterns. The same activity can be performed differently by two persons, or by the same person over time. Some activity domains are more susceptible to personal differences than others. Exercises or activities of daily living are such complex activities performed with more personal nuances compared to ambulatory activities like walking or jogging. Ability to adapt to new persons and personal changes is essential to the successful deployment of a HAR algorithm.

We argue that it is more intuitive to treat a *person-activity* pair as the class label, where we view each person's data as a dataset in its own right. Inspired by the personalised few-shot learning and meta-learning research, in this Chapter, we introduce a meta-learning methodology for personalised HAR. We model personalised HAR as a meta-learning problem and show how to train a generalised meta-model that it is adaptable by an unseen person encountered after deployment.

5.1 Use Case

We present a detailed use case of personalised meta-learners for HAR implemented within a fitness application illustrated in Figure 5.1. The fitness application can recognise four activity classes, walking, dancing, running and sitting using a single accelerometer sensor.

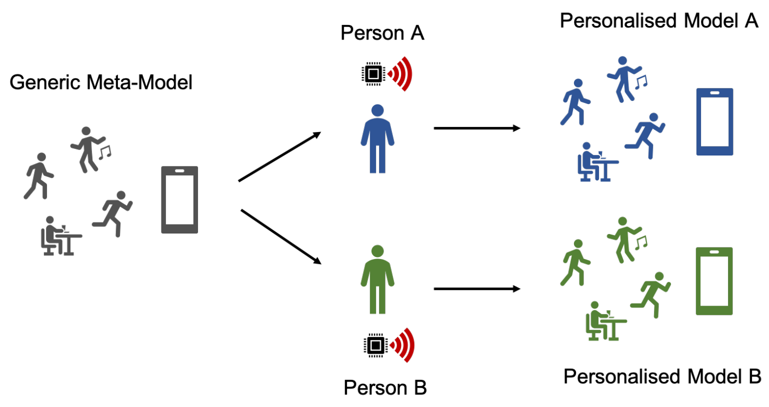


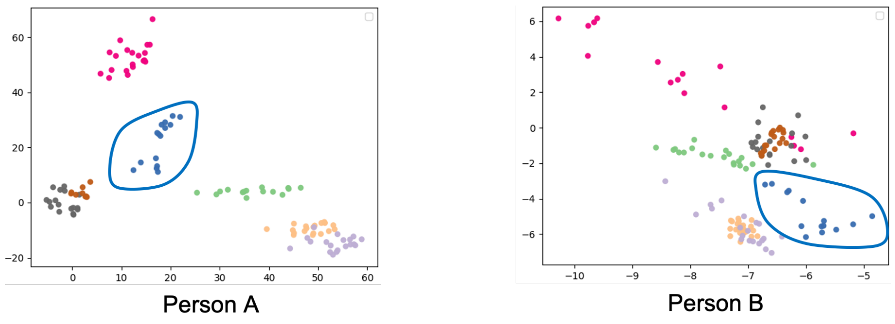
Figure 5.1: A use case of personalising activity recognition

It is embedded with an activity recognition meta-model trained using the personalised meta-learning methodology introduced in this Chapter.

Imagine person A (blue), who is young, physically active and healthy, downloads the fitness application to their mobile phone. They find that the generic model annotates their walking as running most of the time because naturally, they walk faster. Therefore, person A wants to calibrate the generic model such that it is personalised to their walking and running rhythms. They record few seconds of calibration data for each activity using sensors recommended by the fitness application. Subsequently, the fitness application is personalised using the calibration data to recognise these activities in the future.

Person B (green), an elderly who enjoys dancing, finds that the fitness application needs to be personalised to their style of dancing and to their walking rhythm. Similar to person A, they record few seconds of data for both activities, and thereafter, the fitness application is personalised with the new dancing and walking data. Both persons have the opportunity to further personalise the fitness application by adding calibration data for other activities.

Accordingly, when a meta-learner is trained and embedded in the fitness application, there is an initial personalisation step to provide calibration data. The end-user will be instructed to record few seconds of data for each activity using the sensor modalities synchronised with the fitness application. This step is similar to demographic configurations users perform when installing new fitness applications (on-boarding). Thereafter, the personal data will be used by the algorithm for personalisation. Importantly, personalised meta-learners provide the opportunity to provide new calibration data if the physiology or preferences of the person change over time. It is also noteworthy that personalisation of all activities is not required but is recommended over personalising a subset of activities.



(a) Feature space of person A (b) Feature space of person B

Figure 5.2: PCA 2D feature space comparison between two persons performing the same set of 7 exercises

We visualise data from two persons in the MEx dataset performing 7 exercises in Figures 5.2 to highlight personal differences. This data is captured with the pressure mat and applied PCA dimensionality reduction to select the 2 most significant PCA coefficients. Different colours indicate different exercise classes. Notably, the data instances of an exercise (such as the exercise denoted with blue dots) are distributed in different feature spaces for Person A and B. Accordingly, we treat activity recognition of a person as a recognition problem in itself (i.e. person-task), and a *person-activity* pair as a distinct class label. We view each person's data as a HAR dataset with a limited number of data instances per *person-activity* class which resembles a few-shot learning scenario. To learn a personalised reasoning model from few data, we view HAR as a personalised meta-learning problem.

5.2 Personalised Meta-learning

The goal of a personalised meta-learning is to learn a reasoning model (i.e. meta-model) that can adapt to any person encountered after deployment. In the meta-learning setting, recognition of activities of a person is considered a task (i.e. person-task). Accordingly, a meta-model is trained on many person-tasks and is tested on person-task(s) not seen during training. Importantly, a personal HAR task is considered a few-shot classification task to minimise the burden on data collection. This section first presents the personalised meta-learning task design and then how to implement this methodology with three meta-learning algorithms from recent literature.

5.2.1 Personalised Meta-Learning Task Design

The task design for personalised meta-learning is illustrated in Figure 5.3. Given a HAR dataset of population $\mathcal{P}$, we create tasks such that, each *person-task*, $\mathcal{P}_i$, only contains data from a specific person, p where $p \in \mathcal{P}$. A *person-task*, $\mathcal{P}_i$, consists of a set of training and test data referred to as the support set, $\mathcal{D}^s$, and the query set, $\mathcal{D}^q$. The support set consist of k^s amount of representatives for each class randomly selected stratified across activity classes such that $|\mathcal{D}^s| = k^s \times |\mathcal{C}|$. Similarly, the query set, $\mathcal{D}^q$, is selected with k^q amount of representatives for each class such that $|\mathcal{D}^q| = k^q \times |\mathcal{C}|$. Accordingly, a person-task is a few-shot dataset with train and test data that can learn a person-task model using the support set. Instead of learning an independent model, meta-learning aggregates the learning experiences of many such person-task models to produce a generalised meta-model, θ.

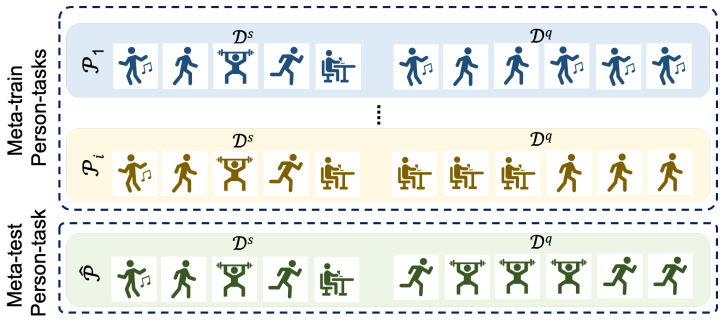


Figure 5.3: Personalised meta-learning task design

A meta-test person-task, $\hat{\mathcal{P}}$, is created similar to a meta-train person-task. In practice, a test person, $\hat{p}$, provides few seconds of labelled data instances for each activity class to form the support set, $\mathcal{D}^s$ using recommended sensor modalities. The generalised meta-model, θ is optimised for the test person using the support set, and the resulting model, $\hat{\theta}$ is personalised to $\hat{p}$. Thereafter, $\hat{\theta}$ is used to predict the class labels for any query data instance presented by $\hat{p}$ (i.e. elements of $\mathcal{D}^q$ in $\hat{\mathcal{P}}$).

We view personalised HAR as a few-shot classification problem with a $|\mathcal{C}| \times |\mathcal{P}|$ number of classes. Notably, each *person-task* is learning to classify the same set of activity classes $\mathcal{C}$, but from different persons. The goal of applying personalised meta-learning to solve personalised HAR is to learn the most generalised model adaptable to any new unseen person.

5.2.2 Personalised Meta-learning Algorithms

This section presents how 3 meta-learning algorithms implement the personalised meta-learning methodology. Adaptation-optimised and similarity-optimised meta-learners are selected to implement the personalised meta-learning methodology for HAR. As described in Section 2.4.1, both categories have the flexibility to use custom feature learners to suit heterogeneous modality settings encountered in HAR applications. Model-optimised meta-learners are not considered for personalised HAR as their performance is reliant on using deep feature learners (Mishra et al., 2018).

Personalised MAML

MAML^p is the variant of MAML (in Section 2.4.2) which implements the personalised meta-learning methodology from Section 5.2.1. MAML^p for HAR learns the generalised model, θ, such that it can be optimised to any new unseen person encountered at test time given only few samples of labelled data.

In comparison to MAML, a meta-model learned from MAML^p is conditioned on person-tasks. At each iteration, a set of persons are sampled from the population, $\mathcal{P}$, to create a set of person-tasks, $\{\mathcal{P}_i\}$, as the meta-train set. Sampling is applied with replacement, where the same person, p, may appear more than once in a meta-train set. For instance, if the sample size, n is larger than the population size, $|\mathcal{P}|$, a person may contribute to more than one *person-tasks*, each with unique support sets and query sets. We consider $n = 32$, and this method emulates a slightly larger population, with the opportunity to learn from few different variations of the same person.

For each person-task, a support set and a query set is selected according to the personalised meta-learning task design discussed in Section 5.2.1. Training of a person-task model, θ_i, is similar to MAML as detailed in Section 2.4.2. At the end of person-task training, θ_i, is now adapted for $\mathcal{P}_i$. In personalised meta-learning, data instances in $\mathcal{D}^q$ is selected stratified across all activity classes such that the learning experience (i.e. loss) is representing all activity classes. Next, similar to MAML, the collective losses from all person-tasks are used to train the meta-model. For an unseen meta-test person $\hat{p}$, a meta-test person-task is created as $\hat{\mathcal{P}}$. A model, $\hat{\theta}$ is initialised from the meta-model and is adapted using the support set, $\hat{\mathcal{D}^s}$. Once $\hat{\theta}$ is optimised for $\hat{\mathcal{P}}$, it is used to predict class labels of the query set, $\hat{\mathcal{D}^q}$. Unlike meta-train tasks, in practice, the composition of $\hat{\mathcal{D}^q}$ is not pre-determined to be stratified across class labels.

Gradient steps (gs) and meta-gradient steps ($meta\_gs$) are the number of epochs that optimise person-task models. If a person carries pronounced personal nuances, learning a fully optimised person-task model can be detrimental during meta-training. Once, fully optimised, the loss calculated with the query set will be smaller, which it is contributing to learning a generalised meta-model. As a result, the collective loss from all person-tasks will be smaller, incorrectly indicating that the meta-model is optimised. Accordingly, we select gs and $meta\_gs$ such that $gs < meta\_gs$. As a result, during training, the person-task model, θ_i is not fully optimised for the person-task. Through this optimisation, the meta-model is learning commonalities between many person-tasks. The resulting meta-model has two properties: it is the best starter model for *any new unseen person*; and once personalised, the person-task model is the best recognition model *for the person*.

Personalised Relation Networks

RN^p, is the variant of RN (in Section 2.4.2) which implements the personalised meta-learning methodology from Section 5.2.1. With the personalisation methodology, the similarity learning of RN is conditioned on personal nuances. Activity representatives for similarity comparison are selected from the same person to provide personalisation context, while through iterative optimisation, the meta-model learns commonalities between different persons.

As discussed in Section 2.4.2, RN learns to find the best matching. Moreover, the goal of a personalised relation network meta-model is to find the best matching given a specific person's data. A person-task, $\mathcal{P}_i$ is formed for the randomly selected person p, with a support set, $\mathcal{D}^s$ and a query set, $\mathcal{D}^q$ according to the personalised meta-learning methodology detailed in Section 5.2.1. Training data instances, $(x_i^q, \mathcal{D}^s)$, and the input to the relation module, (x_i^q, x_j^s) are created similar to original RN, but now against their own data in $\mathcal{P}_i$. Accordingly, with the personalised methodology, the relation score is always predicted against ones own data. Similar to MAML^p, the same person can be sampled as person-tasks with different support sets and query sets, allowing the network to learn from a few variations of the same person. These personalisation constraints will enable the network architecture to capitalise on personal characteristics by learning on conditioned person-tasks, instead of generic tasks.

A meta-test person $\hat{p}$, not seen during training, can use a RN^p meta-model to match a query instance to an instance in their own support set. With RN^p, there is no explicit adaptation step as in MAML^p. However, by using the personal support set in conjunction with every query, personalised relation network is optimising its learning to a new unseen person.

Originally, RN was implemented for few-shot image classification (Sung et al., 2018) with a 2D convolutional network architecture for feature learning (θ_f) and another 2D convolutional architecture for relation learning (θ_r). In our initial experiments we failed to successfully train an RN network with alternatives for the relation learner such as 1D convolutional networks or dense networks. This is an indication that the training of RN is highly dependent on the 2D convolutional relation learner. Accordingly, we use two 2D convolutional architectures for relation learning in single modality and multi-modal settings as in Table 5.1. We refer to Section 3.1 for the notations used in Table 5.1. The output layer of θ_r is of size $\mathbb{R}^1$ with sigmoid activation (logistic) to predict a similarity value between 0 and 1.

Table 5.1: Relation learner architecture details

Setting	Architecture
Single-modal	$conv(3 \times 3)64 \rightarrow maxpool(2 \times 2) \rightarrow bn \rightarrow dense(120, Relu) \rightarrow bn \rightarrow dense(1, Sigmoid)$
Multi-modal	$conv(3 \times 3)64 \rightarrow maxpool(2 \times 2) \rightarrow bn \rightarrow dense(1200, Relu) \rightarrow bn \rightarrow dense(120, Relu) \rightarrow bn \rightarrow dense(1, Sigmoid)$

Personalised Matching Networks as a Meta-learner

By applying the personalised meta-learning methodology introduced in Section 5.2.1, we improve MN^p by Sani et al. (2018) (detailed in Section 2.2.2), such that the model training creates a generic meta-model. During training we use the task design where k^q is $k - k^s$ where k is the total number of data instances available for the *person-activity*. Accordingly, there are $k^q \times |\mathcal{C}|$ amount query instances in $\mathcal{D}^q$. At each epoch, one person-task is randomly selected, which creates a $k^q \times |\mathcal{C}|$ amount of training instances using a fixed support set. This method ensures that the meta-model is optimised for the specific person-task, which is represented by $k^q \times |\mathcal{C}|$ amount of query instances instead of 1. We refer to this version of personalised matching networks as MN^{p*}. With the personalised meta-learning methodology, we have limited the number of variants of the same person seen by the model. Accordingly, the network learns to discriminate the personal traits from activity classes which improves generalisability of the meta-model.

5.3 Evaluation

In this section, we perform an empirical evaluation of design hyper-parameters to fine-tune personalised meta-learners: MAML^p, RN^p and MN^p*. A shallow feature learner is essential to avoid over-fitting in a few-shot classification setting. Accordingly, we explore an array of shallow feature learners with the view to selecting the most effective yet, economical for each meta-learner in Section 5.3.1. A small support set size is a key factor that improves the deployability of a personalised algorithm with limited memory and computational requirements. We explore a range of support set sizes and observe the impact on meta-model learning and personalisation to select the most optimal considering performance and unobtrusive deployment (Section 5.3.2). Since we have access to the demographic data, MEx dataset is selected to perform these fine-tuning evaluations. MEx dataset is compiled with 30 persons from different demographics where each persons data emulates a few-shot dataset. And each person's contribution is approximately balanced, to avoid any personal bias. We refer to Section 7.1.4 for more details.

5.3.1 Comparison of Feature Learners

The three personalised meta-learners introduced in Section 5.2.1 are model agnostic, such that the meta-learning is independent of the feature learners. In literature, meta-learners are not applied in the HAR domain and not evaluated with heterogeneous sensor data to refer to when selecting the most effective feature learners. Accordingly, as a refinement step, we aim to find the best performing feature learner for different modalities. In this section, we consider a list of shallow feature learners as listed in Table 5.2 to observe the impact on performance. We refer to Section 3.1 for the notations used in Table 5.2.

Table 5.2: Feature learner

Model	Architecture
DNN(1)	$dense(1200) \rightarrow bn$
DNN(3)	$dense(1200) \rightarrow bn \rightarrow dense(640) \rightarrow bn \rightarrow dense(120) \rightarrow bn$
1D-CONV(1)	$conv(5)64 \rightarrow bn \rightarrow maxpool(2)$
1D-CONV(3)	$conv(5)64 \rightarrow bn \rightarrow maxpool(2) \rightarrow conv(5)64 \rightarrow bn \rightarrow maxpool(2) \rightarrow conv(5)64 \rightarrow bn \rightarrow maxpool(2)$
2D-CONV(1)	$conv(3,3)64 \rightarrow bn \rightarrow maxpool(2,2)$
2D-CONV(2)	$conv(3,3)64 \rightarrow bn \rightarrow maxpool(2,2) \rightarrow conv(3,3)64 \rightarrow bn \rightarrow maxpool(2,2)$

Implementation Details

Given a dataset, we apply the personalised meta-learning methodology to create person-tasks in the $k^s = 5$ setting. To create a person-task, for each *person-activity* class, we randomly select a support set and use the remaining data instances as the query set (i.e. $k^q = k - k^s$). We use four single modal MEx datasets for this evaluation. Input to each feature learners is adjusted to suit the modality dimensions. Details of the input dimensions of each feature learner for the four MEx modalities are detailed in Table 5.3. Note that 2D convolutional feature learners are not used to learn from DCT feature transformed accelerometer data, and also all convolutional architectures consider input data to have 1 channel.

All models are implemented using the Python library Pytorch (Paszke et al., 2019). MAML^p and MN^{p*} are using the categorical cross-entropy loss, and RN^p uses mean squared error loss. All models are trained using the Adam optimiser without mini-batching. MAML^p, RN^p and MN^{p*} are trained for 100, 300 and 200 epochs respectively. For MAML^p, we use 5 gradient steps and 10 meta-gradient steps (i.e. $gs = 5$ and $meta\_gs = 10$) and 32 person-tasks are sampled for each meta-training epoch ($n = 32$).

We follow the LOPO evaluation methodology, as described in Section 3.4. In a given fold, with a MEx dataset, there are 29 persons in the train set to create meta-train tasks and one person in the test set to create meta-test tasks. To account for intra-personal variations and non-deterministic nature of deep learners, we repeat testing with 100 test-tasks created from the test person. All meta-train and meta-test tasks are created while maintaining class balance, accordingly, we report the accuracy of each experiment averaged over person folds.

Table 5.3: Feature learner input dimensions with MEx modalities

Modality	Architecture		
	DNN	1D-CONV	2D-CONV
MEx$_{ACT}$/MEx$_{ACW}$	(5×180)	$(5, 180)$	-
MEx$_{DC}$	$(5 \times 12 \times 16)$	$(5 \times 12, 16)$	$(5, 12, 16)$
MEx$_{PM}$	$(5 \times 16 \times 16)$	$(5 \times 16, 16)$	$(5, 16, 16)$

MAML^p Results

Table 5.4 details the results obtained for the MAML^p algorithm. All four experiments achieve the best performance using the three-layer dense feature learner, DNN(3). Notably, both visual and time-series modalities achieve significant performance improvements with dense architectures compared to convolutional architectures. All four modalities found an increased number of layers in the dense architectures to be advantageous. Notably, 1D-CONV(1), 1D-CONV(3) and 2D-CONV(1) feature learners fail to successfully learn meta-models.

Table 5.4: MAML^p performance comparison using different feature learners

Feature Learner	MEx$_{ACT}$	MEx$_{ACW}$	MEx$_{DC}$	MEx$_{PM}$
DNN(1)	0.9103	0.6825	0.9774	0.9377
DNN(3)	**0.9713**	**0.8399**	**0.9857**	**0.9618**
1D-CONV(1)	0.2545	0.1827	0.1681	0.2521
1D-CONV(3)	0.4406	0.1668	0.3141	0.3478
2D-CONV(1)	-	-	0.3409	0.4433
2D-CONV(2)	-	-	0.9605	0.9398

RN^p Results

Table 5.5 details the results obtained for the RN^p algorithm. Overall, time-series data achieve the best performance with the three-layer dense architecture, DNN(3). Visual data modalities achieve the best performance with one-layer 2-dimensional convolutional architecture 2D-CONV(1). Similar to MAML^p, dense architectures benefit from an increased number of layers. In contrast, convolutional architectures find multiple layers detrimental to overall performance, indicating that feature learners with multiple convolutional layers over-fit to training data (see 2D-CONV(1) vs 2D-CONV(2)).

Table 5.5: RN^p performance comparison using different feature learners

Feature Learner	MEx$_{ACT}$	MEx$_{ACW}$	MEx$_{DC}$	MEx$_{PM}$
DNN(1)	0.9444	0.6899	0.8533	0.7553
DNN(3)	**0.9596**	**0.7444**	0.8525	0.7857
1D-CONV(1)	0.8972	0.7149	0.1768	0.9059
1D-CONV(3)	0.7368	0.5731	0.8892	0.8931
2D-CONV(1)	-	-	**0.9562**	**0.9229**
2D-CONV(2)	-	-	0.8968	0.8269

MN^{p}* Results

Performance of MN^{p}* using the list of feature learners are presented in Table 5.6. All 4 experiments achieve the best performance using dense architectures: DNN(3) achieves the best performances for time-series modalities; and DNN(1) achieves the best performances for visual modalities. While time-series data benefit from an increased number of layers in the dense architectures, visual data achieve the best performance with the single-layer dense architecture. Convolutional architectures fail to outperform dense architecture and also an increased number of convolutional layers is found to be detrimental to the overall performance.

Table 5.6: MN^{p}* performance comparison using different feature learners

Feature Learner	MEx$_{ACT}$	MEx$_{ACW}$	MEx$_{DC}$	MEx$_{PM}$
DNN(1)	0.9465	0.7438	**0.9606**	**0.9289**
DNN(3)	**0.9695**	**0.7917**	0.8868	0.8593
1D-CONV(1)	0.8625	0.6506	0.9228	0.9150
1D-CONV(3)	0.7775	0.5743	0.8650	0.8704
2D-CONV(1)	-	-	0.9089	0.8641
2D-CONV(2)	-	-	0.8209	0.8255

In summary, we find that all three meta-learners achieve competitive performance with dense architectures as the feature learner. The notable exception is personalised RN with visual data achieving the best performance with a 2D convolutional architecture.

5.3.2 Explore support set size

In this section, an empirical study is performed to observe the effect of the support set size on personalised meta-learners. Finding the balance between k^s and model performance is essential because at deployment, a test-person is expected to provide a k^s amount of data instances per activity class for personalisation. Therefore it is desirable to keep k^s to a required minimum.

We perform a set of experiments using the four MEx single modality datasets for a range of k^s values, 1, 3, 5, 7 and 10. Each experiment uses k^s amount of instance per *person-activity* in the support set and the rest (i.e. $30 - k^s$) in the query set. The performance is measured using the mean accuracy averaged over 30 person-folds. For more implementation and evaluation details we refer to Section 5.3.1.

MAML^p Results

Figure 5.4 plots the meta-test accuracy obtained from MAML^p models trained using different k^s values. Overall, increasing k^s has consistently improved performance up to $k^s = 10$ with all experiments. Significant performance improvement is observed when increasing k^s from 1 to 3 with all experiments, precisely 5.49%, 8.14 %, 12% and 11.73% with ACT, ACW, DC and PM respectively. Highest improvements when increasing the support set from $k^s = 1$ to $k^s = 10$ is observed with the PM and DC experiments; 16.52% and 15.82% respectively. These results show the advantage of learning from many labelled examples, and it is most significant for complex visual modalities. We find $k^s = 5$ is a balanced choice for creating the support set across all modalities. Accordingly, for personalisation, a test-person only need to provide approximately 15 seconds of labelled data for each activity. This is according to the pre-processing steps discussed in Section 3.3.1 where the window size is 5 seconds and the overlap of 3 seconds.

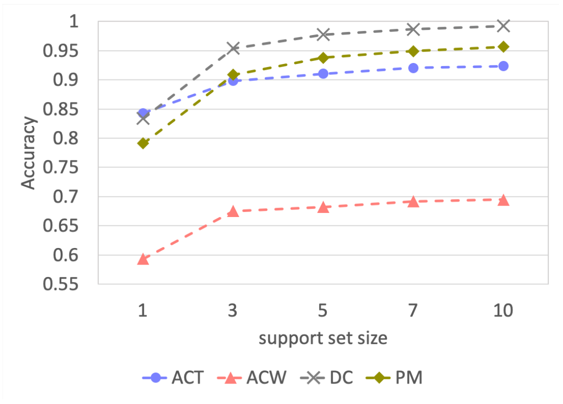


Figure 5.4: Exploration of support set size, k^s for MAML^p

RN^p Results

Performances obtained for RN^p models trained using different k^s sizes are plotted in Figure 5.5. In contract to MAML^p, we do not observe a consistent performance improvement with increasing k^s values. ACT, ACW and DC experiments found more than one representatives per *person-activity* is to be detrimental to the overall performance. For instance, the best performance for ACW is reported when $k^s = 1$ and the best performances for ACT and DC are reported at $k^s = 3$. In contrast, PM experiment is not penalised when increasing k^s. However, the performance gain is not significant when increasing k^s from 3 to 10.

With a larger k^s, the number of elements to be compared at the relation module increase. Moreover, it increases the number of data instances needs to be kept in memory to perform comparisons in the RN^p algorithm. We find $k^s = 3$ to be the most optimal setting across all four experiments which minimise memory requirement, yet achieve significantly improved performance. Accordingly, to personalise an RN^p model, a test-person only need to provide 9 seconds of labelled data for each activity.

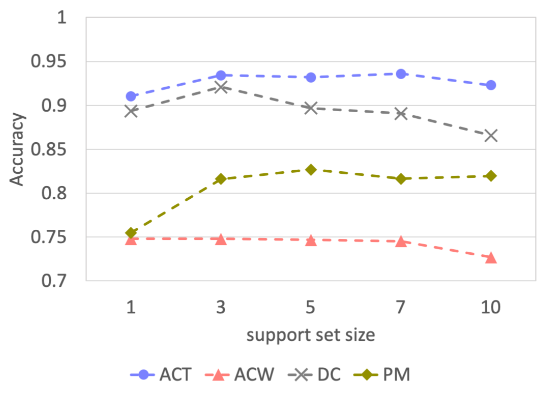


Figure 5.5: Exploration of support set size, k^s for RN^p

MN^p* Results

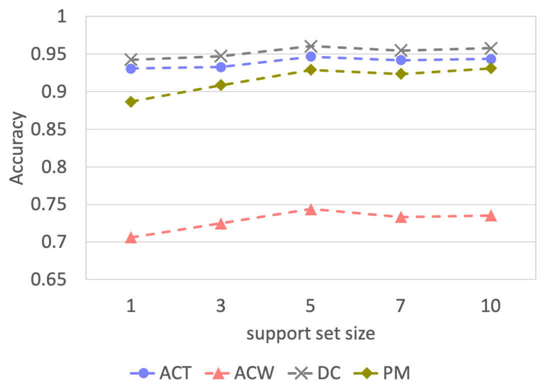


Figure 5.6: Exploration of support set size, k^s for MN^p*

Figure 5.6 presents the mean accuracy of MN^p* models trained with k^s values 1, 3, 5, 7 and 10 for the four experiments All four experiments show a similar pattern with increasing k^s values. We attribute this deterministic behaviour to the use of static similarity function. Overall, the best performance is obtained with $k^s = 5$. In contrast to RN^p, more than 5 data instances per *person-activity* has not been significantly detrimental to the overall performance. Instead, comparable performances are achieved in in $k^s = 5, 7$ and 10 settings. In addition, with all experiments, we observe that performance consistently improve when increasing $k^s = 1$ to $k^s = 5$. For MN^{p}*, we select $k^s = 5$ to be the most optimal setting. Accordingly, a test-person need to provide approximately 15 seconds of labelled data for personalisation.

5.3.3 Optimising Personalised Relation Networks

In this section, we explore three loss functions and observe the optimisation process of personalised RN. Originally RN is modelled as a regression task using mean squared error loss where each pair predicts a similarity score. But RN can be modified to perform a classification task or a metric learning task. Accordingly, we view personalised RN as a classification task using categorical cross-entropy and as a metric learning task using a custom distance-based loss function inspired by the Triplet loss (Hoffer and Ailon, 2015). The loss functions considered are detailed below:

MSE: Originally RN is trained as a regression task using mean squared error. For each *support set element, query instance* pair, the relation module predicts a similarity score (a scalar value) and the expected score is either 1 (matching pair) or 0 (non-matching pair). For a given training data instance, $(\mathcal{D}_s, (x^q, y^q))$, the collective loss is calculated from all pairs as in Equation 5.1. Here $sim(y_i^s, y^q)$ is the true label which is 1 if $y_i^s = y^q$ and 0 if $y_i^s \neq y^q$. Consider, $\mathcal{D}_s$ and $\mathcal{D}_q$ are already transformed to features using θ_f.

$$\mathcal{L} = \sum_{(x_i^s, y_i^s) \sim \mathcal{D}^s} \parallel \theta_r(x_i^s, x^q) - sim(y_i^s, y^q) \parallel_2^2 \tag{5.1}$$

CCE: We train Personalised RN as a classification task, where in an n-way setting, the n number of outputs of the relation pairs are analogous to a one-hot encoding of the predicted class label. Accordingly, there exist one pair where the similarity score should be 1 (matching pair) and others 0 (non-matching pairs). Loss for one training data instance, $(\mathcal{D}_s, (x^q, y^q))$, is calculated in Equation 5.2. Here y^q and y_i^s are transformed to one-hot encoding.

$$\mathcal{L} = \sum_{(x_i^s, y_i^s) \sim \mathcal{D}^s} y_i^s \log \theta_r(x_i^s, x^q) + (1 - y_i^s) \log(1 - \ \theta_r(x_i^s, x^q)) \tag{5.2}$$

Locality Aware Loss: We train Personalised RN as a metric learning task where the output of the relation module is the distance between the *support set element, query instance* pair. In a training data instance, $(\mathcal{D}_s, (x^q, y^q))$, there is 1 positive (i.e. matching) pair, and the rest are negative (i.e. non-matching) pairs. We consider the distance predicted for the positive pair (d^p) should be less than some margin, $\mathcal{M}$. Moreover, the minimum distance predicted for a negative pair (d^n) should be larger than the margin $\mathcal{M}$. Accordingly, personalised RN learns a feature space where furthest like-neighbour is within the distance $\mathcal{M}$ and the nearest unlike-neighbour is at least $\mathcal{M}$ distance away in the feature space. We formalise the Locality Aware Loss (LAL) in Equation 5.3. The sigmoid activation at the relation network output layer ($r^{q,s}$) now predicts the distance of a pair within 0 and 1 (instead of similarity). For class prediction, we select the pair that generates the minimum distance (in contrast to maximum similarity) (Equation 5.4).

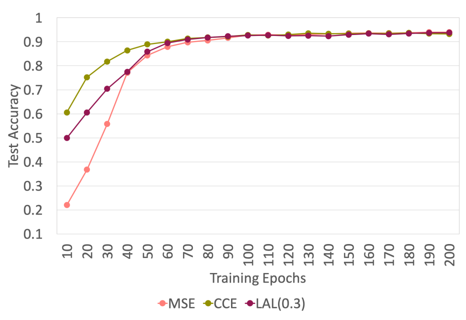
$$\begin{gathered} d^p = r^{q,s} \text{ where } y_i^s = y^q \text{ and } d^n = r^{q,s} \text{ where } y_i^s \neq y^q \\ \mathcal{L} = max(0, \mathcal{M} + d^p - \min(d^n)) \end{gathered} \tag{5.3}$$

$$y^{q'} = \arg\min_{|\mathcal{C}|} \; r_j^{q,s} \tag{5.4}$$

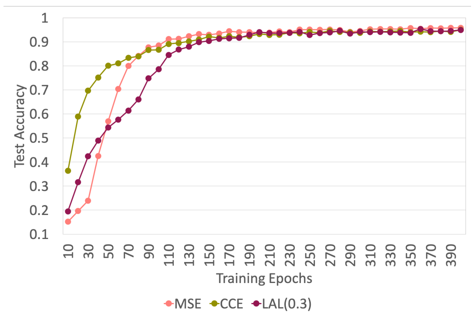
We create experiments using the two MEx modalities, ACT (time-series data) and PM (visual data). We compare the performance in $k^s = 1$ and $k^s = 5$ personalised meta-learning settings and are trained for 200 and 400 epochs respectively. Each model is tested at every 10 epochs using the 100 meta-test tasks created from the test person. Mean accuracy of the meta-test tasks over person-folds are plotted in Figures 5.7a to 5.7d.

Results

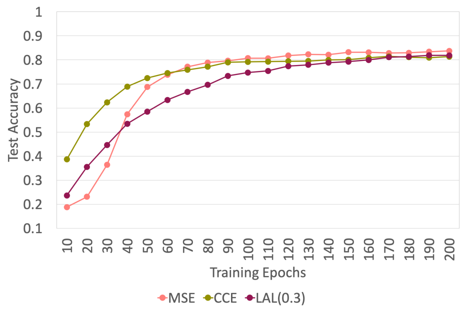
Overall, all three methods achieve comparable performances in both $k^s = 1$ and $k^s = 5$ settings. Importantly, CCE achieves comparable performance with rapid meta-model training in both $k^s = 1$ and $k^s = 5$ settings. For instance, with $k^s = 1$ experiments, at 50 epochs, CCE method outperforms MSE by 4.55% (0.8434∼0.8889) and 3.69% (0.6874∼0.7243) for ACT and PM. Similar, with $k^s = 5$ experiments, at 50 epochs, CCE method outperforms MSE by 23.1% (0.5694∼0.8005) and 28.9% (0.2743∼0.5638) for ACT and PM. Accordingly, we find that CCE is significantly better at training in a



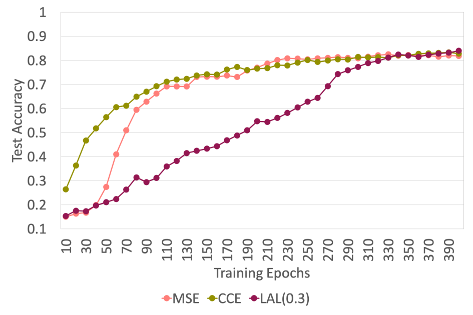
(a) MEx$_{ACT}$ at $k^s = 1$



(b) MEx$_{ACT}$ at $k^s = 5$



(c) MEx$_{PM}$ at $k^s = 1$



(d) MEx$_{PM}$ at $k^s = 5$

Figure 5.7: Personalised RN model performance with different loss functions

few-shot setting ($k > 1$).

Personalised RN as a metric learner using LAL loss trains significantly faster compared to MSE within the first 40 epochs in the $k^s = 1$ setting and achieve comparable performance with CCE and MSE. LAL struggles significantly in the few-shot setting with visual data and takes longer to achieve comparable performances with CCE and MSE methods (at 330 epochs for PM). Accordingly, we find LAL is better suited in a constricted ($k = 1$ and a small number of training epochs) setting compared to MSE and when data cannot be modelled as a classification task (i.e. one-shot deep metric learning tasks).

5.3.4 Prediction Latency of Personalised Meta-learners

A HAR algorithm should be able to recognise activities as they are performed in real-time for the best user experience. The processor and memory requirements are crucial factors that affect the latency, especially on an edge device. MAML^p class prediction is a simple classification task but requires post-deployment model re-training for personalisation. In contrast, RN^p and MN^{p*} do not require model-retraining, however, obtaining the activity class label for a given query involves a more complex inference process. First, each data instance in the support set and the query instance is transformed to feature vectors and then paired to obtain the similarity scores and the predicted class. We compare the latency for obtaining a prediction using the three algorithms, with the ACT dataset. The time elapsed is measured on a computer with 8GB RAM and a 3.1 GHz dual-core processor.

The MAML^p model used here is already personalised using the support set and it takes 0.0011 ms for a single prediction. RN^p takes 2.2124 ms when $K^s = 1$ and 2.6444 ms when $K^s = 5$; and MN^{p*} takes 0.1656 ms when $K^s = 1$ and 0.3122 ms when $K^s = 5$. In comparison, MAML^p has the lowest prediction latency, but the post-deployment re-training calls for specialised deployment environments. On the other hand, RN^p and MN^{p*} algorithms take significantly more time for class prediction. With respect to the MEx dataset, the increment chosen when applying the windowing method is 2 seconds (5 second window with 2 second increments from Section 3.3.1). Accordingly, in real-time, the selected algorithm should be able to make a prediction every 2 seconds. Within this time period, the sensor data streams are communicated from the sensor devices, and pre-processed using the methods discussed in Section 3.3.1 before making a prediction with the selected algorithm. While the time taken for prediction is less than 0.3% of the 2 second time interval, minimising the prediction time is desirable. It allows more time to mitigate issues that emerge in real-time with communication and sensor functionality and to provide an uninterrupted user experience.

5.4 Chapter Summary

In this chapter, we presented the personalised meta-learning methodology for HAR. Our goal was to create personalised HAR models with minimal demand for end-user data. Accordingly, we bring together meta-learning methodologies and few-shot personalisation methods from previous literature to present personalised meta-learning methodology. Personalisation of a meta-model only requires few instances of labelled sensor data that can be obtained via micro-interactions with the end-user.

We presented the implementation of this methodology with three meta-learning algorithms. Personalised MAML algorithm has the lowest latency of the three algorithms, but requires to bootstrap the model using end-user data. In contrast, Personalised RN and Personalised MN algorithms have higher latency but do not require model re-training after deployment. An empirical evaluation showed how the choice of feature learners and support set size affect overall performance using four MEx modalities. We found the best feature learners for each algorithm and the most optimal support set sizes, which we select, considering performance and usability. Additionally, we explored different optimisation methods for personalised RN, to find that rapid meta-model training can be achieved by modelling personalised RN as a classification task or a metric learning task instead of a regression task.

Chapter 6

Open-ended Recognition with Meta-learners

Open-ended recognition models have the ability to recognise new and unseen classes added after model deployment. In the HAR domain, open-ended recognition can be seen as a form of personalisation where the end-user can add their preferred activities to the recognition model. Personalised HAR literature and methods introduced in Chapter 5 showed how incorporating few end-user data improve recognition performance without model re-training. We argue that a new activity is best represented by few instances of data and better yet few instance from the same person (i.e. personalised). In this chapter, we present the open-ended meta-learning methodology where a new unseen class can be represented by few data instances provided by the end-user. Moreover, the methodology is implemented using similarity-optimised meta-learners that dynamically expand the decision layer as new activity classes are added.

6.1 Use Case

Figure 6.1 presents a detailed use case of personalised open-ended fitness application that recognise activities according to user preference. Imagine person A who is a young, physically active, gym enthusiast, downloads the fitness application to their mobile phone. The application has a meta-model that is pre-trained to recognise four activity classes, walking, dancing, running and sitting using a single accelerometer sensor. First, person A personalises the fitness application using the method described in Chapter 5.

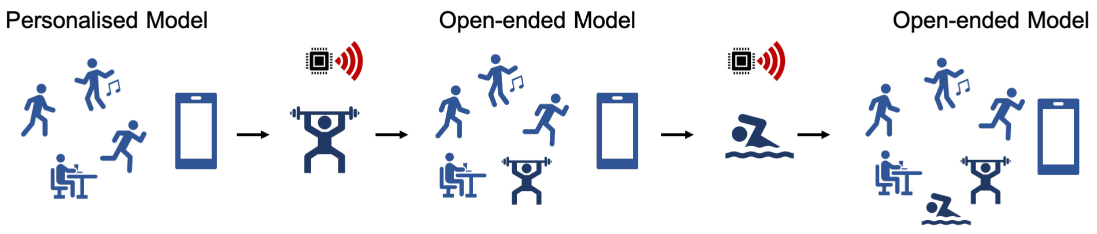


Figure 6.1: A use case of open-ended activity recognition

Imagine, person A wanted the fitness application to recognise activities they perform regularly but are not packaged in the generic model of four activity classes. For example, person A finds that weight lifting is an activity they perform regularly but is not one of the activities automatically recognised by the application. They perform a few seconds of weight lifting while being recorded by the sensors recommended by the application, and at the end, they label the data as weight lifting. Subsequently, the application is personalised and extended to recognise five activities using the calibration data. This model we refer to as personalised and open-ended.

After a while, person A start regular swimming lessons and plan to add swimming to their weekly fitness plan. They record few seconds of calibration data while swimming and the personalised open-ended model can integrate the new activity class swimming using the calibration data to the application. Importantly the new data required is minimal (i.e., knowledge-light) and is integrated with the reasoning model without re-training.

6.2 Open-ended Meta-learning

Traditionally, semantic knowledge exists in the form of a mapping between intermediary features and activity labels, Υ, for all seen classes, $\mathcal{C}$ and unseen classes, $\hat{\mathcal{C}}$. We formalise conventional open-ended recognition in Equation 6.1 where $\mathbb{X}^d$ is the d dimensional raw feature space, $\mathbb{A}^s$ is the s dimensional intermediary feature space. The mapping from raw feature space to intermediary features, Ω, is often learned using the labelled data from seen classes, $\mathcal{C}$. The mapping between the intermediary features and all activity labels, Υ, must be open-ended such that it can accommodate classes discovered during after deployment. Accordingly, Υ is often considered a nearest neighbour feature space.

$$y = \Upsilon(\Omega(x)) \text{ where } \Omega : \mathbb{X}^d \rightarrow \mathbb{A}^s \text{ and } \Upsilon : \mathbb{A}^s \rightarrow \mathcal{C} \cup \hat{\mathcal{C}} \tag{6.1}$$

Following the discoveries of Chapter 5, we argue that the best representation for a new unseen class is few representative instances provided by the test person. Accordingly, we eliminate the intermediary feature, $\mathbb{A}^s$. Instead, we learn the mapping $\mathbb{X}^d \to \mathcal{C}$ such that the mapping is transferable to $\mathbb{X}^d \to \mathcal{C} \cup \hat{\mathcal{C}}$ without model re-training.

6.2.1 Zero-shot Meta-learners in an Open-ended Setting

In the zero-shot setting, after deployment, we assume that we have access to few example instances, for a set of new activity classes, $\hat{\mathcal{C}}$, that were not seen during the training of the model. This data is provided by the end-user to introduce the new activity classes. Thereafter, the model is expected to recognise all activity classes in both $\mathcal{C}$ and $\hat{\mathcal{C}}$. Similar to in few-shot setting, a meta-learners in a zero-shot setting is learning to discriminate a subset of classes, ζ. These classes may belong to seen classes, $\mathcal{C}$, or unseen classes, $\hat{\mathcal{C}}$. Importantly, the meta-learner conforms to the n-way classification such that $|\zeta| = n$.

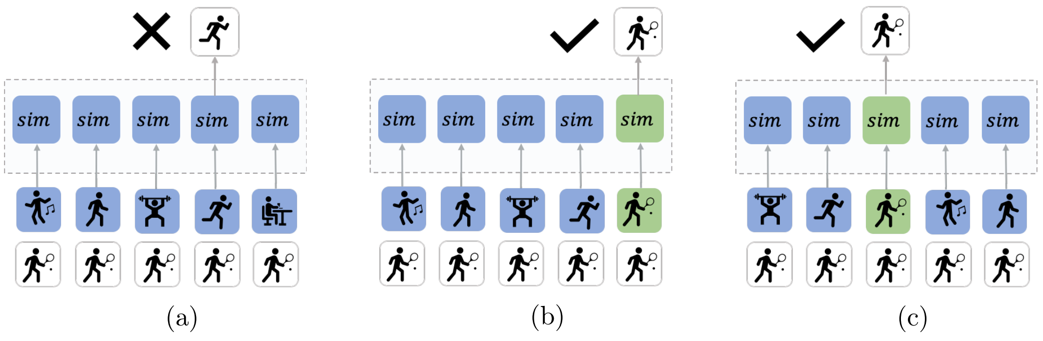


Figure 6.2: A conventional similarity-optimised meta-learner in a open-ended setting

In an Open-ended setting, this restriction forces the model to select a subset of classes from both seen and unseen classes ($\mathcal{C} \cup \hat{\mathcal{C}}$). For instance, in a daily fitness application where every activity has a possibility to take place, selecting a subset is undesirable. The selected support set of classes, ζ, may not include the true class (the true class label of the query instance), resulting in poor performance. For instance, in a 6 class setting ($|\mathcal{C} \cup \hat{\mathcal{C}}| = 6$), for 5-way recognition, there are 6 possible ways ($nCr = n!/r! \times (n-r)!$) to select the support set. Figure 6.2 illustrates three instances of random support set selection in a zero-shot setting. There are five seen classes (denoted by blue icons), and one unseen class (denoted by a green icon) and the black icon denotes the query instance. Evidently, the absence of the expected class in the support set has resulted in an incorrect classification outcome.

To mitigate the limitations of a zero-shot setting, we expand the support set to include as many as the expected number of classes that are available after deployment. Accordingly, the meta-learner no longer conform to an n-way classification after deployment. We refer to this approach as open-ended meta-learning. As we take forward personalisation from Chapter 5, the resulting meta-learner performs open-ended recognition in a personalised manner.

6.2.2 Open-ended Meta-learning Task Design

An open-ended meta-learner is trained on many personal HAR tasks similar to a personalised meta-learner. Each person-task learns to classify a set of *person-activity* classes where the *activity* is part of the set, $\mathcal{C}$. An open-ended meta-learner is tested on a meta-test person-task, that belongs to a person, $\hat{\mathcal{P}}$, with $\mathcal{C}_{te}$ number of activity classes where $\mathcal{C}_{te} = \mathcal{C} \cup \hat{\mathcal{C}}$. Importantly, person, $\hat{\mathcal{P}}$ was not seen during training, and the meta-task contain activity classes, $\hat{\mathcal{C}}$ that were not seen during training.

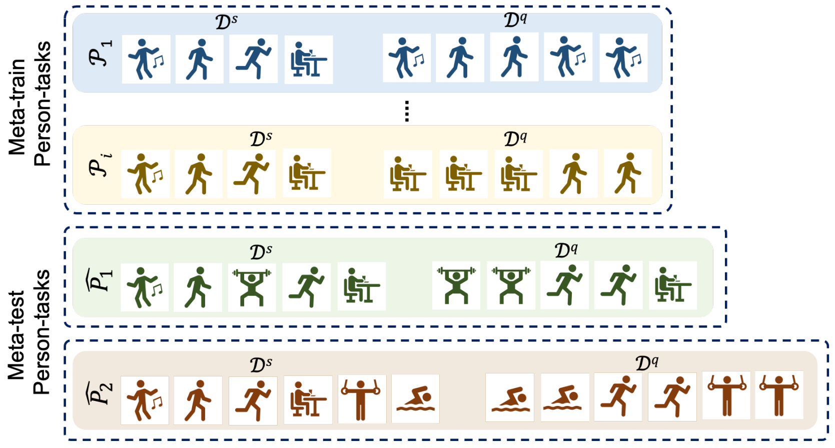


Figure 6.3: Personalised open-ended meta-learning task design

The task design for personalised open-ended meta-learning is illustrated in Figure 6.3. The meta-train person-task configuration is similar to personalised meta-learning methodology, detailed in Section 5.2.1. Each meta-train person-task consists of $\mathcal{C}$ number of classes and k^s number of representatives for each activity class to create the support set, $\mathcal{D}^s$. Furthermore, k^q number of instances for each activity class creates the query set, $\mathcal{D}^q$. Often, $\mathcal{D}^s$ and $\mathcal{D}^q$ are selected to be disjoint.

We illustrate two meta-test tasks: $\hat{\mathcal{P}}_1$ and $\hat{\mathcal{P}}_2$. Person $\hat{p}_1$ has provided k^s number of labelled data instances for each activity class seen during training, $\mathcal{C}$. In addition there are k^s representatives for 1 new unseen class *weight lifting* which is $\hat{\mathcal{C}}$. Accordingly the test query set may contain data instances from activity classes that belongs to both $\mathcal{C}$ and $\hat{\mathcal{C}}$. Person $\hat{p_2}$ has provided representatives for 2 additional unseen classes ($|\hat{\mathcal{C}}| = 2$), *rope jumping* and *swimming*. Accordingly, at test time, the personalised open-ended meta-model recognises 7 activity classes (i.e. $\mathcal{C} \cup \hat{\mathcal{C}}$)

Equation 6.2 formalises meta-test person-task conditions which facilitate the inclusion of all available classes in the support set. As new classes are introduced to the model after deployment the meta-test person-task support set, $\mathcal{D}^s$, has $k^s \times |\mathcal{C}_{te}|$ number of labelled data instances where $\mathcal{C}_{te}$ is the complete set of classes for the test person. With this refinement, we can use the meta-learner trained in a personalised manner to perform open-ended recognition.

$$\begin{gathered}\mathcal{C}_{te} = \mathcal{C} \cup \hat{\mathcal{C}} \\ \mathcal{D}^s = \{(x, y)|y \in \mathcal{C}_{te}\} \text{ where } |\mathcal{D}^s| = k^s \times |\mathcal{C}_{te}|\end{gathered} \tag{6.2}$$

The goal of an open-ended meta-learner is to learn a generalised feature learner such that at test time, the feature space successfully discriminates the set of seen and unseen classes. Importantly the decision layer needs to adapt to an increasing number of unseen classes dynamically. As seen in Sections 2.4.2, similarity-optimised meta-learners predict the class label based on a similarity distribution at the decision layer. We exploit this property to dynamically expand the decision layer with several modifications. We will discuss them next with respect to 3 similarity-optimised meta-learners.

6.2.3 Open-ended Similarity-optimised Meta-learner Algorithms

Similarity-optimised meta-learners highlight the similarities between data instances that belong to the same class. In an open-ended setting, we view a similarity-optimised meta-learner as a parametric k-nearest neighbour classifier where $k = 1$. For instance, a meta-task created in the $k^s = 5$ setting with 7 classes, each of the 7 classes is represented by 5 data instances. A query instance is compared against this support set of 35 labelled instances that collectively form the nearest neighbour feature space.

As detailed in Section 2.4.2, with similarity-optimised meta-learners, the label of the support set element that creates the highest similarity pair with the query instance is selected as the predicted label of the query instance (i.e. the nearest neighbour). This approach can also be seen analogues to attribute-based open-ended recognition methods in the literature that perform k-nearest neighbour classification to predict the class label. We consider three similarity-optimised meta learners: Matching networks (MN) (Vinyals et al., 2016); Prototypical Networks (PN) (Snell et al., 2017); and Relation Networks (RN) (Sung et al., 2018) to implement the open-ended meta-learning methodology.

Let us consider a meta-train person-task, $\mathcal{P}_i$ with a support set, $\mathcal{D}^s$, that consists of $\mathcal{C}$ activity classes and k^s number of instances per class. And a meta-test person-task, $\hat{\mathcal{P}}$ with a support set, $\mathcal{D}^s$ that consists of $\mathcal{C}_{te}$ activity classes and k^s number of instances per class. Training of a open-ended matching networks, MN^{o*}, is similar to as detailed in Section 5.2.2. Cosine similarity for each *query instance, support set element* pair, (x^q, x^s_j) is calculated and transformed into a skewed probability distribution using non-parametric hard attention (i.e. softmax activation).

With the open-ended methodology, a meta-train person-task calculates the similarity distribution against $|\mathcal{C}| \times k^s$ number of similarity values. In contrast, a meta-test person-task calculates the similarity distribution against $|\mathcal{C}_{te}| \times k^s$ number of similarity values. The similarity calculation and class prediction in meta-train and meta-test settings for MN^{o*} are compared in Table 6.1. Here, $y^{q'}$ refers to the predicted class.

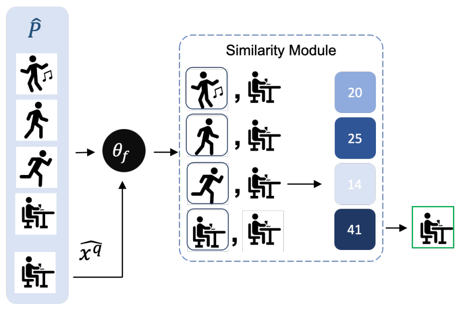
Table 6.1: Similarity and class prediction: MN^{o*}

	Meta-train	Meta-test
Similarity	$a(x^q, x^s_i) = \frac{e^{sim(x^q, x^s_i)}}{\sum^{\lvert\mathcal{C}\rvert \times k^s} e^{sim(x^q, x^s_j)}}$	$a(x^q, x^s_i) = \frac{e^{sim(x^q, x^s_i)}}{\sum^{\lvert\mathcal{C}_{te} \times k^s\rvert} e^{sim(x^q, x^s_j)}}$
Class prediction	$y^{q'} = \arg\max_{y^s_i} \; a(x^q, x^s_i)$ where $a \in \mathbb{R}^{\lvert\mathcal{C}\rvert \times k^s}$	$y^{q'} = \arg\max_{y^s_i} \; a(x^q, x^s_i)$ where $a \in \mathbb{R}^{\lvert\mathcal{C}_{te}\rvert \times k^s}$

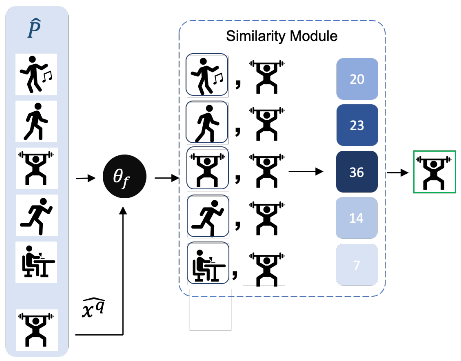
Personalised and open-ended prototypical networks, PN^o, is the implementation of the open-ended meta-learning methodology using Prototypical Networks by Snell et al. (2017). Originally, compared to MN, the only difference is when $k^s > 1$, where PN creates prototypical representative for each class label. Accordingly, with the open-ended methodology, there are only $|\mathcal{C}|$ number of pairs for a meta-train task and only $|\mathcal{C}_{te}|$ number of pairs for a meta-test task.

Both MN and PN similarity modules are non-parametric. In contrast, RN predicts similarity using a parametric regression model (detailed in Section 2.4.2). In addition, RN also create prototypes when $k^s > 1$. Accordingly, once the open-ended meta-learning methodology is adapted, RN^o class prediction is similar to PN^o.

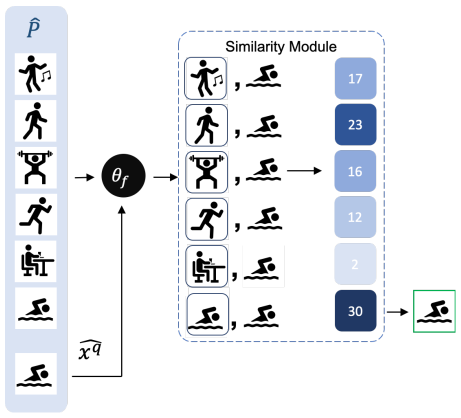
In summary, train conditions and train methodologies of open-ended similarity-optimised meta-learners are similar to those discussed in Sections 5.2.2. However, a meta-test person-task can have a variable amount of activity classes, each represented with a support set. Accordingly, we expand the test conditions and make the necessary changes to each algorithm as described above.



(a) MN^p*, personalised meta-learner



(b) MN^o*, personalised and open-ended meta-learner with one unseen class



(c) MN^o*, personalised and open-ended meta-learner with two unseen class

Figure 6.4: Dynamic adaptation of the personalised and open-ended meta-learners with an increasing number of unseen classes

In Figures 6.4, we visualise the use case introduced in Section 6.1, using a MN^o* model to further clarify our approach. Figure 6.4a show the meta-model available on the fitness application after personalisation (as described in Chapter 5) which can identify four *person-activity* classes. Figures 6.4b and 6.4c show the addition of activities to the personalised meta-model with few calibration data from the end-user. Importantly, all classes (seen during training and introduced after deployment) are represented in the support set. Moreover, the calibration data are not used to update the parametric models but instead uses them as *descriptors* for new classes. As additional classes are introduced, the support set includes them all when matching a query instance for classification. Accordingly, we have eliminated the need for an intermediary semantic space. We refer this method as *knowledge-light* compared to conventional methods that use expert intervention to define semantic attributes for open-ended recognition.

6.3 Evaluation

In this section, we evaluate the three similarity-optimised meta-learners for personalised open-ended activity recognition: MN^{o}*, PN^{o} and RN^{o}. The goal is to find the most robust open-ended meta-learner that perform consistently across different unseen activity classes and with an increasing number of unseen activity classes. To evaluate the former, we present the leave-one-class-out evaluation in Section 6.3.1 and to evaluate the latter we present the leave-N-class-out evaluation in Section 6.3.2.

6.3.1 Performance variability between different unseen activity classes

A set of L1CO experiments are designed using the three datasets MEx, selfBACK and PAMAP2 to find the best open-ended meta-learner across a wide range of activity types. Collectively these datasets contain general fitness activities, activities of daily living and exercises. Open-ended meta-learning tasks are created in the $k^s = 5$ and $k^q = k - k^s$ setting. For every single modal dataset, we design an experiment following the conventional setting described in Section 3.4.1. Accordingly, we create a $\mathcal{C}^*$ number of folds. For instance, ACT experiment will create 7 folds, each fold using 6 classes in train tasks (both support set and query set), 7 classes in the test support set and 1 class in the test query set. Accordingly, In the conventional setting, we can isolate the performance of the unseen class. Each fold will be repeated with different test persons (1/3 of all persons) for 20 iterations. In contrast to LOPO methodology, there is more than one person in the test set, and we will preserve personalisation by creating person-tasks for both train and test.

For comparability, every algorithm use the DNN(1) feature learner architecture from Section 5.3.1. All models are implemented using the Python library Pytorch (Paszke et al., 2019). MN^{o}* and PN^{o} are using the categorical cross-entropy loss, and RN^{p} use mean squared error loss. All models are trained using the Adam optimiser without mini-batching. MN^{o}*, PN^{o} and RN^{o} are trained for 100, 100 and 200 epochs respectively. The meta-task design maintains the class balance. Accordingly, we plot the mean accuracy of each experiment. Error bars are used to indicate the best and worst performances recorded in a fold (for an unseen activity class).

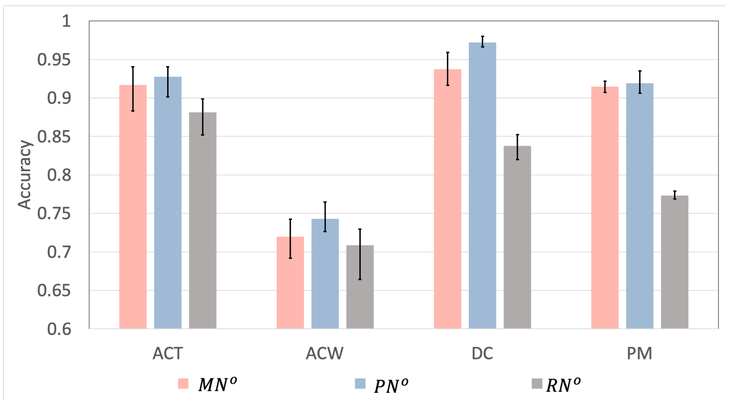
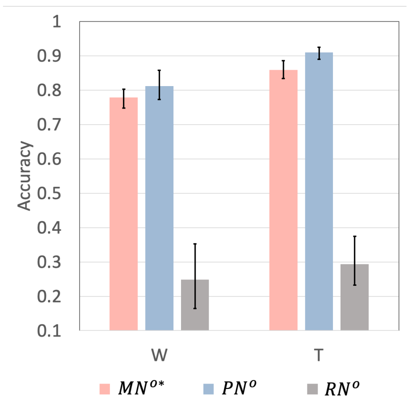


Figure 6.5: MN^o* vs. PN^o vs. RN^o L1CO performance comparison with MEx modalities

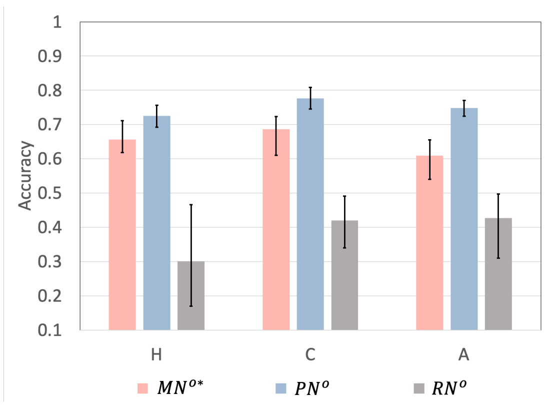
Figure 6.5 plots the results of experiments using the four single modality MEx datasets. Overall, PN^o algorithm outperforms both MN^o* and RN^o with all four MEx datasets. ACT, ACW, DC, and PM achieve mean accuracy of 0.9276, 0.7432, 0.9719 and 0.9193, respectively. PN^o also records the lowest variability between unseen exercise classes for 3 of the 4 experiments: ACT, ACW and DC. The difference between the best and the worst performing exercise classes for ACT, ACW, DC and PM are 3.92% (0.9404∼0.9012), 3.80% (0.7647∼0.7267), 1.3% (0.9799∼0.9661) and 2.91% (0.9351∼0.9060). PM is the only experiment that record the highest variability with PN^o; with MN^o* and RN^o, the differences are 1.48% (0.9218∼0.9070) and 1.01% (0.7790∼0.7689) respectively. Overall, RN^o fails to outperform MN^o* and PN^o, also records high variability between unseen classes.

In an open-ended setting, we are unable to anticipate the activity classes added after deployment. Accordingly, the selected modalities and algorithm should support a wide range of activity classes to ensure robustness. The high variability observed between unseen classes in ACT and ACW suggest that the modality is not able to capture a wide range of exercise classes. In contrast, PM and DC modalities can capture different unseen exercises with similar precision (hence the low variability). Notably, DC modality records the highest mean accuracy and the lowest variability, which suggests DC is the best single modal setting for open-ended exercises recognition.

Figure 6.6a plots the single modality experiment results with the selfBACK datasets.



(a) SELFBACK



(b) PAMAP2

Figure 6.6: $\mathrm{MN}^{o}*$ vs. PN^{o} vs. RN^{o} L1CO performance comparison with SELFBACK and PAMAP2 modalities

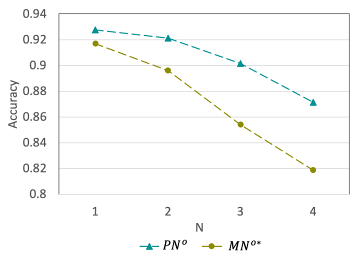
Similar to MEx, the overall best performance is achieved by PN^{o} algorithm; with SB_W and SB_T, PN^{o} achieves performances 0.8120 and 0.9105 respectively. $\mathrm{MN}^{o}*$ fails to outperform PN^{o}, but with SB_W, $\mathrm{MN}^{o}*$ achieves the lowest variability between unseen classes at 5.52% (0.8032∼0.7480). SB_T records the lowest variability with PN^{o}, at 3.48% (0.9248∼0.8900). RN^{o} perform poorly with both datasets, and also records a high variability between unseen classes.

Figure 6.6b plots the single modality experiment results with the PAMAP2 datasets. PN^{o} algorithm records the highest mean accuracy with all three experiments, which is consistent with MEx and SELFBACK results. $\mathrm{PAMAP2}_H$, $\mathrm{PAMAP2}_C$ and $\mathrm{PAMAP2}_A$ record performances of 0.7248, 0.7767 and 0.7483 respectively. In addition, PN^{o} records the lowest variability between unseen classes, 6.42% (0.7565∼0.6922), 6.34% (0.8084∼0.7449) and 4.58% (0.7702∼0.7244) respectively for $\mathrm{PAMAP2}_H$, $\mathrm{PAMAP2}_C$ and $\mathrm{PAMAP2}_A$. Similar to SELFBACK, RN^{o} perform poorly with all three experiments with high variability.

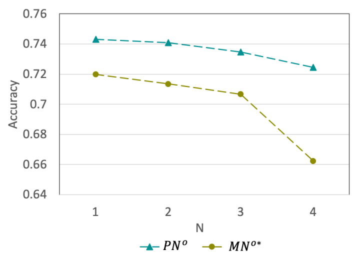
In summary, we find that compared to wearable sensors, visual sensors such as PM and DC are better single modal configurations to capture many activities with high precision in a open-ended recognition setting. In addition, PN^{o} algorithm is robust across a wide range of activity classes, and $\mathrm{MN}^{o}*$ is a close second; RN^{o} fails to maintain performance across different classes. Accordingly, in the next section, we take forward PN^{o} and $\mathrm{MN}^{o}*$ to test which algorithm is the most robust with an increasing number of new unseen activity classes.

6.3.2 Performance variability with increasing unseen activity classes

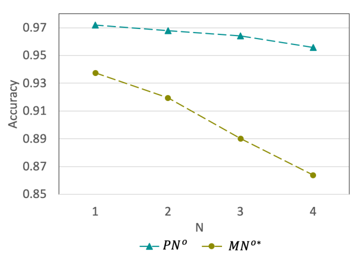
LNCO evaluation measures the robustness of an open-ended recognition algorithm with an increasing number of unseen classes. For instance, in a scenario where a physiotherapist may change the exercise regime often, the model should be able to adapt to multiple new activity classes while maintaining performance. Accordingly, we compare the two algorithms MN^{o}* and PN^{o} with a set of experiments created using the three datasets MEx, SELFBACK and PAMAP2. With each dataset, we are observing the impact in different activity domains. Here we test up to 4 unseen classes (i.e. $1 \leq n \leq 4$) where the total number of activity classes for MEx, SELFBACK and PAMAP2 are 7, 9 and 8 respectively. Implementation details are similar to the previous section, and all results are presented in the generalised setting (Section 3.4.1) where the model is tested for both seen and unseen classes. For instance, ACT ($\mathcal{C}^* = 7$) experiment in the L2CO setting will create 20 folds. Each fold will randomly select 2 exercise classes as $\hat{\mathcal{C}}$ and remaining 5 exercises classes as $\mathcal{C}$. A meta-train person-task will have 5 classes in the support set and the query set; a meta-test person-task will have 7 activity classes in the support set and 7 activity class in the query set. Each fold will be repeated with different test persons (1/3 of all persons) for 20 iterations.



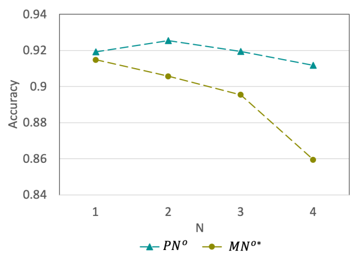
(a) MEx$_{ACT}$



(b) MEx$_{ACW}$



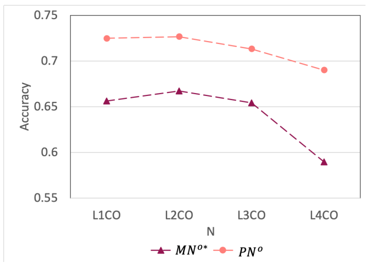
(c) MEx$_{DC}$



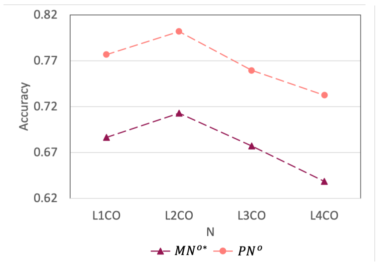
(d) MEx$_{PM}$

Figure 6.7: MN^{o}* vs. PN^{o} LNCO performance comparison with MEx modalities

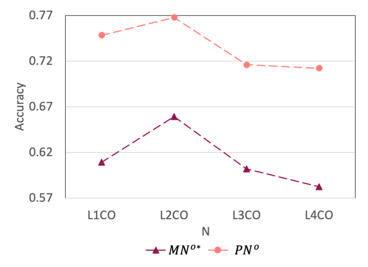
Figures in 6.7 plot the results for MN^{o}* vs. PN^{o} using the MEx datasets that contain exercise activity classes. Similar to L1CO experiments, PN^{o} outperform MN^{o}* with all four experiments. In addition, compared to MN^{o}*, PN^{o} maintains performance with increasing unseen classes. MN^{o}* performance drop from L1CO to L4CO are 9.82%, 5.74%, 7.36% and 5.53% respectively for ACT, ACW, DC and PM. With PN^{o} performance drops are 5.63%, 1.87%, 1.62% and 0.76%, that are significantly lower. With ACT, both MN^{o}* and PN^{o} significantly loose performance when increasing unseen classes. With ACW, DC and PM, MN^{o}* significantly loose performance, but the performance loss in PN^{o} is less severe. These observations also highlight the importance of selecting generalisable modalities to capture unseen classes.



(a) PAMAP2$_H$



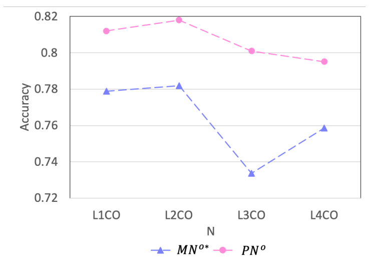
(b) PAMAP2$_C$



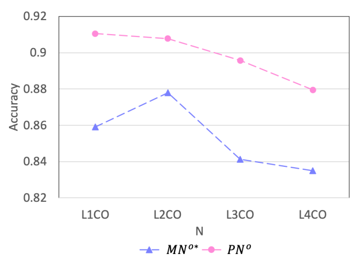
(c) PAMAP2$_A$

Figure 6.8: MN^{o}* vs. PN^{o} LNCO performance comparison with PAMAP2 modalities

Figure 6.8 plot the comparative results for MN^{o}* and PN^{o} using the three PAMAP2 datasets. PAMAP2 includes ambulatory, sedentary and daily living activities. Similar to MEx results, we find that PN^{o} significantly outperform MN^{o}*. MN^{o}* drop performance by 6.64%, 4.79% and 2.67% for PAMAP2$_H$, PAMAP2$_C$ and PAMAP2$_A$ respectively and PN^{o} only drop by 3.47%, 4.44% and 3.62%. We note the performance improvement between L1CO and L2CO, which we attribute to the difference between the two evaluation methods used for L1CO and L2CO experiments.



(a) SB$_W$



(b) SB$_T$

Figure 6.9: MN^{o}* vs. PN^{o} LNCO performance comparison with selfBACK modalities

Comparative results for selfBACK datasets are presented in Figure 6.9. Both experiments exhibit similar patterns to MEx and PAMAP2 where PN^{o} outperform MN^{o}*.

More specifically, MN^{o}* performance is dropped by 2.03% and 2.40% for SB$_{W}$ and SB$_{T}$, and PN^{o} performance is dropped by 1.69% and 3.11%. In summary we find that PN^{o} achieves the best performance when applied personalised open-ended methodology in both L1CO and LNCO settings. In the L1CO setting, PN^{o} exhibits minimal performance variability between different unseen activity classes. In addition, PN^{o} records the lowest performance drop when increasing the number of unseen classes.

Overall performance achieved using a single modality is clearly insufficient for real-world deployment. It is infeasible to have prior knowledge on what activity classes will be added by the end-user when using an open-ended recognition model in practice. The selected sensor modality or modality combination should be able to capture discriminatory features of these activities with sufficient precision. Accordingly, the sensor modality configuration used by the algorithm is a key factor that affect recognition performance in an open-ended setting. We will investigate this further in Chapter 7 when we evaluate PN^{o} in multi-modal settings and in comparison to existing methods. In addition, open-ended meta-learners use few data instances provided by the end-user recorded using the selected modalities. The quality of data is impacted by factors such as modality orientation and configuration which also influences the recognition performance in the real-world.

6.4 Chapter Summary

In this chapter, we presented the *knowledge-light* open-ended meta-learning methodology for HAR. Our goal was to facilitate the addition of new and unseen classes after model deployment with reduced demand for data and expert knowledge. Accordingly, we argued that an unseen activity class is best represented by few data instances of the activity. More specifically, instead of integrating attribute mapping, we acquire a limited amount of raw sensor data from the end-user through micro-interactions. We viewed open-ended recognition as a matching task implemented by similarity-optimised meta-learners where the matching is made between a query and activity class representatives. Consequently, our methodology can also be conveniently evaluated with any HAR dataset with no burden of acquiring semantic attributes from experts.

Our empirical evaluation finds the best similarity-optimised meta-learner for open-ended HAR with two desirable properties: a minimal variability of performance between different unseen classes; and a minimal decline of performance with an increasing number of unseen classes. The first property ensures that our method performs consistently across many activity classes. This property is essential, given that there is no prior knowledge of what activities the end-user will introduce to the model deployment. The second property is necessary to ensure that the addition of multiple new activity classes does not result in a poor performing recognition model. We established that PN^o algorithm owns both these properties in a wide range of HAR application domains using three HAR datasets.

Chapter 7

Exercise Recognition with MEx

In this chapter we perform exercise recognition with the *Heterogeneous* ***M****ulti-modal Physiotherapy* ***Ex****ercises Dataset*, ***MEx***. MEx is compiled with a view to implementing a self-management digital intervention for musculoskeletal pain, and we start this chapter by detailing the data collection process. The main objective of this chapter is to situate the methods introduced in Chapters 4, 5 and 6 among the methods from literature in their respective branch of exercise recognition challenges. We achieve this objective by evaluating our methods against appropriate baselines and the most recent methods from literature. A secondary objective is to establish the effectiveness of each method in other HAR domains which we achieve by evaluating them using three HAR datasets.

7.1 Heterogeneous Multi-modal Physiotherapy Exercises Dataset: MEx

Motivation for data collection is two-fold: the lack of publicly available data in the domain of physiotherapy exercises for machine learning; and the need to identify minimal sensor configurations for unobtrusive deployment. Accordingly, we explore a range of sensor modalities to capture physiotherapy exercises in a sensor-rich environment. The resulting MEx dataset is collected with 4 sensor modalities for 7 exercises with 30 participants. MEx dataset is publicly available at the UCI Machine Learning Data Repository [1].

[1]https://archive.ics.uci.edu/ml/datasets/MEx

7.1.1 Exercises

When selecting exercises for the data collection, we referred to a collection of 71 physiotherapy exercises that are recommended for the self-management of low back pain. This comprehensive list of exercises was compiled during the selfBACK project, for implementing a decision support system to self-manage low back pain [2]. 7 exercises were selected for this data collection: Knee Rolling, Bridging, Pelvic Tilt, Bilateral Clam, Repeated Extension in Lying, Prone Punch and Superman. These were selected to include 6 different target areas: Ab, Glut, Pain relief, Flexibility, Core and Back. The 7 exercises are illustrated in Figure 7.1 and exercises in detail with steps to perform are presented in Appendix B.

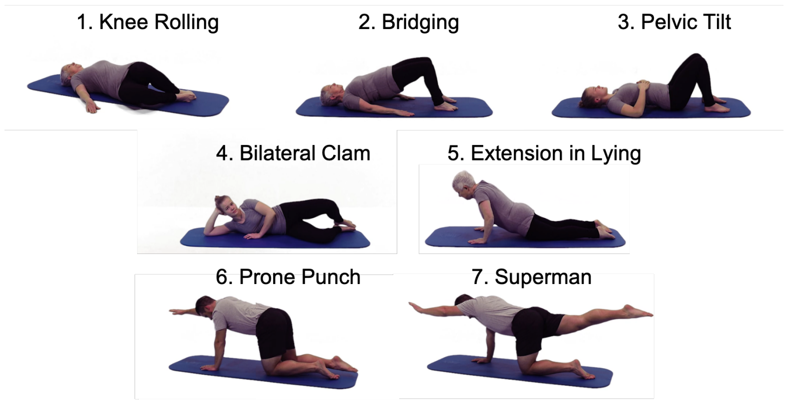


Figure 7.1: MEx exercises

Notably, a *null* activity class is not included in the data collection. The reasoning behind not including the *null* activity class is that different persons have different interpretation of resting during exercises. It can also include transition between exercises which depends on the set of exercises recommended and the order in which they are performed. However, it can be a limitation in a scenario where a recognition algorithm build from this dataset classify "resting in between exercises" as one of the exercises.

7.1.2 Sensors

State-of-the-art sensor technologies were explored for monitoring exercise movements to select following sensors: Obbrec Astra Depth Camera [3]; Sensing Tex Pressure Mat [4];

[2]http://www.selfback.eu
[3]https://orbbec3d.com/product-astra-pro/
[4]http://sensingtex.com/sensing-mats/pressure-mat/

and Axivity AX3 3-Axis Logging Accelerometer [5]. Inertial sensors and depth camera are commonly used in daily fitness activity recognition (Chen et al., 2017) and physiotherapy rehabilitation (Ayoade and Baillie, 2014; Ogonowski et al., 2016). Additionally, we include the pressure mat considering the type of exercises prescribed for MSD, that are often performed on a mat. Table 7.1 summaries the sensor specifications.

Table 7.1: MEx sensor specifications

	Depth Camera	Pressure Mat	Accelerometer
Frame rate(Hz)	$\approx$15	$\approx$15	$\approx$100
Frame size	320$\times$240	32$\times$16	3(x,y,z)
Data range	0-1	0-1	$\pm 8g$

Unobtrusive use in a home environment and the ability to capture not only the 7 exercises selected but many other exercises that are recommended for patients (such as the 71 in the selfBACK collection) are two key considerations when selecting the placement of each sensor. Accordingly, the following sensor placements were selected: two accelerometers will be placed on the dominant wrist and the thigh of the user; the pressure mat will be used as an exercise mat where the user will lay on to perform exercises; and the depth camera will be placed above the user, facing down-words recording an aerial view. The expectation with these placements is that it is not obtrusive to daily activities, and there is less variability when set up by different persons in their home environment. These sensor placements are illustrated in Figure 7.2.

7.1.3 Ethics and Data Collection

There are several ethical considerations when designing a data collection task and need to be approved by an appropriate ethics committee. We succinctly detail the ethics proposal in five main facets: objective, population and sample, data collection, communication and data management protocols (Figure 7.3). The *objective* of this data collection is to create a sensor-rich data set for physiotherapy exercise recognition. This objective defines the scope of the ethics proposal.

[5] https://axivity.com/product/ax3

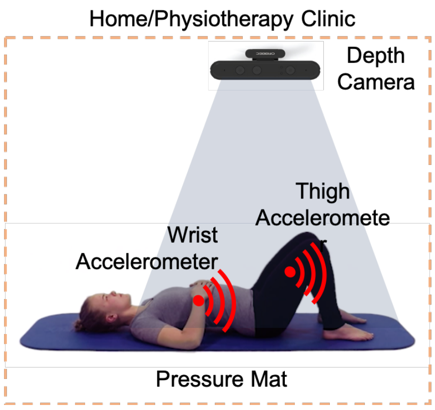


Figure 7.2: MEx modality placement

We defined the *population* using the following inclusion and exclusion criteria: adults of age 18 or over who do not have a physiological condition that prevents them from performing exercises. Participants are asked to complete a PAR-Q prior to participation to measure their physiological readiness to perform exercises (Thomas et al., 1992). Target *sample size* is 30 and will be recruited from the university staff and student population. We expect to include both expert and non-expert participants. We recognise an expert participant as someone who has clinical experience in performing physiotherapy exercises.

In the *data collection protocol* we present the set of exercises selected (Section 7.1.1), the sensor setup (Section 7.1.2) and steps taken to ensure participant safety. We propose the use of hypoallergenic tap to attach wearable sensors and the measures taken to provide medical assistance for participants if needed. The *communication protocol* defines how the researcher and participants communicate during the study. Emails, e-bulletin advert, word of mouth and social media are the methods proposed to publish the call for participants within the university. *Data management protocol* details the process of data storage and privacy. After collecting data with a participant, we plan to anonymise and store data in secure storage within the internal university network. During the research, the data will retain three demographic properties: age, gender and expert, non-expert status for evaluation purposes. The publicly shared dataset will be anonymised with all demographic data removed. This data collection protocol was approved by the ethics committee of School of Health Sciences, Robert Gordon University, Aberdeen, UK (SHS/17/29).

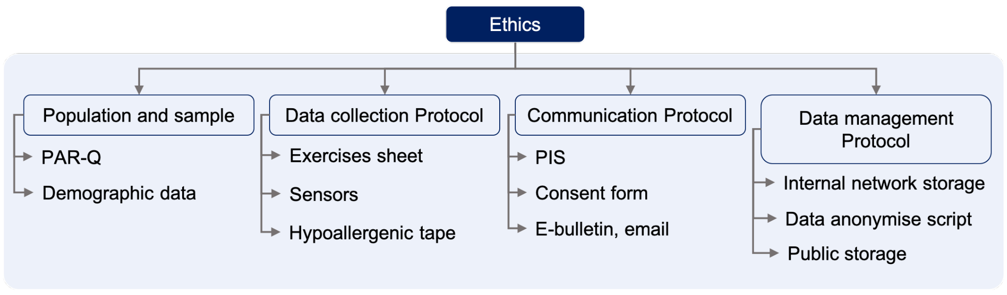


Figure 7.3: Data collection ethics considerations

Data Collection

Once a participant contacts the researcher, they received a participant information sheet (PIS) which includes exercise details, PAR-Q and a consent form. If the participant chooses to accept the terms, and if they are eligible, a 1-hour time slot was booked for the data collection. The signed consent form was collected from the participant, which allowed the researcher to use the data collected during the study and publish the anonymised dataset publicly.

The participant was given an instruction sheet of all the exercises. Each exercise is described with a starting position and set of actions (see Appendix B). The participant is provided with two accelerometers, one worn on the wrist using a wrist band and one on the thigh worn using hypoallergenic tape. The participant is asked to place the wearable on their own, on their dominant side, and not restricted to a specific orientation. This method adds variability to sensor data, which can be challenging for reasoning (Morris et al., 2014), but it correctly emulates an unsupervised scenario (i.e. in a home setting).

During the session, the researcher first demonstrated an exercise, and then the participant performed the exercise for approximately 60 seconds while being recorded with the four sensors. During the recording, the researcher did not provide any advice or counting to enforce rhythm. For exercises where it suggests holding a position for a few seconds, the participant was instructed at the beginning to keep a count by themselves to preserve their natural rhythm. This method captured individual nuances of participants which replicates a scenario where a patient performs these exercises at home without the guidance of the physiotherapist. In addition, this method avoided any discomfort that may occur when following strict instructions.

The data collection was conducted over a 1 year period with 30 participants. 60% of the participants were female, and 40% were male. 47% of the group were in the 18-24

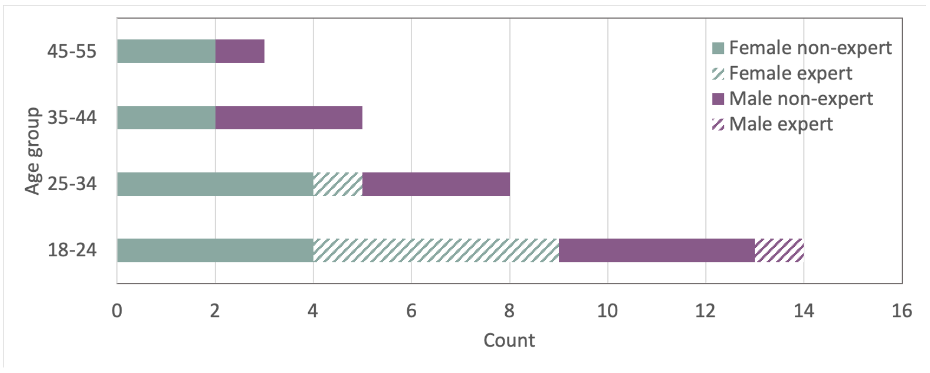


Figure 7.4: MEx data collection demographic

age category and the rest were dispersed among the ages from 25 to 54. 7 of the 30 participants were experts, and the remaining 23 were non-experts. These demographic information is visualised in Figure 7.4.

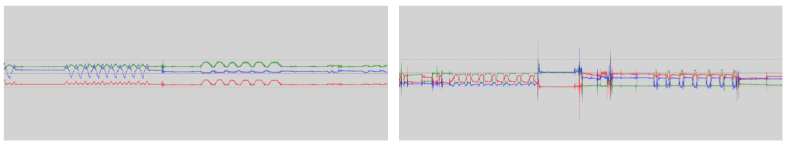
We used convenient sampling to collect data with healthy adults due to the challenges of collecting data with MSD patients. Accordingly, MEx population does not emulate the demographics of MSD. This is a limiting factor of MEx, where the performance measures presented for methods in this thesis can be too optimistic in the real-world. In addition, the objective of this data collection is to develop recognition algorithms using machine learning methods. Accordingly, the dataset is only annotated by the respective exercise label and person index. It is noteworthy that this dataset in its current state cannot be used to develop algorithms for repetition counting or exercise quality assessment. In future MEx can be annotated for performance quality and repetition count to extend to such tasks.

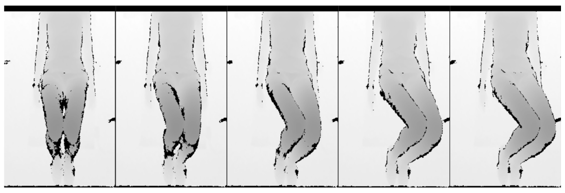
7.1.4 MEx Data Visualisation

In this section, we visualise multi-modal data from the MEx dataset: the first is a visualisation of raw data; and the next is using dimensionality reduction.

Raw Data Visualisation

Figure 7.5 visualises samples of data collected from each sensor. Figures 7.5a is the ACT (left) and ACW (right) sensor data for a person performing 5 exercises. Noisy outliers are found more commonly in the wrist sensor data compared to the thigh sensor data due to higher freedom of movement. Figure 7.5b and Figure 7.5c show five timestamps of depth camera data frames and pressure mat data frames sampled from


(a) Accelerometer data: thigh(left) and wrist(right)


(b) Depth camera video

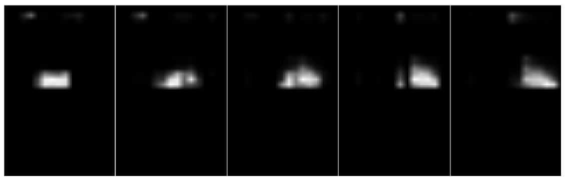

(c) Pressure mat data

Figure 7.5: MEx raw data visualisation

the knee-rolling exercise. Depth camera visibly captures a large amount of visual data compared to the pressure mat. Accordingly, exercises such as knee-rolling where there is a significant visual difference are better captured by the depth camera. In-contrast, an exercise such as pelvic tilt where the visual movement is minimal, is better captured by the pressure mat. There are multiple exercises with zero movements of the wrist. These observations highlight the challenge of recognising a large set of exercises from a single wearable sensor and the need for multiple heterogeneous sensors.

PCA Visualisation

We use Principal Component Analysis (PCA) for dimensionality reduction and visualise PM data for 5 different persons in Figure 7.6. Data instances are created using the sliding window method, as described in Section 3.3.1. Two of the 5 persons are experts, and other 3 are non-experts. We plot the first two PCA components clustered by exercise classes (indicated by different colours). PCA visualisation highlights that different persons have unique data distributions, even when performing the same set of exercises. Physiology, personal nuances, preferences and rhythm attribute to these unique distributions.

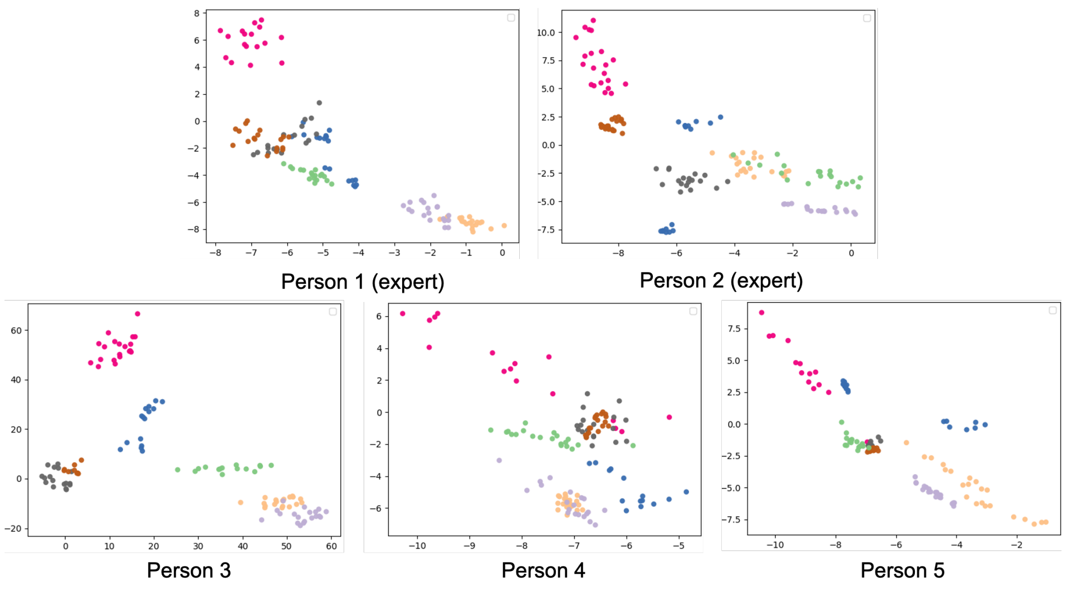


Figure 7.6: MEx 2D-PCA visualisation with the PM modality

MEx Dataset Statistics

We investigate the data distribution of MEx over persons and exercise classes in Figure 7.7. We plot the number of data instances for each person where the error bar indicates the maximum and the minimum number of instances for an exercise class. For example, person 5 contributes 39, 40, 19, 44, 21, 40 and 39 data instances for exercise classes 1 to 7. The maximum is 44 for the exercise *bilateral clam* and the minimum is 19 for *pelvic tilt*. The average contribution of a person to the dataset is between 34.57 and 26.86, which we consider approximately balanced, thus avoid any personal bias.

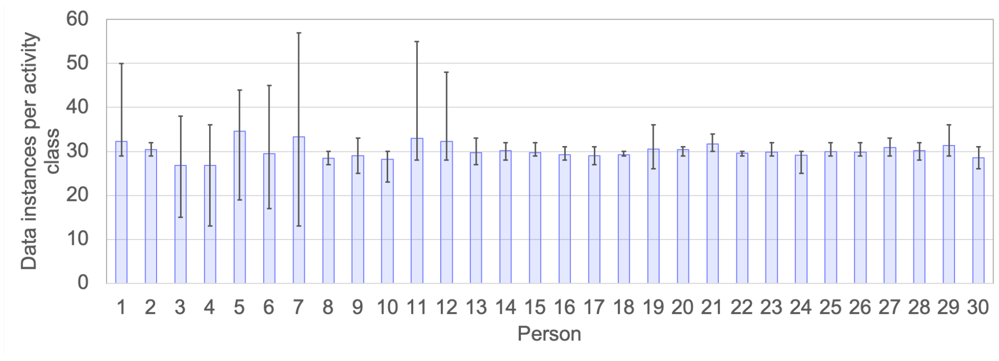


Figure 7.7: MEx data distribution by person

Figure 7.8 plots the data distribution by the exercise class. Here the error bars indicate the maximum and the minimum number of data instances contributed by a person. For instance, the average number of data instances per person for exercise class 4 (bilateral clam) is 30 where the minimum is contributed by person 6, at 22 and maximum is from person 5, at 44. The data instance distribution across exercises classes is approximately balanced (27.73∼32), thus avoids any class bias.

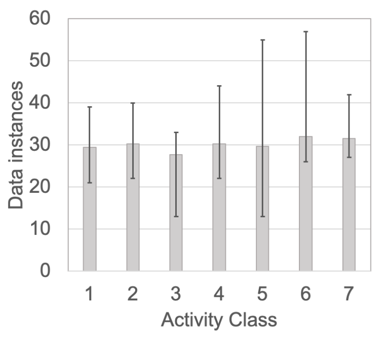


Figure 7.8: MEx data distribution by exercise class

7.2 Multi-modal Recognition with Hybrid Attention Fusion

In this section, we present the evaluation of the multi-modal fusion methods presented in Chapter 4 for exercise recognition using the MEx dataset. The evaluation is presented in three sections: 1) a comparative evaluation against baselines and recent methods from literature in Section 7.2.1; 2) learning multi-modal combinations by mHAF explained using confusion matrices in Section 7.2.2; and 3) learning feature combinations by mHAF explained using attention weights in Section 7.2.3.

7.2.1 Comparative Evaluation

In Chapter 4 we presented the mHAF architecture for heterogeneous multi-modal fusion. In this section we compare the performance of mHAF architecture against a set of baselines and algorithms from recent literature. The aim is to situate mHAF for heterogeneous multi-modal exercise recognition within the recent research landscape. We use the three most effective modality combinations identified in Section 4.6.2: MEx$_{ACT,PM}$ (2M), MEx$_{ACT,PM,DC}$ (3M) and MEx$_{ALL}$ (4M) and compare the performance of the algorithms listed below.

(1) Best single modality performance is selected as a baseline in this comparative evaluation. We find this to be an appropriate baseline given one of the goals of multi-modal fusion is to improve upon the performance achieved by the best single modality. Best performing single modality of MEx is the ACT modality and we refer to the results from Section 4.6.1.

(2) Early fusion refers to the fusion architecture where raw features from multiple sensor modalities are concatenated, to learn a shared feature representation. Here the fusion is applied at the feature axis. This baseline is selected to measure the adverse effect of early fusion on heterogeneous modalities. We re-use the 1D-CNN-LSTM architecture (Table 4.1) as the shared feature learner. The input sizes for MEx$_{ACT,PM}$, MEx$_{ACT,PM,DC}$ and MEx$_{ALL}$ are (1, 5, 180+256), (1, 5, 180+256+192) and (1, 5, 180+256+192+180) where the three dimensions refer to the number of input channels, the number of timestamps and the input feature length.

(3) Mid fusion learns modality specific feature representations and apply fusion at a mid level (refers to Figure 2.3b). Here the fusion is applied at the feature axis. With mid fusion, modalities have the opportunity to learn features independently and together. Accordingly, this baseline will show the trade off between late fusion and mid fusion on heterogeneous modalities. The shared feature learner has two dense layers ($dense(600) \rightarrow bn \rightarrow dense(100) \rightarrow bn$). Modality-specific feature learners from Section 4.6.1 are adapted for mid fusion as below:

- 2D-CNN for DC modality: $conv(3\times3)32 \rightarrow maxpool(2\times2) \rightarrow bn \rightarrow conv(3\times3)64 \rightarrow maxpool(2\times2) \rightarrow bn \rightarrow dense(1200)$; Input size $(1, 5\times12, 16)$.
- 1D-CNN-LSTM for AC and PM modalities: $td(conv(5)32 \rightarrow maxpool(2) \rightarrow bn \rightarrow conv(5)64 \rightarrow maxpool(2) \rightarrow bn) \rightarrow lstm(1200)$; Input sizes $(1, 5, 180)$ and $(1, 5, 256)$.

(4) Mid Fusion+temporal axis fusion architecture can be seen as the mid fusion (3) that applies fusion at the temporal axis. After applying mid fusion at the temporal axis, the resulting fusion features is considered to have multiple channels. Accordingly, we use modality-specific feature learners listed in (3) with a convolutional shared feature learner scripted as $conv(5)64 \rightarrow bn \rightarrow dense(600) \rightarrow bn \rightarrow dense(100) \rightarrow bn$ to form this architecture.

(5) DeepConvLSTM architecture by Ordóñez and Roggen (2016) applied early fusion on the temporal axis. Original DeepConvLSTM architecture consists of four, 1D convolutional layers followed by 2 LSTM layers. After applying early fusion at the temporal axis, the input sizes to DeepConvLSTM with $\text{MEx}_{ACT,PM}$, $\text{MEx}_{ACT,PM,DC}$ and MEx_{ALL} are (180+256, 5, 1), (180+256+192, 5, 1) and (180+256+192+180, 5, 1). The first dimension refers to the number of input channels, and the rest are feature dimensions. Accordingly, we use only one, 1D convolutional layer, and all other parameters kept consistent with the original architecture. DeepConvLSTM architecture for MEx is scripted as $conv(5)64 \rightarrow lstm(128) \rightarrow lstm(128)$.

(6) DeepSense is a deep fusion architecture introduced by Yao et al. (2017) that applies mid fusion on the feature axis. At each timestamp, an independent convolutional feature learner is applied on each modality, and the resulting feature vectors are concatenated to form the fusion feature vector. A shared convolutional feature learner transforms the fusion feature vector before applying 2 GRU layers for temporal fusion over the timestamps. The original DeepSense architecture assumes a homogeneous multi-modal setting where modality-specific feature learner outputs are of the same length. However, in a heterogeneous multi-modal setting, the outputs are different sizes. Accordingly, we adjust the fusion axis, and all other parameters are kept consistent with the original architecture.

(7) mHAF is our multi-modal hybrid attention fusion architecture as described in Chapter 4. mHAF has modality specific feature learners and the modality features are concatenated at a late level using hybrid attention fusion.

Implementation Details

Architectures 1, 2, 3, 4 and 6 were implemented using Keras and TensorFlow libraries for Python and architecture 5 was implemented using PyTorch. Architectures 1, 2, 3 and 4 were trained end-to-end for 100 epochs using the AdaDelta optimiser with default parameters to minimise the categorical cross-entropy loss. Architectures 5 and 6 were trained for 100 epochs using the Adam optimiser. All architecture use 32 as the mini-batch size. We present the mean F1-scores averaged over person folds created using the LOPO evaluation methodology. Statistically significant results are highlighted in bold text.

Table 7.2: mHAF performance comparison with MEx multi-modal datasets

Algorithm	2M	3M	4M
(1) Best single modality (ACT)	0.9015	0.9015	0.9015
(2) Early Fusion	0.8533	0.9224	0.8992
(3) Mid Fusion	0.8953	0.9263	0.9359
(4) Mid Fusion+temporal axis fusion	0.8723	0.9267	0.9323
(5) DeepConvLSTM	0.8887	0.9168	0.9332
(6) DeepSense	0.8670	0.9048	0.9205
(7) mHAF	**0.9354**	**0.9624**	**0.9584**

Results

Table 7.2 presents the results obtained using the three modality combinations from MEx dataset. Overall, mHAF significantly outperformed all baselines and algorithms from recent literature with all three modality combinations.

Architectures (2) and (5) both apply early fusion, (2) along the feature axis and (5) along the temporal axis. Temporal axis fusion has clearly shown an advantage over feature fusion with 2 of the 4 modality combinations. 2M and 4M datasets achieve 3.54% and 3.40% performance improvements with temporal axis fusion. The 3M dataset shows marginal advantage with the feature axis fusion method (2). Importantly, with both (2) and (5) modality features are aggregated before learning independent feature representations. And a comparison of (2) and (5) with mHAF performance provides strong evidence that early fusion is detrimental to learning effective multi-modal fusion.

Mid fusion on the temporal axis, (4), outperforms DeepSense architecture, (6), which apply mid fusion on the feature axis. Specifically, mid fusion on the temporal axis perform 0.53%, 2.19% and 1.18% better with 2M, 3M and 4M. Apart from fusion axis, (4) allow modalities to learn from modality-specific feature learners before applying mid fusion. Accordingly, both the choice of the fusion axis and independent features learners have contributed to this performance improvement. However, similar to early fusion architectures, both (4) and (6) fail to outperform mHAF architecture.

A closer look at the two algorithms from recent literature shows that they are both not effective in a heterogeneous multi-modal setting. DeepSense has a high parametric complexity, yet fails to achieve comparable performance with mHAF. Moreover, it may over-fit to the limited amount of training data available. In contrast, DeepConvLSTM,

(5) with a shallow architecture and early fusion on the temporal axis has outperformed DeepSense, (6).

Notably, architectures (3)-(6) improve performance with the addition of the noisy modality in 4M. In contrast, mHAF and (2) achieve the best performance with the 3M modality combination. The significance of this observation is that mHAF performance indicates the addition of a noisy modality allowing us to discover minimal modality configurations.

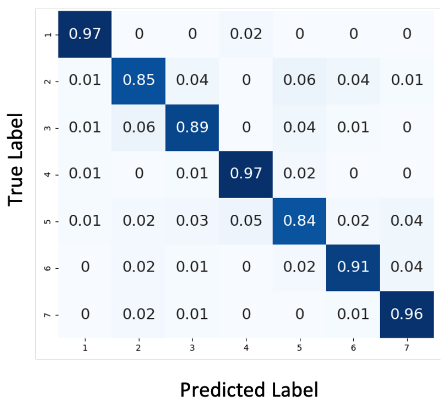
In comparison to the single modality baseline, mHAF outperforms the ACT performance in all three settings. Notably, only mHAF architecture outperforms ACT performance with the 2M modality combination. Late fusion in mHAF has allowed learning the most effective features from the additional PM modality that are complementary to ACT features. This is in contract to more noisy, yet significantly larger set of PM modality features dominating in a early or mid fusion setting.

7.2.2 Confusion Matrices

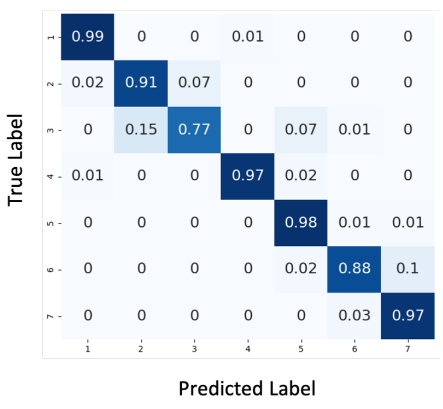
Results in Section 7.2.1 established that mHAF architecture learns an effective multi-modal fusion in a heterogeneous modality setting. In this section, we aim to investigate the mHAF fusion results using confusion matrices. We visualise the best single modality ACT (1M) performance against 2M, 3M and 4M performances using the mHAF architecture. Confusion matrices show how the addition of modalities affected the performance at the exercise class level. F1-score performance of each activity class is averaged over the person folds to create confusion matrices. For implementation details we refer to Section 7.2.1.

Figures 7.9a, 7.9b, 7.9c and 7.9d are the confusion matrices obtained with 1M, 2M, 3M and 4M. Best single sensor ACT achieves performances between 0.84∼0.97 across the 7 exercise classes. We note with prone punch (6), although the thigh remains stationary, it is unique compared to other 6 exercises where others have some unique movements. Consequently, in a single modal setting, the performance may get penalised with a different set of exercises (i.e. in an open-ended setting). Extension in lying (5) exercise record the lowest performance and the error is dispersed across all other classes.

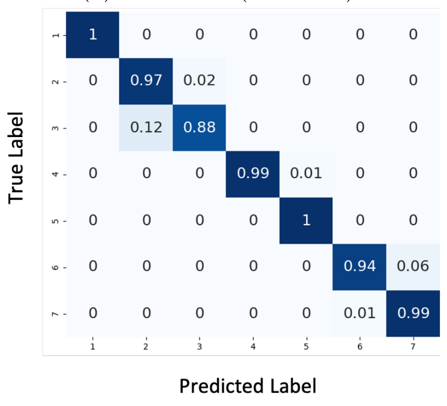
Addition of the PM modality in the 2M setting (Figure 7.9b) shows clear improvements to exercise classes 2 and 5, which are the least performing exercises in the 1M setting. But it is significantly detrimental for exercises 3 and 6. Discriminatory features of ACT for recognising exercise 3 apart from 2 and 5 are depreciated by the introduction of the



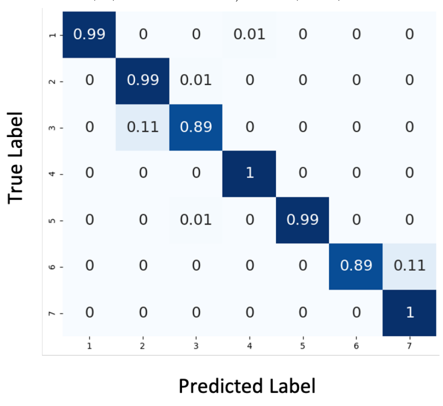
(a) MEx$_{ACT}$ (Best 1M)



(b) MEx$_{ACT,PM}$ (2M)



(c) MEx$_{ACT,PM,DC}$ (3M)



(d) MEx$_{ALL}$ (4M)

Figure 7.9: mHAF confusion matrices with MEx$_{ACT}$, 2M, 3M and 4M

PM modality. Similarly, recognising exercise class 6 is more challenging after introducing the PM modality. Although the overall performance is significantly improved, the performance differences across the 7 exercise classes are also increased to 0.77∼0.99.

Introduction of the DC modality in the 3M setting has significantly improved the performance compared to the 2M setting. The ambiguity between exercises 2 and 3 is reduced, and the performance of recognising exercise 6 is significantly improved by 6%. Addition of ACW modality in the 4M setting has been detrimental to recognising exercise 6. Given that exercises 6 and 7 have the same hand movements, the ambiguity is increased and now less discriminatory from exercise 7.

Overall, 3M is the best modality combination for the 7 exercises where the performance range from 0.88∼1.00. Exercise 3, pelvic tilt is the most challenging to recognise (max

0.88) followed by exercise 6, prone punch (max 0.94). All 5 other exercises achieve performances between 0.97 and 1. Accordingly, confusion matrices has revealed that 3M or 4M multi-modal settings do not recognise pelvic tilt and prone punch exercises with precision required for real-world deployment. An exercise such as pelvic tilt which involve fine yet intricate hip movements can be challenging to capture. The pressure mat modality has not successfully captured the torso movements with the precision required as expected. We also noticed that pelvic tilt is one of the exercises participants failed to perform correctly as instructed. These observation will be investigated further in future with different modality combinations (e.g. with the addition of a wearable on the torso).

7.2.3 Attention Weights

In this section, we investigate how mHAF is learning feature importance and modality combinations at the exercise class level using attention weights. We visualise the attention weights learned by the HA and SA modules (α^h and α^s) for each exercise class in Figure 7.10. The weights are obtained from the best performing 3M mHAF architecture for a set of randomly selected test instances. Each row is an exercise class, and the weights post-processed for clear visualisation. The weights are grouped by the modality and then applied a set of evenly spaced thresholds to create 5 groups. Resulting weights are sorted and stacked where a lighter colour indicates a higher weight.

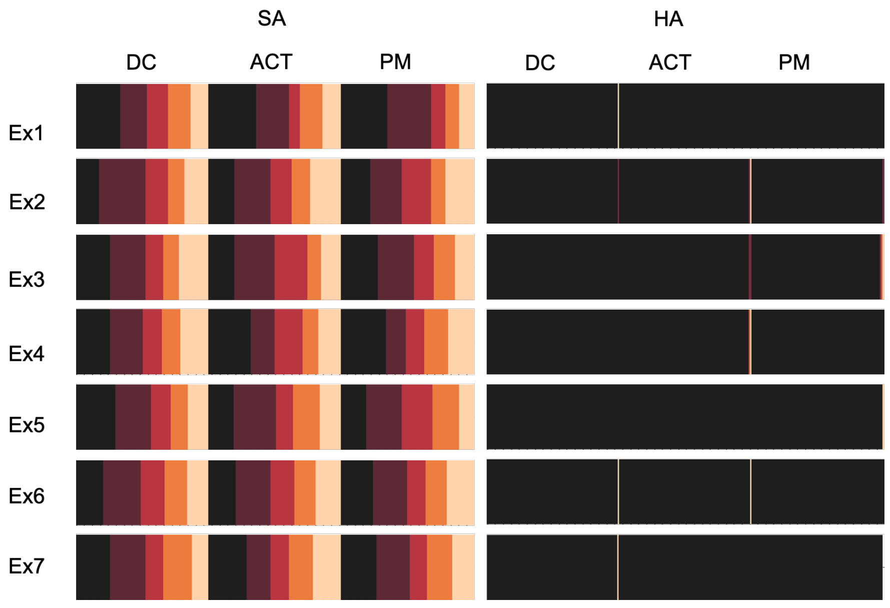


Figure 7.10: Visualising soft and hard attention weights of the 3M mHAF architecture

The hard attention module (HA) has learned skewed weights to highlight only a few features, whilst the soft attention module (SA) has learned a more naturally distributed set of attention weights. HA weights show that some exercises classes are recognised using a combination of all sensor modalities, while others learn to choose a subset of sensor modalities.

Exercise 4, bilateral clam, only moves the thigh and HA weights have correctly recognised that by learning to attend to only ACT features instead of a modality combination. This observation is further verified by confusion matrices in Section 7.2.2, where exercise 4 has achieved performance over 97% with the ACT modality. Exercise 6 is the only exercise that has no thigh movement (unique compared to others) and HA weights show that exercise 6 relies on ACT and DC modalities. This observation again conforms with confusion matrices in Section 7.2.2 where the addition of the PM modality was detrimental for exercise 6 (drops performance from 91% to 88%) and improve up to 94% with the addition of the DC modality.

For Exercise 2, bridging, HA is learning a combination of modalities, with the highest weight on the ACT modality. This is correct in practice as well as aligns with the observations in Section 7.2.2 where exercise 2 performance continuously increase from 85% to 91% and then to 97% with the addition of PM and DC modalities. For Exercise 5, extension in lying, HA is learning the highest weights for the PM modality. This is aligned with the observations in Section 7.2.2 where the performance of exercise 5 increase from 84% to 98% with the introduction of the PM modality.

PM modality should be able to clearly capture exercise 3, pelvic tilt because it involves only low back moments. In contrast, visible movement from an aerial view is minimal to be captured by the DC modality. Notably, in the 3M modality setting, HA shows the highest weight on PM modality features with some weight on ACT, and no weight on DC to correctly align with the expected modality combination. But, the addition of the DC modality has significantly improved the performance of exercise 3 in Section 7.2.2 which we are unable to explain at this time.

Importantly, with all exercises, the SA module learns a normal distribution of weights highlighting additional features that are complementary to the few feature highlighted by the HA module. Overall, there is a significant compatibility between the interpretation of attention weights, domain knowledge and confusion matrices. MHAF architecture is learning to select modality and feature combinations with heterogeneous multi-modal sensors for the ExRec task. These observations also demonstrated the need for maintaining alternative forms of attention within the fusion architecture. Additional features attended by the SA module are compatible with the modalities identified by the HA module. However, we identified that there is room for improvement with some exercise classes. Furthermore, we verify the two design hypothesis we made when selecting feature level granularity for learning attention weights: not all features of a modality equally contribute to improved performance; and more than one modality has significant features for improved performance.

7.3 Personalised Recognition with Meta-learners

In this section, personalised meta-learners from Chapter 5 are evaluated using MEx datasets for personalised exercise recognition. This evaluation is three fold: a comparison against baselines and methods from recent literature in Section 7.3.1; a visualisation of meta-learner training and testing to demonstrate the advantages of personalisation in Section 7.3.2; and a visualisation of the impact of meta-model training with expert users and meta-model adaptation by non-expert users in Section 7.3.3.

7.3.1 Comparative Evaluation

In this section, the performance of personalised meta-learners personalised MAML (MAML^p), personalised RN (RN^p) and personalised MN (MN^p*) are compared against several baselines and methods from recent literature listed below using MEx modalities.

DL: Best performing DL algorithm from Section 4.6.1

MAML: Model-Agnostic Meta-Learner by Finn et al. (2017)

MAML^p: Personalised MAML introduced in Section 5.2.2

RN: Relation Networks by Sung et al. (2018)

RN^p: Personalised Relation Networks introduced in Section 5.2.2

MN: Matching Networks for HAR by Sani et al. (2018)

MN^p: Personalised Matching Networks for HAR by Sani et al. (2018)

MN^p*: Personalised Matching Networks as a meta-learner introduced in Section 5.2.2

Implementation Details

Personalised and conventional meta-learning tasks are created in the $k^s = 5$ and $k^q = 25(30 - k^s)$ setting. For comparability, DNN(1) feature learner from Section 5.3.1 is used with all algorithms where the input is adjusted to suit the modality dimensions (see Table 5.3). For more implementation details we refer to Section 5.3.1. For $MAML$ and RN the same model implementation details from Section 5.3.1 apply. For the implementation details of MN and MN^p we refer to Sani et al. (2018).

We follow the LOPO evaluation methodology, as described in Section 3.4. For more evaluation details we refer to Section 5.3.1. Notably, even conventional meta-learners, MN, $MAML$ and RN preserve the personalisation aspect during testing because there is only one test person to create meta-test tasks. We report the accuracy of each experiment averaged over 30 person folds. Statistically significant results are highlighted in bold text.

Results

Table 7.3: Comparative evaluation of personalised meta-learners using MEx datasets

Algorithm	MEx$_{ACT}$	MEx$_{ACW}$	MEx$_{DC}$	MEx$_{PM}$
DL(Accuracy)	0.9068	0.6457	0.8825	0.7703
$MAML$	0.8673	0.6525	0.9629	0.9283
MAML^p	0.9103	0.6825	**0.9774**	**0.9377**
RN	0.8770	0.5184	0.7628	0.6714
RN^p	**0.9444**	0.6899	0.8533	0.7553
MN	0.9073	0.4620	0.5065	0.6187
MN^p	0.9155	0.6663	0.9342	0.8205
MN^{p*}	**0.9465**	**0.7438**	0.9606	**0.9289**

Table 7.3 presents the comparative results of personalised meta-learners evaluated with MEx modalities. First, we find that at least two out of the three personalised meta-learners outperformed the best performing DL algorithm with all four experiments. In addition, all personalised meta-learners significantly outperformed their conventional counterpart.

In the second section, we show how $MAML$ performance is significantly improved when using the personalised meta-learning methodology. All four experiments achieve performance improvements compared to the best DL algorithms. Moreover, all experiments significantly outperform conventional $MAML$ and the highest improvement is achieved by ACT data with 4.33%. Next, we compare RN and RN^p algorithms. The personalisation methodology has significantly improved the performance of RN with all experiments. Notably, the visual data modalities DC and PM, fail to outperform the best DL algorithm performance using RN^p, while time-series data modalities achieve significant improvements. Lastly, we compare the performances of MN, MN^p against MN^{p}*. The personalised meta-learning methodology has significantly improved the performance of the personalised few-shot classifier MN^p. While ACT and DC improve by 3.1% and 2.64%, ACW and PM improve by 7.75% and 10.84%. These results suggest that modalities that are more sensitive to personal nuances, like the pressure mat and the wrist wearable benefit from the meta-learning approach instead of a few-shot learning approach. In addition, MN^{p}* outperforms the best DL algorithm with all four experiments.

RN^p and MN^{p}* algorithms are both similarity-optimised meta-learners, and with three out of four experiments, MN^{p}* significantly outperform RN^p. It is also noteworthy when comparing the performance of the original algorithms MN and RN, the parametric similarity learning in RN outperforms the static similarity function used in MN in 3 out of the 4 times. However, once we apply the personalisation methodology, RN^p no longer outperforms MN^{p}*. Personalisation has eliminated task compositions that do not occur naturally, and simplify the classification task. Accordingly, a simpler algorithm like MN^{p}* has achieved significant improvement over a more generic and complex algorithm like RN^p.

Compared to RN^p, MN^{p}* is successfully learning feature representations from visual data for similarity prediction and to outperform the best DL algorithms. Yet, in general, complex visual data like DC and PM prefer adaptation-optimised MAML^p algorithm over similarity-optimised RN^p and MN^{p}* algorithms. In contrast, with accelerometer data similarity optimised MN^{p}* and RN^p significantly outperform MAML^p. Raw image data in DC and PM datasets are more complex compared to DCT features of accelerometer data. Accordingly, these results suggest that learning to compare feature representations from simple data types is more effective compared to complex data types.

Table 7.4 presents a summary of best performing personalised meta-learners for MEx modalities and multi-modal combinations. Here 2M, 3M and 4M refers to MEx$_{ACT,PM}$, MEx$_{ACT,PM,DC}$ and MEx$_{ALL}$ datasets (the best performing multi-modal combinations found in Section 4.6.2). The experiments are created in 5-shot setting. Each 1M experiment is using the best performing feature learners found in Section 5.3.1. In multi-modal

Table 7.4: Fine-tuned performance of personalised meta-learners with MEx datasets

Algorithm	MEx$_{ACT}$	MEx$_{ACW}$	MEx$_{DC}$	MEx$_{PM}$	2M	3M	4M
$MAML^p$	**0.9713**	**0.8399**	**0.9857**	**0.9618**	**0.9980**	**0.9999**	**1.0000**
RN^p	0.9596	0.7444	0.9562	0.9229	0.9868	**0.9960**	**0.9960**
MN^p*	**0.9695**	0.7917	0.9606	0.9289	**0.9890**	**0.9922**	**0.9941**

settings, MAML^p, RN^p and MN^p* use architectures DNN(3), 2D-CONV(1) and DNN(1) respectively (the best feature learners for visual data).

Notably, for all single modal experiments, the best performance is achieved with the MAML^p algorithm and only for the ACT, MN^p* achieve comparable performance with MAML^p. While 2M experiment achieves the best performance with MAML^p, 3M, and 4M experiments achieve comparable performances with all three algorithms. Moreover, we find a significant improvement with personalisation in the multi-modal settings compared to best DL algorithms. We refer to Section 4.6.2 where 2M, 3M and 4M experiments achieved an accuracy of 0.9421, 0.9669 and 0.9649 respectively. Importantly, all three algorithms achieve approximately 100% accuracy with the three multi-modal experiments, which is a significant achievement for creating high performance exercise monitoring applications.

7.3.2 Personalised vs Conventional Meta-learners

In this section, we explore the impact of the personalisation in the meta-learning methodology. We visualise the training of the adaptation-optimised meta-learner, MAML and the similarity-optimised meta-learner, RN when applied the two methodologies using the PM dataset.

Comparing MAML vs MAML^p

We first investigate the performance improvements achieved by MAML^p against $MAML$ using the PM dataset. We compare the following three algorithms:

MAML: Original MAML algorithm (Finn et al., 2017) trained and tested using tasks that were created disregarding any personal identifiers;

MAML^p: Personalised MAML introduced in Section 5.2.2; and

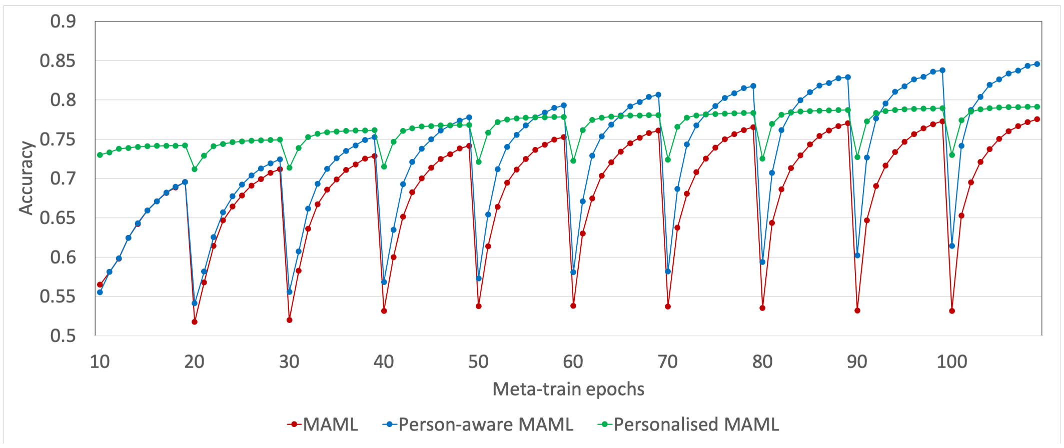
Person-aware MAML: A lazy personalisation of MAML where a meta-train task is comprised of a support set with *person-activity* classes selected from a set of persons. Each *person-activity* class in the support set is represented by k^s data instances that belong to a single person. However, each *person-activity* class may select data from any person in the meta-train set. The query set will have data from a single person who may not have been selected to form the support set. This method still preserves the concept of *person-activity* only at the class label level, but not over the entire support set or at the task level.

We create experiments in both $k^s = 1$ and $k^s = 5$ settings. The meta-model is evaluated using meta-test tasks at every 10 meta-train epochs to create 10 evaluation points over the 100 training epochs. At each evaluation point, the meta-test support set is used to adapt the current meta-model for 10 meta-gradient steps. After each meta-gradient step, the adapted meta-test model performance is evaluated using the meta-test query set. Accordingly, there are 100 evaluation points for each algorithm. Through this process, the adaptation of the generic meta-model is observed at different stages of optimisation. The graphs plot the mean test-accuracy obtained over person folds.

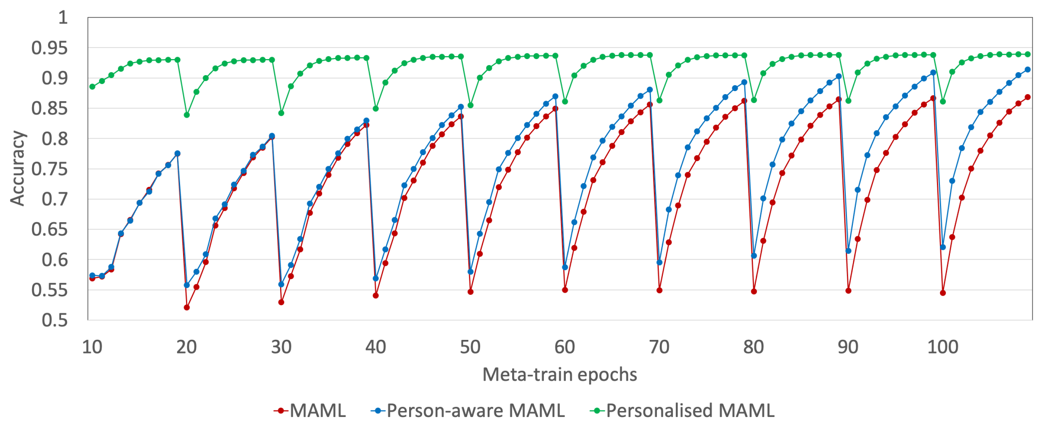
Results obtained for the three algorithms in the $k^s = 1$ setting are plotted in Figure 7.11a. Overall performance is improved over meta-train epochs with all three algorithms. Moreover, both $MAML^p$ and person-aware $MAML$ outperform $MAML$ at each evaluation point. $MAML$ creates a set of meta-train tasks disregarding the person parameter. When $k^s = 1$, there is no significant difference between the support sets of person-aware $MAML$ and original $MAML$, but $MAML$ create query sets that contain data instances from multiple persons. Accordingly, the task loss calculated for meta-model update is not restrained to a single person sourced query set to learn personalisation patterns.

At 30 epochs (and after adaptation, i.e. at the 10th meta-gradient step), person-aware MAML matches the meta-test accuracy of MAML^p and thereafter, at each evaluation point, person-aware MAML outperform both $MAML^p$ and $MAML$. Given there is only 1 labelled data instance per *person-activity* in the support set and only one person's data is in the query set, person-aware $MAML$ in the $k^s = 1$ setting may emulate $MAML^p$, which leads to higher accuracy over $MAML^p$. We observe that $MAML^p$ achieves the best performance before adaptation. In addition, over the 10 meta-gradient steps, $MAML^p$ meta-model adaptation significantly faster compared to both person-aware MAML and $MAML$. This result is attributed to $MAML^p$ creating person-tasks, and the resulting meta-model of $MAML^p$ is the best starter model for any person.

Results obtained for the three algorithm in the $k^s = 5$ setting is plotted in Figure 7.11b.



(a) $k^s = 1$ setting



(b) $k^s = 5$ setting

Figure 7.11: Comparison of meta-model performance: MAML vs. person-aware MAML vs. personalised MAML

Once the number of labelled data instance per *person-activity* class is increased to 5, we observe a clear advantage of the personalised MAML over person-aware MAML or conventional $MAML$. In the $k^s = 5$ setting, with 5 labelled examples per *person-activity* class, a person-aware task has increased the variations within the support set and now resembles a conventional meta-learning task. Accordingly, meta-test accuracy of person-aware $MAML$ outperforms $MAML$, but fails to outperform $MAML^p$.

$MAML^p$ achieves the best performance at each evaluation point. $MAML^p$ exhibit fast learning of the meta-model by achieving significant meta-test accuracy even at 10 meta-train epochs. Moreover, MAML^p exhibit rapid meta-model adaptation only taking 5 meta-gradient steps on average, where person-aware $MAML$ and $MAML$ adaptation takes longer. And, similar to $k^s = 1$, MAML^p achieves the best performance before adaptation. We reiterate that meta-test tasks of all three algorithms are ensured personalisation in the LOPO setting. Therefore, any meta-test adaptation variations or final meta-test accuracy differences are only attributable to the learning of the meta-model. Moreover, the learning is influenced by the compositional differences of the tasks created by each methodology.

Overall, we highlight that personalised meta-learner $MAML^p$ improves meta-model generalisation and rapid adaptation. While person-aware $MAML$ fail to outperform $MAML^p$ when provided with few labelled data instances, the lazy personalisation can be useful in a restricted setting. For instance, if the test person is unable to provide a support set for every activity class, with person-aware $MAML$, a combination of generic labelled data instances and person-specific data instances should form a cohesive support set for successful personalised activity recognition.

Comparing RN vs RN^p

We investigate the performance improvement achieved by RN^p using the personalised meta-learning methodology in this section. The two algorithms compared in this section are:

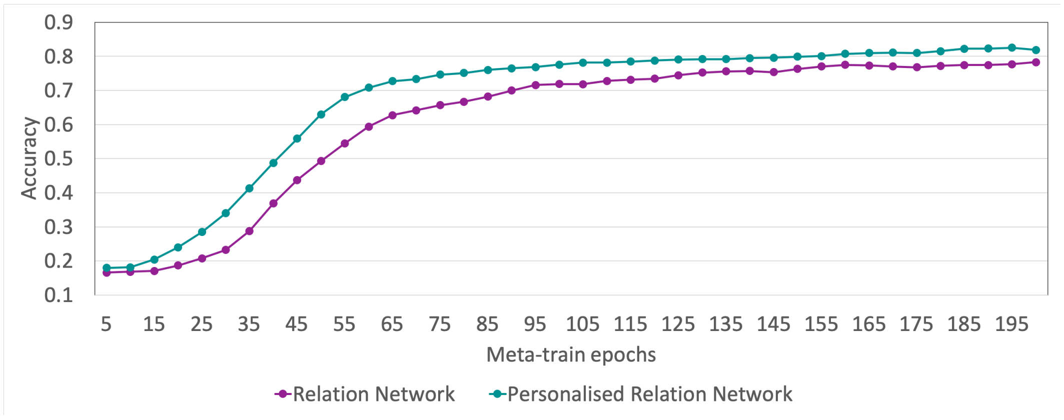
RN: Original RN (Sung et al., 2018); and

RN^p: Personalised RN introduced in Section 5.2.2

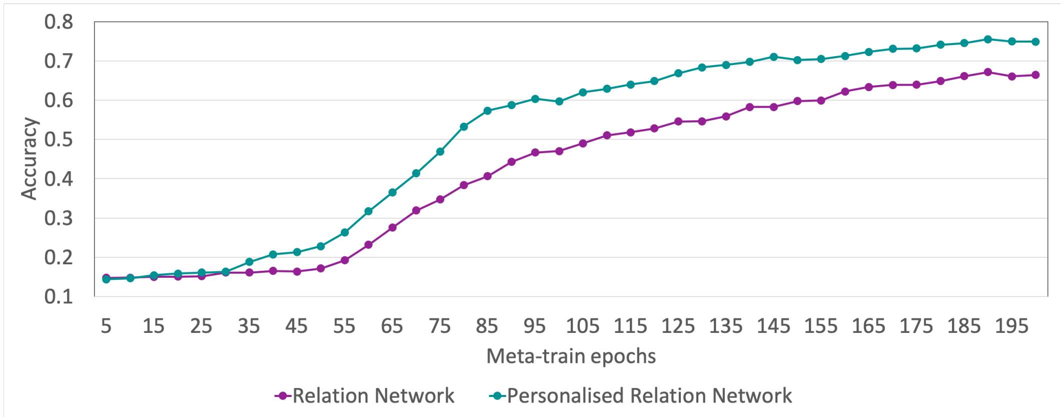
The experiments with the PM dataset are created in two settings $k^s = 1$ and $k^s = 5$. The meta-model is evaluated at every 5 epochs using the meta-test tasks to create 40 evaluation points for each algorithm over the 200 training epochs. The mean accuracy over 30 person folds obtained using the LOPO evaluation methodology are plotted in Figures 7.12a and 7.12b.

Both $k^s = 1$ and $k^s = 5$ settings clearly show that personalisation has improved meta-test performance. In the $k^s = 1$ setting, both methods consistently improve performance over the training epochs, but the personalised methodology significantly outperforms the conventional methodology at each evaluation point. For instance, at 100 epochs, RN achieves 71.57% accuracy and RN^p achieves 77.6% accuracy, RN achieves 77.51% performance (to match RN^p at 100 epochs) only after 160 epochs.

It is noteworthy that overall performance achieved in the $k^s = 5$ is significantly lower



(a) $k^s = 1$ setting



(b) $k^s = 5$ setting

Figure 7.12: Comparison of meta-model performance: RN vs. personalised RN

compared to $k^s = 1$ setting. We also observe a staggered start compared to $k^s = 1$ setting where both algorithms struggle until ∼50 epochs. In Section 7.3 we observed that reasoning with complex visual data is challenging for both RN and RN^p. In addition, the increased number of elements in the support set has made the model optimisation even more challenging. However, this visualisation focus on the comparison of the two algorithms in the $k^s = 5$ setting. After the staggered start, the performance of RN^p and RN consistently improve over the training epochs. The personalised methodology significantly outperforms the conventional methodology at each evaluation point. At 100 epochs, RN achieves 47.05% accuracy while RN^p achieves 59.68% accuracy. At the end of 200 epochs, RN has achieved an accuracy of 66.44% which RN^p achieved at 125 epochs.

When training RN in the $k^s = 5$ setting, a task is created by disregarding the person parameter. As a result, an activity class in the support set has representatives from many persons. Learning similarities from multiple persons have adversely affected the RN meta-model. Accordingly, a performance difference of 8.46% is observed between RN and RN^p in the $k^s = 5$ setting at the end of 200 epochs. In contrast, in the $k^s = 1$ setting, a task contains only one data instance per class; for RN it is from a random person, for RN^p it is from the same person. However, there is no variability within the support set for a given activity class. Thus the performance difference is narrower, at 3.64%.

Overall, the personalised methodology has led to learning a generalised meta-model that is better suited for any meta-test task from a person not seen during training.

7.3.3 Visualising Variability Between Persons

In a real-world application, the data for meta-model training is likely to be provided by physiotherapy experts performing exercises. For instance, a physiotherapy clinic can create their meta-model using the data collected from the physiotherapists. Patients who are registered at the clinic can receive the application which embeds this model to take home and use for exercise monitoring. Accordingly, a reasoning model is trained using data from experts and is personalised and used by a non-expert user. While experts perform exercises uniformly with less variability, non-experts perform exercises with added personal characteristics. This can be viewed as a specialised evaluation setting that follow person-aware methodology presented in Section 3.4. Given MEx recognise persons in the dataset as experts or non-experts, we explore this scenario with the MEx dataset. MEx has data from 7 physiotherapy experts and 23 non-experts.

We select the PM modality and $k^s = 1$ setting (to create meta-train and meta-test tasks) to demonstrate this specialised setting. Three meta-models are trained using MAML^p, RN^p and MN^{p*}. Each model is trained using the data from the 7 experts and tested by each non-expert. The main difference to LOPO evaluation setting is that we fix the train set to only include experts such that we can observe the performance variations of non-experts against a fixed meta-model created by all expert data. Similar to previous evaluations, a meta-model is evaluated at every 10 meta-train epochs. MAML^p algorithm has a re-training step to personalise a meta-model and to demonstrate the performance gain achieved by personalisation we plot both before and after meta-model adaptation. Figure 7.13 present the results. The bold colour line for each algorithm plots the mean accuracy over 23 folds (non-experts) and light colour lines plot individual performances.

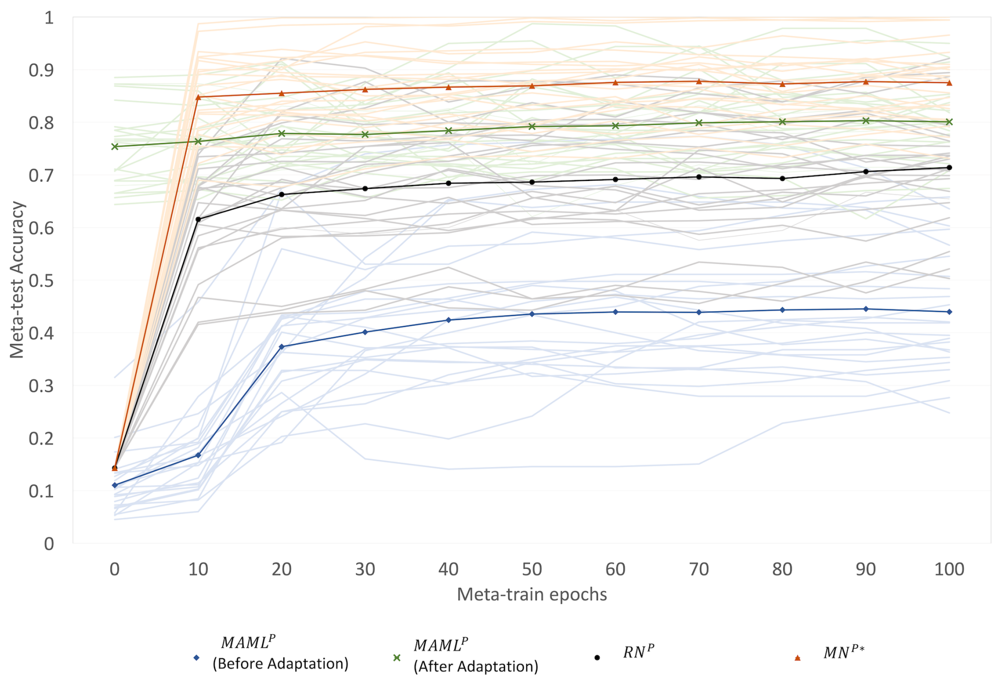


Figure 7.13: Personalisation variability between non-expert users

Overall, MN^{p}* outperform MAML^{p} and RN^{p}. In this discussion, we focus on variable performance between persons. The general trend is that most persons show improvements with increasing training epochs. The differences between the worst performing person and the best performing person are 27.57%, 41.85% and 26.72% for MAML^{p}, RN^{p} and MN^{p}* respectively. Accordingly, we find that MN^{p}* and MAML^{p} learn meta-models that are better generalisable to new, unseen persons compared to RN^{p}. Comparison between the MAML^{p} meta-model performance before and after adaptation highlights the importance of personalisation. In the $k^s = 1$ setting, only one data instance for an exercise is obtained from the non-expert user to personalise the MAML^{p} meta-model. For instance, at the 100th epoch, after re-training the MAML^{p} meta-model with the 7 data instances, non-expert users have achieved a mean performance improvement of 36.1%. We observe that the performance variance in a single modal setting is significantly high with all three algorithms for deployment in the real world. However, we expect the variance to be lower when $k^s > 1$ and in multi-modal settings as shown in Section 7.3.1.

7.4 Open-ended Recognition with Similarity-optimised Meta-learners

In this section, similarity-optimised meta-learners for personalised and open-ended recognition introduced in Chapter 6 are evaluated. This evaluation is threefold: a comparative evaluation against baselines in Section 7.4.1; a detailed evaluation of performance variances between different unseen classes in Section 7.4.2; and a detailed evaluation of performance variances when increasing the number of unseen classes in Section 7.4.4.

7.4.1 Comparative Evaluation

Conventional open-ended recognition algorithms rely on the nearest neighbour algorithm to select the most similar semantic attribute representation at test time for class predictions (Cheng et al., 2013b; Ohashi et al., 2018). Instead of semantic attributes, our open-ended methodology uses few data instances to represent an unseen class. Similarity-optimised meta-learners in an open-ended setting can be seen as parametric nearest neighbour algorithms, as discussed in Section 6.2.3. For instance, given a query instance, a similarity optimised meta-learner finds the best matching instance from a set of labelled data instance (support set) to predict a class label. Accordingly, in this section, we compare how meta-learners in the open-ended setting perform against the k-nearest neighbour algorithm. The algorithms compared are listed below:

k-NN: k-Nearest Neighbour algorithm with cosine similarity.

PN^o: Personalised Open-ended Prototypical Networks, introduced in Section 6.2.3.

We note that it is infeasible to compare the performance of our open-ended methodology against attribute-based open-ended recognition methods from recent literature using MEx. Such methods are knowledge-intensive and require an intermediary attribute mapping that is not available for MEx. Therefore, we perform a comparative evaluation against the methods from recent literature using the HDPoseDS datasets in Section 7.5.3.

Implementation Details

With k-NN, we replicate the personalised open-ended setting where a personal support set is created by sampling 5 data instances per activity class for each person ($k^s = 5$). k-NN is evaluated using the remaining data instances of the same person. Each experiment is repeated 20 times with a different support set. We evaluate k-NN for k=1 and k=5.

We select the best performing open-ended meta-learning algorithm PN^o for this comparative evaluation instead of MN^{o}* and RN^o according to results from Sections 6.3.1 and 6.3.2. Experiments are evaluated in the L1CO setting, and therefore, PN^o performs 6-way training and 7-way testing with $k^s = 5$. Importantly, at test time, both k-NN and PN^o has access to the same amount of data instances when predicting the class label for a given query. Experiments are performed with each individual modality and the best multi-modal combinations of the MEx dataset. We record mean accuracy for all experiments with error bars to show the best and the worst performing unseen classes for each algorithm.

Results

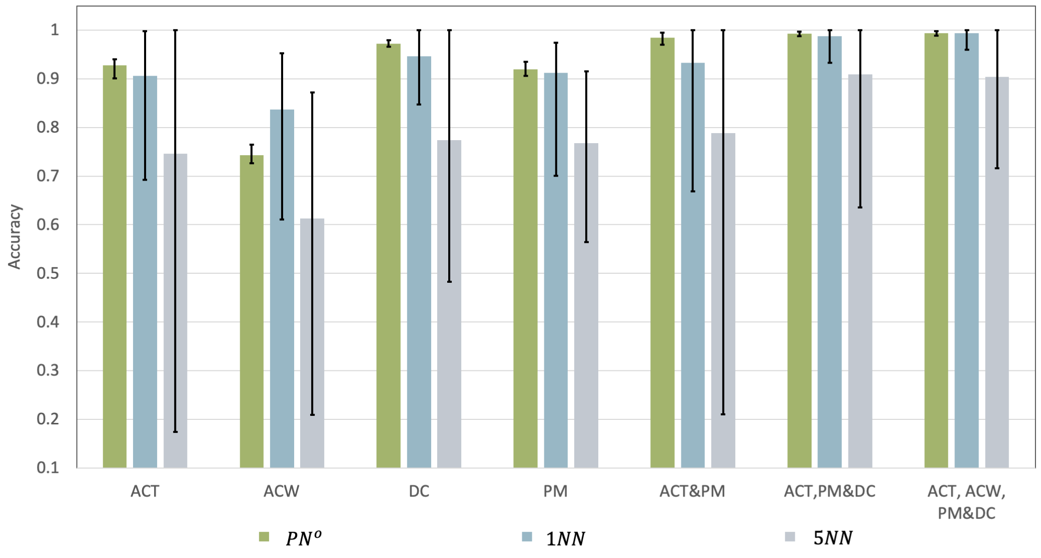


Figure 7.14: PN^o vs. k-NN performance comparison with MEx datasets

Figure 7.14 plot the performance of the three algorithms using the MEx datasets. Overall, PN^o outperform both 1-NN and 5-NN with 6 out of the 7 experiments. Notably, with ACW, 1-NN significantly outperform PN^o. This 1-NN performance can be attributed to the noisiness of the wrist accelerometer data where learning with only a few noisy data instances can be detrimental to learning a successful feature representation using a parametric model.

1-NN achieve comparable performances to PN^o with 3 of the 7 experiments. However, the variability in performance for different unseen classes is significantly high compared to PN^o. For instance, with ACT, the mean accuracy achieved by PN^o and 1-NN are

0.9276 and 0.9058 respectively, and the difference between best performance and worst performances for an unseen class are 3.92% (0.9012∼0.9404) and 30.48%(0.6928∼0.9976). The high variability of performance for different unseen classes is detrimental in the real-world application where the model is unaware of end-user preferences.

Interestingly with all 7 experiments, 5-NN fails to outperform 1-NN. 1-NN is most similar to classification where the top-1 data instance pair is selected based on similarity to obtain a class label. With 5-NN 5 pairs with the highest similarity are selected, and the class label is derived using majority voting. With k-NN, there is no opportunity to learn from the representatives for an activity class. Accordingly, independent similarity calculations lead to misleading results.

7.4.2 Performance variability between different unseen classes

Table 7.5: PN^o performance comparison by activity class with MEx datasets

Test class	MEx$_{ACT}$	MEx$_{ACW}$	MEx$_{DC}$	MEx$_{PM}$	2M	3M	4M
Knee rolling	0.9332	0.7432	0.9741	0.9119	0.9802	0.9874	0.9983
Bridging	0.9335	0.7436	0.9619	0.9060	0.9950	0.9968	0.9978
Pelvic tilt	0.9257	0.7321	0.9760	0.9229	0.9845	0.9925	0.9936
Bilateral clam	0.9341	0.7327	0.9661	0.9190	0.9876	0.9938	0.9900
Extension in Lying	0.9404	0.7267	0.9720	0.9351	0.9832	0.9958	0.9946
Prone Punch	0.9250	0.7594	0.9799	0.9179	0.9915	0.9931	0.9949
Superman	0.9012	0.7647	0.9734	0.9221	0.9700	0.9884	0.9890
conv. mean	0.9276	0.7432	0.9719	0.9193	0.9846	0.9925	0.9940
gen. mean	0.9326	0.7389	0.9720	0.9223	0.9831	0.9928	0.9928

Table 7.5 present the detail results for each unseen class using the PN^o algorithm. Lowest variability among 1M experiments is observed with the DC modality (1.80%) followed by the PM modality (2.91%). This observation suggests that they are the best single modal settings recommended for an open-ended exercise recognition application.

The multi-modal results are presented in the last three columns where 2M, 3M and 4M refer to MEx$_{ACT,PM}$, MEx$_{ACT,PM,DC}$ and MEx$_{ALL}$ datasets. In multi-modal settings, the observed variability among unseen classes is narrower. Multi-modal combinations have increased the overall performance and reduced the variability between unseen classes. These results highlight the need for multiple modalities to account for a wide range of unseen classes that may encounter after deployment.

We observe that the performance on the generalised setting and conventional settings are comparable. Accordingly, we find that the PN^o feature learner that was trained on data from seen classes has not over-fit to seen classes. If the feature learner over-fit to seen classes, the performance in the conventional setting (test performance of only the unseen class) should be lower than the performance in the generalised setting (test performance of all classes).

7.4.3 Few-shot vs Open-ended Recognition

Literature shows that integration of an activity-attribute mapping improves recognition performance, specifically in a few-shot setting Liu et al. (2011). Hybrid classification models introduced by Nguyen et al. (2015b) and Cheng et al. (2013a) exploit this idea to augment conventional classification using learned features with hand-crafted intermediary features to improve recognition performance. Since our method of open-ended recognition stems from few-shot learning, we compare how performances compare in a personalised recognition setting vs personalised open-ended recognition setting. Accordingly, we compare personalised prototypical networks, PN^p, and personalised open-ended prototypical networks PN^o in Table 7.6. Here PN^p is trained in the 5-shot, 7-way setting (i.e. $k^s = 5, |\mathcal{C}| = 7$) and PN^o is trained in the 5-shot generalised L1CO setting (i.e. $k^s = 5, |\mathcal{C}| = 6$). This also allows to further evaluate if the introduction of the unseen class has been detrimental to the performance of seen classes.

Table 7.6: Personalised RN vs. personalised and open-ended RN performance comparison with MEx datasets

Test class	MEx$_{ACT}$	MEx$_{ACW}$	MEx$_{DC}$	MEx$_{PM}$	2M	3M	4M
PN^p	0.9529	0.7641	0.9727	0.9374	0.9899	0.9945	0.9960
PN^o (L1CO)	0.9326	0.7389	0.9720	0.9223	0.9831	0.9928	0.9928

The observed performance drop with 1M datasets, ACT, ACW, DC and PM are 2.03%, 2.52%, 0.07% and 1.51%. With modality combinations, 2M, 3M and 4M it is 0.68%, 0.17% and 0.32% respectively. The difference between PN^p and PN^o here is that PN^p perform 7-way train and test classification and PN^o perform 6-way train and 7-way test classification. This difference makes the two performances not directly comparable. However, the performance decline can be seen as an indication of a modality or a modality combination's capacity to perform in an open-ended setting. This is in addition to the rather intuitive performance drop in the open-ended recognition, when trained in a 6-way setting. Importantly, we highlight that the introduction of the unseen class has not been significantly detrimental to seen classes. Especially using multi-modal data, we achieve comparable performances in the open-ended generalised setting and the few-shot learning setting.

7.4.4 Performance variability with increasing unseen classes

In this section, we evaluate the impact on performance when increasing the number of unseen classes. We follow the LNCO evaluation detailed in Section 3.4.1 and implementation details are similar to Section 6.3.2. The experiments are performed for each modality and the best modality combinations of the MEx dataset. Accordingly, we investigate if improving the sensor modality coverage using multiple modalities can improve the performance of the open-ended recognition algorithm compared to single modal settings. We use the PN^o algorithm as found to be the best out of the three similarity-optimised meta-learners in Section 6.3.2. Table 7.7 detail the results obtained for the LNCO experiments.

Table 7.7: PN^o performance comparison for increasing unseen classes (LNCO) with MEx datasets

Datasets	L1CO	L2CO	L3CO	L4CO
MEx_{ACT}	0.9276	0.9212	0.9015	0.8713
MEx_{ACW}	0.7432	0.7409	0.7347	0.7245
MEx_{DC}	0.9719	0.9679	0.9641	0.9557
MEx_{PM}	0.9193	0.9254	0.9195	0.9117
2M	0.9846	0.9706	0.9690	0.9510
3M	0.9925	0.9892	0.9901	0.9865
4M	0.9940	0.9915	0.9876	0.9821

All datasets exhibit a decline of performance with an increasing number of unseen classes. The performance drop ranges from 5.63% to 0.76% for MEx single modality experiments. However, it is less severe with modality combinations; 3.36% for 2M, 0.6% for 3M and 1.19% for 4M. 3M and 4M multi-modal settings consistently achieve performances higher than 98% even with 4 unseen classes. These results demonstrate that in multi-modal settings, the modality combinations capture discriminatory features to recognise multiple new and unseen activities with high precision. Results suggest that the PN^o model manages to maintain recognition performance despite increasing the number of unseen class in a multi-modal settings. Having this form of robustness is a desirable feature for real-world deployment.

7.5 Transferability to other HAR domains

In this section, the methods introduced in Chapters 4, 5 and 6 are evaluated for the transferability to other HAR domains. We consider everyday fitness activities using the selfBACK Dataset; and daily living activities using the PAMAP2 dataset. In addition, we evaluate methods from Chapter 6 using the HDPoseDS datasets, which allows to compare our methods against the most recent algorithms from literature.

7.5.1 Multi-modal Hybrid Attention Fusion

In this section, we evaluate the performance of the Multi-modal Hybrid Attention Fusion architecture, mHAF introduced in Chapter 4 with two HAR datasets, selfBACK and PAMAP2. The algorithms considered in this comparative evaluation are listed in Section 7.2.1. Relevant model implementation details are also found in Section 7.2.1 and relevant data pre-processing steps are as detailed in Section 3.3.1. All experiments follow LOPO evaluation to report mean F1-score over the person folds. Notably, both datasets are homogeneous multi-modal datasets with multiple accelerometer sensors. Accordingly, we will be evaluating the mHAF architecture for homogeneous modality combinations.

selfBACK Dataset

mHAF architecture for the selfBACK dataset is constructed with optimal feature learners discovered in Section 4.6.1. Considering the comparable performance achieved by 2D-CNN and 1D-CNN-LSTM architectures and performance of accelerometer data in other datasets, we select 1D-CNN-LSTM architecture for both SB_T and SB_W modalities.

In Table 7.8 details the performance results obtained using the selfBACK dataset. The first row refers to the best single modal performance obtained with the thigh sensor at 0.7753. mHAF architecture achieves 0.7919 F1-score to outperform the best single modality performance. mHAF achieves comparable performance with the three baselines numbered (2) to (4). Moreover, mHAF fails to outperform the two algorithms numbered (5) and (6). We attribute the performance achieved by (5) and (6) to applying early and mid-level fusion to the homogeneous modality combination in the selfBACK dataset. Best performance is achieved by the DeepConvLSTM architecture, which in comparison to DeepSense, is shallower and apply early fusion.

Table 7.8: mHAF performance comparison with selfBACK multi-modal datasets

Algorithm	SB_{WT}
(1)Best single modality (Thigh)	0.7753
(2)Early Fusion	0.8091
(3)Mid Fusion	0.7950
(4)Mid Fusion+temporal axis fusion	0.7926
(5)DeepConvLSTM	**0.8472**
(6)DeepSense	0.8156
mHAF	0.7919

PAMAP2 Dataset

PAMAP2 dataset consists of 3 modalities, and in this evaluation, we consider the best 2 modality combination $PAMAP2_{HA}$ and all modality combination $PAMAP2_{HCA}$. mHAF architecture for both modality combinations are created using the 1D-CNN-LSTM architecture discovered as the best modality-specific feature learner for all 3 modalities in Section 4.6.1.

Table 7.9: mHAF performance comparison with PAMAP2 multi-modal datasets

Algorithm	$PAMAP2_{HA}$	$PAMAP2_{HCA}$
(1)Best single modality (Ankle)	0.8034	0.8034
(2)Early Fusion	**0.8905**	**0.8964**
(3)Mid Fusion	**0.8935**	**0.8988**
(4)Mid Fusion+temporal axis fusion	**0.9021**	**0.9007**
(5)DeepConvLSTM	**0.8962**	**0.8985**
(6)DeepSense	**0.8895**	0.8868
mHAF	**0.9004**	**0.9070**

Table 7.9 details the comparative performance results obtained using the PAMAP2 dataset. The first row refers to the best single modality performance achieved by the ankle accelerometer, and both 2M and 3M combinations significantly outperform single modality performance. Similar to selfBACK dataset, the three baselines (2), (3) and (4) achieve comparable performances to mHAF architecture. While DeepConvLSTM architecture achieves comparable performance to mHAF, DeepSense fails to outperform mHAF. Results from both 2M and 3M combinations further affirm that early and mid fusion levels are more suited for homogeneous modality combinations. Moreover, similar

to selfBACK, the shallow and early fusion DeepConvLSTM architecture outperforms DeepSense architecture with both modality combinations.

In summary, selfBACK and PAMAP2 results highlight that homogeneous modality combinations benefit from early and mid level fusion compared to late fusion.

7.5.2 Personalised Meta-learning

In this section, we evaluate the performance of the personalised meta-learners introduced in Chapter 5 using the two HAR datasets selfBACK and PAMAP2. We highlight that compared to MEx, selfBACK and PAMAP2 have significantly more data instances per *person-activity*. Accordingly, with these datasets, we compare how personalised meta-learners for few-shot classification transfer to considerably larger HAR datasets. The algorithms considered in this comparative evaluation are listed in Section 7.3. Both selfBACK and PAMAP2 have accelerometer modalities and the data pre-processing steps are as detailed in Section 3.3.1. For comparability, DNN(1) feature learner from Section 5.3.1 is used with all algorithms and algorithm implementation details can be found in Section 7.3. All experiments follow LOPO evaluation to report mean accuracy over the person folds.

selfBACK Dataset

Table 7.10: Comparative evaluation of personalised meta-learners with selfBACK datasets

Algorithm	SB_W	SB_T	SB_{WT}
DL(Accuracy)	0.6959	0.7908	0.8253
$MAML$	0.7532	0.8398	0.8997
$MAML^p$	0.8075	0.8625	0.9339
RN	0.8276	0.9334	**0.9596**
RN^p	0.8528	**0.9487**	**0.9622**
MN	0.7669	0.8392	0.8733
MN^p	**0.8653**	0.9124	**0.9609**
MN^{p*}	0.8140	0.9219	0.9401

In Table 7.10 we summarise the comparative performances obtained using the selfBACK dataset. Experiments are performed using both single modalities, SB_W and SB_T and the multi-modal combination SB_{WT}. Personalised meta-learner significantly outperform the DL algorithm performance, outperform their conventional counterpart. Similar to observations from MEx modalities, similarity-optimised meta-learners achieve the best performance with the accelerometer modalities and modality combination in the SELFBACK dataset. We observe that RN^p and MN^{p}* methods fail to outperform MN^p few-shot learner with the noisy SB$_W$ modality. SB$_T$ modality achieves the best performance with RN^p, significantly outperforming both MN^p and MN^{p}*. MN^p, RN^p and *RN* achieve comparable performances for the SB$_{WT}$ combination. We attribute the high performance of the few-shot learner MN^p to the availability of a moderate quantity of training data for each *person-activity*. Overall we find RN^p is the most robust algorithm with all three datasets.

PAMAP2 Dataset

Table 7.11: Comparative evaluation of personalised meta-learners with PAMAP2 datasets

Algorithm	PAMAP2$_H$	PAMAP2$_C$	PAMAP2$_A$	2M	3M
DL(Accuracy)	0.75056	0.7878	**0.8075**	**0.9016**	**0.9089**
MAML	0.7593	0.7626	0.6830	0.8573	0.8755
MAML^p	**0.8037**	0.7822	0.7256	0.8912	**0.9031**
RN	0.7818	0.8170	0.7527	0.8824	0.8884
RN^p	0.7868	0.8294	0.7761	0.8824	0.8971
MN	0.6299	0.7243	0.7123	0.8071	0.8847
MN^p	0.7239	**0.8402**	**0.8128**	0.8404	0.8857
*MN^p**	0.7086	0.7922	0.7471	0.8826	**0.9077**

In Table 7.11 we summarise the comparative performances obtained using the PAMAP2 dataset. Experiments are performed using the three single modalities, PAMAP2$_H$, PAMAP2$_C$, and PAMAP2$_A$, best 2 modality combination, PAMAP2$_{HA}$ (2M) and the default 3 modality combination PAMAP2$_{HCA}$ (3M). PAMAP2 dataset has comparably more data instance per *person-activity* class, and it is reflected by achieving the best performance with DL methods by 2M and 3M modality combinations. In contrast to performances of other inertial modalities in SELFBACK and MEx, MAML^p outperforms RN^p with two datasets and achieve comparable performance with one dataset. Overall, we find PAMAP2 results are not as conclusive as SELFBACK or MEx results. Thus, it is difficult to find a single algorithm that fit all modalities and modality combinations.

Compared to SELFBACK and MEx, PAMAP2 has access to a moderate amount of labelled data. In addition, PAMAP2 dataset contains activities of daily living where it can be challenging to find patterns between two persons performing an activity like *ironing* or *vacuum cleaning*.

7.5.3 Open-ended Meta-learning

In this section, we evaluate the performance of the personalised open-ended similarity optimised meta-learners introduced in Chapter 6 using the three HAR datasets HDPoseDS, SELFBACK and PAMAP2.

HDPoseDS Dataset

We compare the performance of the similarity optimised open-ended meta-learner PN^o, against two attribute-based open-ended recognition algorithms from the literature: Direct Attribute Prediction (Lampert et al., 2014); and Attribute Importance (Ohashi et al., 2018). Accordingly, the following algorithms are compared in this section:

- **DAP:** Direct Attribute Prediction method introduced by Lampert et al. (2014)
- **AI:** Attribute Importance method introduced by Ohashi et al. (2018) for open-ended pose classification
- **PN^o:** Personalised Open-ended Prototypical Networks introduced in Section 6.2.3

Both DAP and AI methods rely on hand-crafted activity-attribute mappings that are not available for datasets MEx, SELFBACK or PAMAP2. Accordingly, we include the HDPoseDS dataset in this section to evaluate our method against two attribute-based classifier methods which require expert knowledge. Notably, HDPoseDS dataset has only 27 data instances for a *person-activity*, thus resembles a few-shot scenario similar to MEx.

We experiment with four variants of the HDPoseDS dataset created by selecting subsets of modalities for unobtrusive use in the real-world (see Section 3.3 for details). We remove sensors considering their redundancy and obtrusiveness while maintaining the full-body sensor coverage to create the following four datasets.

- **$HDPoseDS_{17}$** excludes 14 on fingers
- **$HDPoseDS_{13}$** excludes 14 on fingers, 2 on upper legs, 2 on the lower arm

- **HDPoseDS$_{10}$** excludes 14 on fingers, 2 on upper legs, 2 on the lower arm, 2 on the upper arm, 1 on spine
- **HDPoseDS$_6$** excludes 14 on fingers, 2 on upper legs, 2 on the lower arm, 2 on the upper arm, 1 on spine, 2 on lower legs, 2 on shoulders

DAP and AI evaluations are performed in a conventional leave-one-class-out setting which we also follow in this section for comparability. DAP and AI experiments are performed using the full HDPoseDS dataset with 31 IMU modalities. We follow the L1CO evaluation methodology explained in Section 3.4.1 and report mean accuracy over the R-PHO experiments.

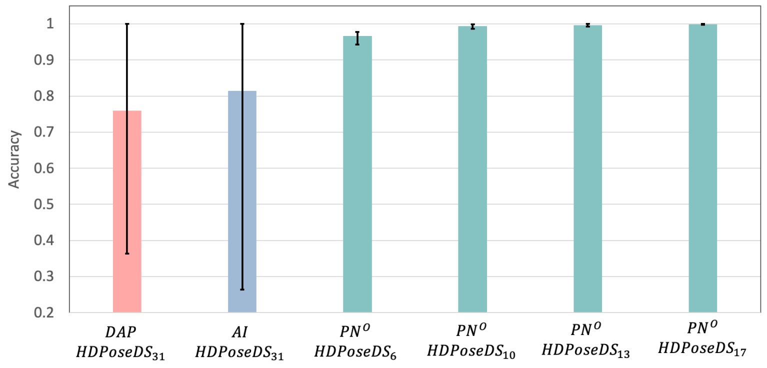


Figure 7.15: PN^o vs. DAP vs. AI performance comparison with HDPoseDS datasets

In Figure 7.15 we plot the mean performances obtained over the 22 L1CO experiments (for HDPoseDS, $|\mathcal{C}^*| = 22$). The column refers to the mean accuracy and error bars indicate the difference between the lowest performance and the best performance obtained by an unseen class using the respective algorithms. Overall, PN^o significantly outperforms DAP and AI algorithms. With the most restricted HDPoseDS$_6$ configuration, PN^o outperforms DAP and AI by 20.63% and 15.09%. Datasets HDPoseDS$_{17}$, HDPoseDS$_{13}$ and HDPoseDS$_{10}$ achieve comparable performances with PN^o while significantly outperforming the HDPoseDS$_6$ dataset.

PN^o exhibits significantly less variability across different unseen activity classes. The differences between the least performing experiment and the best performing experiment for PN^o using HDPoseDS$_{17}$ and HDPoseDS$_6$ are 0.22% (0.9978 $\sim$ 1.0000) and 3.45% (0.9429 $\sim$ 0.9774). In comparison to DAP and AI the differences are 63.61% (0.3639 $\sim$ 1.0000) and 73.59% (0.2642 $\sim$ 1.0000).

In Table 7.12 we present the detailed performances by each unseen class experiment. For the results from DAP and AI, we refer to Ohashi et al. (2018). The table is sorted by lowest to highest performances obtained by the *DAP* algorithm. For PN^o, we report the results in both conventional and generalised settings.

Table 7.12: PN^o performance comparison by pose class with HDPoseDS datasets

Test class	DAP	AI	PN^o with HDPoseDS			
			6	10	13	17
WaistTwistingR	0.3639	0.2938	0.9666	0.9891	0.9974	0.9978
StretchingForward	0.3703	0.8707	0.9692	0.9987	0.9956	0.9988
Sitting	0.4072	0.7438	0.9679	0.9979	0.9926	0.9984
WaistTwistingL	0.4241	0.2642	0.9711	0.9955	0.9934	0.9992
FoldingArm	0.4773	0.4387	0.9653	0.9982	0.9970	0.9997
Skiing	0.5277	0.7830	0.9754	0.9883	0.9971	0.9987
BaseballHitting	0.5856	0.7739	0.9577	0.9859	0.9959	0.9981
Boxing	0.6549	0.7494	0.9593	0.9970	0.9952	0.9995
StretchingCalfL	0.6647	0.8068	0.9758	0.9979	0.9986	0.9999
Standing	0.7148	0.6944	0.9712	0.9959	0.9983	0.9995
Thinking	0.8241	0.8230	0.9622	0.9956	0.9980	0.9997
Squatting	0.8922	1.0000	0.9667	0.9963	0.9934	0.9987
DeepBreathing	0.9061	0.9804	0.9459	0.9890	0.9991	0.9987
StretchingCalfR	0.9570	0.8897	0.9679	0.9963	0.9974	0.9991
PointingR	0.9629	0.9947	0.9774	0.9894	0.9971	0.9996
StretchingUp	0.9913	1.0000	0.9679	0.9930	0.9960	0.9986
HeelToBackR	0.9931	0.9729	0.9747	0.9943	0.9965	0.9997
PointingL	0.9937	0.9721	0.9765	0.9960	0.9961	0.9992
RaiseArmR	0.9973	0.9522	0.9429	0.9864	0.9944	1.0000
WaistBending	1.0000	0.9612	0.9588	0.9971	0.9966	0.9981
HeelToBackL	1.0000	0.9787	0.9668	0.9930	0.9954	0.9998
RaiseArmL	1.0000	0.9854	0.9693	0.9923	0.9996	0.9992
Conv. mean	0.7595	0.8149	0.9991	0.9964	0.9938	0.9658
Gen. mean	-	-	0.9991	0.9962	0.9928	0.9642

PN^o outperforms DAP for 18 unseen classes, and outperform AI for 20 unseen classes with the HDPoseDS$_{17}$ dataset. PN^o outperforms DAP for 15 unseen classes, and outperform AI for 14 unseen classes with the HDPoseDS$_6$ dataset. These results suggest that while many activities are successfully captured with precision with limited sensors, some activities benefit from the sensor-rich setting. It is noteworthy that although the mean performance of AI is significantly higher than DAP, there are two unseen classes where the AI performance is lower than the lowest DAP performance.

To evaluate if PN^o is biased towards seen classes, we present performances in the generalised setting. In the generalised open-ended setting, each experiment is trained with 21 activity classes and tested for all 22 activity classes. The generalised setting achieves comparable performance to the conventional setting, which shows the PN^o is not biased towards seen activity classes.

Unbiased performance against seen classes and less variability across different unseen test classes is two important properties achieved by the PN^o algorithm compared to open-ended algorithms found in the literature. It is noteworthy that a comprehensive sensor coverage is essential to capture any activity that may encounter after deployment. Unbiased performance towards seen classes is essential in a real-world application where both seen and unseen classes need to be recognised.

selfBACK Dataset

In this section, we compare the performance of PN^o against the two baselines 1-NN and 5-NN using the selfBACK datasets SB$_T$, SB$_W$ and SB$_{WT}$. Relevant algorithm implementation details can be found in Section 7.4 and data pre-processing steps are as detailed in Section 3.3.1. Figure 7.16 plots the mean performances obtained in the conventional setting over 9 L1CO experiments (for selfBACK, $|\mathcal{C}^*| = 9$).

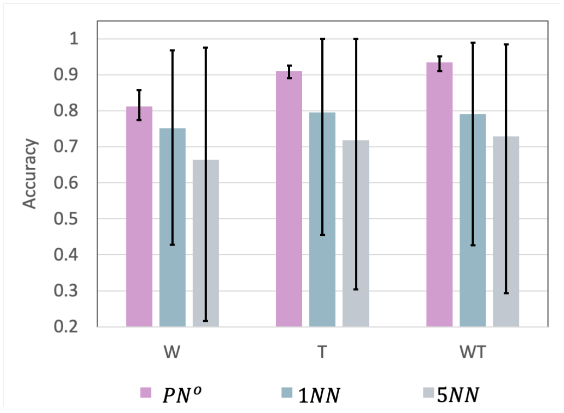


Figure 7.16: PN^o vs. k-NN performance comparison with selfBACK datasets

Overall, PN^o significantly outperform both 1-NN and 5-NN algorithms. Moreover, similar to observations in Section 7.4, we find that PN^o performance is consistent across different unseen classes. The differences between the least performing experiment and the best performing experiment for PN^o with SB$_{WT}$ is 4.00% (0.9108 $\sim$ 0.9508). This is in comparison to the more significant differences observed with 1-NN and 5-NN. Specifically for 1-NN algorithm the difference is 56.09% (0.4270 $\sim$ 0.9883) and for 5-NN algorithm the difference is 69.16% (0.2935 $\sim$ 0.9851). Notably, the overall performance of PN^o with SB$_{WT}$ (0.9352) is improved compared to SB$_T$ (0.9105), but the addition to of the wrist modality has been detrimental to some unseen classes indicated by the increased difference (3.48% and 4.00% for SB$_T$ and SB$_{WT}$).

A detailed summary of performances obtained for each unseen class using the PN^o algorithm is presented in Table 7.13. Similar to MEx and HDPoseDS, the generalised mean performance calculated for PN^o algorithm indicates that the bias towards seen classes or unseen classes is negligible.

Table 7.13: PN^o performance comparison by activity class with SELFBACK datasets

Test class	SB$_W$	SB$_T$	SB$_{WT}$
Walking downstairs	0.8221	0.9008	0.9149
Walking upstairs	0.8257	0.8900	0.9108
Walking fast pace	0.8032	0.9248	0.9455
Walking moderate pace	0.7735	0.9243	0.9497
Walking slow pace	0.7970	0.9107	0.9314
Standing	0.8101	0.9067	0.9366
Jogging	0.8161	0.9153	0.9392
Sitting	0.8577	0.9028	0.9379
Lying	0.8029	0.9194	0.9508
conv. mean	0.8120	0.9105	0.9352
gen. mean	0.8130	0.9103	0.9316

PAMAP2 Dataset

In this section, we compare the performance of PN^o against the two baselines 1-NN and 5-NN using the PAMAP2 datasets, PAMAP2$_H$, PAMAP2$_C$, PAMAP2$_A$, PAMAP2$_{HA}$ and PAMAP2$_{HCA}$. For implementation, evaluation and performance measure details we refer to previous sections. In Figure 7.17 we plot the mean performances obtained in the conventional setting over 8 L1CO experiments (for PAMAP2, $|\mathcal{C}^*| = 8$).

Overall, PN^o significantly outperform both 1-NN and 5-NN algorithms. Moreover, similar to MEx and SELFBACK, we find that PN^o performance is consistent across different unseen classes. The differences between the least performing experiment and the best performing experiment for PN^o with PAMAP2$_{HCA}$ is 4.93% (0.8515 $\sim$ 0.9008). Comparatively, more significant differences were observed for k-NN: 31.51% (0.6085$\sim$0.9636) for 1-NN; and 32.43% (0.6393$\sim$0.9636) for 5-NN. Similar to the HDPoseDS datasets, the

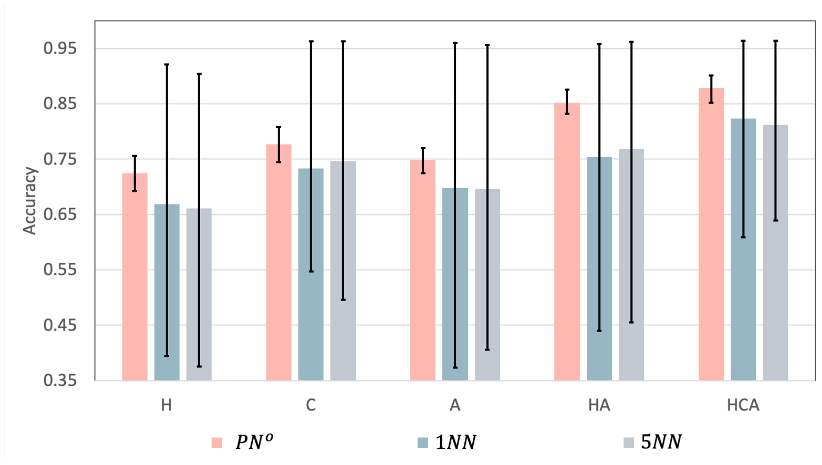


Figure 7.17: PN^o vs. k-NN performance comparison with PAMAP2 datasets

overall performance of PN^o is consistently improved with additional modalities and also significantly reduces the performance difference between unseen classes.

A detailed summary of performances obtained for each unseen class using the PN^o algorithm is presented in Table 7.14. Here, 2M and 3M refer to PAMAP2$_{HA}$ and PAMAP2$_{HCA}$ datasets. Similar to MEx, HDPoseDS and SELFBACK the generalised mean performance calculated for PN^o algorithm indicate that there is no significant bias towards seen classes.

Table 7.14: PN^o performance comparison by activity class with PAMAP2 datasets

Test class	PAMAP2$_H$	PAMAP2$_C$	PAMAP2$_A$	2M	3M
Descending stairs	0.7156	0.7449	0.7366	0.8353	0.8545
Sitting	0.7347	0.8084	0.7554	0.8617	0.8875
Ascending stairs	0.7565	0.7761	0.7320	0.8406	0.8603
Vacuum cleaning	0.7172	0.7826	0.7613	0.8759	0.9008
Ironing	0.7206	0.7861	0.7520	0.8318	0.8844
Standing	0.6922	0.7797	0.7547	0.8520	0.8886
Lying	0.7232	0.7453	0.7702	0.8738	0.8968
Walking	0.7386	0.7908	0.7244	0.8471	0.8515
conv. mean	0.7248	0.7767	0.7483	0.8522	0.8781
gen. mean	0.7233	0.7763	0.7459	0.8515	0.8763

LNCO with HDPoseDS, selfBACK and PAMAP2

Table 7.15 summarises the LNCO performance comparison for HDPoseDS, selfBACK and PAMAP2 datasets using the PN^o algorithm. This evaluation measures the robustness of the PN^o algorithm with an increasing number of unseen classes, and the evaluation setting is detailed in Section 3.4.1. Results for single modality PAMAP2 and selfBACK datasets are a summary from Section 6.3.2 where we compared the two similarity-optimised meta-learners PN^o and MN^o* to find PN^o is the most robust. It is noteworthy that there is no appropriate method in the literature to compare against these results.

Table 7.15: PN^o performance comparison for increasing unseen classes (LNCO)

Datasets	L1CO	L2CO	L3CO	L4CO	L5CO	L6CO	L7CO	L8CO
HDPoseDS$_{17}$	0.9991	0.9991	0.9989	0.9990	0.9983	0.9990	0.9986	0.9992
HDPoseDS$_{13}$	0.9964	0.9964	0.9972	0.9968	0.9940	0.9970	0.9962	0.9958
HDPoseDS$_{10}$	0.9938	0.9928	0.9948	0.9912	0.9945	0.9909	0.9871	0.9941
HDPoseDS$_{6}$	0.9658	0.9654	0.9600	0.9722	0.9717	0.9578	0.9575	0.9685
PAMAP2$_{H}$	0.7233	0.7265	0.7132	0.6901	-	-	-	-
PAMAP2$_{C}$	0.7763	0.8018	0.7593	0.7324	-	-	-	-
PAMAP2$_{A}$	0.7459	0.7678	0.7161	0.7121	-	-	-	-
PAMAP2$_{HA}$	0.8515	0.8476	0.8412	0.8020	-	-	-	-
PAMAP2$_{HCA}$	0.8763	0.8600	0.8352	0.8144	-	-	-	-
SB$_{W}$	0.8130	0.8180	0.8010	0.7952	-	-	-	-
SB$_{T}$	0.9103	0.9078	0.8958	0.8795	-	-	-	-
SB$_{WT}$	0.9316	0.9219	0.9126	0.8978	-	-	-	-

We observe that PN^o maintain consistent performance as we keep introducing up to 8 new classes after deployment with the HDPoseDS datasets. With PAMAP2 and selfBACK datasets, we observe that the performance gradually decrease as new unseen classes are introduced. These results are consistent with the results observed with the MEx single modality datasets in Section 7.4. In addition, similar to MEx multi-modal settings, with HDPoseDS, we observed that PN^o maintains performance with increasing unseen classes. Accordingly, we affirm that it is an added advantage to have a sensor-rich setting when performing open-ended recognition. We attribute minor random increments of performance to the random selection of test classes in the experiment design.

7.6 Chapter Summary

In this Chapter we presented the MEx dataset and a comprehensive evaluation of the methods introduced in Chapters 4, 5 and 6. We presented the sensor and exercise section, ethics processes followed in data collection. We presented a summary of the resulting dataset organised by the demographic information and also discussed few limitations.

Next, we presented an evaluation of the three technical contributions of this thesis for exercise recognition using the MEx dataset. First, we evaluated mHAF architecture for heterogeneous multi-modal fusion to show that our method outperformed several baselines and fusion algorithms from recent literature. We further verified this contribution using visualisations of confusion matrices and attention weights. Secondly, we evaluated the personalised meta-learners against conventional meta-learners and deep learning methods to show the positive effect of the personalisation methodology. We further verified this contribution by comparing the conventional and personalised meta-learner training progression over epochs. Finally, we evaluated the open-ended meta-learners against k-NN, which is the basis of attribute-based open-ended recognition algorithms. We visualised the robustness of our methods by evaluating for different unseen classes and increasing number of unseen classes.

Finally, we evaluated the three contributions using other HAR datasets. Multi-modal fusion evaluation showed that our architecture is less effective in homogeneous multi-modal settings. We evaluated personalised meta-learners in other HAR domains to show that our methods are highly effective when only a limited number of data instances is available. Finally, we evaluated open-ended meta-learners. Our method not only outperformed two methods from recent literature but also demonstrated robust performance across a range of activity domains in multi-modal settings.

Chapter 8

Conclusion

This thesis investigated the personalisation challenges of heterogeneous multi-modal exercise recognition. The following research questions were selected to address the challenges in the context of exercise recognition.

1. How to recognise exercises, given a set of sensor modalities, by learning modality and feature combinations and learning to discard noise?
2. How personalise exercise recognition to end-users with limited data and minimal end-user interaction?
3. How to extend exercise recognition to recognise new unseen exercises with limited data and minimal end-user interaction?

Five objectives were identified to address these research questions. This chapter discusses the contributions of this thesis by revisiting the initial research objectives and summarising key conclusions that emerged from each Chapter. This chapter also presents the limitations, future directions and desirable extensions before finally concluding this thesis.

8.1 Objectives Revisited

1. **Multi-modal Recognition: Develop a fusion algorithm to recognise exercises using a combination of heterogeneous sensor modalities.**

   The emphasis of this objective was to create a fusion architecture that learns to highlight significant features and understate noisy features from multiple modalities for each activity class. Given the limited of availability of labelled data, it is infeasible to learn such feature and modality combinations using a very deep architecture. To address this objective, we presented a Multi-modal Hybrid Attention Fusion architecture, MHAF.

In a heterogeneous multi-modal setting, modalities may prefer different feature learners, and supporting several modality combinations is infeasible with existing methods. To address this, we created a modular architecture where individual modalities learn from their specialist feature learners, and the resulting features were fused using the hybrid attention fusion module. Accordingly, the MHAF architecture was adaptable modality combinations to support to end-user modality preferences and restricted environments. Attention learns to emphasise desirable features, that contributes to reducing the depth of feature learners and consequently reducing the need for excessive quantities of training data. With the combination of hard and soft attention, we ensured that the MHAF learns a modality combination effective to recognise each activity class.

In evaluation, the specialist feature learner for each modality is identified empirically from a group of deep feature learners. Next, given a set of modalities, we identified the best minimal modality combinations for restricted settings. Results of these empirical evaluations informed the MHAF architecture to create bespoke models. We performed an ablation study which established the contribution and necessity of each element of the modular architecture.

2. **Personalised Recognition: Develop Exercise Recognition algorithms that are adaptable to unseen persons or person groups**

   Personalisation methods found in literature either demand end-users to provide labelled data to bootstrap models or need periodical end-user interactions with active learning methods. We propose that the key to achieving effective personalisation is the ability to personalise with few data instances. We investigated personalisation with few-data, where we treat a person as a task and *person-activity* as a class label. We introduced the personalised meta-learning methodology to train meta-learners that are adaptable to any new, unseen person using only few data instances. With this methodology, we learned a meta-model that is the best initialisation point for any person, and it is easily personalised using only a few data instances. We implemented the personalisation methodology with two meta-learners, and we also improved an existing personalised few-shot learning method using this new approach.

In the evaluation, an empirical study was performed to find the best feature learners for each modality for learning a meta-model. We find that simple dense architectures are best at extracting abstract features for a given set of activities from many persons. We explored different few-shot settings to find a balance between performance and usability. We also presented an exploratory evaluation on improving the training of RN^p algorithm. We showed how training RN^p as a metric learning task and a classification task is preferred over the original approach of training as a regression task.

3. **Open-ended Recognition: Develop algorithms to recognise exercises not seen during model implementation.**

   Physiotherapists prescribe multiple exercise plans to a patient, and it is infeasible to design an algorithm to recognise all expected exercise classes. To address this objective we explored a *knowledge-light* approach to introducing new and unseen activity classes, instead of the *knowledge-intensive* methods seen in the literature. We investigate how meta-learners performing zero-shot learning is not suitable in an open-ended setting. Instead, we introduced the personalised and open-ended meta-learning methodology and implemented with similarity-optimised meta-learners. The resulting approach enabled the dynamic expansion of the decision layer to integrate new activity classes as they are introduced to the model without any model re-training. Moreover, similar to personalisation, only few labelled data instances are needed from the end-user to integrate a new activity class to the existing model. This new data is not used in model re-training, instead used as representatives of the unseen classes.

   In evaluation, we compared three similarity-optimised meta-learners for personalised open-ended recognition. First, we evaluated the robustness of each algorithm by introducing different unseen activity classes using three HAR datasets. Given that the end-user can introduce any activity class after deployment, we highlighted the algorithm which achieves overall highest performance but also performed well across different activity classes. An end-user can introduce more than one new activity class, and the model needs to maintain performance in such a scenario. Accordingly, we evaluated the robustness of the meta-learners with an increasing number of unseen activity classes. We find that personalised and open-ended prototypical networks, PN^o, achieved the best performance across different activity classes, different modalities and with multiple unseen classes.

4. **Create an open, sensor-rich dataset for physiotherapy exercise recognition**

   An investigation of the literature showed the lack of publicly available datasets impedes the research and development of exercise recognition algorithms. Accordingly, we explore the sensors, exercises and data collection protocols that can be used in monitoring physiotherapy exercises. The resulting MEx dataset is collected for 7 physiotherapy rehabilitation exercises using 4 sensor modalities: 2 accelerometers, a depth camera and a pressure mat. Accordingly, this is a sensor-rich dataset with heterogeneous modalities. MEx has data recorded for 30 persons each performing 7 exercise, each for approximately 60 seconds. As a result, this dataset is not large compared to other general fitness HAR datasets found in the literature. We share the dataset as an open-research resource, to encourage further research in multi-modal fusion and to standardise activity recognition evaluation methods. This dataset was used in this thesis to fine-tune and evaluate our methods to establish the significance of our methods compared to methods found in recent literature.

5. **Conduct a comprehensive evaluation of all developed methods for physiotherapy exercise recognition**

   Finally we performed a comprehensive evaluation of all methods introduced for Objectives 1, 2 and 3 using the MEx dataset.

   A comparative evaluation showed that mHAF architecture outperformed several baselines and two methods from recent literature in the heterogeneous multi-modal setting for exercise recognition. By visualising confusion matrices and attention weights, we further verified that mHAF learned modality and feature combinations that are intuitive for each activity class. Notably, mHAF architecture is most effective in a heterogeneous multi-modal setting and achieved comparable performance in a homogeneous multi-modal setting.

   A comparative evaluation showed that personalised meta-learners outperform deep learners and conventional meta-learners. We also find that our methodology significantly improves the performance of a few-shot learning methods for personalised

HAR in literature. By visualising the training of meta-learners, we show that personalisation has improved the generalisability of the resulting meta-models. We also presented how different persons adapt meta-learners using their few personal data to find that adaptation-optimised meta-learners are the most robust across different persons and different modality combinations.

Finally, we perform a comparative evaluation to find that our open-ended meta-learning methodology implemented using PN^o outperformed kNN baseline and two attribute-based open-ended recognition algorithms found in literature. PN^o performance is consistent across different unseen classes compared to attribute-based methods. Accordingly, open-ended meta-learners are more effective for open-ended activity recognition with no model re-training after deployment. Furthermore, our method requires only few data instances from the end-user to integrate new classes to the existing recognition model.

This objective is evaluated using the MEx dataset collected in a lab environment with healthy participants, which is not an accurate representation of the real-world. Accordingly, we highlight that the performances presented in this objective can be too optimistic in practice with MSD patients.

8.2 Limitations and Future Work

This section highlights the limitations and implications of the work presented in this thesis and outlines some areas for consideration in future extensions.

We presented the heterogeneous multi-modal physiotherapy exercises dataset, MEx in Chapter 7. A notable limitation of the dataset is that the population of 30 persons does not emulate a population with musculoskeletal disorders. Due to the challenges of recruiting patients with musculoskeletal pain within a limited time, the data collection was conducted with participants recruited from the university. For instance, 100% of the sample were healthy individuals and 47% were in the age group of 18-24. As we visualised in Section 7.3.3, personal differences affect the recognition performance. Accordingly, there is a significant possibility that the performance measures we have reported in this thesis may vary significantly in a real-world setting with a different population.

Another limitation of the MEx dataset is the exclusion of a *null* activity class. In the recent past, only few HAR datasets were published with a *null* activity class to indicate that the person is not doing any of the intended activities (Anguita et al., 2013;

Koskimäki and Siirtola, 2014; Reiss and Stricker, 2012). For physiotherapy exercises, a *null* activity class can be interpreted as when the person is taking rests between exercises and performing warm-up or cool-down exercises. The initial intuition behind not including the *null* activity class was the large number of possibilities that may or may not be captured from all modalities. However, we find it is a significant limitation for deep learning models introduced in Chapter 4. This issue can be alleviated when using the methods presented in Chapters 5 and 6, where the end-user has the opportunity to integrate the *null* class as a new unseen activity class. The end-user can record few data instances of their interpretation of a *null* activity class using the preferred modalities, and thereafter the personalised and open-ended recognition model will consider *null* activity class as one of the possible activity classes.

In Chapter 7 we find how methods introduced in Chapters 4, 5 and 6 perform in both single-modal and multi-modal settings. While our methods outperform methods from recent literature and other baselines, we observe that performance in single-modal settings is not sufficient for real-world deployment. Recognition performance of methods introduced in Chapters 4 in a restricted home setting with one modality is between 63%∼90%. Performance is improved to between 74%∼97% when using personalisation methods introduced in Chapters 5. However, these methods require the patient to provide few data instances for personalisation. Also, we find that performance is again penalised when adding new unseen exercise classes using methods in Chapters 6. Evidently, a single modality fails to sufficiently capture discriminative features of exercises to achieve high accuracy. In addition, modalities with a high degree of freedom capture noise which penalises the recognition performance. In contrast, with 2M, 3M and 4M settings (multi-modal) we observed that recognition performance achieved is between 93%∼96% using fusion methods and 99%∼100% using personalisation. Also, in these multi-modal settings, the performance is between 95%∼98% with the addition of new unseen exercise classes. Importantly, multi-modal configurations have improved the coverage to capture features essential to recognise exercises with high accuracy. These observations confirm that a multi-modal setting is required for physiotherapy exercise recognition.

Integration of the exercise recognition algorithms introduced in this thesis with a digital self-management application is the most impactful future extension of this work. Such integration requires to design a hardware and software solution with many design considerations for unobtrusive and user-friendly deployment in a home environment. We find several design challenges that need to be addressed to implement such a solution. An embedded system needs to be implemented to connect and synchronise the multi-modal sensors. In addition the embedded system needs to maintain a consistent connection to a user interface in the form of a web or a mobile to transfer sensor data streams. Computational requirements and end-to-end latency will be contributing factors when selecting to host reasoning algorithms either remotely or on the edge device. Addressing these design challenges require investigation of research on edge computing and embedded systems.

A complete digital intervention for self-managing musculoskeletal pain should consist of three main components: monitoring exercises in real-time; recognition of the exercise being performed; and the evaluation of the quality of the performance. In this thesis, we investigated the first two steps to introduce effective monitoring methods with multi-modal configurations and recognition algorithms for improved performance. Evaluation of exercise performance quality remains an open research challenge. The performance quality of an exercise can be seen as how much actual execution deviates from the correct execution of the exercise performed under the supervision of a physiotherapist. Measuring this deviation can be either objective (i.e. measurable) or subjective (i.e. open to interpretation). Existing literature models the measurement of performance quality as a regression task in the domains of weight-lifting and rehabilitation exercises (Chen et al., 2013; Rybarczyk et al., 2017; Velloso et al., 2013). A rule-based model matches a query execution against a set of rules derived from correct execution to measure deviation. These methods are non-transferable to other exercise domains and need expert intervention to define the rules. We believe that the research in assessing performance quality can benefit from advances deep learning.

Addressing this challenge can be investigated in two directions. Firstly the performance quality can be viewed as alternative tasks such as a classification task, clustering task or a reconstruction task. Performance quality as a classification task or a clustering task is data-driven where data for multiple incorrect executions are needed to train a model. An interesting avenue is to investigate exercise performance quality as an anomaly detection task. The deep models, such as Auto-encoders can be trained using data from correct executions. The reconstruction loss of a query task can be considered as the measurement of quality where higher loss indicates a more deviation from the correct execution. Secondly, to support such research, we identify the lack of data in the literature. Accordingly, we plan to extend the MEx dataset by recording data in non-lab environments to capture performances that naturally deviate from the correct performance.

In conclusion, this thesis investigated three personalisation challenges in exercise recognition. The methods presented to address these challenges were motivated by the need to minimise data or knowledge requirements to improve usability. In future, these methods will contribute towards automating exercise monitoring in fitness applications that promote self-management of MSD.

Bibliography

Abadi, M., Barham, P., Chen, J., Chen, Z., Davis, A., Dean, J., Devin, M., Ghemawat, S., Irving, G., Isard, M., et al. (2016). Tensorflow: A system for large-scale machine learning. In *12th {USENIX} symposium on operating systems design and implementation ({OSDI} 16)*, pages 265–283.

Abdallah, Z. S., Gaber, M. M., Srinivasan, B., and Krishnaswamy, S. (2018). Activity recognition with evolving data streams: A review. *ACM Computing Surveys (CSUR)*, 51(4):1–36.

Al Machot, F., R Elkobaisi, M., and Kyamakya, K. (2020). Zero-shot human activity recognition using non-visual sensors. *Sensors*, 20(3):825.

Anguita, D., Ghio, A., Oneto, L., Parra, X., and Reyes-Ortiz, J. L. (2013). A public domain dataset for human activity recognition using smartphones. In *Esann*, volume 3, page 3.

Anjum, A. and Ilyas, M. U. (2013). Activity recognition using smartphone sensors. In *2013 ieee 10th consumer communications and networking conference (ccnc)*, pages 914–919. IEEE.

Antón, D., Goni, A., and Illarramendi, A. (2015). Exercise recognition for kinect-based telerehabilitation. *Methods of information in medicine*, 54(02):145–155.

Antos, A., Devroye, L., and Gyorfi, L. (1999). Lower bounds for bayes error estimation. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 21(7):643–645.

Ar, I. and Akgul, Y. S. (2014). A computerized recognition system for the home-based physiotherapy exercises using an rgbd camera. *IEEE Transactions on Neural Systems and Rehabilitation Engineering*, 22(6):1160–1171.

Ayoade, M. and Baillie, L. (2014). A novel knee rehabilitation system for the home. In *Proceedings of the SIGCHI conference on human factors in computing systems*, pages 2521–2530.

Bahdanau, D., Cho, K., and Bengio, Y. (2014). Neural machine translation by jointly learning to align and translate. *arXiv preprint arXiv:1409.0473*.

Bai, S., Kolter, J. Z., and Koltun, V. (2018). An empirical evaluation of generic convolutional and recurrent networks for sequence modeling. *arXiv preprint arXiv:1803.01271*.

Bendale, A. and Boult, T. E. (2016). Towards open set deep networks. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 1563–1572.

Berchtold, M., Budde, M., Gordon, D., Schmidtke, H. R., and Beigl, M. (2010). Actiserv: Activity recognition service for mobile phones. In *International Symposium on Wearable Computers (ISWC) 2010*, pages 1–8. IEEE.

Bleser, G., Steffen, D., Reiss, A., Weber, M., Hendeby, G., and Fradet, L. (2015). Personalized physical activity monitoring using wearable sensors. In *Smart health*, pages 99–124. Springer.

Blom, P. M., Bakkes, S., Tan, C. T., Whiteson, S., Roijers, D., Valenti, R., and Gevers, T. (2014). Towards personalised gaming via facial expression recognition. In *Tenth Artificial Intelligence and Interactive Digital Entertainment Conference.*

Bromley, J., Guyon, I., LeCun, Y., Säckinger, E., and Shah, R. (1994). Signature verification using a" siamese" time delay neural network. In *Advances in neural information processing systems*, pages 737–744.

Burns, D. M., Leung, N., Hardisty, M., Whyne, C. M., Henry, P., and McLachlin, S. (2018). Shoulder physiotherapy exercise recognition: ML the inertial signals from a smartwatch. *Physiological measurement*, 39(7):075007.

Chapron, K., Plantevin, V., Thullier, F., Bouchard, K., Duchesne, E., and Gaboury, S. (2018). A more efficient transportable and scalable system for real-time activities and exercises recognition. *Sensors*, 18(1):268.

Chavarriaga, R., Sagha, H., Calatroni, A., Digumarti, S. T., Tröster, G., Millán, J. d. R., and Roggen, D. (2013). The opportunity challenge: A benchmark database for on-body sensor-based activity recognition. *Pattern Recognition Letters*, 34(15):2033–2042.

Chen, C., Jafari, R., and Kehtarnavaz, N. (2017). A survey of depth and inertial sensor fusion for human action recognition. *Multimedia Tools and Applications*, 76(3):4405–4425.

Chen, P.-C., Huang, C.-N., Chen, I.-C., and Chan, C.-T. (2013). A rehabilitation exercise assessment system based on wearable sensors for knee osteoarthritis. In *International Conference on Smart Homes and Health Telematics*, pages 267–272. Springer.

Chen, S. and Jin, Q. (2016). Multi-modal conditional attention fusion for dimensional emotion prediction. In *Proceedings of the 24th ACM international conference on Multimedia*, pages 571–575.

Cheng, H.-T., Griss, M., Davis, P., Li, J., and You, D. (2013a). Towards zero-shot learning for human activity recognition using semantic attribute sequence model. In *Proceedings of the 2013 ACM international joint conference on Pervasive and ubiquitous computing*, pages 355–358.

Cheng, H.-T., Sun, F.-T., Griss, M., Davis, P., Li, J., and You, D. (2013b). Nuactiv: Recognizing unseen new activities using semantic attribute-based learning. In *Proceeding of the 11th annual international conference on Mobile systems, applications, and services*, pages 361–374. ACM.

Cho, K., Van Merriënboer, B., Bahdanau, D., and Bengio, Y. (2014a). On the properties of neural machine translation: Encoder-decoder approaches. *arXiv preprint arXiv:1409.1259.*

Cho, K., Van Merriënboer, B., Gulcehre, C., Bahdanau, D., Bougares, F., Schwenk, H., and Bengio, Y. (2014b). Learning phrase representations using rnn encoder-decoder for statistical machine translation. *arXiv preprint arXiv:1406.1078*.

Crema, C., Depari, A., Flammini, A., Sisinni, E., Haslwanter, T., and Salzmann, S. (2017). Imu-based solution for automatic detection and classification of exercises in the fitness scenario. In *2017 IEEE Sensors Applications Symposium (SAS)*, pages 1–6. IEEE.

Das, D., Busetty, S. M., Bharti, V., and Hegde, P. K. (2017). Strength training: A fitness application for indoor based exercise recognition and comfort analysis. In *2017 16th IEEE International Conference on Machine Learning and Applications (ICMLA)*, pages 1126–1129. IEEE.

Das, S., Thonnat, M., Sakhalkar, K., Koperski, M., Bremond, F., and Francesca, G. (2019). A new hybrid architecture for human activity recognition from rgb-d videos. In *International Conference on Multimedia Modeling*, pages 493–505. Springer.

Dash, M. and Liu, H. (1997). Feature selection for classification. *Intelligent data analysis*, 1(3):131–156.

Ding, Y., Tian, X., Yin, L., Chen, X., Liu, S., Yang, B., and Zheng, W. (2019). Multi-scale relation network for few-shot learning based on meta-learning. In *International Conference on Computer Vision Systems*, pages 343–352. Springer.

Escalera, S., Gonzàlez, J., Baró, X., Reyes, M., Lopes, O., Guyon, I., Athitsos, V., and Escalante, H. (2013). Multi-modal gesture recognition challenge 2013: Dataset and results. In *Proceedings of the 15th ACM on International conference on multimodal interaction*, pages 445–452.

Finn, C., Abbeel, P., and Levine, S. (2017). Model-agnostic meta-learning for fast adaptation of deep networks. In *Proceedings of the 34th International Conference on Machine Learning-Volume 70*, pages 1126–1135. JMLR. org.

Geng, C., Tao, L., and Chen, S. (2020). Guided cnn for generalized zero-shot and open-set recognition using visual and semantic prototypes. *Pattern Recognition*, 102:107263.

Ghosal, D., Akhtar, M. S., Chauhan, D., Poria, S., Ekbal, A., and Bhattacharyya, P. (2018). Contextual inter-modal attention for multi-modal sentiment analysis. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pages 3454–3466.

Gjoreski, H. and Roggen, D. (2017). Unsupervised online activity discovery using temporal behaviour assumption. In *Proceedings of the 2017 ACM International Symposium on Wearable Computers*, pages 42–49. ACM.

Gomes, J. B., Krishnaswamy, S., Gaber, M. M., Sousa, P. A., and Menasalvas, E. (2012). Mars: a personalised mobile activity recognition system. In *2012 IEEE 13th International Conference on Mobile Data Management*, pages 316–319. IEEE.

Gómez, D. and Rojas, A. (2016). An empirical overview of the no free lunch theorem and its effect on real-world machine learning classification. *Neural computation*, 28(1):216–228.

Goodfellow, I., Bengio, Y., and Courville, A. (2016). *Deep learning*. MIT press.

Gu, Y., Yang, K., Fu, S., Chen, S., Li, X., and Marsic, I. (2018). Hybrid attention based multimodal network for spoken language classification. In *Proceedings of the conference. Association for Computational Linguistics. Meeting*, volume 2018, page 2379. NIH Public Access.

Gui, L.-Y., Wang, Y.-X., Ramanan, D., and Moura, J. M. (2018). Few-shot human motion prediction via meta-learning. In *Proceedings of the European Conference on Computer Vision (ECCV)*, pages 432–450.

Guo, M., Wang, Z., and Yang, N. (2018). Aerobic exercise recognition through sparse representation over learned dictionary by using wearable inertial sensors. *Journal of Medical and Biological Engineering*, 38(4):544–555.

He, Z. and Jin, L. (2009). Activity recognition from acceleration data based on discrete consine transform and svm. In *2009 IEEE International Conference on Systems, Man and Cybernetics*, pages 5041–5044. IEEE.

Hochreiter, S. and Schmidhuber, J. (1997). Long short-term memory. *Neural computation*, 9(8):1735–1780.

Hoffer, E. and Ailon, N. (2015). Deep metric learning using triplet network. In *International Workshop on Similarity-Based Pattern Recognition*, pages 84–92. Springer.

Hori, C., Hori, T., Wichern, G., Wang, J., Lee, T.-y., Cherian, A., and Marks, T. K. (2018). Multimodal attention for fusion of audio and spatiotemporal features for video description. In *CVPR Workshops*, pages 2528–2531.

Hospedales, T., Antoniou, A., Micaelli, P., and Storkey, A. (2020). Meta-learning in neural networks: A survey. *arXiv preprint arXiv:2004.05439*.

Inoue, M., Inoue, S., and Nishida, T. (2018). Deep recurrent neural network for mobile human activity recognition with high throughput. *Artificial Life and Robotics*, 23(2):173–185.

Ioffe, S. and Szegedy, C. (2015). Batch normalization: Accelerating deep network training by reducing internal covariate shift. *arXiv preprint arXiv:1502.03167*.

James, S. L., Abate, D., Abate, K. H., Abay, S. M., Abbafati, C., Abbasi, N., Abbastabar, H., Abd-Allah, F., Abdela, J., Abdelalim, A., et al. (2018). Global, regional, and national incidence, prevalence, and years lived with disability for 354 diseases and injuries for 195 countries and territories, 1990–2017: a systematic analysis for the global burden of disease study 2017. *The Lancet*, 392(10159):1789–1858.

Jiang, M.-x., Deng, C., Shan, J.-s., Wang, Y.-y., Jia, Y.-j., and Sun, X. (2019). Hierarchical multi-modal fusion fcn with attention model for rgb-d tracking. *Information Fusion*, 50:1–8.

Kahou, S. E., Bouthillier, X., Lamblin, P., Gulcehre, C., Michalski, V., Konda, K., Jean, S., Froumenty, P., Dauphin, Y., Boulanger-Lewandowski, N., et al. (2016). Emonets: Multimodal deep learning approaches for emotion recognition in video. *Journal on Multimodal User Interfaces*, 10(2):99–111.

Keogh, E., Chu, S., Hart, D., and Pazzani, M. (2001). An online algorithm for segmenting time series. In *Proceedings 2001 IEEE international conference on data mining*, pages 289–296. IEEE.

Ketkar, N. (2017). Introduction to keras. In *Deep learning with Python*, pages 97–111. Springer.

Khaire, P., Kumar, P., and Imran, J. (2018). Combining cnn streams of rgb-d and skeletal data for human activity recognition. *Pattern Recognition Letters*, 115:107–116.

Khurana, R., Ahuja, K., Yu, Z., Mankoff, J., Harrison, C., and Goel, M. (2018). Gymcam: Detecting, recognizing and tracking simultaneous exercises in unconstrained scenes. *Proceedings of the ACM on Interactive, Mobile, Wearable and Ubiquitous Technologies*, 2(4):1–17.

Koskimäki, H. and Siirtola, P. (2014). Recognizing gym exercises using acceleration data from wearable sensors. In *2014 IEEE Symposium on Computational Intelligence and Data Mining (CIDM)*, pages 321–328. IEEE.

Krizhevsky, A., Sutskever, I., and Hinton, G. E. (2012). Imagenet classification with deep convolutional neural networks. In *Advances in neural information processing systems*, pages 1097–1105.

Lampert, C. H., Nickisch, H., and Harmeling, S. (2014). Attribute-based classification for zero-shot visual object categorization. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 36(3):453–465.

Lara, O. D. and Labrador, M. A. (2012). A survey on human activity recognition using wearable sensors. *IEEE communications surveys & tutorials*, 15(3):1192–1209.

Lawrence, S., Giles, C. L., Tsoi, A. C., and Back, A. D. (1997). Face recognition: A convolutional neural-network approach. *IEEE transactions on neural networks*, 8(1):98–113.

Lin, H.-C., Chiang, S.-Y., Lee, K., and Kan, Y.-C. (2015). An activity recognition model using inertial sensor nodes in a wireless sensor network for frozen shoulder rehabilitation exercises. *Sensors*, 15(1):2181–2204.

Liu, J., Kuipers, B., and Savarese, S. (2011). Recognizing human actions by attributes. In *CVPR 2011*, pages 3337–3344. IEEE.

Longstaff, B., Reddy, S., and Estrin, D. (2010). Improving activity classification for health applications on mobile devices using active and semi-supervised learning. In *2010 4th International Conference on Pervasive Computing Technologies for Healthcare*, pages 1–7. IEEE.

Losing, V., Yoshikawa, T., Hasenjaeger, M., Hammer, B., and Wersing, H. (2019). Personalized online learning of whole-body motion classes using multiple inertial measurement units. In *2019 International Conference on Robotics and Automation (ICRA)*, pages 9530–9536. IEEE.

Luong, M.-T., Pham, H., and Manning, C. D. (2015). Effective approaches to attention-based neural machine translation. *arXiv preprint arXiv:1508.04025*.

Ma, H., Li, W., Zhang, X., Gao, S., and Lu, S. (2019). Attnsense: Multi-level attention mechanism for multimodal human activity recognition. In *IJCAI*, pages 3109–3115.

Madotto, A., Lin, Z., Wu, C.-S., and Fung, P. (2019). Personalizing dialogue agents via meta-learning.

In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 5454–5459.

McMahan, B., Moore, E., Ramage, D., Hampson, S., and y Arcas, B. A. (2017). Communication-efficient learning of deep networks from decentralized data. In *Artificial Intelligence and Statistics*, pages 1273–1282. PMLR.

Mendiola, V., Doss, A., Adams, W., Ramos, J., Bruns, M., Cherian, J., Kohli, P., Goldberg, D., and Hammond, T. (2019). Automatic exercise recognition with machine learning. In *International Workshop on Health Intelligence*, pages 33–44. Springer.

Mirza, M. and Osindero, S. (2014). Conditional generative adversarial nets. *arXiv preprint arXiv:1411.1784*.

Mishra, N., Rohaninejad, M., Chen, X., and Abbeel, P. (2018). A simple neural attentive meta-learner. In *International Conference on Learning Representations*.

Miu, T., Missier, P., and Plötz, T. (2015). Bootstrapping personalised human activity recognition models using online active learning. In *2015 IEEE International Conference on Computer and Information Technology; Ubiquitous Computing and Communications; Dependable, Autonomic and Secure Computing; Pervasive Intelligence and Computing*, pages 1138–1147. IEEE.

Morris, D., Saponas, T. S., Guillory, A., and Kelner, I. (2014). Recofit: using a wearable sensor to find, recognize, and count repetitive exercises. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*, pages 3225–3234.

Münzner, S., Schmidt, P., Reiss, A., Hanselmann, M., Stiefelhagen, R., and Dürichen, R. (2017). Cnn-based sensor fusion techniques for multimodal human activity recognition. In *Proceedings of the 2017 ACM International Symposium on Wearable Computers*, pages 158–165.

Nanni, L., Ghidoni, S., and Brahnam, S. (2017). Handcrafted vs. non-handcrafted features for computer vision classification. *Pattern Recognition*, 71:158–172.

Nardi, P. (2019). Human activity recognition: Deep learning techniques for an upper body exercise classification system.

Ngiam, J., Khosla, A., Kim, M., Nam, J., Lee, H., and Ng, A. Y. (2011). Multimodal deep learning. In *ICML*.

Nguyen, L. N. N., Rodríguez-Martín, D., Català, A., Pérez-López, C., Samà, A., and Cavallaro, A. (2015a). Basketball activity recognition using wearable inertial measurement units. In *Proceedings of the XVI international conference on Human Computer Interaction*, page 60. ACM.

Nguyen, L. T., Zeng, M., Tague, P., and Zhang, J. (2015b). Recognizing new activities with limited training data. In *Proceedings of the 2015 ACM International Symposium on Wearable Computers*, pages 67–74.

Nichol, A., Achiam, J., and Schulman, J. (2018). On first-order meta-learning algorithms. *arXiv preprint arXiv:1803.02999*.

Ogonowski, C., Aal, K., Vaziri, D., Rekowski, T. V., Randall, D., Schreiber, D., Wieching, R., and Wulf, V. (2016). Ict-based fall prevention system for older adults: qualitative results from a long-term field study. *ACM Transactions on Computer-Human Interaction (TOCHI)*, 23(5):1–33.

Ohashi, H., Al-Naser, M., Ahmed, S., Nakamura, K., Sato, T., and Dengel, A. (2018). Attributes' importance for zero-shot pose-classification based on wearable sensors. *Sensors*, 18(8):2485.

Ordóñez, F. and Roggen, D. (2016). Deep convolutional and lstm recurrent neural networks for multimodal wearable activity recognition. *Sensors*, 16(1):115.

Oreshkin, B., López, P. R., and Lacoste, A. (2018). Tadam: Task dependent adaptive metric for improved few-shot learning. In *Advances in Neural Information Processing Systems*, pages 721–731.

Parmar, P. and Morris, B. T. (2016). Measuring the quality of exercises. In *2016 38th Annual International Conference of the IEEE Engineering in Medicine and Biology Society (EMBC)*, pages 2241–2244. IEEE.

Paszke, A., Gross, S., Massa, F., Lerer, A., Bradbury, J., Chanan, G., Killeen, T., Lin, Z., Gimelshein, N., Antiga, L., et al. (2019). Pytorch: An imperative style, high-performance deep learning library. In *Advances in neural information processing systems*, pages 8026–8037.

Pernek, I., Kurillo, G., Stiglic, G., and Bajcsy, R. (2015). Recognizing the intensity of strength training exercises with wearable sensors. *Journal of biomedical informatics*, 58:145–155.

Qi, J., Yang, P., Hanneghan, M., Waraich, A., and Tang, S. (2018). A hybrid hierarchical framework for free weight exercise recognition and intensity measurement with accelerometer and ecg data fusion. In *2018 40th Annual Int. Conf. of the IEEE Engineering in Medicine and Biology Society (EMBC)*, pages 3800–3804. IEEE.

Radu, V., Lane, N. D., Bhattacharya, S., Mascolo, C., Marina, M. K., and Kawsar, F. (2016). Towards multimodal deep learning for activity recognition on mobile devices. In *Proceedings of the 2016 ACM International Joint Conference on Pervasive and Ubiquitous Computing: Adjunct*, pages 185–188.

Radu, V., Tong, C., Bhattacharya, S., Lane, N. D., Mascolo, C., Marina, M. K., and Kawsar, F. (2018). Multimodal deep learning for activity and context recognition. *Proceedings of the ACM on Interactive, Mobile, Wearable and Ubiquitous Technologies*, 1(4):1–27.

Rahmani, H. and Bennamoun, M. (2017). Learning action recognition model from depth and skeleton videos. In *Proceedings of the IEEE International Conference on Computer Vision*, pages 5832–5841.

Ramya, H. and Bhatt, M. R. (2019). Personalised emotion recognition utilising speech signal and linguistic cues. In *2019 11th International Conference on Communication Systems & Networks (COMSNETS)*, pages 856–860. IEEE.

Ravi, N., Dandekar, N., Mysore, P., and Littman, M. L. (2005). Activity recognition from accelerometer data. In *Aaai*, volume 5, pages 1541–1546.

Ravi, S. and Larochelle, H. (2017). Optimization as a model for few-shot learning. In *International Conference on Learning Representations.*

Reiss, A. and Stricker, D. (2012). Introducing a new benchmarked dataset for activity monitoring. In *2012 16th International Symposium on Wearable Computers*, pages 108–109. IEEE.

Reiss, A. and Stricker, D. (2013). Personalized mobile physical activity recognition. In *Proceedings of the 2013 international symposium on wearable computers*, pages 25–28.

Rybarczyk, Y., Deters, J. K., Gonzalo, A. A., Esparza, D., Gonzalez, M., Villarreal, S., and Nunes, I. L. (2017). Recognition of physiotherapeutic exercises through dtw and low-cost vision-based motion capture. In *International Conference on Applied Human Factors and Ergonomics*, pages 348–360. Springer.

Sani, S., Massie, S., Wiratunga, N., and Cooper, K. (2017a). Learning deep and shallow features for human activity recognition. In *International Conference on Knowledge Science, Engineering and Management*, pages 469–482. Springer.

Sani, S., Wiratunga, N., Massie, S., and Cooper, K. (2016). Selfback—activity recognition for self-management of low back pain. In *International Conference on Innovative Techniques and Applications of Artificial Intelligence*, pages 281–294. Springer.

Sani, S., Wiratunga, N., Massie, S., and Cooper, K. (2017b). knn sampling for personalised human activity recognition. In *International conference on case-based reasoning*, pages 330–344. Springer.

Sani, S., Wiratunga, N., Massie, S., and Cooper, K. (2018). Personalised human activity recognition using matching networks. In *International Conference on Case-Based Reasoning*, pages 339–353. Springer.

Santoro, A., Bartunov, S., Botvinick, M., Wierstra, D., and Lillicrap, T. (2016). Meta-learning with memory-augmented neural networks. In *International conference on machine learning*, pages 1842–1850.

Sariyildiz, M. B. and Cinbis, R. G. (2019). Gradient matching generative networks for zero-shot learning. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 2168–2178.

Satorras, V. G. and Estrach, J. B. (2018). Few-shot learning with graph neural networks. In *International Conference on Learning Representations.*

Snell, J., Swersky, K., and Zemel, R. (2017). Prototypical networks for few-shot learning. In *Advances in neural information processing systems*, pages 4077–4087.

Su, X., Tong, H., and Ji, P. (2014). Activity recognition with smartphone sensors. *Tsinghua science and technology*, 19(3):235–249.

Sun, X., Kashima, H., and Ueda, N. (2012). Large-scale personalized human activity recognition using online multitask learning. *IEEE Transactions on Knowledge and Data Engineering*, 25(11):2551–2563.

Sundholm, M., Cheng, J., Zhou, B., Sethi, A., and Lukowicz, P. (2014). Smart-mat: Recognizing and counting gym exercises with low-cost resistive pressure sensing matrix. In *Proceedings of the 2014 ACM UbiComp.*, pages 373–382. ACM.

Sung, F., Yang, Y., Zhang, L., Xiang, T., Torr, P. H., and Hospedales, T. M. (2018). Learning to compare: Relation network for few-shot learning. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 1199–1208.

Tapia, E. M., Intille, S. S., Haskell, W., Larson, K., Wright, J., King, A., and Friedman, R. (2007). Real-time recognition of physical activities and their intensities using wireless accelerometers and a heart rate monitor. In *2007 11th IEEE international symposium on wearable computers*, pages 37–40. IEEE.

Thiam, P., Kessler, V., and Schwenker, F. (2017). Hierarchical combination of video features for personalised pain level recognition. In *ESANN*.

Thomas, S., Reading, J., and Shephard, R. J. (1992). Revision of the physical activity readiness questionnaire (par-q). *Canadian journal of sport sciences*.

Triantafyllidis, A., Filos, D., Buys, R., Claes, J., Cornelissen, V., Kouidi, E., Chatzitofis, A., Zarpalas, D., Daras, P., Chouvarda, I., et al. (2018). A computer-assisted system with kinect sensors and wristband heart rate monitors for group classes of exercise-based rehabilitation. In *Precision Medicine Powered by pHealth and Connected Health*, pages 237–241. Springer.

Vakanski, A., Jun, H.-p., Paul, D., and Baker, R. (2018). A data set of human body movements for physical rehabilitation exercises. *Data*, 3(1):2.

Vanschoren, J. (2019). Meta-learning. In *Automated Machine Learning*, pages 35–61. Springer.

Vaswani, A., Shazeer, N., Parmar, N., Uszkoreit, J., Jones, L., Gomez, A. N., Kaiser, Ł., and Polosukhin, I. (2017). Attention is all you need. In *Advances in neural information processing systems*, pages 5998–6008.

Velloso, E., Bulling, A., Gellersen, H., Ugulino, W., and Fuks, H. (2013). Qualitative activity recognition of weight lifting exercises. In *Proc. 4th Augmented Human Int. Conf.*, pages 116–123. ACM.

Verma, V. K., Brahma, D., and Rai, P. (2020). Meta-learning for generalized zero-shot learning. In *AAAI*, pages 6062–6069.

Vinyals, O., Blundell, C., Lillicrap, T., Wierstra, D., et al. (2016). Matching networks for one shot learning. In *Advances in neural information processing systems*, pages 3630–3638.

Vox, J. P. and Wallhoff, F. (2017). Recognition of human motion exercises using skeleton data and svm for rehabilitative purposes. In *2017 IEEE Life Sciences Conference (LSC)*, pages 266–269. IEEE.

Wahl, F. and Amft, O. (2014). Personalised phone placement recognition in daily life using rfid tagging. In *2014 IEEE International Conference on Pervasive Computing and Communication Workshops (PERCOM WORKSHOPS)*, pages 19–26. IEEE.

Xian, Y., Lampert, C. H., Schiele, B., and Akata, Z. (2018). Zero-shot learning—a comprehensive evaluation of the good, the bad and the ugly. *IEEE transactions on pattern analysis and machine intelligence*, 41(9):2251–2265.

Xiao, F., Chen, J., Xie, X. H., Gui, L., Sun, J. L., and none Ruchuan, W. (2018). Seare: A system for exercise activity recognition and quality evaluation based on green sensing. *IEEE Transactions on Emerging Topics in Computing.*

Xu, J., Yao, T., Zhang, Y., and Mei, T. (2017a). Learning multimodal attention lstm networks for video captioning. In *Proceedings of the 25th ACM international conference on Multimedia*, pages 537–545.

Xu, X., Hospedales, T., and Gong, S. (2017b). Transductive zero-shot action recognition by word-vector embedding. *International Journal of Computer Vision*, 123(3):309–333.

Xue, H., Jiang, W., Miao, C., Yuan, Y., Ma, F., Ma, X., Wang, Y., Yao, S., Xu, W., Zhang, A., et al. (2019). Deepfusion: A deep learning framework for the fusion of heterogeneous sensory data. In *Proceedings of the Twentieth ACM International Symposium on Mobile Ad Hoc Networking and Computing*, pages 151–160.

Yang, J., Nguyen, M. N., San, P. P., Li, X. L., and Krishnaswamy, S. (2015). Deep convolutional neural networks on multichannel time series for human activity recognition. In *Twenty-Fourth International Joint Conference on Artificial Intelligence.*

Yao, S., Hu, S., Zhao, Y., Zhang, A., and Abdelzaher, T. (2017). Deepsense: A unified deep learning framework for time-series mobile sensing data processing. In *Proceedings of the 26th International Conference on World Wide Web*, pages 351–360.

Yao, S., Zhao, Y., Hu, S., and Abdelzaher, T. (2018). Qualitydeepsense: Quality-aware deep learning framework for internet of things applications with sensor-temporal attention. In *Proceedings of the 2nd International Workshop on Embedded and Mobile Deep Learning*, pages 42–47.

Young, A. D., Ling, M. J., and Arvind, D. (2010). Distributed estimation of linear acceleration for improved accuracy in wireless inertial motion capture. In *Proceedings of the 9th ACM/IEEE international conference on information processing in sensor networks*, pages 256–267.

Young, A. D., Ling, M. J., and Arvind, D. K. (2011). Imusim: A simulation environment for inertial sensing algorithm design and evaluation. In *Proceedings of the 10th ACM/IEEE International Conference on Information Processing in Sensor Networks*, pages 199–210. IEEE.

Zang, C., Pei, M., and Kong, Y. (2020). Few-shot human motion prediction via learning novel motion dynamics. In *Proceedings of the 29th International Joint Conference on Artificial Intelligence.* AAAI Press.

Zhang, W., Wang, W., Wang, J., and Zha, H. (2018). User-guided hierarchical attention network for multi-modal social image popularity prediction. In *Proceedings of the 2018 World Wide Web Conference*, pages 1277–1286.

Zhou, B., Sundholm, M., Cheng, J., Cruz, H., and Lukowicz, P. (2016). Never skip leg day: A novel wearable approach to monitoring gym leg exercises. In *2016 IEEE International Conference on Pervasive Computing and Communications (PerCom)*, pages 1–9. IEEE.

Zhu, Z.-A., Lu, Y.-C., You, C.-H., and Chiang, C.-K. (2019). Deep learning for sensor-based rehabilitation exercise recognition and evaluation. *Sensors*, 19(4):887.

Appendix A

Dissemination

Journal Publications:

- Wijekoon, A., Wiratunga, N., Sani, S., & Cooper, K. (2020). A knowledge-light approach to personalised and open-ended human activity recognition. Knowledge-Based Systems, 192, 105651.

Conference Publications:

- Wijekoon, A., Wiratunga, N., Cooper, K., & Bach, K. (2020, May). Learning to Recognise Exercises in the Self-Management of Low Back Pain. In The Thirty-Third International Flairs Conference (FLAIRS). AAAI Press.
- Wijekoon, A., & Wiratunga, N. (2020, June). Evaluating the Transferability of Personalised Exercise Recognition Models. In International Conference on Engineering Applications of Neural Networks. Springer, Cham.
- Wijekoon, A., Wiratunga, N., Cooper, K. (2020, July). Heterogeneous Multi-Modal Sensor Fusion with Hybrid Attention for Exercise Recognition. In International Joint Conference on Neural Networks (IJCNN). IEEE.
- Wijekoon, A., & Wiratunga, N. (2020, December). Personalised Meta-Learning for Human Activity Recognition with Few-data. In International Conference on Innovative Techniques and Applications of Artificial Intelligence. Springer, Cham.

Workshops Publication:

- Wijekoon, A., Wiratunga, N., & Sani, S. (2018, July). Zero-shot learning with matching networks for open-ended human activity recognition. CEUR Workshop Proceedings.

MEx Exercise Recognition Dataset:

- UCI Machine Learning Repo: https://archive.ics.uci.edu/ml/datasets/MEx

Code: https://github.com/anjanaw

Appendix B

MEx Dataset

In this appendix, we present the empirical study for pre-processing MEx modalities and a detailed overview of the selected physiotherapy exercises.

B.1 Pre-processing MEx Modalities

This empirical evaluation is two-fold; firstly, we explore different window and overlap values to find the most optimal sliding window parameters for exercise recognition; secondly, we explore frame selection, frame rate and frame compression parameters to find the most optimal pre-processing steps for MEx visual modalities PM and DC.

B.1.1 Window and Overlap

An empirical study was conducted to understand how the classification performance vary with different window and overlap settings. We create a set of experiments with the PM modality using the 1D-CNN-LSTM architecture from Section 4.3. 3, 5 and 8 seconds were considered as window sizes, and a number of overlap values were considered for each window size. Experiments were performed using the LOPO evaluation methodology, and mean F1-measure is presented in Table B.1. Best performing overlap value for each window setting is highlighted in bold.

Overall we observe that classification performance improves with the window size, which is intuitive given that with larger window size, the data instance carry more information to recognise the exercise. We observe no strict pattern with the overlap where two window

Table B.1: Window and overlap with PM data

Classifier	Window	Overlap (s)					
	Size (s)	6	4	3	2	1	0
1D-CNN-LSTM	3	-	-	-	0.6884	0.6914	**0.7102**
	5	-	-	**0.7408**	0.7370	0.7388	0.7397
	8	**0.7557**	0.7537	-	0.7286	-	0.7431

sizes 5 and 8 performed best with highest overlap values 3 and 6 respectively, and window size 3 performed best with no overlap.

In practice, there are limitations to using a larger window size; firstly a larger window size results in a smaller number of training instances. A limited number of training instances may cause parametric models to over-fit or not optimise. Secondly, at test time, larger data instances may cause a delay in predictions; also, with a smaller overlap, the time elapsed between two predictions is higher, both of which are not desirable for real-time use. Therefore it is important to find a window and overlap size that are both practical in real-world applications while preserving performance. From this study, we select the window size 5 and overlap 3.

B.1.2 Frame Selection, Frame Rate and Compression

In this section we observe the impact on performance when frames are down-scaled to reduce frame sizes and frame rates. A set of experiments were designed to explore three frame sizes and three frame rates with following frame selection techniques.

- Average (AVG): A new frame is created by pixel-wise averaging all frames within the time period.
- Increment (INC): Increment timestamp by the time-period and select the nearest timestamp and respective frame.

We select the PM modality and the 1D-CNN-LSTM architecture from Section 4.3 for these experiments. LOPO evaluation methodology was followed, and the mean F1-measure is presented in Table B.2. Best performing frame size and frame selection method combinations for each frame rate are highlighted in bold.

Considering the frame size, we observe that the 1D-CNN-LSTM model achieves the best

Table B.2: Frame selection, frame rate and compression results for the PM modality

Frame Size	Frame Selection	Frames per second					
	Method	1	2	3	4	5	6
32 × 16	INC	0.7408	0.7413	0.7462	0.7259	**0.7530**	0.7093
	AVG	0.7401	0.7335	0.7370	0.7457	**0.7482**	0.7249
16 × 16	INC	0.7632	0.7753	0.7780	0.7423	0.7812	**0.7832**
	AVG	0.7442	0.7528	0.7926	0.7561	**0.7949**	0.7560
8 × 8	INC	0.7226	0.7116	0.7101	0.7210	**0.7369**	0.7075
	AVG	0.7076	0.7129	0.7157	0.7174	**0.7186**	0.7128

performance with 16 × 16. Down-scaling to frame size 16 × 16 has improves the performance significantly compared to the original frame size at every frame rate. However, once the frame size is reduced to size 8 × 8, the performance is heavily penalised even with higher frame rate due to the loss of spatial information.

It is observed that increased frame rate does not necessarily contribute towards improved performance since all frame sizes achieve the best performance with frame rate 5, but degrades performance at frame rate 6 (with the exception at frame size 2 and frame selection method INC). Overall, INC and AVG frame selection methods yielded similar performances across all three frame sizes and all frame rates. Based on the above results, we select 16 × 16 frame size and 5 frames per second using the INC frame selection method for the PM modality.

Table B.3: Frame rate and compression results for the DC modality

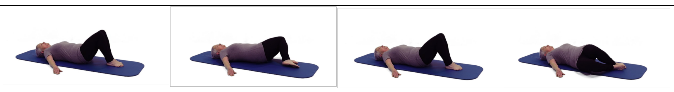
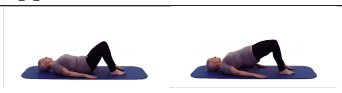
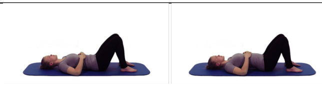
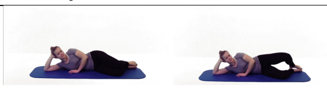
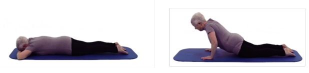
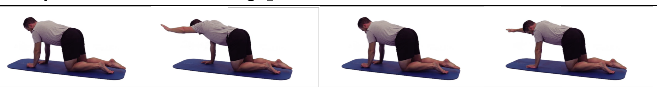
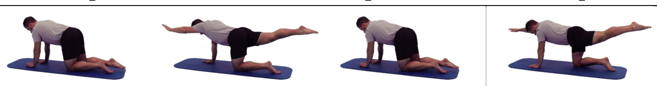
Frame Size	Frame Selection	Frames per second					
	Method	1	2	3	4	5	8
32 × 24	INC	0.8746	**0.8991**	0.8778	0.8655	0.8713	0.8374
16 × 12	INC	0.8720	**0.8929**	0.8837	0.8764	0.8853	0.8450

We extend the empirical study of frame rates and frame sizes with the DC modality using the *INC* frame selection method. We select two frame sizes by downscaling the original frame size of the DC modality which was 320 × 240. We are using the 1D-CNN-LSTM architecture from Section 4.3 for these experiments. LOPO evaluation methodology was followed, and the mean F1-measure is presented in Table B.3. We observe similar performances with both frame sizes. The best performance for each frame size is achieved at 2 frames per second, but manage to maintain performance up to 5 frames per second. Performance is dropped significantly at 8 frames per second.

The important take away from this study is that frame size and frame rate can be selected such that they complement each other. And importantly, selecting a higher frame rate and a larger frame size does not naturally improve performance. Lazy feature augmentation techniques such as frame down-scaling and frame selection can be pivotal to achieving higher performance with reduced memory and computational power.

B.2 MEx Exercises

Table B.4: MEx exercises

Knee Rolling Ab	 Lying on back, knees together and bent, feet flat on floor. Slowly roll knees to the right, back to the centre, then to the left keeping upper trunk still
Bridging Glut	 Lying on back with knees bent and slightly apart, feet flat on floor and arms by side. Squeeze buttock muscles and lift hips off floor. Hold approximately 5-seconds and lower slowly.
Pelvic tilt Pain relief	 Lying on back with knees bent and slightly apart, feet flat on floor and arms by side. Tighten stomach muscles and press small of back against the floor letting your bottom rise. Hold approximately 5 seconds then relax.
Bilateral Clam Glut	 Lying on right side with hips and shoulders in straight line. Bend knees so thighs are at 90 degrees angle. Rest head on top arm (stretched overhead or bent depending on comfort). Bend top arm and place hand on floor for stability. Stack hips directly on top of each other (same for shoulders). Keep big toes together and slowly rotate leg in hip socket so the top knee opens. Open knee as far as you can without disturbing alignment of hips. Slowly return to starting position
Extension in Lying Flexibility	 Lying face down, place palms on floor and elbows under shoulders (press-up position). Straighten elbows as far as you can and push top half of body up as far as you can. Pelvis, hips and legs must stay relaxed. Maintain position for approximately 2-seconds then slowly lower to starting position.
Prone punches Core	 On all 4's with hands directly beneath shoulders, knees slightly apart and straight back. Punch right arm in front and lower to floor. Repeat with left arm. Keep trunk as still as possible
Superman Back	 On all 4's with hands directly beneath shoulders, knees slightly apart and straight back. Extend right arm straight in front of you and left leg straight behind you, keeping trunk as still as possible. Hold approximately 5-seconds then lower and repeat with other arm and leg.

Appendix C

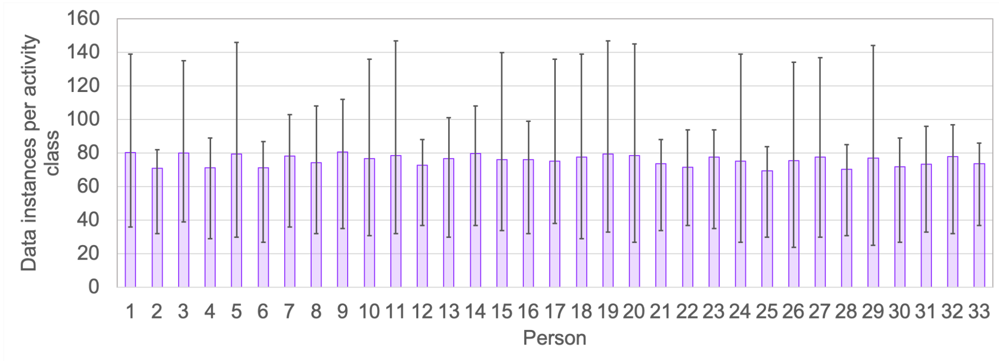
HAR Datasets

In this appendix, we present an extended view on data distribution properties of HAR datasets selfBACK, PAMAP2 and HDPoseDS. We plot the data distribution by person and by activity class to observe the characteristics of each dataset. The error bars on by person plot indicates the maximum, and the minimum number of instances for a class and the error bars on by class plot indicate the maximum and the minimum number of instances from a person.

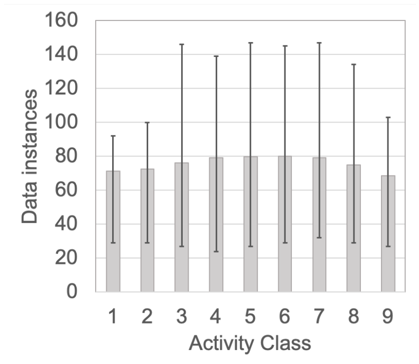
selfBACK Dataset: Figure C.1a plots the number of data instances for each person of the selfBACK dataset. On average, a person contributes 75.7 ± 3.25 data instances per class, but there is a significant imbalance between the number of data instances per class. Figure C.1b plots the data distribution by exercise class. While the data instance distribution across exercises classes is approximately balanced (75 ± 4.21), there exist significant outliers for personal contributions. It is noteworthy that compared to MEx, selfBACK is comparably a larger dataset.

PAMAP2 Dataset: Figure C.2a plots the number of data instances for each person on the PAMAP2 dataset. On average, a person contributes 88.4 ± 2.8 data instances per class, but there is a significant imbalance between the number of data instances per class. Figure C.2b plots the data distribution by exercise class. There is a significant class imbalance (88.4 ± 24.35) in addition to the significant outliers for personal contributions. Similar to selfBACK, PAMAP2 is also a large dataset.

HDPoseDS Dataset Figure C.3a plots the number of data instances for each person on the HDPoseDS dataset. On average, a person contributes 27.05 ± 2.3 data instances per class. There is a significant imbalance between the number of data instances per class for person 6. Figure C.3b plots the data distribution by exercise class. On average, a pose class has 27.05 ± 1.42) data instance with significant outliers for only pose class 22 (Thinking). In numbers, this dataset closely resembles a few-shot dataset.

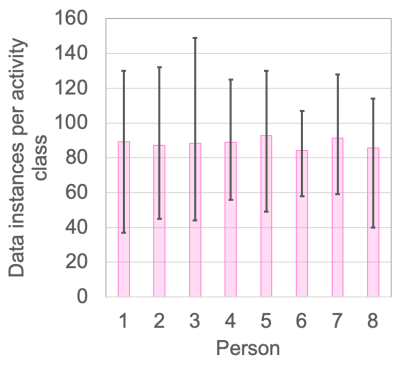


(a) By person

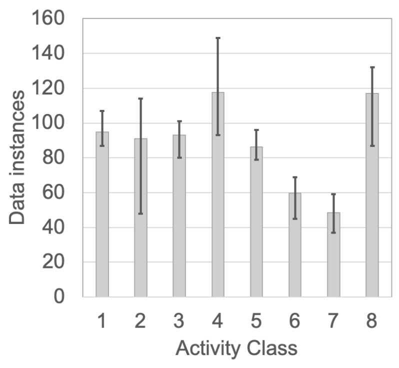


(b) By activity class

Figure C.1: SELFBACK data distribution

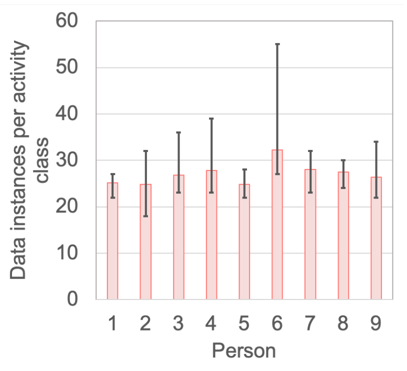


(a) By person

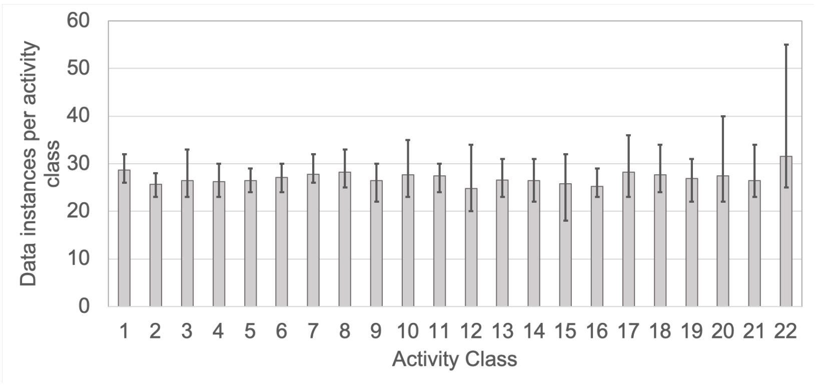


(b) By activity class

Figure C.2: PAMAP2 data distribution



(a) By person



(b) By pose class

Figure C.3: HDPoseDS data distribution