Yves Luduvico Coelho

Energy-Efficient Human Activity Recognition Framework for Wearable Devices: Development, Deployment, and Analysis

Vitoria

2022

Yves Luduvico Coelho

Energy-Efficient Human Activity Recognition Framework for Wearable Devices: Development, Deployment, and Analysis

Doctoral thesis presented to the Postgraduate Program in Electrical Engineering (PPGEE), Federal University of Espirito Santo (UFES), in partial fulfillment of the requirements for the degree of Doctor in Electrical Engineering

Federal University of Espirito Santo

Postgraduate Program in Electrical Engineering

Supervisor: Prof. Teodiano Freire Bastos Filho, PhD (UFES, Brazil)

Co-supervisors: Prof. Anselmo Frizera Neto, PhD (UFES, Brazil) and Prof. Sridhar Krishnan, PhD (Toronto Metropolitan University, Canada)

Vitoria

2022

Ficha catalográfica disponibilizada pelo Sistema Integrado de
Bibliotecas - SIBI/UFES e elaborada pelo autor

Coelho, Yves Luduvico, 1989-
C672e Energy-efficient human activity recognition framework for wearable devices : Development, Deployment, and Analysis / Yves Luduvico Coelho. - 2022.
169 f. : il.

Orientador: Teodiano Freire Bastos Filho.
Coorientadores: Anselmo Frizera Neto, Sridhar Krishnan.
Tese (Doutorado em Engenharia Elétrica) - Universidade Federal do Espírito Santo, Centro Tecnológico.

1. Inteligência artificial. 2. Redes Neurais (Computação). 3. Sistemas de computação. 4. Sistemas embarcados (Computadores). 5. Sensoriamento remoto. I. Bastos Filho, Teodiano Freire. II. Frizera Neto, Anselmo. III. Krishnan, Sridhar. IV. Universidade Federal do Espírito Santo. Centro Tecnológico. V. Título.

CDU: 621.3

Yves Luduvico Coelho

Energy-Efficient Human Activity Recognition Framework for Wearable Devices: Development, Deployment, and Analysis

Doctoral thesis presented to the Postgraduate Program in Electrical Engineering (PPGEE), Federal University of Espirito Santo (UFES), in partial fulfillment of the requirements for the degree of Doctor in Electrical Engineering

Approved, August 17th, 2022.

[illegible]

Prof. Teodiano Bastos Filho, PhD
Supervisor - UFES, Brazil

[illegible]

Prof. Anselmo Frizera Neto, PhD
Co-Supervisor - UFES, Brazil

[illegible]

Prof. Sridhar Krishnan, PhD
Co-Supervisor - TMU, Canada

Documento assinado digitalmente
gov.br CARLOS EDUARDO PEREIRA
Data: 21/11/2022 13:09:31-0300
Verifique em https://verificador.iti.br

Prof. Carlos Eduardo Pereira, PhD
External examiner - UFRGS, Brazil

[illegible]

Prof. Denis Delisle Rodriguez, PhD
External examiner - IIN-ELS, Brazil

[illegible]

Prof. Patrick Marques Ciarelli, PhD
Examiner - UFES, Brazil

[illegible]

Prof. Eliete Maria Caldeira, PhD
Examiner - UFES, Brazil

Vitoria
2022

Acknowledgements

First of all, I would like to thank my parents for all the support they have always been, and for their enormous contribution to the path I have been building in life. I would also like to thank my beloved companion, Meire, for her encouragement, and for always being there in difficult times.

I wish to express my sincere gratitude to my supervisor Prof. Dr. Teodiano Freire Bastos Filho for the opportunity to work with him on my master's and doctoral studies. His guidance and teachings contributed enormously to achieving the objectives of this research. To my co-supervisor Prof. Dr. Anselmo Frizera Neto, I thank him for all the recommendations, teachings, and inspiration that helped me to complete this work. I also thank my co-supervisor Prof. Dr. Sridhar Krishnan for his support and guidance during my stay at Toronto Metropolitan University, and for his contribution to this research.

I would like to thank all my colleagues at NTA/UFES, with whom I had the opportunity to live, learn and grow together during my master's and doctoral studies. To my dear colleagues at Kunumi, with whom I have shared my days for the last few months, thank you for always creating a great atmosphere that helped me deal with challenging times. I am grateful to all my friends, old and new, who have been around and supported me whenever I needed it.

I thankfully acknowledge the professors from the Postgraduate Program in Electrical Engineering for the knowledge and experience shared along these years, and the Federal University of Espirito Santo for the structure of laboratories and equipment, which were essential for the realization of this research. I extend my thanks to all the teachers and professors I have had during my educational journey who have contributed to my career.

Finally, I wish to acknowledge and thank CNPq (Conselho Nacional de Desenvolvimento Científico e Tecnológico, Brazil) for the scholarship received during my doctoral studies to support this research, and ELAP (Emerging Leaders in the Americas Program, Canada) for the scholarship during my stay in Canada.

I am very grateful to all of you.

Abstract

The rapid population aging has significant implications for society, especially in healthcare. At the same time, technological advancement and the spread of artificial intelligence provide new possibilities to meet these challenges and improve health and well-being, preventing a future collapse of the healthcare system. Wearable devices such as smart bands and smartwatches, already widely adopted today to monitor physical activities, will be important allies to promote better healthcare services. One area of research that combines these devices with machine learning is Human Activity Recognition (HAR), which has been widely discussed in the literature. The accelerated progress of deep learning techniques has also contributed to the development of models that surpassed the classification performance of the state-of-the-art until then. However, such models are highly complex to be deployed on resource-constrained wearable devices, which demand applications with low power consumption, real-time response, and privacy. Edge computing along with lighter and more efficient HAR systems is a viable solution to meeting these requirements. Given this scenario, this doctoral research aims to develop, implement and test an energy-efficient HAR framework for classifying human activities. The developed models were extensively analyzed in different configurations, and compared with reference and state-of-the-art models. In addition, the proposed framework was deployed and tested on a microcontroller, providing important analysis regarding computational performance. With the MHEALTH dataset, using only data sampled at 10 Hz from a wrist-worn sensor, the proposed framework achieved accuracy of 79.07%, surpassing reference classifiers, with a model of 44.58 kB, which consumes only 1.59 mW. In the multi-sensors scenario, the framework obtained an accuracy of 91.02%, close to the accuracy level of state-of-the-art complex deep learning-based models. The results achieved show it is possible to develop a model more efficient and simple enough to be embedded in wearable devices. Additionally, the findings also show that HAR systems can be simplified by reducing the sampling rate and discarding irrelevant data. Finally, the conclusions highlight the key contributions, the research limitations, and provide recommendations for further research.

Key-words: deep learning; edge computing; human activity recognition; machine learning; smartwatches; wearable devices.

Resumo

O envelhecimento populacional traz consigo desafios importantes para a sociedade, principalmente na área de saúde. Ao mesmo tempo, o avanço tecnológico e a disseminação da inteligência artificial oferecem novas possibilidades para atender as demandas para melhorar a saúde e o bem-estar, evitando um futuro colapso do sistema de saúde. Os dispositivos vestíveis, como *smart bands* e *smartwatches*, já amplamente adotados hoje para monitorar atividades físicas, serão importantes aliados para promover melhores serviços de cuidado à saúde. Uma área de pesquisa que combina esses dispositivos com aprendizado de máquina é o Reconhecimento de Atividades Humanas (HAR, do inglês *Human Activity Recognition*), que vem sendo amplamente discutido na literatura. O rápido progresso do aprendizado profundo contribuiu para o desenvolvimento de modelos que têm superado o desempenho de classificação do estado da arte até então. No entanto, tais modelos são altamente complexos para serem implantados em dispositivos vestíveis, os quais demandam aplicações com baixo consumo de energia, resposta em tempo real, e privacidade. A computação de borda em conjunto com sistemas HAR mais leves e mais eficientes é uma solução viável para atender esses requisitos. Diante desse cenário, o presente trabalho tem como objetivo desenvolver, implantar e testar um *framework* eficiente em termos de consumo de energia para classificação de atividades humanas. Os modelos desenvolvidos foram amplamente analisados em diversas configurações, e comparados com modelos de referência e do estado-da-arte. Além disso, o *framework* proposto foi implantado e testado em um microcontrolador, fornecendo análises importantes a respeito do desempenho computacional. Os resultados alcançados mostram que é possível desenvolver um modelo mais eficiente e simples o suficiente para ser embarcado em dispositivos vestíveis. Adicionalmente, os achados desta pesquisa também permitem observar que sistemas HAR podem ser simplificados com redução de taxa de amostragem e descarte de dados irrelevantes. Na base de dados MHEALTH, utilizando apenas dados de um sensor de punho coletados a 10 Hz, o *framework* proposto obteve acurácia de 79,07%, superando classificadores de referência, com um modelo de 44,58 kB, que consome apenas 1,59 mW. No cenário com múltiplos sensores, o *framework* obteve acurácia de 91,02%, situando-se próximo aos modelos complexos baseados em aprendizado profundo (do inglês *deep learning*) do estado-da-arte. Finalmente, as conclusões destacam os principais achados desta pesquisa, as limitações do trabalho, e oferece recomendações para continuação da pesquisa.

Palavras-chave: aprendizado de máquina; aprendizado profundo; computação de borda; dispositivos vestíveis; reconhecimento de atividades humanas; smartwatches.

List of Figures

List of Tables

List of abbreviations and acronyms

ADL	Activities of Daily Living
ANN	Artificial Neural Network
Bi-LSTM	Bidirectional LSTM
BNN	Binarized Neural Network
CART	Classification and Regression Tree
CNN	Convolutional Neural Network
DFT	Discrete Fourier Transform
DNN	Deep Neural Network
ECG	Electrocardiography
EFB	Exclusive Feature Bundling
EMG	Electromyography
FC	Fully-connected
FFT	Fast Fourier Transform
FOG	Fiber Optic Gyroscopes
GA	Genetic Algorithm
GRU	Gated Recurrent Unit
GBDT	Gradient Boosting Decision Tree
GOSS	Gradient-Based One-Side Sampling
GPU	Graphic Processing Unit
HAR	Human Activity Recognition
ICA	Independent Component Analysis
IDE	Integrated Development Environment
IMU	Inertial Measurement Unit

IoT	Internet of Things
IoMT	Internet of Medical Things
KNN	K-Nearest Neighbors
LDA	Linear Discriminant Analysis
LGBM	Light Gradient Boosting Machine
LOSO	Leave-one-subject-out
LOTO	Leave-one-trial-out
LSTM	Long Short Term Memory
MAC	Multiply-and-accumulate
MEMS	Microelectromechanical
ML	Machine Learning
MLP	Multilayer Perceptron
NB	Naïve Bayes
PAMAP2	Physical Activity Monitoring for Aging People 2
PCA	Principal Component Analysis
RFID	Radio-frequency Identification
ReLU	Rectified Linear Unit
RLG	Ring Laser Gyroscope
RNN	Recurrent Neural Network
SBS	Sequential Backward Selection
SFS	Sequential Forward Selection
SGD	Stochastic Gradient Descent
SMA	Signal Magnitude Area
SNOW	Semi Non-overlapping Temporal Window
STDFT	Short-time Discrete Fourier Transform
UART	Universal Asynchronous Receiver/Transmitter
WISDM	Wireless Sensor Data Mining

Contents

1 Introduction

This chapter provides an introduction to the research by first discussing the motivation and the problem statement. Next, the hypotheses are presented, followed by the research aim, the contributions, the justification, and the publications. Finally, the thesis structure is presented.

1.1 Motivation

The global life expectancy at birth will rise from 71 to 77 years by 2050, according to the United Nations report (UNITED-NATIONS, 2017), causing the elderly population to grow increasingly faster in next years. This significant growth will have important health and social impact in many regions of the world, and in most of them the health system may not be prepared for this change.

Efficient strategies will be mandatory requirements to meet these challenges, and technological innovative movements are already essential to promote better healthcare. Monitoring clinical signs and physical activities of the older population is fundamental to detect early warning signs of serious medical conditions or abnormal situations, offering them the opportunity to live a longer independent life with autonomy to perform their activities of daily living. In addition, encouraging the young and adult population to maintain a healthy life style is critical to preventing an ill elderly population in the future, and relieving the health system.

According to Lu et al. (2020), wearable devices will play a greater role in healthcare and become better integrated into people's daily lives. Research has already found positive effects in the use of wearables on health. A cross-sectional study suggested wearable devices motivate and engage users to practice physical activity through habit formation, and are recommended to increase physical activity and decrease sedentary behavior (YEN; HUANG, 2021). Jakicic et al. (2020) reviewed research that indicate there are favorable effects of wearable technologies on physical activity and weight loss intervention for adults with obesity. Also, wearable sensors have great potential as a continuous monitoring system for mental health (LONG et al., 2022).

The recent advancement of sensors, wireless communication, and machine learning techniques promote the establishment of assistive and monitoring technologies which can provide independent, active and healthy aging. Among these technologies, the wearable sensor-based recognition and tracking of activities become one of the most promising solutions to assist elderly daily life. Moreover, several commercial wrist-worn wearable devices are currently available to monitor physical activity, and are an interesting strategy to promote self-monitoring exerc ise, workout, and sleeping.

Furthermore, the rapid proliferation of smartphones, smartwatches, and wearable devices have contributed to investments and advances in commercial applications for monitoring sports, physical and daily living activities. This context also has motivated researchers to develop activity recognition systems for practical applications, such as gesture recognition (OYEDOTUN; KHASHMAN, 2017), patient monitoring systems (CAPELA; LEMAIRE; BADDOUR, 2015), biometrics (WEISS; YONEDA; HAYAJNEH, 2019), gym activities (QI et al., 2019), behavior analysis (BATCHULUUN et al., 2017), and a variety of healthcare systems (RODRIGUES et al., 2018).

The monitoring and comprehension of human activities from input data sources, such as sensors, and multimedia contents is named Human Activity Recognition (HAR) (DANG et al., 2020). HAR is a very active area of research, and a lot of discussion are going on, however many challenges are still faced, including complexity and variety of daily activities, intra-subject and inter-subject variability for the same activity, computational efficiency in embedded and portable devices, and design and development of small, lightweight, powerful and low-cost smart sensor nodes (LABRADOR; YEJAS, 2013; MUKHOPADHYAY, 2015).

1.2 Problem Statement

HAR is a typical pattern recognition problem and, traditionally, is based on handcrafted features, extracted from raw sensor data, applied to a supervised machine learning model for classifying the activities. However, this approach presents important drawbacks, such as human expertise dependence for feature selection (YANG et al., 2015), not suitable for recognizing complex activities (YANG, 2009), and difficulty in handling inter-subject variability for the same activity (ZHU et al., 2019a).

Therefore, recently, HAR researchers have shifted towards deep learning approaches, motivated by their state-of-the-art performance in many applications area. Most of deep models learn features automatically during the training process, achieving more significant and representative features than traditional methods. However, these models were designed to maximize accuracy without much consideration of the implementation complexity, which can lead to designs that are challenging to implement and deploy (SZE et al., 2017), mainly in wearable sensors with low computational power. According to Sze et al. (2017), deep learning models demand significantly higher computational cost, number of memory accesses, storage, and energy consumption than most of traditional machine learning approaches.

At the same time that deep learning is becoming the trend in state-of-the-art HAR techniques, real-world solutions require energy efficiency, real-time response, and privacy. These prerequisites are not met by complex deep models, as they require data transmission to the cloud to run on more powerful processors. Data transmission consumes more energy, increases latency, and reduces privacy. Therefore, to meet real-world application requirements, edge computing emerges as a viable solution. For deploying and running HAR systems in resource constrained wearable devices, there are some important preconditions such as low computational cost, small memory footprint, and low power consumption, which are not achievable by most of state-of-the-art deep learning approaches.

In light of the above, the following question arises: how can one take advantage of the gain brought by the advancement of deep learning for HAR, while meeting the requirements demanded by resource constrained wearable devices? Are deep complex models needed for one to achieve high enough performance for real-world applications?

1.3 Hypotheses

Recently, HAR researchers have adopted complex deep neural networks techniques inspired by their state-of-the-art achievements in several studies. Although these approaches have shown relevant gain in activity classification performance, they are not appropriate for deployment on resource-constrained wearable devices. Additionally, most of the works, when dealing with public datasets, use all available data as raw input data for their deep models, without proper study of the impact of each sensor, and the influence of lower sampling rates. Therefore, when developing the proposed energy-efficient HAR framework, three hypothesis were stated:

1. Real-world HAR applications do not need highly complex deep learning models to achieve high performance.

2. Reducing the sampling rate has benefits for computational performance and also for activity classification.

3. In a multi-sensor scenario, a large volume of motion data generated by irrelevant devices or sensors can be discarded in order to generate a more efficient HAR model, maintaining classification performance.

1.4 Research Aim and Objectives

Given the lack of studies regarding the development of HAR models suitable for embedded systems, this doctoral research aims to develop an energy-efficient HAR framework for resource constrained devices, in addition to deploying and testing the framework on a microcontroller. For accomplishing the research aim, the following objectives are defined:

1. Investigate state-of-the-art deep learning-based approaches for HAR and analyze their complexity levels for edge computing devices.
2. Identify energy-efficient strategies for deploying low-power HAR systems on resource-constrained devices.
3. Develop a lightweight HAR framework based on shallow Decision Trees and compact Convolutional Neural Network (CNN).
4. Conduct an extensive analysis of the framework performance in relation to parameters such as sampling rate, device placement, and sensors used.
5. Deploy and test the proposed framework on a microcontroller and evaluate its computational performance.
6. Assess the impact of the embedded framework in different configurations on computational cost, power consumption, and battery life.
7. Compare the proposed framework with reference models and state-of-the-art proposals in terms of their strengths and weaknesses.

1.5 Research Contributions

Throughout the development of this research, the following contributions were achieved:

1. Development of an energy-efficient HAR framework suitable for resource-constrained devices, based on 2-level classifier approach using shallow Decision Trees and a compact CNN.
2. Extensive analysis of the classifiers used in the framework, demonstrating that simple models can be quite efficient, achieving high performance, and reducing computational cost.
3. Deployment of the proposed framework on a microcontroller, and evaluation of processing time and memory usage in a real-world scenario.
4. Study of the impact of model complexity and system parameters on estimated energy consumption and battery life.

1.6 Justification

This work will contribute to the development of strategies and approaches for the implementation of HAR systems for resource-constrained wearable devices. The extensive study carried out in this research will provide great value for the adoption of more efficient techniques by HAR researchers and even by real-world applications.

The deployment, testing and analysis of the proposed HAR framework on the microcontroller will help to address the current gap in literature in this practical context. There is a shortage of works that run practical tests of the implemented models, thus the findings from the tests carried out in this research will provide a more realistic view of computational performance.

1.7 Publications

The following publications resulted from this doctoral research:

1. COELHO, Y., SANTOS, F., FRIZERA-NETO, A. and BASTOS-FILHO, T. A Lightweight Framework for Human Activity Recognition on Edge Computing Devices. IEEE Sensors Journal, v. 21, pp. 24471-24481, 2021.

2. NGUYEN, B., COELHO, Y., BASTOS-FILHO, T., KRISHNAN, S. Trends in human activity recognition with focus on machine learning and power requirements, Machine Learning with Applications, v. 5, 2021.

3. COELHO, Y., NGUYEN, B., SANTOS, F., KRISHNAN, S. and BASTOS-FILHO, T. A Lightweight Model for Human Activity Recognition Based on Two-Level Classifier and Compact CNN Model. CBEB 2020. IFMBE Proceedings, v. 82, 2022.

4. COELHO, Y., RANGEL, L., SANTOS, F. and BASTOS-FILHO, T. Evaluation of low-level quantization of acceleration data on human activity recognition system. In: Proceedings of International Workshop on Assistive Technology 2019.

5. COELHO, Y., RANGEL, L., SANTOS, F., FRIZERA-NETO, A. and BASTOS-FILHO, T. Human activity recognition based on convolutional neural network. IFMBE Proceedings of XXVI Brazilian Congress on Biomedical Engineering, Springer, v. 2, p. 247–252, Oct 2018.

1.8 Structure of the Thesis

In Chapter 1, the context of the study has been introduced. The hypotheses, research objectives, and contributions have been identified, and the justification of such research argued.

Chapters 2 approaches the theoretical background on Human Activity Recognition and Machine Learning. The key concepts and definitions are introduced. In addition, the main techniques for HAR model development, and the methods for model evaluation are addressed, specially the ones employed in this research.

In Chapter 3, the existing literature is reviewed to investigate and discuss state-of-the-art deep learning approaches, and energy-efficient strategies for HAR. The strengths and weaknesses of several studies are highlighted, as well as their main contributions.

Chapter 4 presents the proposed energy-efficient HAR framework and the model development. The software and hardware tools, the datasets, and every step of the framework design process are explained. Furthermore, methods for model testing and evaluation are detailed.

Chapter 5 approaches the materials and methods used for deploying and testing the proposed HAR framework on the microcontroller. The software and hardware tools are presented, in addition to the description of the framework implementation in C code on the device. Also, this chapter details the methods for testing and evaluating the embedded framework.

In Chapter 6, the results are presented and discussed in three parts. First, the framework classification performance is extensively analyzed and compared with reference models. Next, the computational performance of the framework embedded on the microcontroller is discussed. Lastly, the chapter approaches a comparison with state-of-the-art techniques.

Finally, Chapter 7 provides the research conclusion. The key research findings are summarized, discussing the main contributions achieved. This chapter also highlights the limitations of the study, and propose recommendations for future work.

2 Theoretical Background

2.1 Human Activity Recognition

This chapter presents a theoretical background on Human Activity Recognition (HAR), with concepts, definitions, and main techniques and devices used for this purpose, for better understanding of the research developed. The Internet of Medical Things (IoMT) paradigm, the problem of HAR, and the different approaches and sensors used in this context are discussed. Finally, the concept of edge computing is introduced.

2.1.1 Introduction

Human Activity Recognition (HAR) is the automatic detection and understanding of human motion behaviour based on data extracted from sensors. Recognizing user activity provides valuable contextual information to help user-centered applications to better adapt to the user needs in many different areas (MUNOZ-ORGANERO, 2019). In addition, HAR records people's behaviors with data that allows computing systems to monitor, analyze, and assist their daily life (CHEN et al., 2020).

The recognition of human activities has become a task of high interest for medical, military, and security applications. Interactive games or simulators might require information about which activity the user is performing in order to respond accordingly. Furthermore, in tactical scenarios, precise information about the soldiers' activities along with their locations and health conditions is highly beneficial for their performance and safety (LABRADOR; YEJAS, 2013).

One of the most attractive fields in HAR is the recognition of Activities of Daily Living (ADLs) for health or fitness tracking. ADLs are the activities people perform daily, such as eating, bathing, dressing, working, cooking, house cleaning, driving, biking, and all of these activities involving physical movement (DEMROZI et al., 2020). According to Labrador and Lara (2013), recognizing ADLs from patients with some chronic disease is quite useful to provide feedback to the caregiver about the patient's exercise routine. In the same way, HAR systems could monitor patients with dementia and other mental ailments to detect abnormal activities and thereby avoid undesirable outcomes.

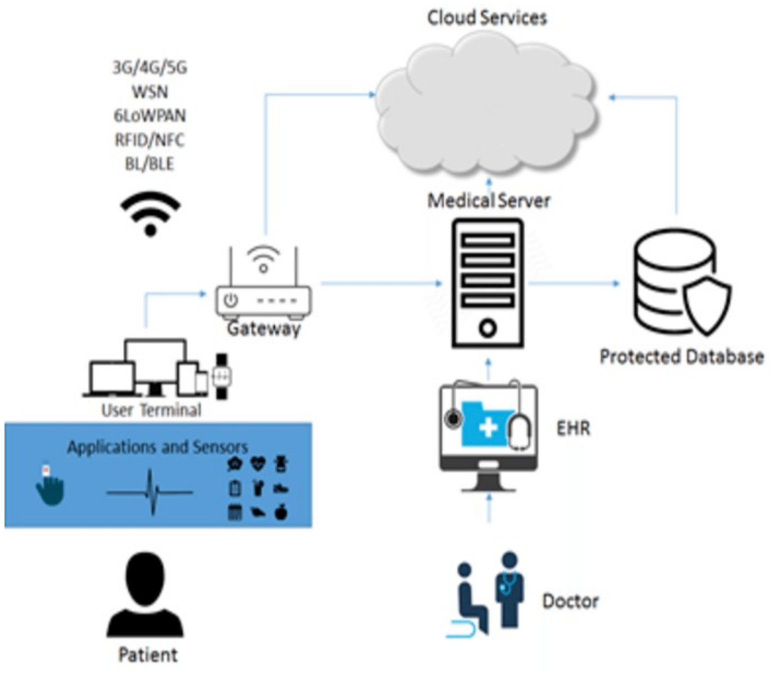
Among the ADLs, the static activities are those where the subject is steady, such as standing still, sitting or lying down. On the other hand, dynamic activities are those where the individual is moving, ambulating, cycling, taking stairs, or interacting with objects. Although there is no rigid definition, simple activities usually refers to the ambulatory or static activities (running, biking, standing still), while complex activities are those in which the person interacts with objects, such as cooking, brushing teeth, typing etc.

2.1.2 Internet of Medical Things

In ubiquitous computing environments, HAR has become an interesting tool for the efficient tracking of ADLs in a smart living environment within an Internet of Things (IoT) architecture. The IoT aims for a world where everything, including houses, home appliances, cars, urban furniture, and industrial machinery are connected to the Internet at any given time and place, creating real-time data that facilitates decision making, offering the user comfort while saving time and money (GARZóN-CASTRO, 2021). By integrating various sensors and data analytics, IoT offers various solutions that can be used in smart grids, smart towns, smart homes, among other applications (JABBAR, 2021).

The Internet of Medical Things (IoMT) is the interconnection of various medical devices and their applications connected with Internet, and it is rapidly changing the healthcare sector. IoMT paired with enabling technologies, such as artificial intelligence, cloud computing, and wireless sensor networks, can effectively monitor people's health continuously (JABBAR, 2021). A typical architecture of the Internet of Medical Things is presented on Fig. 1.

Figure 1 – Internet of Medical Things architecture.



Source: (RODRIGUES et al., 2018)

Data from patients can be collected by using a wide range of sensors and processed by applications on computers, smartphones, smartwatches or, even, a specific embedded device. The user terminal is connected to a gateway through short coverage communication protocols. This gateway connects to a (clinical) server or cloud services for data processing and storage (RODRIGUES et al., 2018).

Recognizing and monitoring ADLs are fundamental functions to provide healthcare and assistance services to elderly people living alone, and physically or mentally disabled people (ATTAL et al., 2015), and it can be totally boosted by the combination of HAR and IoMT features.

2.1.3 Problem definition

The recognition of human activity is crucial for monitoring of ADLs, however accurate recognition is challenging because human activity is complex and highly diverse (KIM; HELAL; COOK, 2010). HAR can be treated as a typical pattern recognition problem. First, raw input data are obtained from several types of sensors (smartphones, smartwatches, Wi-Fi, Bluetooth, sound etc.). Second, features are manually extracted from those readings based on human knowledge, such as mean, variance, and amplitude, in traditional machine learning approaches. Finally, those features serve as inputs to train a model to infer activities in real life settings (WANG; HE; ZHANG, 2019).

In the context of wearable sensors, data are indexed over the time dimension, which allows to define the HAR problem as follows, according to Labrador and Lara (2013). Given a set W (Equation 2.1) of m equally sized time windows, totally or partially labeled, and such that each W_i contains a set of time series S_i (Equation 2.2) with k measured attributes, and a set A of activity labels (Equation 2.3), the goal is to find a mapping function $f : S_i \to A$ that can be evaluated for all possible values of S_i, such that $f(S_i)$ is as similar as possible to the actual activity performed during W_i.

$$W = \begin{bmatrix} W_0 & W_1 & W_2 & \cdots & W_{m-1} \end{bmatrix} \tag{2.1}$$

$$S_i = \begin{bmatrix} S_{i,0} & S_{i,1} & S_{i,2} & \cdots & S_{i,k-1} \end{bmatrix} \tag{2.2}$$

$$A = \begin{bmatrix} a_0 & a_1 & a_2 & \cdots & a_{n-1} \end{bmatrix} \tag{2.3}$$

In addition to the use of wearable sensors in HAR, there are also other approaches that use other types of sensors. There are two primary approaches for HAR according to the type of data being processed: vision-based and sensor-based, as further described in next subsection.

2.1.4 Approaches

This subsection presents the characteristics, and advantages and disadvantages of each HAR approach. Video-based HAR analyzes videos or images containing human motions from cameras, whereas sensor-based HAR focuses on the motion data from on-body or ambient sensors such as accelerometer, gyroscope, sound sensors, wireless radio-frequency modules, and so on.

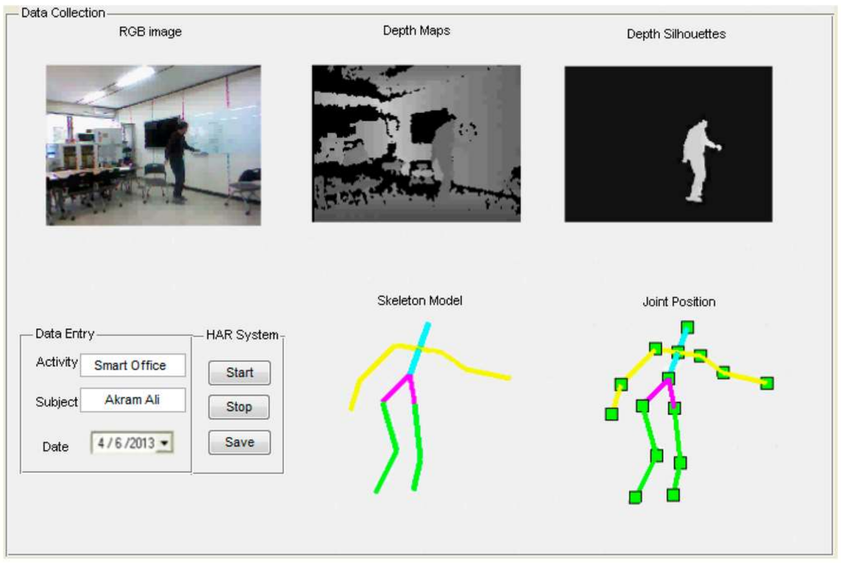
2.1.4.1 Video-Based Approach

Vision-based HAR systems use visual sensing devices to monitor an individual's behaviour and its environment changes. The generated data are video sequences or digitized visual data. This approach exploits computer vision techniques, including feature extraction, structural modelling, movement segmentation, action extraction, and movement tracking, for pattern recognition. Researchers have used a wide variety of modalities, such as a single or multi-camera, stereo and infrared, to investigate a diversity of application scenarios, for single or groups of individuals (CHEN; NUGENT, 2019).

Typically HAR-based systems use monocular RGB videos (Zhang et al. 2019), which makes it hard to comprehensively represent actions in 3D space. Other techniques have emerged more recently, such as RGB-depth-based, skeleton analysis, and thermal images. In recent years, low cost and high mobility sensors, like Microsoft Kinect are being widely adopted. Kinect's ability to track skeleton joints has attracted significant attention from HAR computer vision researchers (YADAV et al., 2021). An example of video-based HAR is shown on Fig. 2, in which silhouettes are captured from Microsoft Kinect and further processed for body joint identification, motion feature generation, and training using Hidden Markov Models (JALAL; KAMAL; KIM, 2014).

Although video-based systems can provide high recognition rate, the use of cameras is not practical in many indoor environments, due to privacy issues (WANG et al., 2016; LABRADOR; YEJAS, 2013; CHEN; NUGENT, 2019). Furthermore, it is not a pervasiveness approach, since video recording devices are difficult to attach to individuals, then they should stay within a perimeter defined by the position and the capabilities of the cameras (LABRADOR; YEJAS, 2013). Finally, vision-based techniques often suffer from illumination variations, ambient occlusion and background change that greatly limits their actual use (WANG et al., 2016). It is a complex approach, since video processing techniques are computationally expensive, decreasing scalability and reusability (LABRADOR; YEJAS, 2013; CHEN; NUGENT, 2019).

Considering the aforementioned issues, sensor-based systems have dominated the applications of monitoring people's daily activities (WANG et al., 2018; CHEN et al., 2020; MURAD; PYUN, 2017). Additionally, sensor miniaturization and advances in mobile



Activity: Smart Office
Subject: Akram Ali
Date: 4/6/2013

Figure 2 – An example of video-based HAR approach using Microsoft Kinect.

Source: (JALAL; KAMAL; KIM, 2014)

technologies made wearable and ambient sensors even more popular.

2.1.4.2 Sensor-Based Approach

The development and the rapid dissemination of wireless sensor network, smart devices, and IoT have contributed to sensor embedding in devices such as smartphones, smartwatches, goggles, and non-portable objects such as cars, walls, and furniture, and have led to a large amount of data being gathered from those sensors (DANG et al., 2020; CHEN et al., 2020). Sensor-based HAR approaches collect raw data from wearable sensors, object sensors, and ambient sensors, and use machine learning to infer the activities performed by individuals and the interactions with the environment.

As wearable sensors can directly and efficiently capture body movements, they are the most commonly used for sensor-based HAR. Three standard wearable sensors include accelerometer, gyroscope, and magnetometer, which can be conveniently worn by users or integrated into portable devices, such as smartphones, smartwatches, smart bands, glasses, or helmets. In addition, electromyography (EMG) and electrocardiography (ECG) sensors are less commonly used, but they can add physiological information (CHEN et al., 2020; DANG et al., 2020).

Object sensors are attached to specific objects in order to allow the recognition of human interactions with that particular object, such as cooking, taking medications, and watching TV. Radio-frequency identification (RFID) sensors are the most widely used for identifying object usage. When acting as object sensors, RFID tags are needed to be attached to the target objects such as mugs, books, computers, and toothpaste (CHEN et al., 2020).

Ambient sensors are used to capture the interaction between humans and the environment. They are usually embedded in users' smart environment. The most common ambient sensors are Wi-Fi, RFID, radar, sound, pressure, humidity, and temperature sensors. Different from object sensors which measure interactions with objects, ambient sensors are used to capture changes in the environment when physical activities occur (WANG; HE; ZHANG, 2019). According to Chen et al. (2020), the unique advantages of ambient sensors is that they can be used to detect multi-occupant activities, and can also be adopted for indoor localization, which is difficult for wearable sensors to achieve.

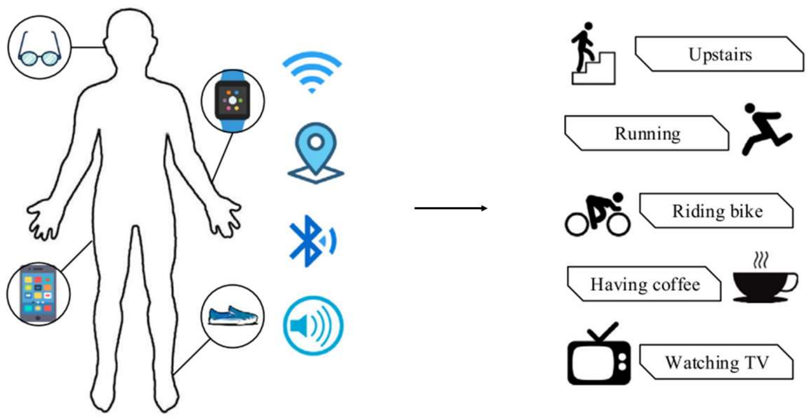


Figure 3 – Sensor-based HAR approach.

Source: Adapted from (WANG; HE; ZHANG, 2019)

2.1.5 Wearable Devices

From smartwatches that monitor your heart rate, to robotic suits that give you more strength to walk and carry heavy loads, wearable technologies are some of the hottest, fastest-growing, popular, emerging technologies on the market today (JOINER, 2018).

Wearable devices are small electronic and mobile devices, with wireless communications capability, that continuously monitor real-time information of the physiological conditions and movements of a person. Wearable sensors are incorporated into gadgets, accessories, clothes, elastic bands, directly attached to the human body, or even invasive versions such as micro-chips, and smart tattoos (MAJUMDER; MONDAL; DEEN, 2017; LUCZAK et al., 2019).

These devices are constructed in different forms to meet their function requirements. They should be able to collect data continuously, process them, and communicate in a secure way while keeping their power consumption as low as possible (OMETOV et al., 2021). Furthermore, some other requirements for wearable sensors are: (i) fashionable and attractive, (ii) compact enough to fit onto the human body, (iii) comfortable, and (iv) and water tolerant (KUMARI; MATHEW; SYAL, 2017).

As reported by Ometov et al. (2021), electronic miniaturization trend, portability, wireless communication, energy-efficient computing, and advanced display technologies have been combined to create state-of-the-art smart wearable devices, which can be divided in different categories according to the body placement, as presented in Fig. 4. Some of these devices are smart bands, posture correcting devices, smart clothes, tattoos, smart glasses, smart headsets, neural interfaces, smart rings, smartwatches, smart shoes, among other devices.

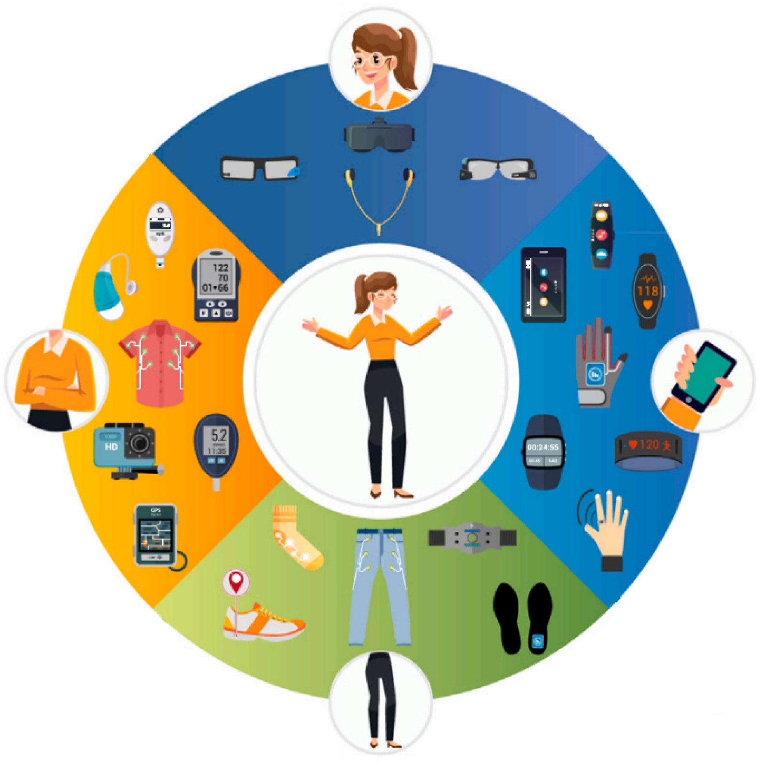


Figure 4 – Wearable devices categorized by body location.

Source: Adapted from (OMETOV et al., 2021)

In healthcare context, wearable devices can be used for early diagnosis and management of medical conditions as well as measuring the vital signs such as body and skin temperature, blood pressure, heart rate, ECG, among others, and may also take a step further and inform the doctor automatically (OMETOV et al., 2021).

More specifically in HAR context, although several kind of sensors can be used, including pressure, body temperature, EMG, ECG, and oximetry (DEMROZI et al., 2020), the inertial sensors are more commonly employed as they are less power demanding compared to the previous sensors (CRUCIANI et al., 2019), and carry information of human movement. Motion data collected from various types and quantities of motion sensors placed on different parts of body are normally used (ZHU et al., 2019b) in current state-of-the-art HAR researches.

Usually, the use of multiple wearable sensors in different body parts (e.g. head, wrist, waist, legs, feet) can improve the performance of wearable sensor-based HAR systems. However, multiple sensors imply higher costs, practical deployment difficulties and obtrusion (WANG et al., 2018). On the other hand, HAR systems with only one wearable device also has some limitations, which probably decreases recognition rate for certain activities.

Fitness trackers keep track of the number of steps users take and advanced versions are able to monitor heart rate and calories burned (JOINER, 2018), and some measure oxygen saturation. Some of the currently commercially available fitness trackers are shown on Fig. 5.

Figure 5 – Examples of commercial wrist-worn fitness trackers bands in the market currently. (a) Fitbit Alta 101; (b) Huawei Band 4; (c) Xiaomi Mi Band 6; (d) Amazfit Band 5.

Source: (FITBIT, 2022a; HUAWEI, 2022; XIAOMI, 2022; AMAZFIT, 2022)

Smartwatches can track physical activities, number of calories burned, and heart rate. When connected to mobile devices, they can notify users about social media messages, emails, and calls (JOINER, 2018). Newer versions can measure oxygen saturation and even blood pressure, track some sport activities automatically (GARMIN, 2022b; SAMSUNG, 2022c; APPLE, 2022c; FITBIT, 2022c), recognize falls (SAMSUNG, 2022a; APPLE, 2022b), and identify signs of atrial fibrillation (STRIK et al., 2021; APPLE, 2022a).

2.1.6 Inertial Sensors

Wearable sensors are widely used for HAR, mainly using inertial sensors, which are sensors composed of accelerometers and gyroscopes. Inertial sensors exploit inertial forces acting on an object to determine its dynamic behavior. External forces acting on a body can cause an acceleration or a change of its orientation (angular position). The rate of change of the angular position is the angular velocity. Accelerations and angular velocities

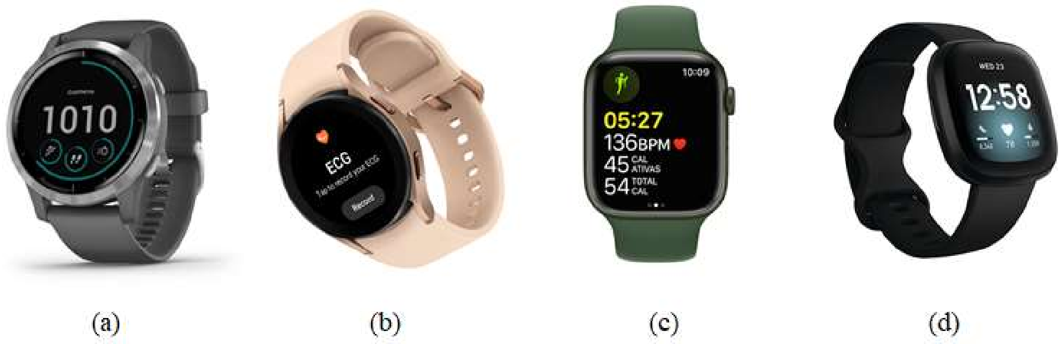


Figure 6 – Examples of commercial smartwatches in the market currently. (a) Garmin Vivoactive 4; (b) Samsung Galaxy Watch4; (c) Apple Watch Series 7; (d) Fitbit Versa 3.

Source: (GARMIN, 2022a; SAMSUNG, 2022b; APPLE, 2022c; FITBIT, 2022b)

are vectorial signals possessing absolute values and orientations, and are denoted 1D, 2D, or 3D according to the number of axial components measured (KEMPE, 2011b).

Today a microelectromechanical (MEMS)-based accelerometer or gyroscope is understood as a complete product that is packaged, calibrated, and tested, and has to be delivered to the customer, who wants to integrate this component with minimal effort into a higher-level measurement or control system (KEMPE, 2011b).

2.1.6.1 Accelerometers

An accelerometer is a basic technology that converts mechanical motion into an electrical signal. It is an electromechanical device that measures acceleration force, whether caused by gravity or motion. There are many different types of mechanisms involved in the accelerometers, including piezoelectric, piezoresistive, capacitive, Hall effect, magnetoresistive, and temperature sensors. Piezoelectric, piezoresistive, and capacitive types are the most common in commercial devices (SUH, 2015)

A common sensing approach used in accelerometers is capacitance sensing in which acceleration is related to change in the capacitance of a moving mass. This sensing technique is known for its high accuracy, stability, low power dissipation, and simple structure to build. The displacement of the movable mass (micrometer) is caused by acceleration, and it creates an extremely small change in capacitance, resulting in a sensor output whose amplitude is proportional to acceleration (MAXIM INTEGRATED, 2022).

As acceleration is proportional to external force, this measurement can reflect both the intensity and frequency of human movement. Body postures (orientations) can be classified by accelerometers, which rotate with objects and can respond to gravity, providing tilt sensing with respect to reference planes. Accelerometers have been widely accepted as useful sensors for wearable devices in either clinical/laboratory settings or free-living environments (MATHIE et al., 2004; CREAN; MCGEOUGE; O'KENNEDY, 2012).

For recognizing human activities, accelerometers can be mounted on various parts of a body, such as the waist, arm, ankle, wrist etc. There are three axes in an often-used accelerometer. Therefore, a tri-variate time series would be captured through an accelerometer (CHEN et al., 2020).

2.1.6.2 Gyroscopes

A classical gyroscope consists of a flying wheel, which, together with friction-compensating mechanisms, allows one to approximate its rotation as a force-free movement within an inertial space. When the platform rotates, the principle of conservation of angular momentum keeps the wheel's motion constant within the inertial system, causing the angular velocity vector to change orientation within the platform system. From this change, the platform's rotation angle can be derived (KEMPE, 2011a).

There are many classes of gyroscopes, with different operating physical principles and technology involved. In particular, mechanical, optical, including Fiber Optic Gyroscopes (FOGs) and Ring Laser Gyroscopes (RLGs), and MEMS gyroscopes are the more commercially diffused categories (PASSARO et al., 2017).

According to Kempe (2011a), recently, successful structures for building MEMS gyroscopes are based on simple flat, rigid proof masses that are particularly well suited for implementation in micromachining technologies. The proof mass can be either vibrated in a translational movement or excited into rotational oscillations forming a vibrating-wheel gyroscope.

In the HAR and motion analysis context, a gyroscope is usually integrated with an accelerometer and mounted on the same body parts. In addition, a gyroscope has three axes and consequently provides three time sequences as well (CHEN et al., 2020).

2.1.7 Edge Computing

A typical approach to process and analyze data collected from sensors in the IoT context is to send data to cloud and use powerful resources to handle the data. However, according to Chen and Nugent (2019), this potential solution of moving data from source to cloud introduces the following challenges: (i) latency, (ii) scalability, and (iii) privacy.

Regarding latency, real-time inference is critical to many health applications. In respect to scalability, sending data to cloud introduces scalability issues, since network access can become a bottleneck as the number of connected devices increases. Finally, privacy can also be an issue, since users may be cautious of sending sensitive information to the cloud and of how the application would use that (CHEN; NUGENT, 2019).

As a result, to meet these challenges, the concept of edge computing was proposed as a new computing paradigm in which the computation is mainly performed on the edge devices (VOGHOEI et al., 2019). In this approach, the data are processed in the device close to the data source. Also, the data source device itself can process the data. Therefore, edge computing is the least vulnerable form of decentralized processing and storage. Some of the advantages of edge computing in relation to cloud computing are as follows: (i) faster data processing, (ii) reduced network traffic, (iii) real-time data analysis, (iv) reduced network operating cost, (v) decentralized data processing and storage, and (vi) reduced network vulnerability (VOGHOEI et al., 2019).

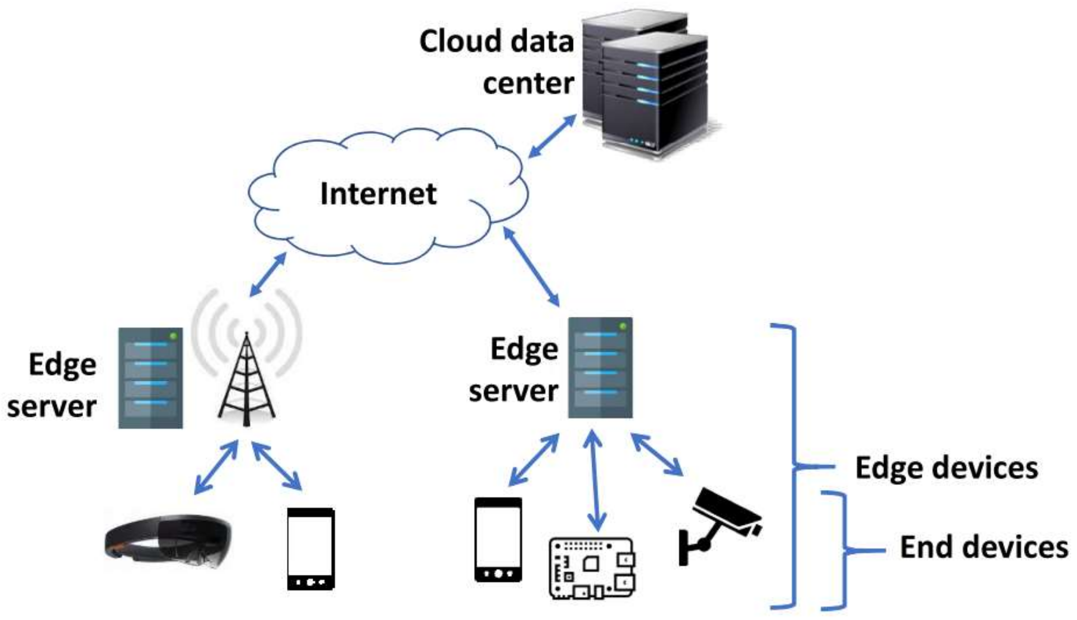


Figure 7 – Illustration of edge devices.

Source: Adapted from (CHEN; NUGENT, 2019)

2.2 Machine Learning Fundamentals

This chapter covers the main concepts and fundamentals related to machine learning and pattern recognition applied in this research. The steps involved in model design are presented, followed by the description of the important models for this research, in addition to the explanation of model evaluation methods.

2.2.1 Introduction

Machine Learning (ML) is a branch of artificial intelligence that applies algorithms to define the underlying relationships among data and information. Supervised learning is an ML approach in which the learning mechanism uses labeled training data to synthesize the model function that attempts to generalize the relationship between the feature vectors (input) and the output (AWAD; KHANNA, 2015). When the goal is to classify images, signal waveforms or any type of measurements into different classes, it is a Pattern Recognition problem (THEODORIDIS; KOUTROUMBAS, 2009).

The steps followed for designing a Pattern Recognition system are shown in Fig. 8. They are interrelated and, depending on the outcomes achieved, the redesign of earlier stages may be needed in order to improve the overall performance. In addition, some methods usually combine steps. The model hyper-parameter optimization can be executed combining the feature selection and the classifier design stages (THEODORIDIS; KOUTROUMBAS, 2009).

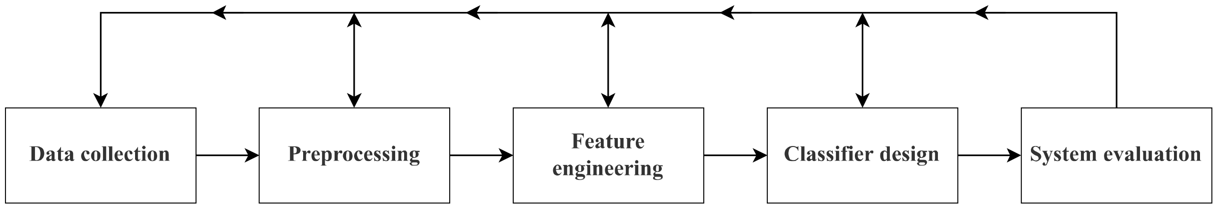


Figure 8 – Process of classification system design.

2.2.2 Preprocessing

Preprocessing is the first step after data acquisition. It is responsible for handling missing and incorrect values, data sparsity, and outliers which can introduce noise to ML models. Accurate predictive analytics of big data can be used to estimate missing values, such as replacing incorrect readings due to malfunctioned sensors or broken communication channels (ZHOU et al., 2017). In some cases, data needs to be normalized, discretized, averaged, smoothened, or differentiated for efficient usage. In other cases, data may need to be transmitted as integers, double precisions, or strings (AWAD; KHANNA, 2015).

In HAR system, due to the nature of inertial sensors, the data collected may need a preprocessing step to reduce noise and eliminate possible artifacts resulting from unwanted sensor movements. A low-pass filter, such as a median, Gaussian, or Laplacian, is usually applied to eliminate high-frequency noise. On the other hand, a high-pass filter can be used when it is desired to distinguish the acceleration of the individual's body from the gravitational acceleration (AVCI et al., 2010).

2.2.3 Segmentation

Time series segmentation is also considered as a discretization problem and might be included in the preprocessing stage. Unlike transactional databases with discrete items, time series data are characterized by their numerical and continuous nature (FU, 2011). Segmentation corresponds to the process of dividing sensor signals into smaller data segments. This process has been performed in different ways in HAR. Most of the segmentation techniques can be categorized into three groups, namely activity-defined windows, event-defined windows and sliding windows (BANOS et al., 2014a).

The slide window technique Semi-Non-Overlapping-Window (SNOW), is the most employed process to yield samples (temporal windows) for activity recognition (JORDAO et al., 2018). This technique considers an overlap of 50% between windows, which is adopted to reduce information loss at the edges of the window (WANG; HE; ZHANG, 2019). Fig. 9 shows the SNOW segmentation process. Each segmented window has a class associated with it, corresponding to the activity performed at that moment. The class can be assigned to the window according to the majority class in the window interval, or by the state of the class at the end of the window, for example.

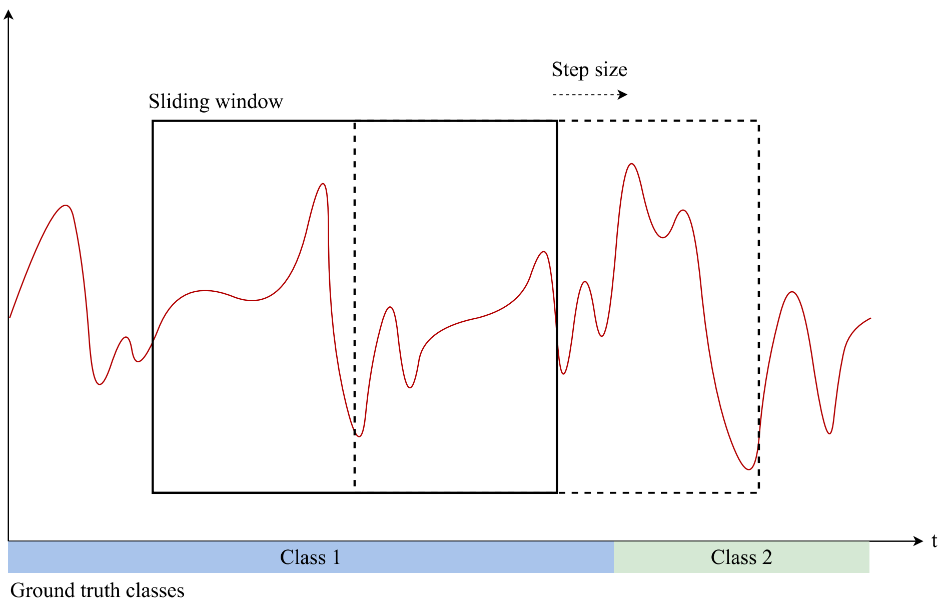


Figure 9 – Sliding window segmentation. The classes shown on time axis represent the activity that was being carried out during that interval.

The window length is a key factor, which depends on the sampling rate of the acquisition device as well as the type of activities being recognized (EHATISHAM-UL-HAQ et al., 2020). Short windows may split an activity signal into multiple separated windows. On the other hand, larger window size may contain multiple activity signals which could also lead to misinterpretation of activities (NOOR; SALCIC; WANG, 2017). Usually, for simple ADLs a window length between two and five seconds has proved to be sufficient, while complex and non-repetitive activities require a larger window from 15 to 30 seconds to reach satisfactory results (SHOAIB et al., 2016).

2.2.4 Feature Engineering

A feature is a reduced and significant representation of raw data, while feature engineering is the process of formulating the most appropriate features given the data, the model, and the task. The number of features is also important. If there are not enough informative features, then the model will be unable to perform the ultimate task. If there are too many features, or if most of them are irrelevant, then the model will be more expensive and tricky to train (ZHENG; CASARI, 2018).

2.2.5 Feature Extraction

The feature extraction consists of finding an efficient way to represent the original data with a reduced set containing its most relevant features, which can be referred as feature vector. According to Guyon and Elisseeff (2006), finding a good data representation is very domain specific and related to available measurements, therefore, human expertise, which is often required to convert raw data into a set of useful features, can be complemented by automatic feature extraction methods.

Sensor data feature extraction methods are usually divided into time domain, frequency domain, and hybrid approaches. Time domain features include basic waveform characteristics and signal statistics. Frequency domain features, on the other hand, concentrate on the periodic structures of the signal, which include coefficients such as those derived from Fourier transforms. For hybrid features, they employ both time and frequency information for complex signals, and a good example of this is the wavelet representation (YANG et al., 2014).

Time domain features include mean, median, standard deviation, variance, range, maximum and minimum value, kurtosis, and skewness features, zero crossing rate, among others (YANG et al., 2014; AVCI et al., 2010; DANG et al., 2020). This approach is more suitable for real-time computation because no transformation is required on embedded systems. In addition, the amplitude and phase of a signal at any given instant can be analyzed quickly using extracted time domain features (DANG et al., 2020). Mean and variance are particularly highly employed, due to their efficient implementation. The mean tends to detect the local posture of the body, while the variance describes how much movement is present in the signal (LAERHOVEN; BERLIN; SCHIELE, 2009).

On the other hand, some examples of frequency domain features are Fourier coefficients, Chebyshev coefficients, spectral peaks, and power spectral density (YANG, 2009; AVCI et al., 2010). While time domain approach shows how a signal changes over time, the frequency domain approach presents how much of a signal remains inside each frequency band over a range of frequencies (GU et al., 2018). For this approach, various transform operators are necessary to convert a signal into a variety of frequencies, thus frequency domain feature extraction need high computational power, so it is not suitable for low-power wearable devices (DANG et al., 2020).

Automatic feature extraction techniques can be applied to transform the high-dimensional data into meaningful data of fewer dimensions by combining attributes (AWAD; KHANNA, 2015). Some strategies are Principal Component Analysis (PCA), Linear Discriminant Analysis (LDA), and Independent Component Analysis (ICA).

PCA is a linear technique that consists of transforming the original features into new mutually uncorrelated features, remapping the original features into a low dimensional space in which the principal components are arranged according to their variance. LDA projects the original features points into a new space of lower dimension that maximizes the between-class separability while minimizing their within-class variability. The ICA algorithm aims at finding independent components, such as the original features that can be expressed as a linear combination of those components (ATTAL et al., 2015).

2.2.6 Feature Selection

Feature selection techniques prune away non useful variables, i.e., redundant or irrelevant features, to reduce the complexity of the resulting model. The end goal is a more efficient model that is faster to compute, with little or no degradation in predictive performance (AWAD; KHANNA, 2015; ZHENG; CASARI, 2018). The proper selection of features may significantly improve the performance in terms of classification and computational performance, while redundant and irrelevant variables may require more processing and deteriorate the performance.

Guyon and Elisseeff (2006) highlighted potential benefits provided by feature selection: (i) reduction of measurement and storage requirements, (ii) reduction of training and prediction processing time, (iii) improvement of prediction performance, and (iv) simplification of data understanding.

Some of the techniques usually employed for feature selection are the well-known Sequential Forward Selection (SFS) and its backward counterpart Sequential Backward Selection (SBS). These methods obtain a chain of nested subsets of features by adding (or removing) the locally best (or worst) feature in the set. Other important approaches are based on Genetic Algorithms (GA). In this case, it allows a stochastic search process guided by a fitness measure (FERRI et al., 1994).

The Python library scikit-learn provides a feature selection tool which uses a meta-transformer that assigns importance to each feature after fitting a model. The features are considered unimportant and removed if the corresponding importance of the feature values are below a provided threshold (SCIKIT-LEARN, 2022).

2.2.7 K-Nearest Neighbors

Neighbors-based classification is an instance-based or non-generalizing learning technique. Instead of constructing a general internal model, it simply stores instances of the training data. K-Nearest Neighbors (KNN) classification technique identifies a group of k objects in the training set which are closest to the test object and assigns a label based on the most dominant class in this neighborhood. The three fundamental elements of this approach are: (i) an existing set of labeled objects; (ii) a distance metric to estimate distance between objects; and (iii) the number of nearest neighbors k.

The value of k should be carefully defined. A small value can result in noisy behavior, whereas a large one may include excessive points from other classes (AWAD; KHANNA, 2015). All the points in the neighborhood can be weighted equally, or they can be weighted by the inverse of their distance, for example. The distance metric commonly used are the Euclidean Distance and the Manhattan Distance, as shown respectively in Equations 2.4 and 2.5, where x is a training object, $\hat{x}$ is a test object, and N is the dimension of the objects.

$$d(x, \hat{x})_{euclidean} = \sqrt{\sum_{i=0}^{N-1} (x_i - \hat{x}_i)^2} \tag{2.4}$$

$$d(x, \hat{x})_{manhattan} = \sum_{i=0}^{N-1} |x_i - \hat{x}_i| \tag{2.5}$$

2.2.8 Tree-Based Classifiers

2.2.8.1 Decision Tree

Decision Tree is a non-parametric supervised learning method used for classification and regression. The goal is to create a model that predicts the value of a target variable by learning simple decision rules inferred from the data features. Decision Trees are invariant to any monotonic transformation (SEGAL, 1988), and require little data preparation. Furthermore, they are simple and intuitive to interpret and can be visualized. Fig. 10 shows an ordinary Decision Tree example.

A Decision Tree classifies instances by sorting them down the tree from the root to some leaf node, which provides the classification of the instance. Each node in the tree specifies a test of some attribute of the instance, and each branch descending from that node corresponds to one of the possible values for this attribute. A new instance is classified by starting at the root node of the tree, testing the attribute specified by this node, then moving down the tree branch corresponding to the value of the attribute in the given example. This process is then repeated for the sub-tree rooted at the new node (MITCHELL, 1997).

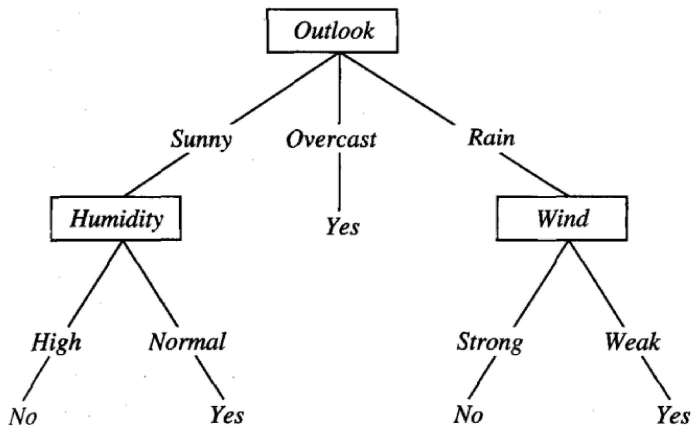


Figure 10 – Example of a Decision Tree that classifies Saturday mornings according to whether or not they are suitable for playing tennis.

Source: (MITCHELL, 1997)

Classification and Regression Trees (CART) is an algorithm for building a Decision Tree based on Gini impurity index as splitting criterion (BREIMAN et al., 1984). The Gini index measures the probability of a particular variable being wrongly classified when it is randomly chosen. It is used by the CART algorithm and is defined as in Equation 2.6, where p_i is the proportion of samples belonging to class i. The variable with the lowest Gini index value defines the node attribute. If all instances in each branch after a node belong to a single class, then Gini index would be zero and the respective node is called "pure".

$$Gini = 1 - \sum_{i=1}^{c} p_i^2 \tag{2.6}$$

CART uses a binary recursive partitioning scheme by splitting two child nodes repeatedly, starting with the root node, which contains the complete learning sample (BREIMAN et al., 1984). The process of growing the tree splits among all the possible splits at each node, such that the resulting child nodes are the "purest". Once a maximal tree is reached, it examines the smaller trees obtained by pruning away the branches of the maximal tree to determine which contribute least to the overall performance. The most suitable tree depth is achieved by evaluating the predictive performance of every tree in the pruning sequence (AWAD; KHANNA, 2015). Some criteria for stop growing the tree can be adopted. For example, the number of cases in the node is less than some specified value, the purity of the node is higher than a threshold or the defined maximum depth of the tree is reached.

2.2.8.2 Random Forest

Random Forest is an ensemble learning technique based on bagging, which combines tree predictors such that each tree uses a random vector sampled independently from the training set, and with the same distribution for all trees in the forest (BREIMAN, 2001). Also, features are randomly selected to the splitting process of each node, and the trees grown are not pruned.

For classification, the n tree predictors are combined, and the majority vote defines the output class. Although individual classifiers are weak learners, all the classifiers combined form a strong learner. Whereas a single Decision Tree experiences high variance and high bias, Random Forest averages multiple Decision Trees to improve estimation performance. By averaging across the ensemble of trees, one can reduce the variance of the final estimation (AWAD; KHANNA, 2015).

2.2.8.3 LightGBM

LightGBM, short for Light Gradient Boosting Machine (LGBM), is a gradient boosting framework that uses tree based learning algorithms (LIGHTGBM, 2022). It is also an ensemble model of Decision Trees, however, differently from Random Forests, the trees are grown sequentially and each new tree is built considering the errors of previous trees. More precisely, LGBM uses the Gradient Boosting Decision Tree (GBDT) algorithm.

GBDT trains the Decision Trees by fitting the negative gradients (or residual errors). The main cost in GBDT is to find the best split points when learning each tree. LGBM uses a sampling method for GBDT that can achieve a good balance between reducing the number of data instances and keeping the accuracy for learned Decision Trees: Gradient-based One-Side Sampling (GOSS) and Exclusive Feature Bundling (EFB) to deal with large number of data instances and large number of features, respectively (KE et al., 2017).

GOSS can obtain quite accurate estimation of the information gain with a much smaller data size, because it uses only larger gradients in the computation of information gain, since they play a more important role in that. While with EFB, LGBM uses histogram-based algorithms to bundle mutually exclusive features into discrete bins to reduce the number of features. These techniques speed up training and reduces memory usage (KE et al., 2017).

The LGBM framework is designed to be distributed and efficient with the following advantages in relation to other GBDT approaches: (i) faster training speed and higher efficiency; (ii) lower memory usage; (iii) better accuracy; (iv) capable of handling large-scale data (LIGHTGBM, 2022).

2.2.9 Artificial Neural Networks

This subsection presents fundamentals concepts of Artificial Neural Networks (ANNs) for better introduction to Deep Learning, approached in next subsection. Here, the neuron model, the activation functions used, the network learning process, and the main optimizers are described.

ANN is a massively parallel distributed processor made up of simple processing units able to store knowledge and make it available for use. The knowledge is acquired by the network by means of a learning process, and the neuron connection strengths, known as synaptic weights, are used to store the acquired knowledge (HAYKIN, 2009).

2.2.9.1 Neuron Model

A neuron is an information-processing unit essential to the operation of an ANN, and is formed by three basic elements: (i) a set of synapses, each of which characterized by its weight; specifically, a signal x_j at the input of synapse j connected to neuron k is multiplied by the synaptic weight w_{kj}; (ii) a linear combiner for summing the input signals weighted by the respective synaptic strengths of the neuron; and (iii) an activation function for limiting the amplitude of the output. Fig. 11 shows the model of a neuron. The model also includes an externally applied bias b_k, which has the effect of increasing or lowering the net input of the activation function (HAYKIN, 2009).

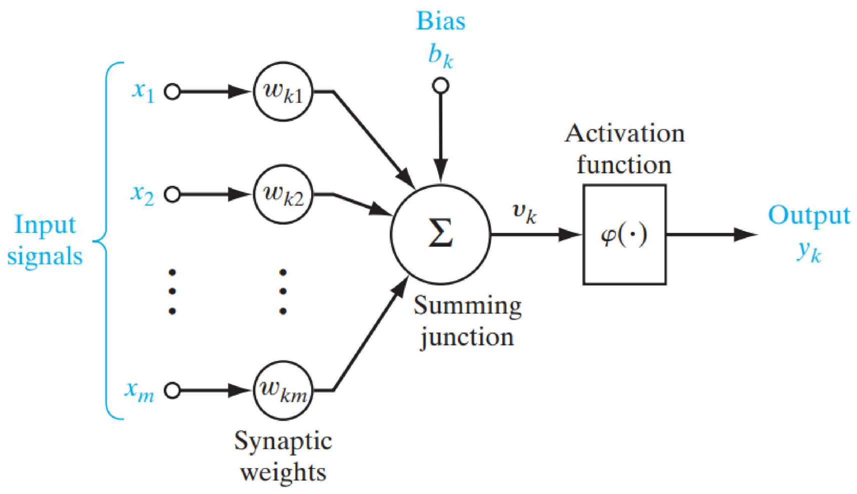


Figure 11 – Neuron model.

Source: (HAYKIN, 2009)

Mathematically, the neuron model can be described as the Equation 2.7, where $x_1, x_2, ..., x_m$ are the input signals, $w_{k1}, w_{k2}, ..., w_{km}$ are the respective synaptic weights of neuron k; b_k is the bias, $\varphi(\cdot)$ is the activation function, and y_k is the output signal of the neuron. The use of bias b_k has the effect of applying an affine transformation to the output of the linear combiner.

$$y_k = \varphi(\sum_{j=1}^{m} w_{kj}x_j + b_k) \tag{2.7}$$

2.2.9.2 Multilayer Perceptron

The Multilayer Perceptron (MLP) is characterized by the presence of at least one hidden layer of neurons, and by the use of a nonlinear and differentiable activation function. The output layer of an MLP network can be composed of multiple neurons for dealing with multiclass classification problems. Fig. 12 shows the structure of an MLP network with two hidden layers. MLP is also denoted as fully connected network, since a neuron in any layer of the network is connected to all the neurons in the previous layer.

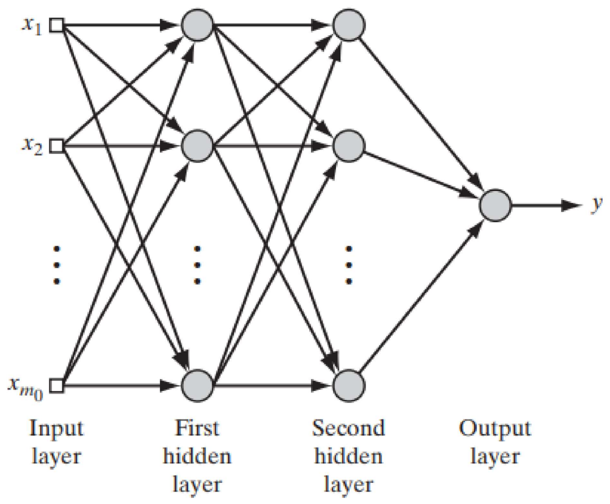


Figure 12 – A Multilayer Perceptron network with two hidden layers.

Source: (HAYKIN, 2009)

2.2.9.3 Activation Functions

The sigmoid function is the most common activation function used with ANN. It is defined as a strictly increasing function that exhibits a balance between linear and nonlinear behavior. An example of the sigmoid function is the logistic function, defined by Equation 2.8.

$$\varphi_{sigmoid}(v) = \frac{1}{1 + e^{-v}} \tag{2.8}$$

The Rectified Linear Unit (ReLU) activation function (Equation 2.9) is a simple variation of a linear function where the negative part is zeroed out. In other words, it trims away the negative values, but leaves the positive part unbounded. The range of ReLU extends from 0 to ∞ (ZHENG; CASARI, 2018).

$$\varphi_{relu}(v) = max(0, v) \tag{2.9}$$

Softmax activation function converts a vector of values to a probability distribution. The elements of the output vector are in range (0, 1) and sum to 1 (KERAS, 2022). Equation 2.10 shows the softmax function equation for M output neurons.

$$\varphi_{softmax}(v) = \frac{e^{v}}{\sum_{k=1}^{M} e^{v_k}} \tag{2.10}$$

Softmax functions are most often used as the output of a classifier, to represent the probability distribution over M different classes (GOODFELLOW; BENGIO; COURVILLE, 2016). The output layer with the softmax activation function is often called softmax layer.

2.2.9.4 Network Learning Process

The back-propagation algorithm is a popular method for the training MLP networks and a highly efficient methodology to find good parameters. It is divided in forward and backward steps. In the forward phase, the synaptic weights are fixed and the input signal is propagated through the network, layer by layer, until the output. In the backward phase, a loss function (the cost function) is used to compute the error between the output of the network and the expected output, and successive adjustments are made to the synaptic weights to minimize the loss function (HAYKIN, 2009).

The back-propagation algorithm is considered to have converged when the absolute rate of change in the loss function between two consecutive iterations is sufficiently small. Each iteration is called an epoch. The cross-entropy loss function is commonly used as cost function for training neural networks for classification problems. Equation 2.11 shows the cross-entropy loss equation for M neurons in the output layer (number of classes), where y and $\hat{y}$ are respectively the expected and the predicted output.

$$\ell = -\sum_{c=1}^{M} y_c \log \hat{y}_c \tag{2.11}$$

Gradient descent is one of the most popular algorithms to perform optimization and by far the most common way to optimize neural networks (RUDER, 2016). The objective is to minimize the loss function $\ell(\theta)$ parameterized by model's parameters θ by updating the parameters in the opposite direction of the gradient of the loss function $\nabla_{\theta}\ell(\theta)$ in relation to the model parameters. The learning rate η determines the size of the steps the process takes to reach a (local) minimum. In summary, the training process follows the direction of the slope of the surface created by the loss function downhill until reaching a valley (RUDER, 2016). The standard gradient descent method computes the gradient of the cost function to the parameters θ for the entire training dataset, as shown in Equation 2.12.

$$\theta = \theta - \eta \nabla_\theta \ell(\theta) \tag{2.12}$$

The Stochastic Gradient Descent (SGD) optimizer performs a parameter update for each training example, or for every subset (mini-batch) of n training examples. The parameters update is calculated as in Equation 2.13, where i is the current iteration of training process.

$$\theta = \theta - \eta \nabla_\theta \ell(\theta; y^{i:i+n}; \hat{y}^{i:i+n}) \tag{2.13}$$

Some strategies are used to enhance the parameter update performance. Momentum, for example, is a method that helps accelerate SGD in the relevant direction and dampens oscillations, by adding a fraction of the update vector of last iteration to the current update vector.

Other algorithms are also proposed for this purpose. Adaptive Gradient Algorithm, or AdaGrad for short, is an algorithm for gradient-based optimization which adapts the learning rate to the parameters, performing larger updates for infrequent and smaller updates for frequent parameters (RUDER, 2016). The RMSprop, short for Root Mean Squared Propagation, is an optimization algorithm which modifies AdaGrad to perform better in the nonconvex setting by changing the gradient accumulation into an exponentially weighted moving average (GOODFELLOW; BENGIO; COURVILLE, 2016).

Finally, Adam optimizer algorithm stores an exponentially decaying average of past squared gradients like RMSprop, and also keeps an exponentially decaying average of past gradients, similar to momentum. According to Goodfellow, Bengio, and Courville (2016), Adam is generally regarded as being fairly robust to the choice of hyperparameters.

2.2.10 Deep Neural Networks

Traditional MLP networks often contains very few hidden layers, and for that reason they are usually denoted as shallow networks. By adding more layers and more units within a layer, a Deep Neural Network (DNN) can represent functions of increasing complexity. Most tasks that consist of mapping an input vector to an output vector, and that are easy for a person to do rapidly, can be accomplished via deep learning (GOODFELLOW; BENGIO; COURVILLE, 2016).

Deep Learning techniques usually combine both feature extraction and classification tasks into a single body unlike traditional ANNs. While conventional Machine Learning methods often use handcrafted features which are not sub-optimal, CNN-based methods can learn and extract the features directly from the raw data.

Diverse structures of DNNs encode features from multiple perspectives. For example,CNNs are competent in capturing local connections of multimodal sensory data, whereas Recurrent Neural Networks (RNNs) extract temporal dependencies and incrementally learn information through time intervals, so are appropriate for streaming data (CHEN et al., 2020).

2.2.10.1 Convolutional Neural Networks

CNN is a neural network which uses convolution in place of general matrix multiplication in at least one of its layers. It works similarly to 2D CNN, except for the different filter dimensionality (height is always 1) and the sliding occurs only in one direction. The convolutional layers are formed by kernels, or filters, while the layer output is referred to as the feature map. Equation 2.14 describes the discrete convolution, where x is the input, w is the kernel, and t refers to each sample (GOODFELLOW; BENGIO; COURVILLE, 2016), and Fig. 13 shows the application of 2D convolution. The boxes with arrows indicate how the upper-left element of the output feature map is formed by applying the kernel to the corresponding upper-left region of the input matrix.

$$s(t) = \sum_{a=-\infty}^{\infty} x(a)w(t-a) \tag{2.14}$$

In convolutional networks, each layer generates a successively higher-level abstraction of the input data, called a feature map, which preserves essential yet unique information. Modern CNNs are able to achieve superior performance by employing a very deep hierarchy of layers (SZE et al., 2017). The CNN input is usually a multidimensional array of data, and the kernel is usually a multidimensional array of parameters that are adapted by the learning algorithm during the model training. After convolution, there is usually a pooling layer, which replaces the output of the net at a certain location with a summary statistic of the nearby outputs. For example, the max pooling operation calculates the maximum output within a rectangular neighborhood (GOODFELLOW; BENGIO; COURVILLE, 2016).

The convolutional layers are responsible for learning and extracting features from the input feature maps and creating the output feature maps. Usually, after passing through these layers, the features extracted are flattened and form the feature vector which is fed into the following fully connected and softmax layers, which are responsible for the classification. An example of 2D CNN is shown in Fig. 14.

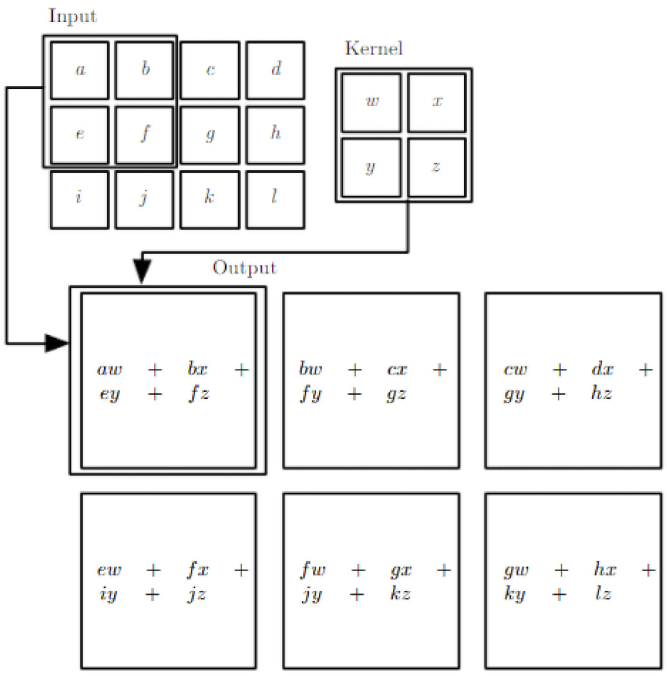


Figure 13 – Example of 2D convolution.

Source: (GOODFELLOW; BENGIO; COURVILLE, 2016)

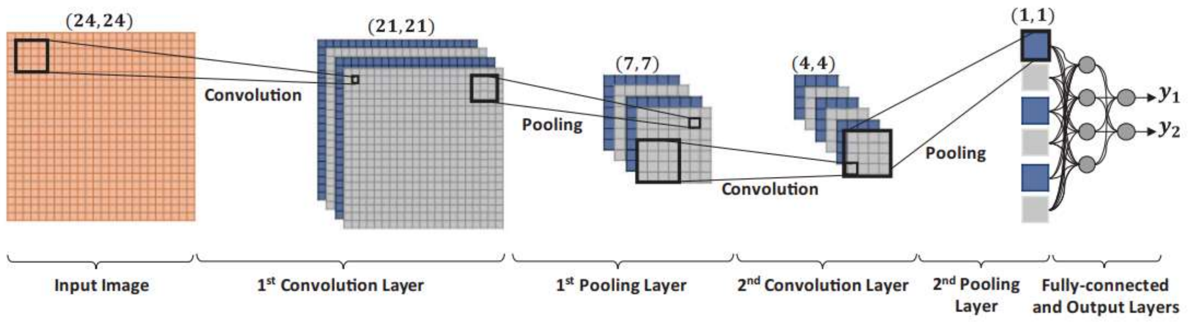


Figure 14 – Example of 2D CNN with two convolutional layers and a fully connected layer.

Source: (KIRANYAZ et al., 2021)

Recently, 1D CNN became a promising method for unidimensional time-series signals in several applications. An advantage is that a real-time and low-cost hardware implementation is feasible due to the simple and compact configuration of 1D CNNs that perform only 1D convolutions (KIRANYAZ et al., 2021). The main advantage of this technique over previous traditional ANN is that 1D CNN extracts features of a signal by considering local information instead of the whole signal in each network layer. This results in faster training of the network with a smaller number of trainable parameters, which cause less computational cost and power (MOZAFFARI; TAY, 2020). Fig. 15 shows an example of 1D CNN.

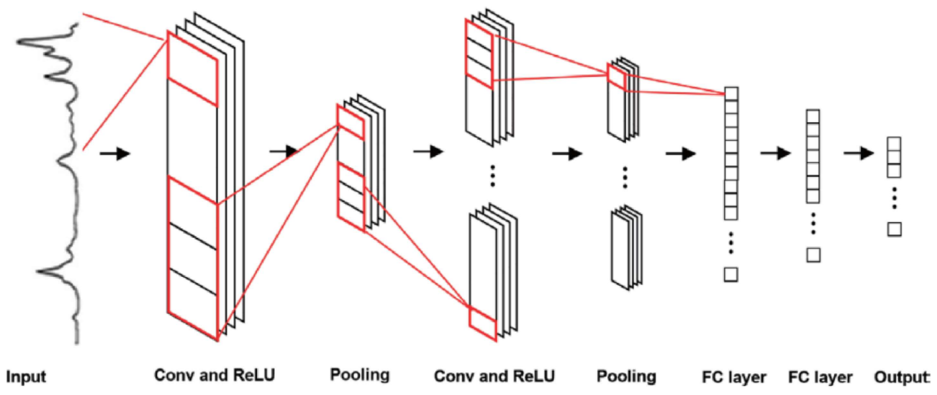


Figure 15 – Example of 1D CNN with two convolutional layers (Conv) and two fully connected (FC) layers.

Source: (MOZAFFARI; TAY, 2020)

2.2.10.2 Recurrent Neural Networks

Recurrent Neural Network (RNN) was developed to model sequential data such as time series or raw sensor data (NWEKE et al., 2018). Just as CNN can scale to images with large width and height, RNN can scale to much longer sequences than would not be practical for networks without sequence-based specialization (GOODFELLOW; BENGIO; COURVILLE, 2016). RNNs incorporate a temporal layer to capture sequential information and then learns complex changes using the hidden unit of the recurrent cell. The hidden unit cells can change based on the information available to the network, and this information is constantly updated to reflect the current status of the network (NWEKE et al., 2018).

A basic model of RNN is shown in Fig. 16. For time t, the following equations define the structure, where x, S, and o are the input, hidden, and output units, respectively, while U, V, and W are the weight matrices, b_1 and b_2 are the biases (not shown in figure), $\varphi(\cdot)$ refers to the activation function, and $\hat{y}$ is the predicted output value.

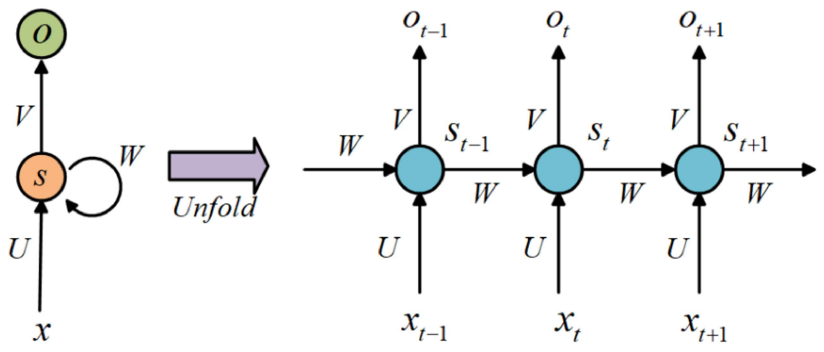


Figure 16 – Basic RNN structure.

Source: (ZHU et al., 2019a)

$$S(t) = \varphi(Ux(t) + WS(t-1) + b_1) \tag{2.15}$$

$$o(t) = \varphi(VS(t) + b_2) \tag{2.16}$$

$$\hat{y}(t) = \varphi(o(t)) \tag{2.17}$$

RNNs deal with the challenge of long-term learning, since the gradients are propagated over many layers and tend to either vanish or explode (GOODFELLOW; BENGIO; COURVILLE, 2016). To overcome this issue, variations of RNN such as Long Short Term Memory (LSTM) and Gated Recurrent Unit (GRU), which integrate varieties of memory cells and gates to capture temporal sequence, were proposed.

LSTM use self-loops, with weights conditioned on the context, to produce paths where the gradient can flow for long duration (GOODFELLOW; BENGIO; COURVILLE, 2016). However, LSTM networks have too many parameters, leading to high computational complexity. To reduce parameter update, GRU was introduced with fewer parameters that make it faster and less complex to implement (NWEKE et al., 2018).

In many applications that may depend on the whole input sequence, the combination of an RNN that moves forward through time with another RNN that moves backward through time could improve the model performance (GOODFELLOW; BENGIO; COURVILLE, 2016). The Bidirectional LSTM (Bi-LSTM) structure has two LSTM layers for extracting temporal dynamics from both forward and backward directions, and is an important variant of the RNN in various domains, including HAR (CHEN et al., 2020).

2.2.10.3 Complexity Analysis

Deep learning models were designed to maximize accuracy without much consideration of the implementation complexity. However, this can lead to models that are challenging to implement and deploy on resource constrained devices. To address this issue, recent work has shown that DNN models and hardware can be codesigned to jointly maximize accuracy and throughput while minimizing energy and cost (SZE et al., 2017). Some strategies include increase sparsity, reduce data precision and choose more compact architectures.

The number of multiply-and-accumulate (MAC) operations is a good metric for understanding the complexity of a neural network. According to Sze et al. (2017), these operations account for 99% of total operations in convolutional and fully connected layers of state-of-the-art CNNs. In fact, computation and memory accesses, including both weights and feature maps, are the main sources of energy consumption in those networks (YANG et al., 2017).

Using convolutional layers with many and large filters might not be a good approach when creating a model for running in embedded devices. According to Yang et al. (2017), convolutional layers consume more energy than fully connected layers, due to significantly more feature map from data movements, which induce higher feature map-related energy consumption. On the other hand, some approaches try to use long sequences of CNN layers, reducing the number and size of filters in each layer. However, when a CNN becomes deeper, it usually generates more feature-map data movement, and the corresponding energy increase outweighs the energy reduction in moving weights (YANG et al., 2017).

Energy consumption, latency and cost are key metrics for machine learning in edge computing. Higher model dimensionality increases the amount of data generated, which is a challenge for energy-efficiency solutions since data movement costs more than computation (SZE et al., 2017; HOROWITZ, 2014). Two techniques for optimizing those models are pruning and quantization. Pruning can reduce the number of MACs by removing weights, where those with minimal impact on the output are removed (SZE et al., 2017). This process gradually zeroes out weights during the training process to achieve model sparsity. Specialized hardware has been proposed to use sparse weights for increasing speed and reducing energy consumption (SZE et al., 2017). Quantization reduces hardware accelerator latency, computational cost, energy consumption and model size with little degradation in model accuracy. Models based on 8-bit integer parameters run faster than the usual ones based on 32-bit floating-point, due to simpler MAC operations and reduced data movement (SZE et al., 2017).

2.2.11 Model Evaluation

When evaluating the predictive performance of a model, there are usually three steps: (i) model selection, (ii) tuning the model hyperparameters, and (iii) estimation of model performance on unseen data.

Hyperparameters are the tuning parameters of a Machine Learning algorithm, such as the maximum depth of a Decision Tree, the learning rate of a neural network, or the number of filters of a CNN. They can be adjusted when optimizing for performance, finding the right balance between bias and variance (RASCHKA, 2018). Bias is the difference between the average prediction and the ground truth value. High bias is related to under-fitting. On the other hand, variance refers to variations model prediction when using different portions of the training data set. High-variance models are over-fitting.

A conventional method for hyperparameters optimization is the grid search, which makes a complete search over a given subset of the hyperparameters space (LIASHCHYNSKYI; LIASHCHYNSKYI, 2019). For tuning the hyperparameters and evaluating the model performance, the dataset must be divided into training, validation, and test subsets. A straightforward approach is a simple three-way split, separating a portion of the dataset for each one of these subsets. The training-validation pair is used for hyperparameter tuning and model selection, while the test subset is kept "independent" for model evaluation (RASCHKA, 2018). However, when the dataset has limited examples, this strategy may not be enough, as the model will be trained and evaluated with little data.

For handling this limited data issue, the k-fold cross-validation method is widely adopted to evaluate the model predictive performance. This method creates partitions from the dataset to form k subsets of approximately equal size. In the first round, one partition is used for testing, while the others are used for model training and hyperparameter optimization. This process is repeated until all partitions are used for testing, and the evaluation metric is averaged. However, this approach introduces an optimistic bias into the performance estimation since if the optimization is performed together with model evaluation. This could be solved by separating a portion of the dataset for testing before partitioning it into k folds, however the test subset would be small.

This method of nested cross-validation can reduce the bias, compared to regular k-fold cross-validation when used for both hyperparameter tuning and model evaluation. This method is relatively straight-forward as it merely is a nesting of two k-fold cross-validation loops: the inner loop is responsible for the model selection and hyperparameter optimization, and the outer loop is responsible for estimating the generalization accuracy (RASCHKA, 2018). Fig. 17 shows the nested cross-validation process. This method can reduce the bias, compared to regular k-fold cross-validation when used for both hyperparameter tuning and model evaluation.

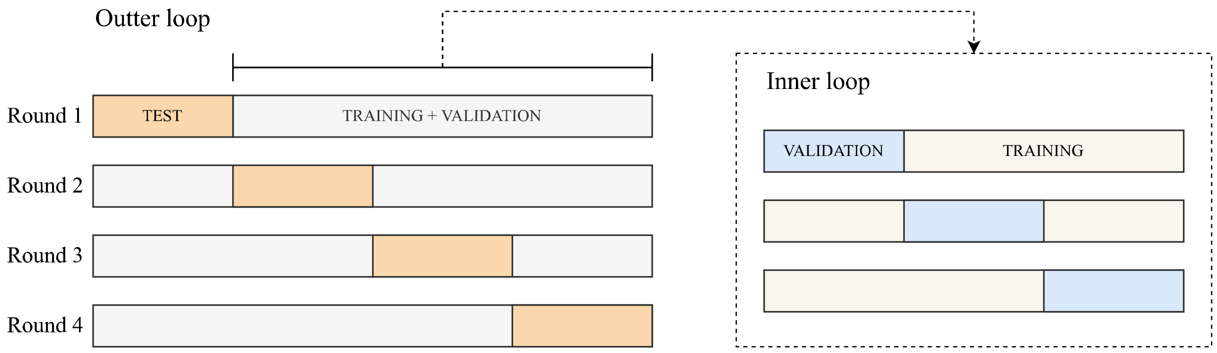


Figure 17 – Nested cross-validation process.

3 Literature Review

This chapter presents a review of the literature on Human Activity Recognition. First, the publicly available datasets are described. Next, the main research findings are highlighted, from early approaches using traditional machine learning to state-of-the-art research using complex models based on deep learning. Finally, the proposals for lightweight models and optimized systems for resource constrained devices are discussed.

3.1 Publicly Available Datasets

Before exploring the most relevant works in the field of Human Activity Recognition, the main datasets explored by HAR researchers are presented here: (i) UCI-HAR, (ii) WISDM-2010, (iii) WISDM-2019, (iv) PAMAP2, (v) MHEALTH, (vi) Opportunity, and (vii) Skoda datasets.

UCI-HAR. The Human Activity Recognition using Smartphones Dataset, also known as UCI-HAR dataset (ANGUITA et al., 2013), consists of 3-axis accelerometer and gyroscope smartphone sensor readings sampled at 30 Hz from a group of 30 volunteers performing six basic activities: standing, sitting, lying, walking, walking downstairs and walking upstairs.

WISDM-2010. Another dataset formed by smartphone data is the Wireless Sensor Data Mining (WISDM) dataset (KWAPISZ; WEISS; MOORE, 2010) which comprises 3-axis accelerometer data collected at 20 Hz from 36 users while performing six activities: standing, sitting, jogging, walking, walking downstairs and walking upstairs.

WISDM-2019. More recently, a new version (WEISS; YONEDA; HAYAJNEH, 2019) of the WISDM dataset was published by the same research group. In the newer version, data were collected by a smartwatch, in addition to the smartphone, from 51 participants during 18 physical activities performed in everyday life. The list of activities comprises from basic activities to more complex such as eating, drinking, and typing. Each of the activities was performed separately for approximately 3 minutes, and the data acquisition was realized at a rate of 20 Hz. In the literature, the nomenclature WISDM is found referring to the same dataset, therefore, to avoid misunderstanding, in this work WISDM-2010 is adopted for the first version, and WISDM-2019 for the latest version.

PAMAP2. The PAMAP2 (Physical Activity Monitoring for Aging People 2) dataset (REISS; STRICKER, 2012b; REISS; STRICKER, 2012a) is composed of data from three inertial measurement units (IMUs) (at 100 Hz) placed on wrist, chest, and ankle, and a heart-rate-monitor (at 9 Hz) collected from nine subjects performing 18 activities, from basic to complex ones such as watching TV, vacuum cleaning, and rope jumping.

MHEALTH. The MHEALTH dataset (BANOS et al., 2014b) consists of data (3-axis acceleration, angular velocity, magnetic field orientation, and electrocardiogram) collected at 50 Hz from 10 volunteers by using three wearable sensors placed on the subject's chest, right wrist, and left ankle, while performing the following activities: standing still, sitting and relaxing, lying down, walking, climbing stairs, bending waist forward, frontal elevation of arms, bending knees, cycling, jogging, running, and jumping front and back.

Opportunity. This dataset is composed of activities performed in a home environment using multiple wearable, object, and ambient sensors. They collected, from four subjects, 17 morning activity data such as preparing a sandwich, drinking coffee, and opening then closing the fridge (ROGGEN et al., 2010; CHAVARRIAGA et al., 2013).

Skoda. This dataset comprises of 3-axis accelerometer data collected at 64 Hz for about three hours from a car maintenance assembly-line workers performing activities of ten manipulative gestures, and wearing sensors on right and left arms (ZAPPI et al., 2008).

3.2 Traditional Machine Learning Approach

Several studies have investigated different handcrafted features as well as feature selection techniques (JANIDARMIAN et al., 2017). Among the handcrafted features traditionally used in HAR systems, time domain features are more often used and include statistical features (i.e., mean, variance, skewness), or other simple features such as maximum and minimum values in the data window (CRUCIANI et al., 2019; MORALES; AKOPIAN, 2017).

Traditionally, handcrafted features are widely employed in HAR, and its classification methods often rely on supervised machine learning approaches such as Hidden Markov Model (LEE; CHO, 2011; SOK et al., 2018; RONAO; CHO, 2017), Naïve Bayes (NB) (LONG; YIN; AARTS, 2009), K-Nearest Neighbors (kNN) (FOERSTER; SMEJA; FAHRENBERG, 1999), Decision Tree (FAN; WANG; WANG, 2013), Random Forest (RF) (UDDIN; BILLAH; HOSSAIN, 2016; HU et al., 2018), Support Vector Machine (SVM) (CHEN et al., 2017; RAVI et al., 2005), shallow Artificial Neural Networks (LEE; CHO, 2011), and, more recently, eXtreme Gradient Boosting (XGBoost) (AYUMI, 2016; VONG et al., 2021).

Still in the first decade of this century, Ravi et al. (2005) developed an HAR system based on 3-axial acceleration data to classify between the following activities: standing, walking, running, climbing upstairs, climbing downstairs, sit-ups, vacuuming and brushing teeth. They used their own dataset collected from two subjects in different days, with a sensor placed near the pelvic region acquiring data at 50 Hz. They used a data window of 5.12 s with 50% of overlapping and extract mean, standard deviation, energy and correlation as features. An accuracy of 99.82% (with a plurality voting method) was obtained when using 10-fold cross-validation with all data shuffled, while a maximum of only 73.33% (with boosted SVM) when training and testing with different subjects. It is notable that using a shuffled dataset that has samples with 50% overlap contributed to the high accuracy in the former case, because it is highly likely that subsequent samples (and therefore 50% equal) will be in the training and testing sets.

In (LONG; YIN; AARTS, 2009), the authors used NB to classify between five activities in an uncontrolled environment with the sensor placed on the user's waist. They used a dataset constructed with a 3-axial accelerometer collected at 20 Hz and with a 16-s window from 24 subjects. An accuracy of 79.3% was achieved using five PCA components derived from 19 features from time, frequency and space domains, and the NB classifier. According to the authors, cycling had the lowest accuracy (about 50%), which can be explained by the less representative acceleration generated when a sensor is placed on the waist. The accuracy obtained in this work was relatively low, however, the fact that it was carried out with data collected in an uncontrolled environment should be highlighted. However, the main drawback of this system is the use of a 16-s window, which generates a large volume of data to be processed, and introduces delay in the response time.

Lee et al. (2011) developed a hierarchical two-level classifier system based on neural networks to classify between seven activities. In the first classification level, they used mean, standard deviation, spectral entropy, and correlation among three axes to classify between static and dynamic activities. Then, in the second level, they extracted auto-regressive coefficients, signal magnitude area and tilt angles, which were further processed by LDA, to classify between the activities. The dataset was formed by data collected at 20 Hz from 10 subjects using a 3-axial accelerometer placed on their chest. The use of a two-level hierarchical classifier is an interesting approach for classifying human activities, given the different nature between static and dynamic activities. However, the authors used the same features for both types of activities, while they could have adopted simpler features that carry posture characteristics for static activities.

Fan, Wang and Wang (2013) developed a Decision Tree to classify five activities (standing still, walking, running, climbing stairs and descending stairs), using eight parameters from the time domain and two from the frequency domain. These parameters are extracted from 3-axial accelerometer data collected with a smartphone carried in different positions (bag, trousers and hand). The authors obtained 80.29% of accuracy for activity recognition, and 61.31% for smartphone position identification, which would be needed as a first stage.

Uddin, Billah and Hossain (2016) used importance score driven to select best features for random forests to recognize activities and postural transitions on the smartphone. They reached 100% recognition accuracy on a benchmark dataset and indicated that features created from accelerometer signals are more relevant for recognition than features created from gyroscope signals.

A smartwatch-based HAR system (BALLI; SAğBAş; PEKER, 2019) was proposed for classifying daily activities such as brushing teeth, walking, running, and vacuuming. Authors used the Moto 36012 smartwatch for collecting data and evaluated recognition rate performance of different models. The PCA-Random Forest was the winning model, as it achieved 98% accuracy and F1-score of 98.1% on a local dataset.

In other work (VONG et al., 2021), an XGBoost classifier combined with the mutual information method reached the highest accuracy value of 91.22% amongst five other classifiers and three other feature selection methods on UniMiB SHAR dataset (MICUCCI; MOBILIO; NAPOLETANO, 2017). Authors found that RF and XGBoost classifiers resulted in generally higher scores. Meanwhile, the KNN, DT, NB and SVM classifiers are simpler classification methods, consequently, resulting in lower scores compared to the RF and XGBoost classifiers.

Another HAR research proposed a smartwatch-based HAR system with a two-level classifier (KONGSIL; SUKSAWATCHON; SUKSAWATCHON, 2020). First, they used a simple SVM to infer if the activity is energetic or dormant. In the second level, the energetic activities model was built by using 30 features from both time and frequency domain, and a SVM for classification. The dormant activities classifier is simpler, since they only need accelerometer data to recognize the activity based on sensor position and orientation. They used six statistical features and a Naïve Bayes for the classification task. They achieved 90.62% of accuracy on the MHEATLH dataset using only the wrist-worn sensor, and 87.41% on the WISDM-2019 dataset using only the smartwatch data.

Despite the good performance found in most of the previous works cited above as well as shown in other comparative studies and reviews (ATTAL et al., 2015; GAO; BOURKE; NELSON, 2014; LARA; LABRADOR, 2013), handcrafted features have some drawbacks, as discussed below. Firstly, they heavily relies on human experience or domain knowledge and only shallow features can be learned according to human expertise (YANG et al., 2015). Therefore, it is highly dependent on experts in the context. Moreover, those shallow features often refer to some statistical information and in general are used to recognize low-level activities like walking or running, as they are hardly able to infer high-level or context-aware activities (YANG, 2009), such as eating, brushing teeth, sweeping the floor etc.

Additionally, traditional pattern recognition approach often focus on static data, and require a large amount of well-labeled data to train the model, while most of the real world applications data are unlabeled and come in stream, requiring robust online and incremental learning (WANG et al., 2018; DANG et al., 2020). Furthermore, differences in behavioral habits, gender and age influence in motion patterns of people, which vary greatly among different individuals, making it even more difficult for recognizing similar activities by using the conventional ML approach. The recognition accuracy tends to be limited due to confusing activities which generate similar motion signals (ZHU et al., 2019b).

Compared to traditional ML approach, deep learning models considerably reduces the effort of selecting the best features by automatically extracting abstract features through several hidden layers, achieving much more high-level and representative features. In addition, deep learning can work well with unsupervised learning (SEYFIOğLU; ÖZBAY-OğLU; GüRBüZ, 2018; NGUYEN et al., 2019) and Reinforcement Learning (IJJINA; CHALAVADI, 2017). Therefore, an increasingly number of deep learning-based HAR frameworks has been proposed recently (DANG et al., 2020).

3.3 Deep Learning-Based Approaches

Deep learning applications have developed rapidly in recent years, achieving singular performance in many areas such as computer vision (WEI et al., 2018) and natural language processing (LE-HONG; LE, 2018). To automate the complicated feature extraction process, various types of deep neural networks have been studied in the literature to recognize human activity from wearable sensor data, as will be presented in this section. Most of these approaches directly employ the collected raw sensor data for automated feature extraction using deep neural network, such as convolutional neural network (CNN), recurrent neural network (RNN), and hybrid network (MAHMUD et al., 2020). The following subsections present and discuss wearable based-HAR researches based on deep learning techniques.

3.3.1 Convolutional Neural Networks

Convolutional neural networks (CNNs) are expert in capturing local connections of multimodal sensory data, and the translational invariance introduced by locality leads to accurate recognition (CHEN et al., 2020; HAMMERLA; HALLORAN; PLöTZ, 2016). When applied to time series classification like HAR, CNN has two advantages over other models: it captures local dependency and it is scale invariant. Local dependency means the nearby signals in HAR are likely to be correlated, while scale invariance refers to the scale-invariant for different paces or frequencies (WANG et al., 2018).

One can take advantage of these CNN features in two ways: (i) using one-dimensional convolutional neural networks (1D-CNN) to extract features directly from the raw motion signals (acceleration, angular velocity) collected from wearable sensors; and (ii) employing a transformation to convert the one-dimensional motion signal into images (in general that carry time and frequency information) and then using two-dimensional convolutional neural networks (2D-CNN).

Zeng et al. (2014) used one uni-dimensional convolutional and two fully-connected layers (with 1024 and 30 neurons) to classify activities from Skoda, and Opportunity datasets. They achieved accuracy of 88.19%, and 76.83% on the former, and the latter, respectively, which were 4.41%, and 1.2% higher than the second best algorithm - PCA based on empirical cumulative distribution function. They showed that CNN outperformed the state-of-the-art approaches at that time, and the confusion matrices showed that CNN was able to better separate walking, walking upstairs and walking downstairs activities, indicating that it extracted more representative features for these activities.

Zebin, Scully and Ozanyan (2016) proposed a 3-layer 1D-CNN for classifying six activities from a local dataset. They reached accuracy of 97.01%, outperforming other models evaluated (96.7% with SVM and 91.7% with MLP), and also achieved lower computational cost than the traditional approaches. Furthermore, they showed that increasing the number of convolutional layers increases the computational load, but there is a significant improvement in performance when a third layer is added. Tuning the filter and pooling sizes revealed that wider filter size and lower pooling size architecture improves the recognition performance.

Ignatov (2018) proposed a CNN composed of a single layer with 196 filters of size 1x16, and a fully connected layer with 1024 neurons, which adds statistical features to the convolutional layer output. They achieved an accuracy of 93.32% on the WISDM-2010 dataset and 97.63% on UCI-HAR dataset, reaching a significant higher accuracy than other approaches in both datasets. According to the author, due to a relatively shallow architecture, the proposed algorithm has a short running time and can be efficiently executed on mobile devices in real time.

Another work (WAN et al., 2020) proposed a 3-layer 1D-CNN with 64 filters in each one and fully connected layer with 512 neurons. They achieved accuracy of 93.21% on UCI-HAR dataset, and 91.00% on WISDM-2019 dataset. Authors stated that the proposed CNN provides a way to automatically and data-adaptively extract relevant and robust features without the need for advanced preprocessing or time-consuming handcrafted feature extraction.

Cruciani et al. (2019) evaluated a 3-layer 1D-CNN against a shallow neural network with one dense layer with 64 neurons and the output layer, using the UCI-HAR dataset. For the shallow network a set of 348 accelerometer features was fed to the network in the first test and 561 accelerometer and gyroscope features in the second test, reaching 89.95% and 95.80% of accuracy, respectively. The CNN achieved 90.55% when using only raw acceleration data and 93.31% with accelerometer and gyroscope data. Even having been outperformed by the approach with handcrafted feature extraction in the second test, the CNN proved to be extremely efficient, as extracting 561 features in computational terms would be extremely costly, especially when considering wearable sensors.

In other research (WANG; HE; ZHANG, 2019), an attention-based HAR with five convolutional layers (with 32, 64 and the last three with 128 filters) was proposed and achieved 93.41% of accuracy on the UCI-HAR dataset. In their proposal, the attention sub-modules compute the compatibility between global feature and local feature for generating the weighted feature map. They demonstrated that the model can achieve significant accuracy improvement on weakly labeled dataset, compared to other deep learning approaches.

Jiang and Yin (2015) applied the Short-time Discrete Fourier Transform (STDFT) to temporal acceleration and angular velocity signals and constructed a time-frequency-spectral image. Then, a 2D-CNN was used to handle the activity image for recognizing simple daily activities from the UCI-HAR dataset, reaching 97.59% of accuracy. The authors achieved an extremely high accuracy with this technique, however an additional step is required to transform the sensor signals into an image, increasing the computational cost, making it difficult to deploy in wearable sensors.

Another 2D-CNN was proposed by Ha, Yun and Choin (2016), in which they transform the input one-dimensional acceleration and angular velocity signals into an activity image vertically concatenating the input data forming a matrix. They stated that the method captures spatial dependency over sensors and axes as well as local dependency over time at the same time. They reached accuracy of 98.30% on MHEALTH dataset (BANOS et al., 2014b) and 97.92% on Skoda dataset.

3.3.2 Recurrent Neural Networks

Recurrent Neural Networks (RNNs) are neural networks that contain cyclic connections, which enable them to learn the temporal dynamics of sequential data (MURAD; PYUN, 2017). This characteristic makes RNN-based models a good approach for dealing with time series data such as acceleration and angular velocity collected by wearable devices.

Murad and Pyun (2017) proposed a unidirectional LSTM model consisting of four layers, which achieved accuracy of 96.7%, 92.5%, and 92.6%, respectively on UCI-HAR, Opportunity, and Skoda datasets. The authors concluded that the performance results of the proposed model clearly demonstrate that deep RNN-based models are very suitable for HAR, and effective for a broad range of activity recognition tasks, since the datasets evaluated are very diverse.

An ensemble of multiple deep LSTM learners was proposed by Guan and Plötz

(2017), demonstrating superior performance to individual networks on three benchmark datasets. They achieved 72.6%, 85.4%, and 92.4%, respectively on Opportunity, PAMAP2 and Skoda datasets.

Edel and Köppe (2016) proposed a Bi-LSTM model, in which the forward and backward LSTM hidden layers are fully connected to the input layer and consist on five LSTM units each with full recurrent connections. They achieved 78% of accuracy on Opportunity dataset and 93% on PAMAP2 dataset. In another bidirectional LSTM proposal, (ALJARRAH; ALI, 2019) used a PCA-Bi-LSTM, which consists of two stages. The first is a standard PCA method to reduce the number of features, and the second is a Bi-LSTM neural network with one bidirectional and one unidirectional LSTM layers, both with 150 units. They achieved 97.64% of accuracy on MHEALTH dataset.

Moreover, Sun et al. (2019) proposed a single layer LSTM with attention mechanism, which can automatically focus and capture the most important temporal dependencies from the input data, without using extra handcrafted features and human domain knowledge. They reached 90.9% of accuracy on Opportunity dataset.

More recently, Tufek et al. (2020) achieved state-of-the-art accuracy on the UCI-HAR dataset, obtaining 97.4% of accuracy, with a 3-layer LSTM model composed of 32 cells each. According to the authors, it was demonstrated that 3-layer LSTM model is the best solution for sensor-based activity recognition problems in real time applications. Additionally, KNN may not be an efficient option for classification of activities for big datasets because of lower accuracy rate and computational cost.

3.3.3 Hybrid Techniques

Multiple researches in the literature propose a combination of CNN and LSTM (ORDóñEZ; ROGGEN, 2016; SU et al., 2019; SINGH et al., 2021), by taking advantage of CNN for capturing the spatial domain information and LSTM for temporal domain information.

Ordóñez and Roggen (2016) proposed the DeepConvLSTM deep learning framework, composed of four consecutive CNN layers, to extract local contextual features, and two LSTM layers, for modelling the temporal dependencies between their activation. They obtained 93% of accuracy on Opportunity dataset.

In other research (SU et al., 2019), the authors proposed a Hierarchical Deep Learning, which is composed of a 3-layer Deep Bidirectional LSTM (Bi-LSTM) model and a 2-layer CNN model. They achieved accuracy of 97.95% and F1-score of 97.27% on the UCI-HAR dataset.

In (MEKRUKSAVANICH; JITPATTANAKUL, 2020), the authors used only smartwatch data from the WISDM-2019 dataset. They proposed a hybrid CNN-LSTM architecture for the improvement of recognition performance, and a Bayesian optimization technique to find the best model hyperparameters. The model comprises two convolutional layers and a single LSTM layer, and outperformed all of the other networks evaluated (a single layer CNN and a non very well described LSTM) with an accuracy of 96.2% and an F1-score of 96.3%.

Singh et al. (2021) proposed a DeepConvLSTM with self-attention that not only captures the spatio-temporal features of multiple sensor time-series data but also learns important time points by utilizing a self-attention mechanism. They compared their approach with the baseline DeepConvLSTM and outperformed it in all dataset evaluated. They achieved 94.86% and 90.41% of accuracy, respectively, on MHEALTH and WISDM-2010 datasets, while reached 93.80% and 89.83% with the baseline DeepConvLSTM.

Most of state-of-the-art deep learning approaches, like the ones explored in this section, use complex networks, with many layers and many weights for reaching great accuracy, however, they are not suitable for wearable devices and smartwatches. How can one take advantage of the main gains offered by deep learning-based models, without compromising the performance and autonomy required by resource constrained wearable activity trackers? The next section explores some alternatives found in the literature.

3.4 Efficient Design for Wearable Devices

This section presents strategies to develop models based on deep learning for Human Activity Recognition on edge computing devices or wearable devices. In addition, other techniques that contribute to a lighter and more efficient system in terms of computational cost are also discussed.

3.4.1 Lightweight Deep Learning Models

Although deep learning models have shown great performance in sensor-based HAR systems, they are typically resource-intensive. For non-portable applications, Graphic Processing Units (GPUs) are usually leveraged to accelerate computation. However, GPUs are very expensive and power-hungry so that they are not suitable for real-time applications on mobile devices. Moreover, current research has demonstrated that making a neural network deeper by introducing additional layers and nodes is a critical approach for improving the model performance, which inevitably increases computational complexity (CHEN et al., 2020). Therefore, it is crucial to solve the concern of high computational cost to deploy real-time and reliable HAR systems on wearable devices using deep learning models. Recently, HAR researchers have focused on optimized and lightweight systems, and some of these works are presented below.

Edel and Köppe (2016) proposed a binarized–Bi-LSTM–RNN model, in which the weight parameters (input and output of all hidden layers) are binary values. The model is especially suitable for resource-constrained environments since it replaces either floating or fixed-point arithmetic with significantly more efficient bitwise operations. The main line of RNN based HAR models is dealing with resource-constrained environments while still achieving good performance.

A Hybrid Neural Network (HNN) was proposed by Vita et al. (2020). The HNN takes advantage of the low complexity of Binarized Neural Network (BNN) (COURBARIAUX et al., 2016), while applies non-binarized output activation to some layers in order to preserve the accuracy compared to a full floating-point implementation. The proposed design was compared to (JAFARI et al., 2019), in which a CNN for multimodal data classification is proposed, and obtained an accuracy slightly lower (97.5% against 98.0%), but achieved a much lower power consumption (6.3 µW when the sensor output data rate is 25 Hz).

Tang et al. (2021) proposed a lightweight CNN using Lego filters for HAR. They used a set of lower-dimensional filters as Lego bricks to be stacked for conventional filters. They showed that the novel Lego CNN with local loss can greatly reduce memory and computation cost over CNN, while achieving high accuracy (96.90% on UCI-HAR, 93.50% on PAMAP2, 98.82% on WISDM-2010, and 88.09% on Opportunity datasets). The model was implemented on a Huawei Honor 20i smartwatch for practical computational cost performance comparison, and they concluded that the Lego CNN with 4x theoretical complexity reduction results in a 1.7x actual speedup in the implementation.

Rashid et al. (2022) proposed a low power Adaptive CNN for HAR which can use two or just one convolutional layer according to a statistical-based decision block. For most of the samples only one layer was needed, reducing model complexity and making it more suitable for resource constrained devices. They evaluated energy and memory consumption of their proposed architecture in comparison with the CNN proposed in (ZHU; SAN-SEGUNDO; PARDO, 2017) on the EFM32 Giant Gecko microcontroller, and achieved a memory reduction by 20 times and processing time by almost 500 times.

3.4.2 Optimization of System Parameters

The activity recognition system proposed by Zheng et al. (2017) achieved an energy-efficient system by using low sampling rates while obtaining a high accuracy on a smartphone-based local dataset with accelerometer and barometer data. They showed the accuracy has only improved slightly with the increase of the sampling rate from 1 Hz to 50 Hz (96.2% with 1 Hz, 97.2% with 5 Hz, 97.6% with 10 Hz, and 98.0% with 50 Hz), while the system consumes low energy by using a sampling rate of 1 Hz compared to 5 Hz and 50 Hz, saving 17.3% and 59.6% of energy, respectively.

The varying sampling frequency and window size for HAR were evaluated in terms of energy efficiency and accuracy by Lee and Kim (2016). They used an adaptive SVM-based HAR system, in which the sampling rate and window size are selected according to the type of activity being performed (still or moving) and the characteristic of the activity (short or long duration). Thus, in many cases they can use low sampling rate and large window size, reducing energy consumption. The strategy resulted in an energy reduction of a minimum of 44.23% and a maximum of 78.85%, while achieving a competitive classification accuracy of 92.17%, close to their baseline SVM (about 95%).

Another approach to meet the requirements of low energy consumption is data compression. Shi et al. (2016) used the integral of the modulus of acceleration (IMA) for feature extraction and SVM for classification. Authors showed that the classification task effectively executed (accuracy $> 90\%$) with a sampling frequency of 0.39 Hz and a bit resolution of 5 bits. According to the authors, this configuration reduced the amount of generated data by three orders of magnitude without compromising the classification performance. They demonstrated that embedded machine learning has the potential to reduce the communication and processing cost by several orders of magnitude and, thus, substantially extend the battery life of wearable sensors.

In (JEONG; KIM, 2019), a segment-level change detection, based on signal area magnitude threshold, is proposed to determine activity change with low computational complexity. Following the detection of activity change, a fully connected model is applied to classify the new activity. They achieved accuracy slightly lower than their baseline CNN model (93.80% versus 94.64%), however, the significant reduction of 90% in the number of model parameters confirmed the proposed approach is good in terms of computational efficiency and power consumption (6.5 time less energy).

Zebin et al. (2019) adopted the model quantization strategy for achieving model size and execution time reduction. They used per-channel quantization of weights and per-layer quantization of activation to 8 bits of precision, and reached 87.60%, 93.51%, and 93.60% of accuracy using dense neural network, LSTM, and CNN, respectively, while the floating-point models obtained 86.55%, 92.20%, and 96.40%. The model size was reduced by 5.98, 4.24, and 7.62 times for the three models evaluated.

An activity-aware 2-level classifier was proposed by Bhat et al. (2019) to enable power and energy savings. The first level classifier uses simple statistical features and a SVM classifier to infer if the activity is static or dynamic. For static activities, a relatively simple Decision Tree is used, while for the dynamic ones, a two-layer dense neural network is used to maintain high accuracy and facilitate future online learning. The 2-level classifier resulted in reduced power consumption by up to two orders of magnitude compared to conventional HAR systems. They achieved accuracy of 95% for the locomotion activities in the Opportunity dataset.

4 Energy-Efficient Human Activity Recognition Framework

This chapter presents the proposed energy-efficient HAR framework, the methodology adopted for the development, testing and evaluation of the models, and the reference models used for the comparative study.

4.1 Proposed Framework

The energy-efficient framework for HAR proposed in this work was developed for resource-constrained devices, therefore, contributes greatly to the deployment of HAR systems on smartwatches or other wearable devices. The following characteristics of the proposed framework are based on the study of the literature discussed in Section 3.4, and also on analysis and tests carried out throughout this doctoral research (COELHO et al., 2018; COELHO et al., 2019; COELHO et al., 2022; COELHO et al., 2021): (i) using a reduced volume of input data; (ii) reading data at low sampling frequency; (iii) using a two-level classifier; (iv) employing a light and fast classifier for static activities; and (v) using a memory and code-optimized compact CNN for the dynamic activities classifier.

It is hypothesized that, with such characteristics, the proposed framework will be suitable for running on wearable devices, in terms of computational cost, memory usage, and battery consumption, as well as with capacity to achieve satisfactory performance in terms of activity recognition rate. Next, the features of the framework are briefly described, and will be further discussed in this and the following chapters.

Reduced volume of input data. A more efficient data processing can be achieved by reducing the input data volume and the number of sensors employed. This optimization may reduce memory demand, computational cost and, consequently, energy consumption.

Low sampling rate. Reading data at a reduced sampling frequency directly impacts energy consumption, as the device can remain in a low-power sleep mode for longer, and storage capacity, since the input data window will be formed by fewer data points.

Two-level classifier. The use of a two-level classifier is a key feature of the proposed HAR framework. This first classification level has a quite simple classifier whose function is to infer whether the activity is static or dynamic (see Subsection 2.1.1 for definition). The second classification level has two classifiers, one for static activities and another one for the dynamic activities, that work according to the output of the first level, and never simultaneously. Thus, a lightweight, fast, and low-power classifier can be used for recognizing simpler activities, such as the static ones. Meanwhile, the dynamic activities classifier, which is more computationally costly and more energy-expensive, will only be requested when the user is performing dynamic activities, remaining off otherwise.

Light and fast classifier for static activities. Using a specific classifier for static activities allows it to be much simpler and lighter than a generic classifier and to achieve superior performance for such activities.

Memory and code-optimized compact CNN for dynamic activities. By using a compact CNN, one can take advantage of its ability to extract significant features from dynamic activity patterns, and at the same time deploy it on a resource-constrained device. To ensure efficiency, the model must be compressed and optimized for the target hardware.

The proposed energy-efficient HAR framework, shown in Fig. 18, is composed of the following modules: (i) Data Acquisition; (ii) First Level Classification; (iii) Static Activities Classification; and (iv) Dynamic Activities Classification. The framework was developed for the STM32 family of 32-bit microcontrollers based on Arm Cortex-M from STMicroelectronics (STMICROELECTRONICS, 2021a), and the integrated development environment STM32CubeIDE was used for development and implementation of the framework in embedded C code.

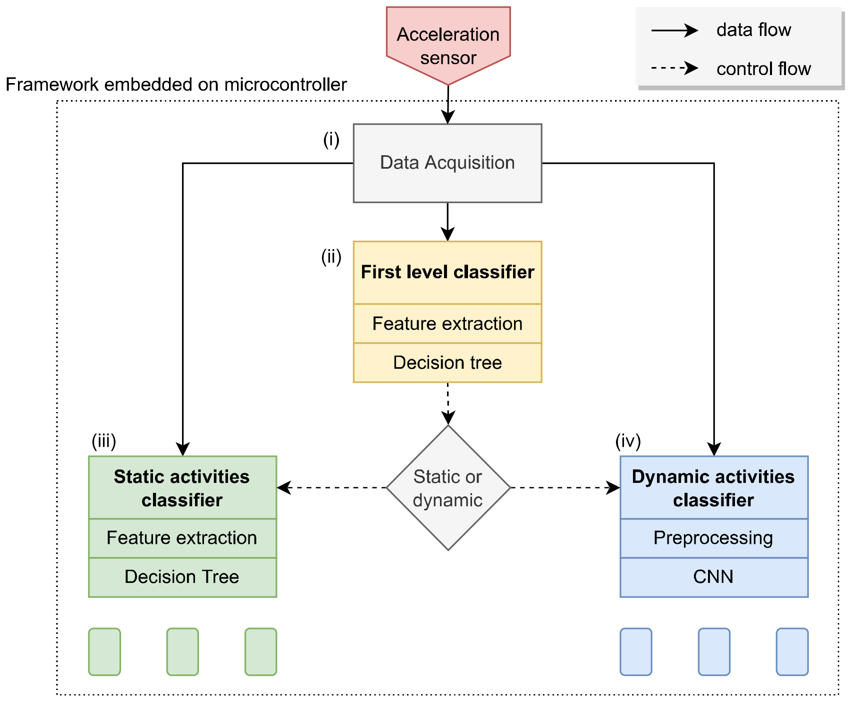


Figure 18 – Proposed energy-efficient HAR framework.

The following subsections in this chapter present each one of the modules that compose the framework, and addresses the signal processing steps and the development of each classifier, and Chapter 5 covers the framework firmware implementation and deployment on microcontroller.

4.1.1 Data Acquisition

The Data Acquisition module (i) (Fig. 18) reads the input data from wearable sensors at a predefined sampling rate. The framework was built to work with a 2-s data window, which is formed by N samples, according to the sampling rate set. Equations 4.1, 4.2, and 4.3 represent the input signals read from a 3-axis accelerometer. As an example, for a sampling rate of 50 Hz, N would be 100, whereas for sampling rate of 5 Hz it would be 10. The difference between four 2-s input data windows of acceleration data collected at different sampling rates at the same interval is shown in Fig. 19. Reducing the sampling frequency significantly reduces computational cost and power consumption, as discussed later.

$$\bar{x} = \begin{bmatrix} x_0 & x_1 & x_2 & \cdots & x_{N-1} \end{bmatrix} \tag{4.1}$$

$$\bar{y} = \begin{bmatrix} y_0 & y_1 & y_2 & \cdots & y_{N-1} \end{bmatrix} \tag{4.2}$$

$$\bar{z} = \begin{bmatrix} z_0 & z_1 & x_2 & \cdots & z_{N-1} \end{bmatrix} \tag{4.3}$$

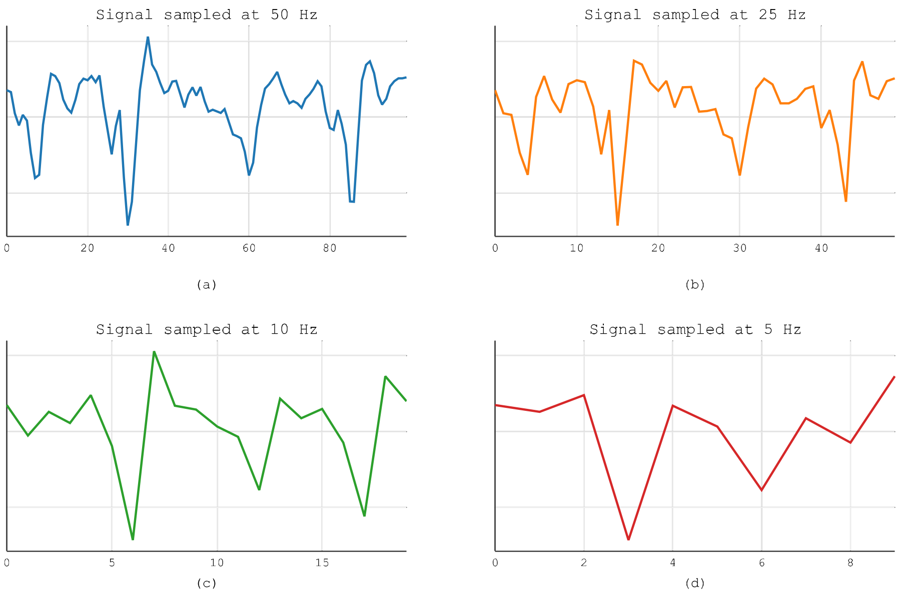


Figure 19 – Example of a 2-s window of x-axis acceleration from wrist-worn device extracted from MHEALTH dataset: (a) sampled at 50 Hz; (b) sampled at 25 Hz; (c) sampled at 10 Hz; (d) sampled at 5 Hz.

Regarding the segmentation process used in datasets, in HAR systems, usually the data sliding window segmentation process is applied to divide the time series signal into consecutive windows of fixed length, which may or may not overlap. As presented in Subsection 2.2.3, choosing the optimal window size must take into account several factors. In this research, the window size of 2 s was adopted, with 50% overlap between subsequent windows. It is understood that this window size is short enough to reduce the computational cost, while having a suitable length to comprise sufficient information and patterns for a HAR system with a wide range of activities.

As stated before in this subsection, the number of data points in the input data window depends on the sampling rate. For the datasets used in this work, the 2-s segmented input data window has 100, 200 and 40 data points, respectively for the MHEALTH, PAMAP2, and WISDM-2019 datasets, according to their signal sampling rate. Equation 4.4 shows a generic segmented input data window with N data points, where $\bar{a}$ is the input data array, and a_i represents each data point.

$$\bar{a} = \begin{bmatrix} a_0 & a_1 & a_2 & \cdots & a_{N-1} \end{bmatrix} \tag{4.4}$$

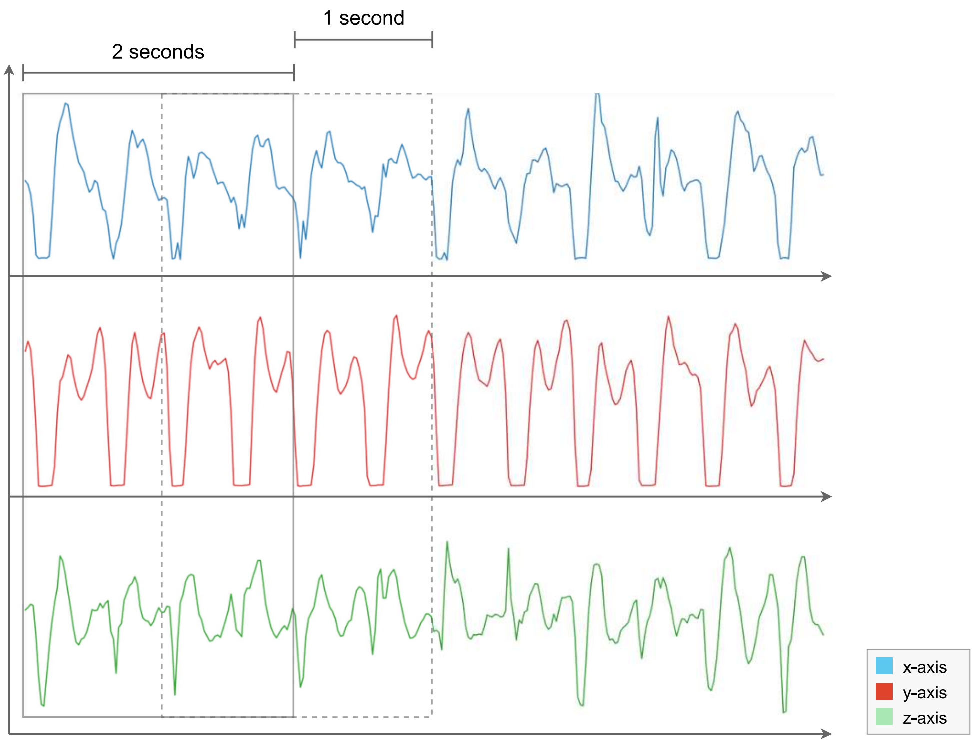


Figure 20 – Sliding window segmentation of 2 s with 50% overlap.

The segmentation process is applied to each time series signal in the dataset, separately for each subject and for each activity. Each one of the three axes (x, y, z) signal from each sensor (accelerometer and gyroscope) and from each device (placed on chest, wrist and ankle) is a particular time series signal, and therefore is segmented separately.

In addition to applying the segmentation with the original sampling rate of each dataset, it was also performed with lower rates, in order to allow the tests and analysis with low sampling rate carried out in this research. Therefore, the signals were downsampled to the following sampling rate: 25, 10, and 5 Hz for the MHEALTH dataset; 50, 25, 10, and 5 Hz for the PAMAP2 dataset; and finally 10, and 5 Hz for the WISDM-2019 dataset.

The downsampling process applied here consists in reducing the sampling rate by simply dropping data points, and in this way, one can simulate the data collection at different sampling rates. As an example, a 2-s input data array collected at 20 Hz ($N = 40$ data points, Equation 4.5) would be formed as in Equation 4.6, with 10 data points, when downsampled to 5 Hz.

$$\bar{a} = \begin{bmatrix} a_0 & a_1 & a_2 & a_3 & \cdots & a_{38} & a_{39} \end{bmatrix} \tag{4.5}$$

$$\bar{a}_{\downarrow 4} = \begin{bmatrix} a_0 & a_4 & a_8 & a_{12} & \cdots & a_{32} & a_{36} \end{bmatrix} \tag{4.6}$$

4.1.2 Feature Extraction

In the proposed HAR framework, feature extraction is needed for the first level and the static activities classifiers, in order to create the feature vector that is inputted to the respective Decision Tree for inference. The initial selection of features was based on the study of the state-of-the-art researches, as well as on the computational cost to calculate them.

Frequency domain features depend on the application of Fast Fourier Transform (FFT), which would be costly for the light classifiers targeted in this work. In the time domain, statistical features that require many exponential calculations, such as skewness and kurtosis, were discarded.

Finally, the following five simple, yet efficient, statistical features are extracted from the segmented input data window: (i) mean (Equation 4.7), (ii) standard deviation (Equation 4.8), (iii) maximum amplitude (Equation 4.9), (iv) minimum amplitude (Equation 4.10), and (v) signal magnitude area (SMA) (Equation 4.11). In the following equations, $\bar{a}$ refers to each input data window, while a_i represents each data point from the window, and N the number of data points.

$$\mu_a = \frac{1}{N} \sum_{i=0}^{N-1} a_i \tag{4.7}$$

$$\sigma_a = \sqrt{\frac{1}{N} \sum_{i=0}^{N-1} (a_i - \mu_a)^2} \tag{4.8}$$

$$max_a = max(\bar{a}) \tag{4.9}$$

$$min_a = min(\bar{a}) \tag{4.10}$$

$$SMA_a = \sum_{i=0}^{N-1} |a_i| \tag{4.11}$$

These features are extracted from each time series signal of the dataset, forming the set of features that is used for training and testing the Decision Tree models of the first level classifier and the static activities classifier. For a scenario with three wearable devices with accelerometer and gyroscope sensors each, the initial feature vector would have 90 features ($3\,devices \times 6\,signals \times 5\,features$) for each input data window. This feature vector is optimized by the feature selection process described in Subsection 4.3.2.3.

4.1.3 First Level Module

The second module (ii) that composes the framework is the First Level module (Fig. 18), which is responsible for classifying the input data as static or dynamic activity. This module is the only classifier module that is executed continuously, while the other two switch according to the activity type identified in the first level. However, this is the lightest and fastest one among the three classifiers, and offers many benefits to the system as a whole. One advantage of using a first level of classification is to be able to separate static activities from dynamic ones and use a much simpler and more suitable model to classify them, but with high performance, in terms of accuracy.

The First Level module is comprised of two tasks: feature extraction and classification. Firstly, a very small set of statistical features (often one or two features are sufficient) are extracted from the input signal, then they are applied to the classifier, a lightweight Decision Tree. Finally, its output (static or dynamic activity) determines which module will be active at the next classification level, while the other will remain off, reducing power consumption, especially when the Dynamic Activities module is not running.

Given that static and dynamic activities have a different nature (non-variable and variable signal, respectively), as shown in Fig. 21, only one measure of acceleration variability could be sufficient for determining a classification rule. However, to guarantee an accurate classification in different datasets and environments, the model tries more features, not exceeding three, to ensure fast processing.

Since this step is relatively simple, it is a straightforward classification task, and may be performed efficiently by a light classifier algorithm as a shallow Decision Tree. The tree adopted in this framework has a max depth between 1 and 5, uses Gini impurity

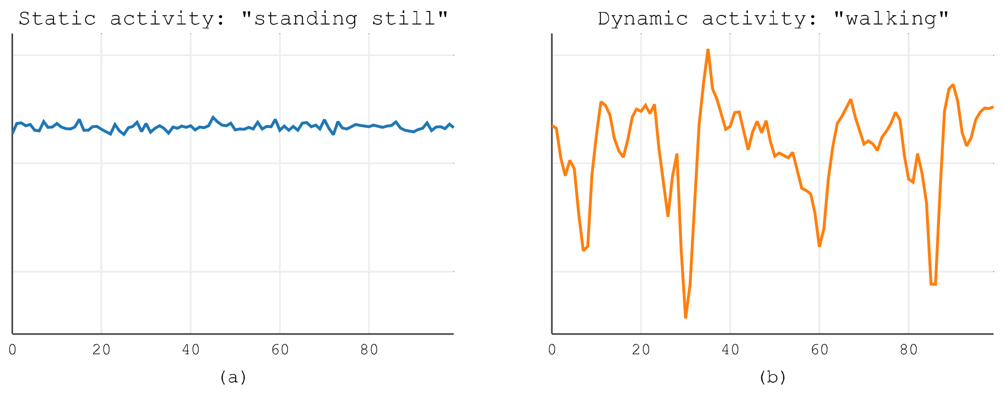


Figure 21 – Waveform difference between static and dynamic activities from MHEALTH dataset: (a) static activity (standing); (b) dynamic activity (walking).

or entropy for the split criterion, and has a maximum of three features as input. The mentioned hyperparameters are tuned and the features are selected for best performance, as will be further described in Subsection 4.3.2.2.

As described in Subsection 2.2.8.1, Decision Trees are invariant to monotonic transformations, therefore, there is no need to normalize or re-scale the feature vector before inputting it to the classifier. This characteristic makes the Decision Trees very efficient in resource-constrained devices, since they can work with integer data read directly from the sensor, avoiding a floating-point calculation from normalization step and making operations simpler.

4.1.4 Static Activities Module

The classification of static activities is performed in the module Static Activities (iii) of the proposed HAR framework (Fig. 18). This module is activated when the First Level Classifier module detects a static activity. Otherwise, it remains off. The main purpose of building a specific classifier for static activities is to be able to use a simple model with low computational cost, since this type of activity does not require a complex model. In addition, separating such activities from the others also allows the dynamic activities classifier to achieve better performance, since it will have fewer classes to handle.

The Static Activities module is composed of two tasks: feature extraction, and classification using a Decision Tree. First, a small set of statistic features are extracted from the input data window. Then, the feature vector is fed into the Decision Tree for classification between static activities (often standing, sitting, and lying).

For static activities recognition, features that provide waveform pattern or signal variability information are not of much interest, as the user is not moving, which means the acceleration variance is close to zero. In addition, static activities usually comprise only the activities standing still, sitting, and lying down, forming a group with small variation in the target, even in different datasets.

On the other hand, accurate information about the individual's posture is essential for a good performance, unlike the case of dynamic activities, where one looks for patterns in acceleration signal caused by the sequence of movements and actions. In this regard, acceleration data carry information of individual's body parts inclination in relation to gravitational acceleration direction, leading to postural information. Fig. 22 shows examples of the three static activities from the MHEALTH dataset: standing still, sitting and relaxing, and lying down.

Given the characteristics described above, the classifier for static activities does not require complexity, but demands quality of data in such a way they can provide posture information. However, more accurate data are usually achieved when using multiple sensors on the body, while with a single sensor the classification becomes much more challenging.

Considering everything exposed in this subsection, the static activity classifier in this framework uses a Decision Tree with a maximum depth between 2 and 10, and Gini or entropy as split criterion. In addition, the model can adopt between two and five features, according to the feature selection process, described in Subsection 4.3.2.2. The goal of limiting to five features is to ensure efficient processing in terms of computational cost.

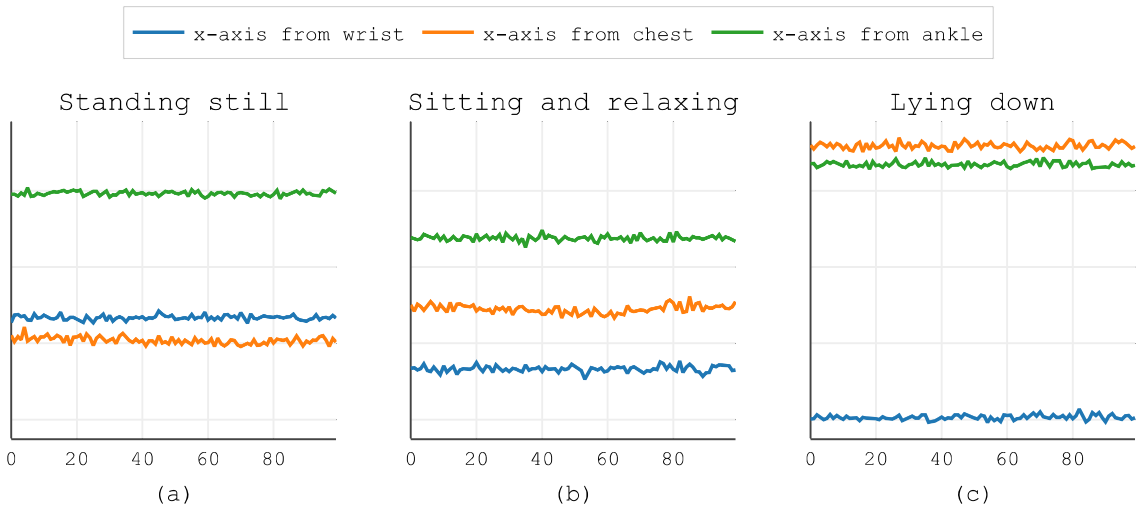


Figure 22 – Example of static activities waveform from MHEALTH dataset: (a) standing; (b) sitting; (c) lying down.

4.1.5 Dynamic Activities Module

The module Dynamic Activities (iv in Fig. 18) performs the classification of dynamic activities. Similarly to the Static Activities module, this module is activated only when the module First Level indicates that a dynamic activity has been recognized, remaining off otherwise.

The module comprises two steps: preprocessing and inference using an optimized compact CNN model. In the preprocessing step, the raw input data are normalized and concatenated. First, the data are normalized to zero mean and unit variance, for each sensor and each device. For instance, the data from the three axes of the accelerometer placed on the wrist-worn device will be normalized according to the mean and variance between the three axes of this sensor. Next, the normalized input data windows from all available signals are concatenated to form a single array, which is required by the embedded CNN model function in the microcontroller. Finally, in classification step, the normalized and concatenated array is fed into the CNN model function.

For dynamic activities, unlike static ones, searching for complex features is important due to significant pattern variations in the waveform of different activities. Furthermore, the dynamic activities group is extensive, and may comprise a variety of activities, from simple ambulatory activities to more complex ones such as drinking a cup of coffee or sweeping the floor. For that reason, often handcrafted features are unable to provide enough representativeness for the data.

Thus, deep learning approaches have been widely applied to HAR systems, as discussed in Section 3.3, due to their effectiveness on extracting more representative features from the raw input data. However, as presented in Subsection 2.2.10.3, state-of-the-art deep learning approaches are generally extremely costly for edge devices. For that reason, for the dynamic activities classification, the framework adopts a compact CNN model optimized for embedded C code, which is suitable for running on embedded systems of wearable devices and robust enough to reach a good performance on feature learning.

Fig. 23 shows the CNN architecture used in the framework. The model is composed of an input layer, two convolutional layers (Conv1 and Conv2) with ReLU activation function, a max pooling layer, a flattening layer, a fully connected layer (FC) with ReLU activation function, and a softmax layer. A dropout rate of 20% was used in the convolutional layers. The cross-entropy was adopted as loss function, and the Adam algorithm was chosen for the optimization. In the figure, the input data have three channels, derived from the three axes of a single motion sensor. However, they can have up to 18 channels when three accelerometer axes, and three gyroscopes axes from three different sensors are used.

4.2 Datasets

For the HAR framework development and the comparative study realized in this doctoral research, three datasets were selected among those publicly available presented in

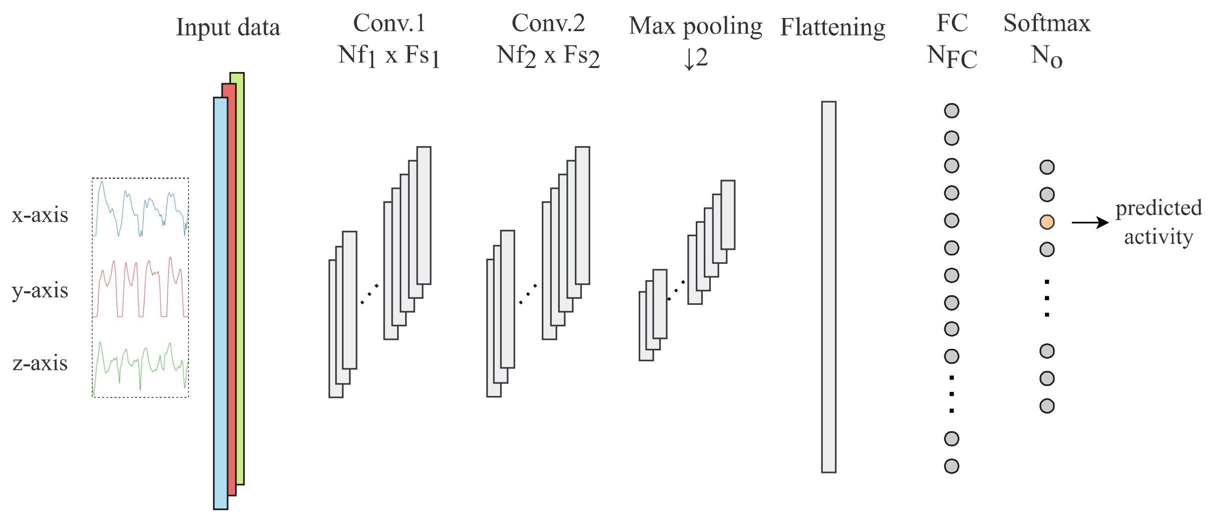


Figure 23 – CNN model for classification of dynamic activities.

Section 3.1: MHEALTH, PAMAP2, and WISDM-2019. The datasets selection was based on the type of sensors used (wearable sensors), number of subjects, quantity and diversity of activities, characteristics of these activities (daily living), and for having benchmark studies for comparison. Next, each dataset is described in more detail, and the necessary data treatments applied to each dataset are reported.

MHEALTH. This dataset (BANOS et al., 2014b) was constructed in 2013, in the Department of Computer Architecture and Computer Technology, at the University of Granada. Body motion data (acceleration, angular velocity and magnetic field orientation) and electrocardiogram were collected at 50 Hz from 10 subjects while performing 12 physical activities with wearable sensors placed on their chest, wrist and ankle, attached by using elastic straps. The activities were realized in an out-of-lab environment with no constraints on execution. The MHEALTH dataset has a “null” activity, which corresponds to 72% of the raw data. After discarding this activity, a well-balanced dataset is obtained, except for the activity “jump front and back”, which has around a third of the average number of samples per activity. The dataset activities, the execution time or number of executions per subject, and the total number of samples per activity are shown on Table 1.

PAMAP2. Three inertial measurement units (IMUs), with accelerometer, gyroscope, and magnetometer, and a heart rate monitor, sampled at 100 Hz, were used for construction of PAMAP2 dataset (REISS; STRICKER, 2012b; REISS; STRICKER, 2012a). The sensors were placed on three different body positions: chest, dominant wrist, and dominant ankle. Nine subjects, mainly employees or students at their research institute, aged 27.22 ±3.31 years, participated in dataset construction. One of them was left-handed, while all the others were right-handed. Over 10 hours of data were collected in total from 18 different activities, from which nearly 8 hours were labeled as 1 of the 18 activities performed. The authors reported some problems of connection loss during the protocol,

Table 1 – Activities from MHEALTH dataset.

Type of activity	Activity	#Data points	%Dataset
Static	Standing still	30,720	8.95 %
	Sitting and relaxing	30,720	8.95 %
	Lying down	30,720	8.95 %
Dynamic	Walking	30,720	8.95 %
	Climbing stairs	30,720	8.95 %
	Cycling	30,720	8.95 %
	Jogging	30,720	8.95 %
	Running	30,720	8.95 %
	Bending waist forward	28,315	8.25 %
	Frontal elevation of arms	29,441	8.58 %
	Bending knees	29,337	8.55 %
	Jumping front and back	10,342	3.01 %

Table 2 – Activities from PAMAP2 dataset.

Type	Activity (#subjects)	Collection time	#Data points	%Dataset
Static	Lying (8)	32 min	192,523	9.91 %
	Sitting (8)	31 min	185,188	9.53 %
	Standing (8)	32 min	189,931	9.78 %
Dynamic	Walking (8)	40 min	238,761	12.29 %
	Running (6)	16 min	98,199	5.05 %
	Cycling (7)	27 min	164,600	8.47 %
	Nordic walking (7)	31 min	188,107	9.68 %
	Ascending stairs (8)	20 min	117,216	6.03 %
	Descending stairs (8)	17 min	104,944	5.40 %
	Vacuum cleaning (8)	29 min	175,353	9.03 %
	Ironing (8)	40 min	238,690	12.29 %
	Role jumping (5)	8 min	49,360	2.54 %

and for this reason some activities for certain subjects are partly or completely missing. In this research, to avoid the problem of class imbalance, only activities with data from at least five of the nine subjects were included. In addition, Subject 9 was excluded from the dataset, as there are available data from only five of the 18 activities for this subject. Table 2 shows the list of activities, the total duration of collection and the number of samples for each activity.

WISDM-2019. Researchers from the WISDM (Wireless Sensor Data Mining) laboratory in the Department of Computer and Information Science of Fordham University created the WISDM Smartphone and Smartwatch Activity and Biometrics Dataset (WEISS; YONEDA; HAYAJNEH, 2019). They collected accelerometer and gyroscope data at 20 Hz from smartphone and smartwatch sensors. Fifty-one subjects participated executing 18 diverse activities of daily living. Each activity was performed for three minutes, therefore

Table 3 – Activities from WISDM-2019 dataset.

Type of activity	Activity	#Total data points	%Dataset
Static	Sitting	209,418	7.24 %
	Standing	212,928	7.36 %
Dynamic	Walking	205,127	7.10 %
	Jogging	202,186	6.99 %
	Using stairs	203,711	7.05 %
	Typing	201,536	6.97 %
	Brushing teeth	205,119	7.09 %
	Eating or drinking	204,838	7.09 %
	Kicking (soccer ball)	205,890	7.12 %
	Playing catch (tennis ball)	206,506	7.14 %
	Dribbling (basketball)	209,210	7.24 %
	Writing	211,765	7.32 %
	Clapping	205,134	7.10 %
	Folding clothes	207,734	7.19 %

each subject contributed with 54 minutes of data. During the data collection protocol the subjects had a smartphone in their pocket and a smartwatch placed on their dominant wrist. The dataset raw data are separated by sensor and device, and rely on synchronization based on a timestamp to be used together. However, some inconsistency was observed in the timestamps between the data collected from the smartwatch and the smartphone, then only the smartwatch data was included in this research, which is the main type of device of interest here. Finally, there are five very similar activities for classification using only the smartwatch motion sensors - eating soup, eating chips, eating pasta, drinking from cup, and eating sandwich. Such activities were grouped into a single class named "eating or drinking", and were proportionally subsampled to avoid dataset imbalance. Table 3 shows the dataset activities and the number of samples per activity.

For the three datasets used in this research, only acceleration and angular velocity data were used, although other types of data are included in the original dataset. This decision was taken because these are the most significant data for the application studied here, and the main data modalities used in the research surveyed in the literature review, in addition to being data collected from the sensors most commonly found in wearable devices. Table 4 summarizes the dataset information as used in this research. The data was segmented into 2-s windows, with 50% overlap, as will be further described in the next section, and the number of examples for each dataset is also included in the table.

4.3 Model Development

This section presents the process for developing the three models used by the framework: first level, static activities, and dynamic activities classifiers. Initially, the

Table 4 – Summary of datasets description as used in this work.

	MHEALTH	PAMAP2	WISDM-2019
Number of subjects	10	8	50
Number of activities	12	12	13
Sensors	Acc. and gyroscope	Acc. and gyroscope	Acc. and gyroscope
Placement	Chest, wrist, ankle	Chest, wrist, ankle	Wrist
Sampling rate	50 Hz	100 Hz	20 Hz
Number of data points	343,195	1,942,872	4,132,800
Window size	2 s	2 s	2 s
Number of examples	6,849	19,353	153,797

Table 5 – Software, tools and libraries used for framework development.

Tool	Definition	Version
Python	Programming language	3.6.3
JupyterLab	Web-based interactive development environment	3.3.2
Spyder	Integrated development environment	5.0.0
Numpy	Numerical computing library	1.19.5
Scipy	Scientific computation library	1.5.2
Matplotlib	Data visualization library	3.3.4
Scikit-learn	Machine learning library	0.24.2
Pandas	Easy-to-use data structures and data analysis library	1.1.5
Plotly	Data visualization library	5.7.0
LightGBM	Library for LightGBM implementation	3.3.2
TensorFlow	End-to-end platform for machine learning	2.0.0
Keras	Deep learning API running on top of TensorFlow	2.2.4
KerasTuner	Hyperparameter optimization framework for Keras	1.1.0

software and hardware tools used for model development. Next, the complete model development process is presented, including hyperparameter optimization, feature selection, and model evaluation with appropriate metrics.

4.3.1 Software and Hardware Tools

For developing and testing the model, the Python programming language was used, together with appropriate libraries for digital signal processing, machine learning - including deep learning -, and data processing, analysis, and visualization. The codes were developed in JupyterLab and Spyder IDE. Table 5 lists the main tools and libraries used and their respective versions.

All the steps of framework development and model testing were executed on a personal computer with the following configuration: AMD Ryzen 5 2600 processor; 16 GB RAM; 256 GB SSD; NVIDIA GeForce GTX 1660 Ti graphic card; and running Windows 10 Professional.

4.3.2 Model Development Process

The model development process is carried out in a sequence of rounds, applying the leave-one-subject-out (LOSO) cross-validation method, described in the next subsection. In each round, training and validation subsets are used to optimize the hyperparameters, and select the most representative features and the most appropriate number of features. Next, the model is trained with the training and validation subsets, and tested with the test subset, to evaluate the model based on metrics. Fig. 24 shows the flowchart of the model development process for the first level and the static activities classifiers.

For the dynamic activities classifier, the development process has a variation. Since a CNN model is used in this case, feature extraction and selection are intrinsic to model training, and there is no feature selection step in the process. On the other hand, a sensor selection stage is added to determine the best combination of sensors to be used as the CNN input. This step is executed prior to the hyperparameter optimization, and is responsible for training and testing the model using data from different sensors and devices. For example, using only acceleration data from wrist and ankle-worn sensors, discarding angular velocity data, and chest device data. Fig. 25 shows the flowchart of the model development process for the dynamic activities classifier.

4.3.2.1 Nested Cross-Validation

The LOSO nested cross-validation was used in this research for hyperparameter optimization, feature selection, and model evaluation. In this approach, each fold comprises a specific subject, and in each testing round one subject is used for testing, while the others are divided between training and validation.

Fig. 26 illustrates the application of the method for 10 subjects (MHEALTH dataset scenario). For the PAMAP2 dataset the same approach was adopted, however for a total of eight subjects. Finally, for the WISDM-2019 dataset, a variation of the method was used, since this dataset is quite large and the regular LOSO nested cross-validation technique would be computationally expensive. For that reason, in this case each fold comprises 10 subjects, instead of a single one.

4.3.2.2 Hyperparameter Optimization

To search for the best model hyperparameters, the grid-search method was used. In this step, the training subset was used for training the model, while the validation subset was used for testing the model with each hyperparameter configuration.

For the first level classifier and the static activities classifier, the parameters searched for the corresponding Decision Tree models are the splitting criterion and the max depth, presented, respectively, in Tables 6 and 7. For the CNN model of the dynamic

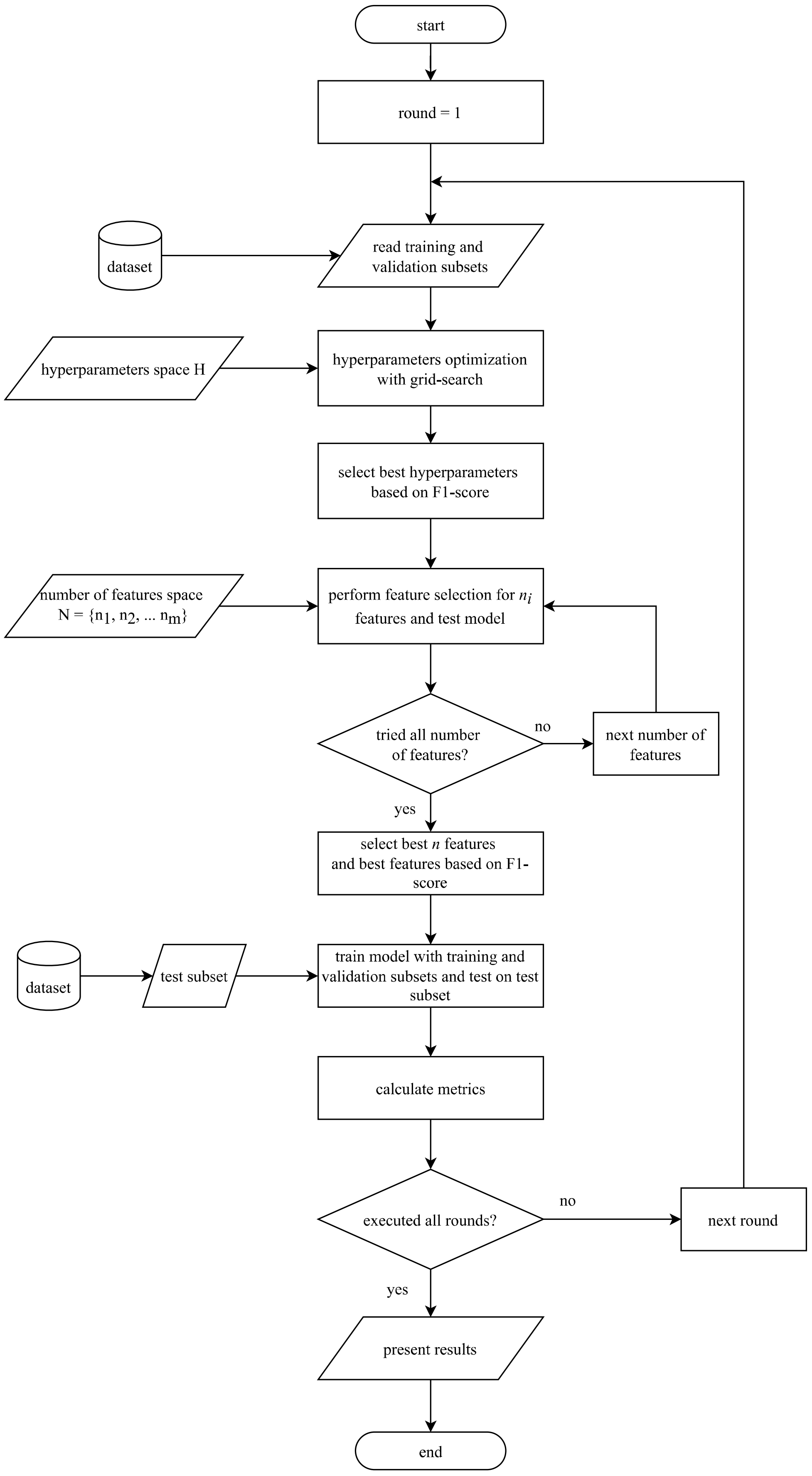

start
round = 1
dataset
read training and validation subsets
hyperparameters space H
hyperparameters optimization with grid-search
select best hyperparameters based on F1-score
number of features space N = {n1, n2, ... nm}
perform feature selection for n_i features and test model
tried all number of features?
no
next number of features
yes
select best n features and best features based on F1-score
dataset
test subset
train model with training and validation subsets and test on test subset
calculate metrics
executed all rounds?
no
next round
yes
present results
end

Figure 24 – Flowchart of hyperparameter optimization, feature selection, and model evaluation for the first level and static activities classifiers.

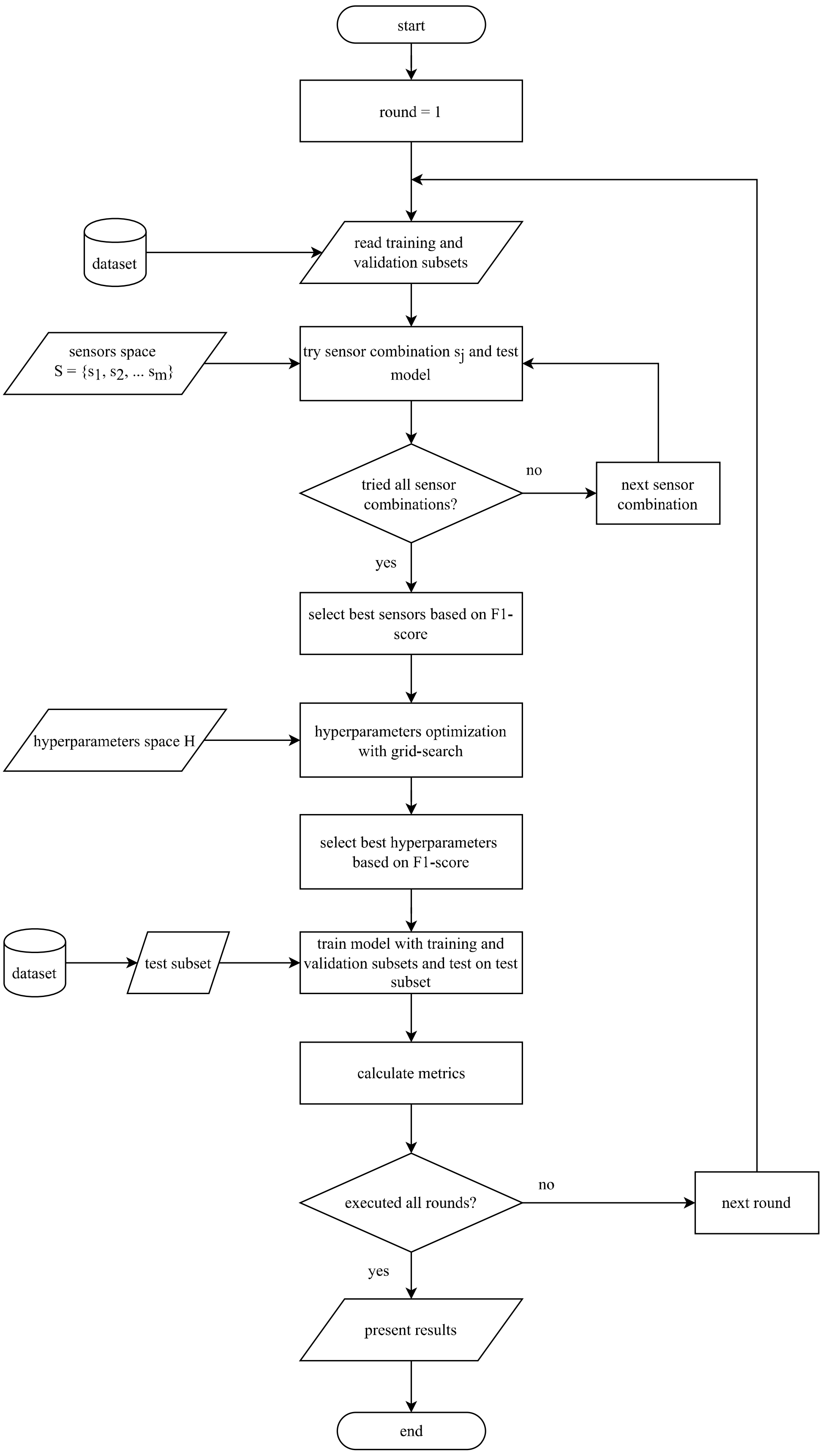

start
round = 1
dataset
read training and validation subsets
sensors space S = {s1, s2, ... sm}
try sensor combination sj and test model
tried all sensor combinations?
no
next sensor combination
yes
select best sensors based on F1-score
hyperparameters space H
hyperparameters optimization with grid-search
select best hyperparameters based on F1-score
dataset
test subset
train model with training and validation subsets and test on test subset
calculate metrics
executed all rounds?
no
next round
yes
present results
end

Figure 25 – Flowchart of sensors selection, hyperparameter optimization, and model evaluation for the dynamic activities classifier.

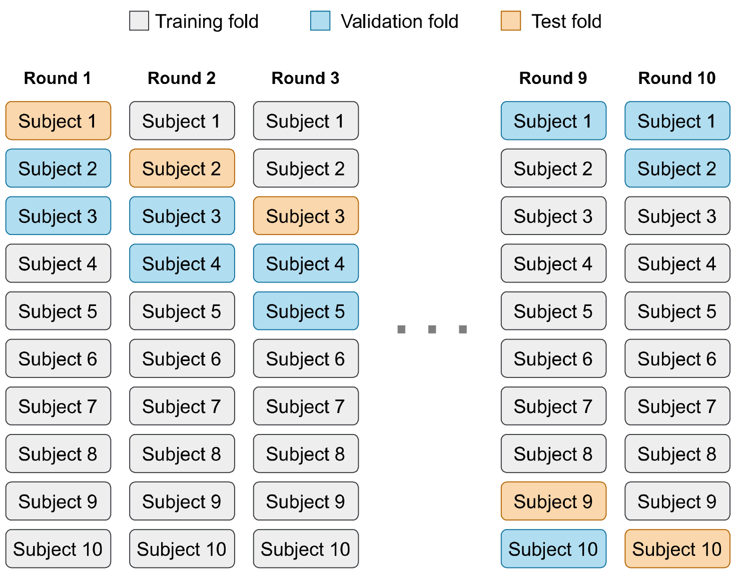


Figure 26 – Leave-one-subject-out nested cross-validation method adopted for training and testing the models.

Table 6 – Parameters for grid search of the first level model

Parameter	Values
Split criterion	{gini, entropy}
Max depth	{1-5}

Table 7 – Parameters for grid search of the static activities model

Parameter	Values
Split criterion	{gini, entropy}
Max depth	{2-10}

activities classifier, the parameters searched are the number of filters and kernel size for both convolutional layers, the stride of first layer, and the number of neurons of the fully connected layer, presented in Table 8.

Table 8 – Parameters for grid search of the dynamic activities model

Denotation	Parameter	Values
NF_1	Number of filters in 1st convolutional layer	{32, 64}
FS_1	Filter size in 1st convolutional layer	{3, 5}
St_1	Stride in 1st convolutional layer	{1, 3}
NF_2	Number of filters in 2nd convolutional layer	{16, 32}
FS_2	Filter size in 2nd convolutional layer	{3, 5}
N_{FC}	Number of neurons in the fully connected layer	{20, 30}

4.3.2.3 Feature Selection

For the first level and static activities classifiers, the hyperparameter tuning is executed using all the available features. After that, a feature selection technique is used for determining the most significant features for the model. In this work, a technique based on feature importance was adopted to eliminate features with low impact on the model output, in order to discard irrelevant features.

Decision tree algorithms, like the one used in the proposed framework classifiers, offer importance scores based on the reduction in the splitting criterion, such as Gini or entropy. By using the scikit-learn library, the fitted model provides a feature importances property, which contains the relative importance scores for each input attribute.

Thus, during the development process, the model performance was evaluated in the validation subset with the features that obtained the highest feature importance values after being trained with the training subset. Different number of features were evaluated.

4.3.2.4 Model Evaluation

After defining the best hyperparameters and best features for the model, it is finally trained with training and validation subsets, tested with test subset, and evaluated based on the following metrics for m classes: accuracy (Equation 4.12), macro precision (Equation 4.13), macro recall (Equation 4.14), and macro F1-score (Equation 4.15). The confusion matrix is also computed. In the aforementioned equations, TP, TN, FP, and FN stand for, respectively, true positive, true negative, false positive, and false negative.

$$accuracy = \frac{TP + TN}{TP + FN + FP + TN} \tag{4.12}$$

$$precision = \frac{\sum_{i=1}^{m} \frac{TP_i}{TP_i + FP_i}}{m} \tag{4.13}$$

$$recall = \frac{\sum_{i=1}^{m} \frac{TP_i}{TP_i + FN_i}}{m} \tag{4.14}$$

$$F1 = 2 \times \frac{precision \times recall}{precision + recall} \tag{4.15}$$

4.4 Reference Models for Comparison

Three different models were selected for an initial comparison of the performance achieved by the framework. KNN, for its ease of construction and tuning, works as a good baseline. Random Forest, for being quite efficient and capable of being embedded

Table 9 – Parameters for grid search of the KNN model

Parameter	Values
Number of neighbors	{3, 5, 7, 9, 11, 13, 15}
Weight	{uniform, distance}
Distance metric	{Euclidean, Manhattan}

Table 10 – Parameters for grid search of the Random Forest model

Parameter	Values
Number of estimators	{20, 40, 60, 80, 100}
Max depth	{3, 5, 7, 9}

Table 11 – Parameters for grid search of the LightGBM model

Parameter	Values
Number of estimators	{30, 60, 100}
Max depth	{3, 5, 7}

in wearable devices in straightforward way. And finally, LightGBM, a recently developed algorithm, which has achieved outstanding results in several applications. This section presents how these models were built, tuned, and tested.

The model development process was similar to the framework models, however, without the feature selection step, thus all features were used as input. The LOSO nested cross-validation method was also used to define the training, validation and test subsets in each round. In the first step, the model hyperparameters are tuned using the training and validation subsets. Next, the model is trained with the best hyperparameters, and evaluated on the test subset. The hyperparameters searched for optimization are presented in Tables 9, 10, and 11, respectively for KNN, Random Forest, and LightGBM.

5 Deployment on Microcontroller

Since the objective of this work is to develop a HAR framework for running on wearable devices, it is essential to deploy and test it on a microcontroller. The previous chapters presented the framework, and described the process of model development. This chapter approaches the materials and methods used for developing, deploying, and testing the proposed HAR framework on a microcontroller.

First, the hardware platform and the Integrated Development Environment (IDE) used to run the tests on the microcontroller are presented. Next, the structure of the embedded HAR framework is described, as well as the implementation of each of the modules on the device. Furthermore, the methods used to analyze the performance are presented, in terms of processing time, model complexity, and energy consumption. Finally, the results achieved are presented.

5.1 Development Tools

The software and hardware development tools used to implement the framework, program it in the microcontroller, run the program, and evaluate the performance are part of the STM32 family of 32-bit microcontrollers based on the Arm Cortex-M processor from STMicroelectronics.

5.1.1 Hardware Platform

The development kit named 32F411EDISCOVERY from STMicroelectronics (STMICROELECTRONICS, 2021b), shown in Figure 27, was the hardware platform used for the tests on microcontroller. This kit allows the development of applications with an STM32 series microcontroller, specifically the STM32F411VET6 in this board, which is based on Arm Cortex-M4 core with 72 MHz as maximum clock frequency, 512 kB of program memory (Flash), and 128 kB of RAM.

This device has a comprehensive set of power-saving mode which allows the design of low-power applications. In run mode, it consumes 100 µA/MHz, and only 9 µA in stop mode with the Flash memory in deep power down mode. In addition, the batch acquisition mode enables data acquisition through any communication peripherals directly to memory using the direct memory access in reduced power consumption as well as data processing while the rest of the system is in low-power mode.

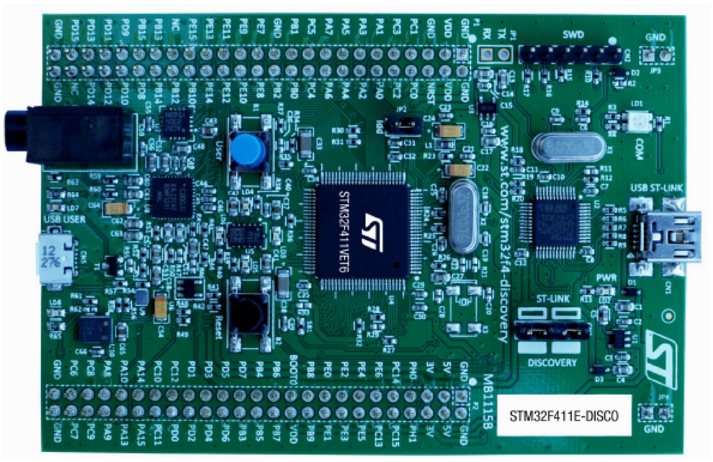


Figure 27 – The development kit 32F411EDISCOVERY from STMicroelectronics.

Source: (STMICROELECTRONICS, 2021b).

5.1.2 Firmware Development Platform

The software tools used to develop the framework firmware were STM32CubeIDE, STM32CubeMX, and X-CUBE-AI. The first is the IDE of the STM32Cube software ecosystem. This is an advanced C/C++ development platform with peripheral configuration, code generation, code compilation, and debug features for STM32 microcontrollers and microprocessors (STMICROELECTRONICS, 2022b).

STM32CubeMX is a graphical tool that simplify the configuration of STM32 microcontrollers, as well as the generation of the corresponding initialization C code for the microcontroller. It allows to configure the general purpose inputs and outputs (GPIOs), and the clock setup for the whole system. Furthermore, the default software can be extended by integrating expansion packages into the project, such as the X-CUBE-AI (STMICROELECTRONICS, 2022c).

X-CUBE-AI is an STM32Cube expansion package that enables the automatic conversion of pre-trained Machine Learning algorithms, including neural networks models. The user's model is converted to a generated optimized library into the user's project (STMICROELECTRONICS, 2022d). The CNN model developed in this research can take advantage of the native support for the Deep Learning framework Keras, used to develop the model. Fig. 28 shows the core engine of this expansion package, in which the user only needs to upload the pre-trained model and set the parameters name, compression factor, and targeted STM32 device. The optimized code and a complexity report is then generated.

In Fig. 28, PINNR refers to platform-independent neural network representation, which is a file generated by the X-CUBE-AI core importer to have a common and portable internal representation of the uploaded model for the next stages (optimizer and C-code generator).

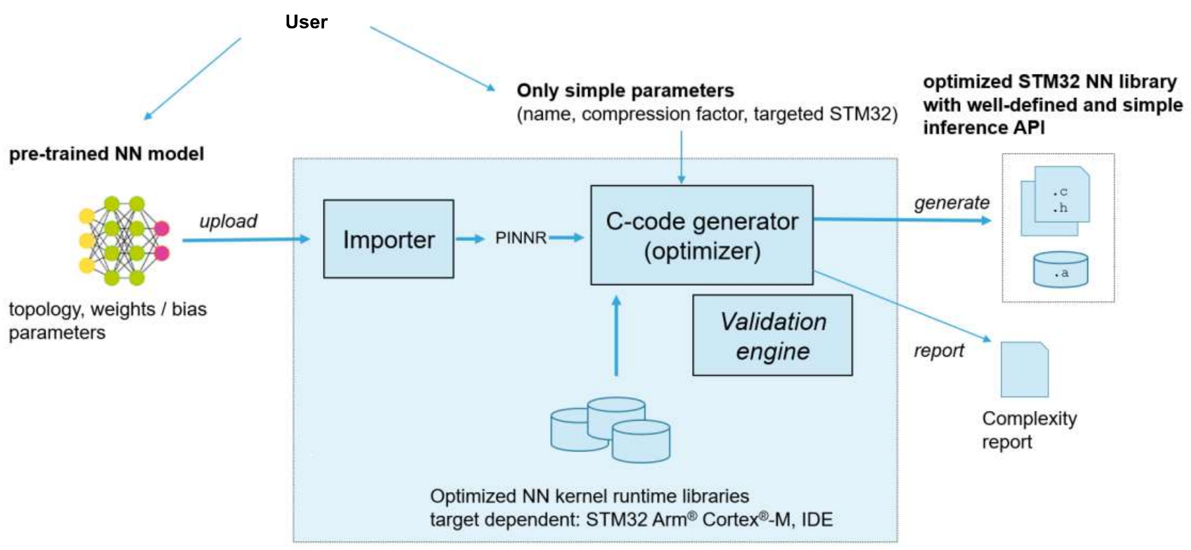

User
pre-trained NN model
upload
topology, weights / bias parameters
Only simple parameters
(name, compression factor, targeted STM32)
optimized STM32 NN library with well-defined and simple inference API
Importer
PINNR
C-code generator (optimizer)
Validation engine
generate
.c
.h
.a
report
Complexity report
Optimized NN kernel runtime libraries
target dependent: STM32 Arm® Cortex®-M, IDE

Figure 28 – X-CUBE-AI core engine.

Source: Adapted from (STMICROELECTRONICS, 2022a)

5.2 HAR Framework Implementation

The proposed HAR framework and its modules were introduced in Chapter 5. Here, the implementation of the framework in the microcontroller is presented. The framework's simplified algorithm is presented as a flowchart in Fig. 29.

The first stage is the declaration of necessary variables and functions, and the initialization of variables. Next, the data are read and the classifiers are executed as described in the next subsections. Finally, the recognized activity is obtained and processed by the appropriate function, and the microcontroller enters sleep mode to save energy.

5.2.1 Data Acquisition

The module Data Acquisition, as presented in Subsection 4.1.1, reads the data from the sensors at the configured sample rate, and applies the segmentation forming the input data window.

In this module, a timer interrupt is used to wake up the microcontroller, which was in sleep mode, every t milliseconds, according to the sampling rate set, and read a sample from the sensor. Each sample fills an incremental position in the input data window, an

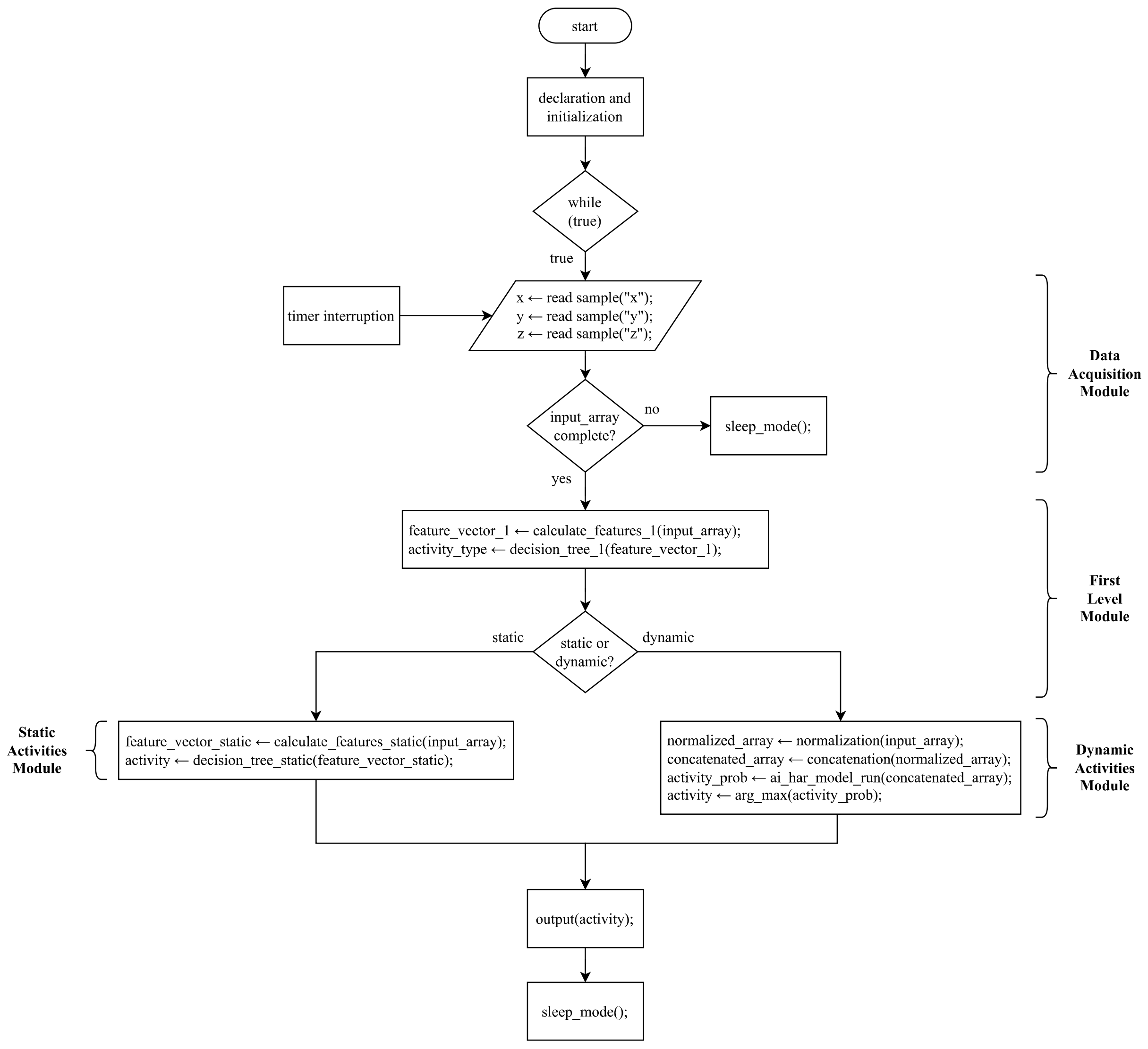

start
declaration and initialization
while (true)
true
timer interruption
x ← read sample("x");
y ← read sample("y");
z ← read sample("z");
input_array complete?
no
sleep_mode();
yes
Data Acquisition Module
feature_vector_1 ← calculate_features_1(input_array);
activity_type ← decision_tree_1(feature_vector_1);
static or dynamic?
static
dynamic
First Level Module
Static Activities Module
feature_vector_static ← calculate_features_static(input_array);
activity ← decision_tree_static(feature_vector_static);
normalized_array ← normalization(input_array);
concatenated_array ← concatenation(normalized_array);
activity_prob ← ai_har_model_run(concatenated_array);
activity ← arg_max(activity_prob);
Dynamic Activities Module
output(activity);
sleep_mode();

Figure 29 – Flowchart of the HAR framework firmware.

array called *input_data*. When the array is complete, the module Data Acquisition outputs it to the next module.

5.2.2 First Level Classification

The module First Level Classification, as described in Subsection 4.1.3, separates between static and dynamic activities. For that purpose, this module receives the data input arrays and outputs the activity type. The functions that form the module are constructed based on the topology and features defined in the model development stage. For tests on the microcontroller, the model was designed with data transformed into 16-bit integers, since the device will handle this type of data when reading samples from sensors.

The first task is the feature extraction. The function *calculate_features_1* receives the input arrays with data from sensors and axes, and calculates the features defined in the model development stage, forming the feature vector. It comprises calculations of the predefined statistics features from the input arrays, among mean, standard deviation, maximum, minimum, and SMA. The calculated features fill the *feature_vector_1* array. The fragment of code below shows an example of function definition and call for the feature extraction of standard deviation from x-axis and minimum value from z-axis.

```
void calculate_features_1(int input_array_x[], int input_array_z[], int *feature_vector):
    feature_vector[0] = calculate_std_dev(input_array_x);
    feature_vector[1] = calculate_min(input_array_z);
    return;

calculate_features_1(input_array_x, input_array_z, feature_vector_1);
```

The obtained feature vector is fed into the Decision Tree, which outputs the activity type identified. If it is a static activity, the program will run the Static Activities module next, otherwise the Dynamic Activities Module will be executed. The function *decision_tree_1* is formed by a sequence of *if...else* statements, and it returns 0 or 1 for static or dynamic activity, respectively. The following fragment of code shows an example of function definition and call for the Decision Tree of the first level classifier.

```
int decision_tree_1(int feature_vector[]) {
        if (feature_vector[0] <= 300.52) {
                if (feature_vector[1] <= 50.93) {
                        return 0;
                }
                else {
                        return 1;
                }
        }
        else {
                if (feature_vector[1] <= 65.6) {
                        return 0;
                }
                else {
                        return 1;
                }
        }
}

activity_type = decision_tree_1(feature_vector_1);
```

5.2.3 Static Activities Classification

When a static activity is recognized by the first level classifier, the module Static Activities is activated. This module is based on two functions which execute the same tasks as the prior module. As in the case of the previous module, the model development used data transformed into 16-bit integers, since it will receive this type of data input when operating with motion sensors.

The first function, *calculate_features_static*, receives the sensors and axes input arrays, and calculates the statistics parameters as defined in the model development step. The fragment of code below shows an example of function definition and call for the feature extraction of SMA from y-axis and mean from z-axis.

```
void calculate_features_static(int input_array_y[], int input_array_z[], int *feature_vector):
    feature_vector[0] = calculate_sma(input_array_y);
    feature_vector[1] = calculate_mean(input_array_z);
    return;

calculate_features_static(input_array_x, input_array_z, feature_vector_static);
```

Next, the Decision Tree receives the feature vector, and outputs the activity recognized. The function *decision_tree_static* is formed by a sequence of *if...else* statements, and it returns integer values corresponding to expected static activities. The fragment of code below shows an example of function definition and call for the Decision Tree which classifies three different static activities.

```
int decision_tree_static(int feature_vector[]) {
        if (feature_vector[0] <= 15498.34) {
                if (feature_vector[1] <= 594981.5) {
                        return 1;
                }
                else {
                        return 0;
                }
        }
        else {
                if (feature_vector[1] <= 1275279.0) {
                        return 1;
                }
                else {
                        return 2;
                }
        }
}

activity = decision_tree_static(feature_vector_static);
```

5.2.4 Dynamic Activities Classification

In case of the module First Level classifies the activity as dynamic, the module Dynamic Activities is activated. This module basically performs a sequence of four functions. Firstly, the function *concatenation* combines the input arrays into a single array. This step is required by the CNN model function. Next, the function *normalization* transforms the array values to a scale with zero mean and unit variance.

After these preprocessing steps, the inference is performed to classify the dynamic activity from the concatenated and normalized input data array. The CNN model function is created by using the X-CUBE-AI tool, which generates the optimized code from the CNN model built in the development stage. The generated CNN model function *ai_har_model_run* receives the normalized array, performs the inference, and outputs the *cnn_output_array*, which gives the probability distribution of activities. Finally, the *get_max_index* gets the index with maximum probability in the array, which gives the dynamic activity recognized. The Dynamic Activities module functions are shown in the following fragment of code.

```
concatenation(input_array_x, input_array_y, input_array_z, concatenated_array);
normalization(concatenated_array, normalized_array);
ai_har_model_run(har_model, normalized_array, cnn_output);
activity = get_max_index(cnn_output);
```

5.3 Performance Analysis

This section describes the methods used to evaluate the performance of the proposed framework, both in terms of computational cost and in relation to the complexity of the model. The objective is to show that with this framework, a HAR system can be implemented in wearable devices, with the requirements met and a satisfactory activity recognition rate.

5.3.1 Simulation of Sensor Reading

To perform the performance analysis on the microcontroller, the same datasets used in the development of the models were used here. In this way, the framework was implemented and executed on the microcontroller with a little modification in the module Data Acquisition. Instead of collecting data from real sensors, samples from the datasets were sent from a personal computer to the microcontroller via Universal asynchronous receiver/transmitter (UART) serial communication. The other framework modules were executed normally.

A script was developed in Python to run on the computer performing the simulated data reading from sensors. In this script, a sequence of samples is obtained from the dataset, forming a data window of 2 s, which is sent to the development kit through a USB/serial UART converter.

5.3.2 Data Preparation

Usually, sensors such as accelerometer and gyroscope provide output data in a resolution between 8 and 16 bits. However, the samples provided by the datasets used in this research are real numbers represented as floating-point, probably due to preprocessing stages possibly applied after the data collection protocol.

In order to simulate the sensor data reading, an expanded resolution of 16 bits was adopted, since there is no information about the datasets' sensor resolution. For this purpose, the floating-point samples were scaled to the range $[0, 2^{16} - 1]$.

5.3.3 Computation of Processing Time

The computational cost is strongly related to the execution time of tasks by the microcontroller. Therefore, measuring the time interval that the device takes to perform the tasks of each module of the framework provides important information about how much each task contributes to the processing cost. To compute these measurements, the microcontroller's timers were used and extra functions were added to calculate and report the elapsed time for each relevant task.

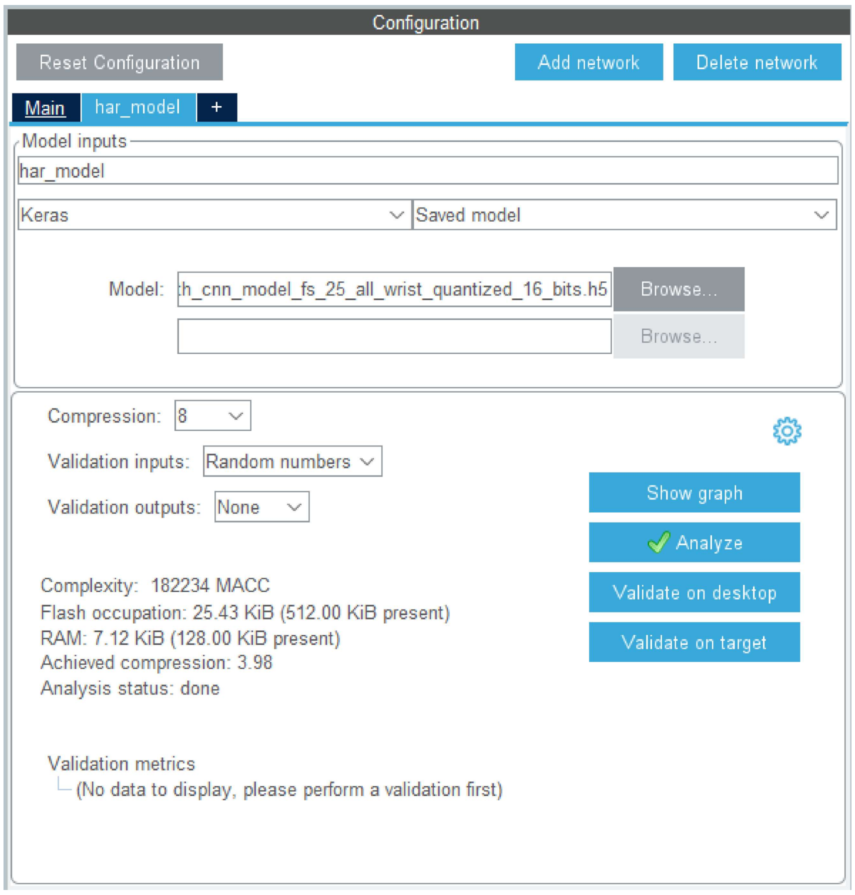
5.3.4 Model Complexity

The analysis of model complexity evaluates specifically the dynamic activities classifier, which uses a CNN model. This analysis gives information about the model regarding number of MAC operations, Flash memory occupied, and RAM memory required for execution. When the pre-trained CNN model is uploaded to X-CUBE-AI tool, it gives information on the these aspects, as shown in Fig. 30.

5.3.5 Estimation of Energy Consumption

Power consumption is critical for resource-constrained devices such as wearable sensors. For this reason, an estimation of energy consumption was performed with the power consumption analysis tool from STM32CubeMX.

This tool allows us to define a sequence of tasks and the duration of these tasks. In addition, it is also possible to include information on the operating mode (execution, sleep, stop), supply voltage, clock frequency, among others. With this information, the



Model inputs
har_model
Keras
Saved model
Model: :h_cnn_model_fs_25_all_wrist_quantized_16_bits.h5

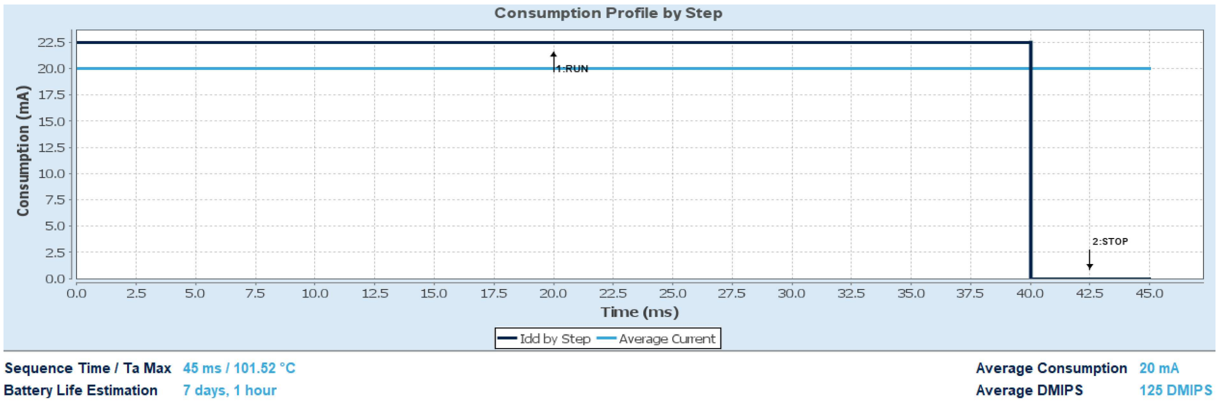
Compression: 8
Validation inputs: Random numbers
Validation outputs: None

Complexity: 182234 MACC
Flash occupation: 25.43 KiB (512.00 KiB present)
RAM: 7.12 KiB (128.00 KiB present)
Achieved compression: 3.98
Analysis status: done

Validation metrics
(No data to display, please perform a validation first)

Figure 30 – X-CUBE-AI core engine.

tool provides the total operating time, average power consumption, and estimated battery life, as shown in Fig. 31.



Sequence Time / Ta Max 45 ms / 101.52 °C
Battery Life Estimation 7 days, 1 hour
Average Consumption 20 mA
Average DMIPS 125 DMIPS

Figure 31 – Power consumption analysis tool from STM32CubeMX.

6 Results and Discussion

This chapter presents the results achieved in this doctoral research, and the discussion about the findings. In the first part, the analysis of the framework recognition performance and the comparison with reference models are presented, in terms of metrics such as accuracy, precision, recall, and F1-score. Next, in the second part, the evaluation of the computational performance in embedded devices with respect to processing time, model complexity, and energy consumption is presented and discussed. Finally, in the third part, the results achieved by the proposed HAR framework are compared with the results from state-of-the-art techniques.

6.1 Evaluation of Activity Recognition Performance

This section presents the results obtained from tests carried out to evaluate the performance of the proposed energy-efficient HAR framework in the classification of human activities. The achievements of each model developed that compose the framework are discussed here, as well as the performance of the framework compared to reference models.

The results and discussion are divided into three subsections, corresponding to each one of the datasets used in this doctoral research: MHEALTH, PAMAP2, and WISDM-2019. Furthermore, the models are evaluated in different scenarios, with multiple wearable sensors, and using only a single sensor on the wrist. In each of these scenarios, they were tested using accelerometer and gyroscope data, and accelerometer-only data, as well as different sampling rates. In the sequence, the summary of results are presented. For better comprehension, Fig. 32 presents a tree of the results presented in this section.

For accomplishing the results presented here, the tests were performed using the LOSO nested cross-validation, as described in Subsection 4.3.2. The hyperparameters and features with the best average performance are presented, as well as the mean performance results in terms of accuracy, precision, recall, and F1-score.

6.1.1 Results on MHEALTH Dataset

In this subsection, the results achieved on the MHEALTH dataset are presented and discussed. First, the results accomplished when using multiple sensors are presented. Further on, in the second scenario, the results achieved by using only the sensor on the wrist are evaluated.

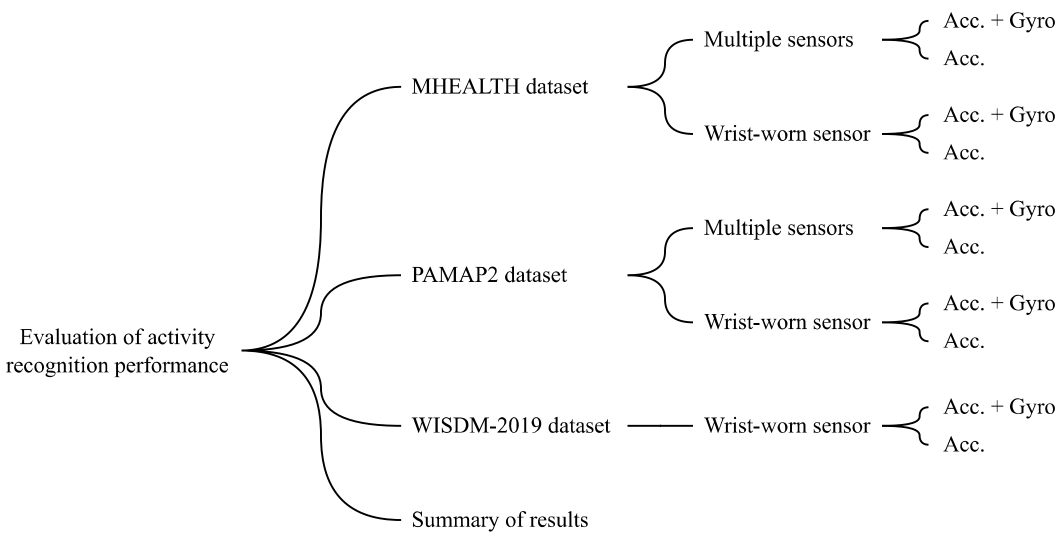


Figure 32 – Organization of the presentation of results in this section.

Table 12 – Framework performance on MHEALTH dataset using accelerometer and gyroscope data from multiple wearable devices.

Model	Accuracy (%)	Precision (%)	Recall (%)	F1-score (%)
First level classifier	98.91	98.93	98.91	98.91
Static activities classifier	89.46	84.47	89.46	86.12
Dynamic activities classifier	92.97	95.22	92.97	91.86
Framework	91.02	91.34	91.02	89.33

6.1.1.1 Multiple Sensors Scenario

Accelerometer and gyroscope data. In the first tests of this scenario, the accelerometer and gyroscope sensors from chest, wrist, and ankle-worn devices were used. The results of tests performed to evaluate the classifiers and the framework are presented in Table 12.

The framework mean accuracy with the three models integrated was 91.02%. Among the framework classifiers, the highest accuracy was achieved by first level Decision Tree, with 98.91%. On the other hand, the static activities classifier had a quite satisfactory performance with some subjects, while much lower than expected with others, greatly impairing the mean accuracy, which was 89.46%. One justification for this variation is that static activities are constant postures and have low variability for the same subject, which makes the model's ability to generalize difficult. The confusion matrix, presented in Fig. 33, shows that there was important misclassification between the activities "standing still" and "sitting and relaxing", and between "sitting and relaxing" and "lying down". The figure also highlights the confusion matrices of three subjects with the worst results, for which the model demonstrated difficulty in generalizing, never being able to infer one of the three activities.

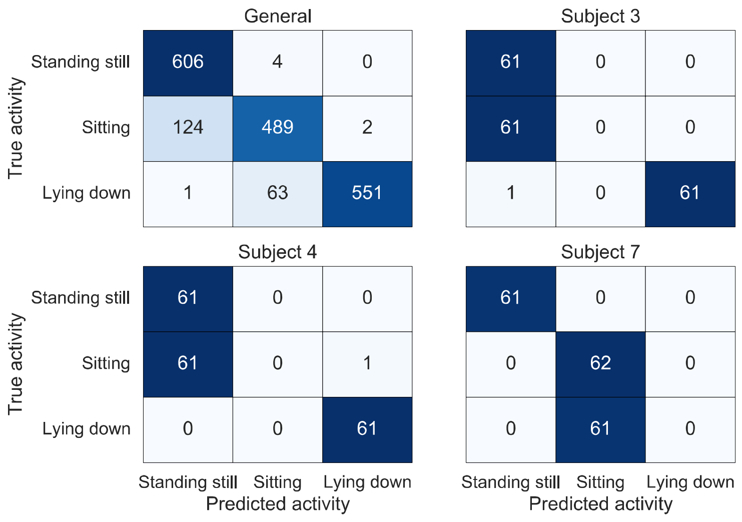


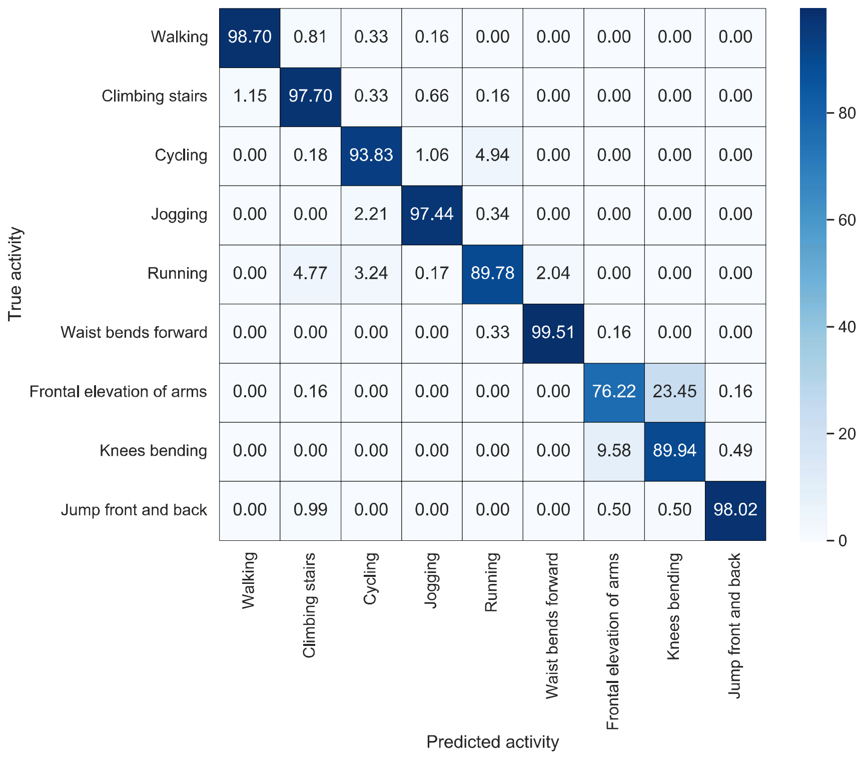
Figure 33 – Confusion matrices of static activities classifier performance, in general and for 3 subjects with the worst results, on MHEALTH dataset within the multiple sensor scenario, using acceleration and gyroscope data.

More promising was the dynamic activities classifier performance, reaching 92.97% of mean accuracy. The confusion matrix in Fig. 34 shows that the most challenging activities to classify were the “frontal elevation of arms” and “bending knees”. These activities are more challenging for the model as they are more dependent on the subject’s ability to follow instructions, and the movements are more susceptible to artifacts than ambulatory activities.

This result was greatly affected by the accuracy of the model with the 3 worst performing subjects. Fig. 35 shows the confusion matrix for these specific subjects. Note the high misclassification between the two aforementioned activities. The confusion between “cycling” and “running” should also be highlighted.

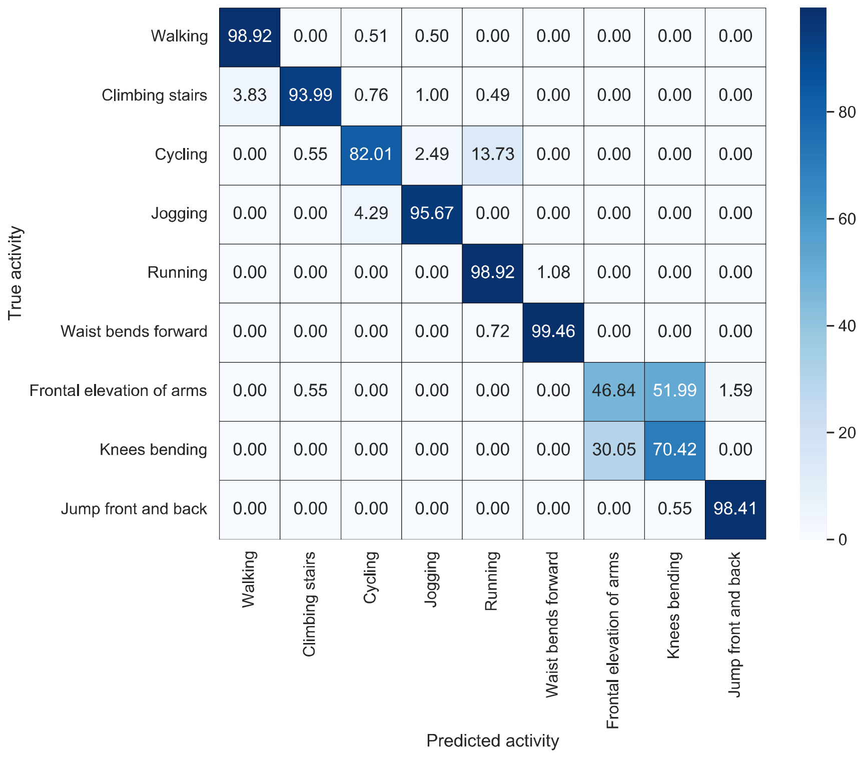
In average of the 10 subjects tested, the first level Decision Tree mostly used entropy as the splitting criterion, a mean max depth of 3.1, and a mean of 2.2 features, mainly y-axis acceleration standard deviation from wrist-worn device, and z-axis acceleration mean from ankle-worn device. Even with gyroscope data available, the best features to separate between static and dynamic activities were derived from accelerometer data. The variation of the standard deviation of wrist acceleration is closely related to the presence or absence of movement, while the mean acceleration of the ankle is linked to the positioning of this body part.

The Decision Tree for static activities classification used the Gini splitting criterion for most subjects, and a mean max depth of 2.2, using 2 features in average, being the acceleration SMA from wrist sensor in y-axis, and the acceleration SMA from chest sensor in x-axis the most selected ones. In the case of static activities, SMA carries information about the position of the body part in relation to the direction of gravity acceleration.



True activity \ Predicted activity	Walking	Climbing stairs	Cycling	Jogging	Running	Waist bends forward	Frontal elevation of arms	Knees bending	Jump front and back
Walking	98.70	0.81	0.33	0.16	0.00	0.00	0.00	0.00	0.00
Climbing stairs	1.15	97.70	0.33	0.66	0.16	0.00	0.00	0.00	0.00
Cycling	0.00	0.18	93.83	1.06	4.94	0.00	0.00	0.00	0.00
Jogging	0.00	0.00	2.21	97.44	0.34	0.00	0.00	0.00	0.00
Running	0.00	4.77	3.24	0.17	89.78	2.04	0.00	0.00	0.00
Waist bends forward	0.00	0.00	0.00	0.00	0.33	99.51	0.16	0.00	0.00
Frontal elevation of arms	0.00	0.16	0.00	0.00	0.00	0.00	76.22	23.45	0.16
Knees bending	0.00	0.00	0.00	0.00	0.00	0.00	9.58	89.94	0.49
Jump front and back	0.00	0.99	0.00	0.00	0.00	0.00	0.50	0.50	98.02

Figure 34 – Confusion matrix of dynamic activities classifier performance on MHEALTH dataset within the multiple sensor scenario, using acceleration and gyroscope data.



True activity \ Predicted activity	Walking	Climbing stairs	Cycling	Jogging	Running	Waist bends forward	Frontal elevation of arms	Knees bending	Jump front and back
Walking	98.92	0.00	0.51	0.50	0.00	0.00	0.00	0.00	0.00
Climbing stairs	3.83	93.99	0.76	1.00	0.49	0.00	0.00	0.00	0.00
Cycling	0.00	0.55	82.01	2.49	13.73	0.00	0.00	0.00	0.00
Jogging	0.00	0.00	4.29	95.67	0.00	0.00	0.00	0.00	0.00
Running	0.00	0.00	0.00	0.00	98.92	1.08	0.00	0.00	0.00
Waist bends forward	0.00	0.00	0.00	0.00	0.72	99.46	0.00	0.00	0.00
Frontal elevation of arms	0.00	0.55	0.00	0.00	0.00	0.00	46.84	51.99	1.59
Knees bending	0.00	0.00	0.00	0.00	0.00	0.00	30.05	70.42	0.00
Jump front and back	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.55	98.41

Figure 35 – Confusion matrix of dynamic activities classifier performance for three specific subjects on MHEALTH dataset within the multiple sensor scenario, using acceleration and gyroscope data.

Table 13 – Framework performance on MHEALTH dataset using only accelerometer data from multiple wearable devices.

Model	Accuracy (%)	Precision (%)	Recall (%)	F1-score (%)
First level classifier	98.96	98.97	98.96	98.95
Static activities classifier	86.20	86.19	86.20	84.03
Dynamic activities classifier	93.11	95.38	93.11	92.00
Framework	90.29	91.96	90.29	88.91

Therefore, in this case, the positioning of the user's chest and wrist demonstrated to be the most important attributes for the activity classification.

The optimized CNN for dynamic activities classification used only acceleration data from wrist and ankle sensors, and its architecture had 32 filters of size 5x1 in the two convolutional layers, stride of 1 in both layers, and 30 neurons in the fully connected layer. In this scenario, the accelerometer proved to be more efficient for classifying dynamic activities. In addition, the use of the chest sensor was neglected, and the best results were obtained with the wrist and ankle sensors, which makes a lot of sense for dynamic activities, in which the user displacement and the movement of the limbs greatly affect these parts of the body.

Accelerometer-only data. Next, in a second evaluation within this scenario with multiple wearable sensors, models were developed and tested using only the devices' accelerometer sensor, and Table 13 shows the results. The findings were very similar to the results of the previous case in this scenario, when the gyroscope data were included, since the main features in that case were derived from the acceleration signal. The first level classifier achieved 98.96% of accuracy, while the static activities classifier reached 86.20%, and the dynamic activities classifier obtained 93.11%. The framework performance was slightly lower than the previous case, being the F1-score only 0.42% lower when using only acceleration data, reducing the volume of processed data by half.

In this case, the first level Decision Tree used, in average, Gini as splitting criterion, and a mean max depth of 2.9, while using 2.1 features, being the best features the same as in prior case: y-axis acceleration standard deviation from wrist-worn device, and z-axis acceleration mean from ankle-worn device. The static activities Decision Tree used mostly Gini splitting criterion, and a mean max depth of 2.1, using only 2 features on average and again the same as in the previous case: y-axis acceleration SMA from wrist sensor, and x-axis acceleration SMA from chest sensor. The CNN model used the same sensors and architecture as in the first case.

Fig. 36 shows a comparison between the two cases evaluated in this scenario with multiple wearable sensors. The graph shows a boxplot chart of accuracy for each model and the framework, with the mean value represented by a dashed line. It can be observed that the performance of the first level classifier remained quite high for all subjects in both cases. On the contrary, for static activities, the classifier performed very well with part of the subjects, however the much lower performance for other subjects pulled the average down. This shows a difficulty in the model's generalization ability to classify inter-subject static activities.

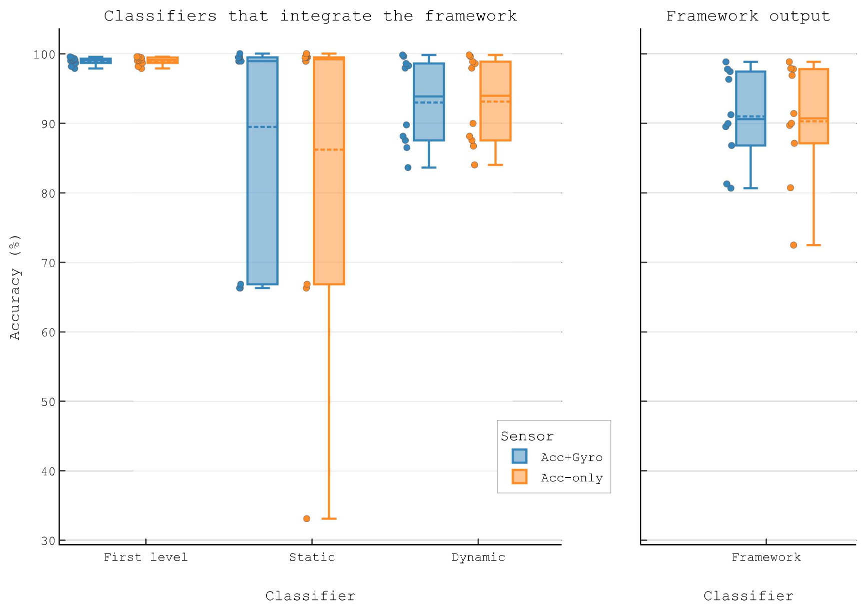


Figure 36 – Boxplot charts of models and framework accuracy on MHEALTH dataset within the multiple sensors scenario. Comparison between results using accelerometer and gyroscope data and using accelerometer-only data.

As stated before, a slight decrease in the performance achieved by the framework can be noticed when using only the accelerometer sensor, with the accuracy decreasing from 91.02% to 90.29%. However, considering the reduction in the volume of input data used by half, this result indicates a high efficiency when using only the accelerometer. Furthermore, it can be seen that this reduction in accuracy is essentially due to the approximately 3% decrease in the static activity classifier accuracy.

Lower sampling rates. The framework was also evaluated with the input data downsampled in order to evaluate the system performance with lower sampling rates. For this purpose, the downsampling was performed as described in Subsection 2.2.3, and the models were trained and tested with the resampled dataset for different rates. Fig. 37 shows the models performance behaviour for different sampling rates, and different sensors used, in the multiple wearable devices scenario.

Results show the first level classifier had slight variations on performance, achieving higher accuracy with lower sampling rates, however it is less significant than the changes in the other classifiers. The static activities classifier achieved the highest performance when using only 10 Hz as sampling rate, which may be due to the natural loss of signal

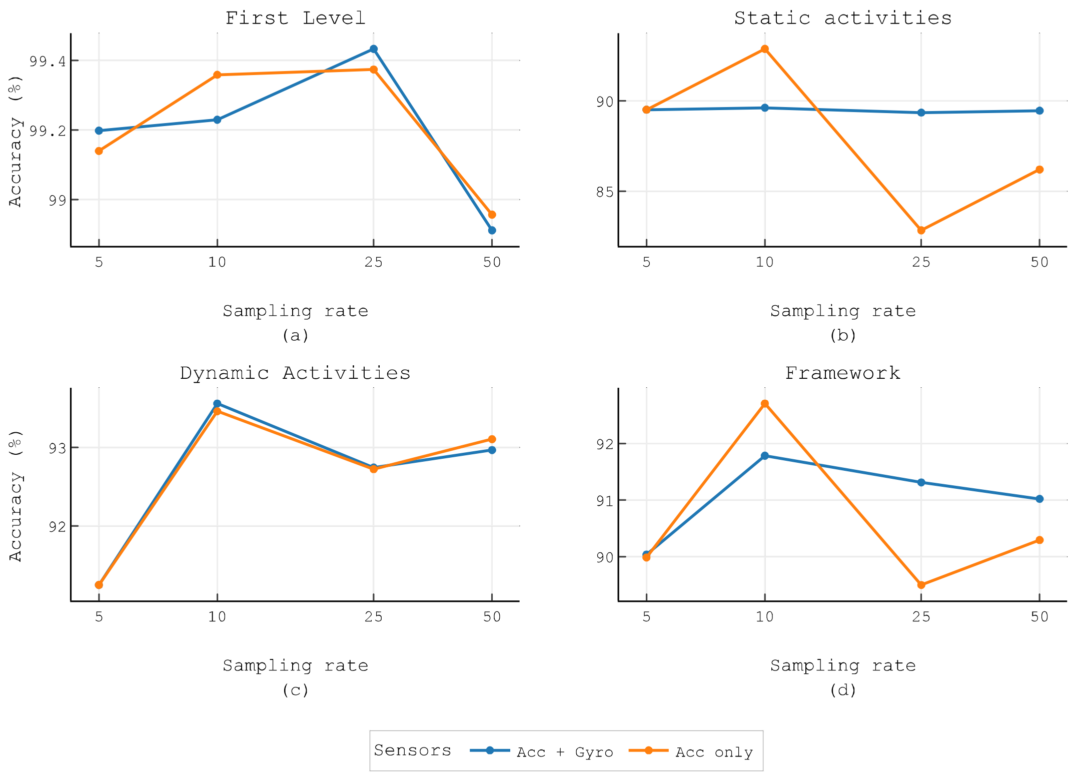


Figure 37 – Classifiers and framework accuracy on MHEALTH dataset for different sampling rates within the multiple sensors scenario. Comparison between using accelerometer and gyroscope data and using accelerometer-only data.

fidelity resulting from reduced sampling rate, and the consequent beneficial missing of minor oscillations. Since the mean value of acceleration is more relevant for static activities classification, the less noisy oscillations in signal the better. The dynamic activities classifier, in turn, also reached the best accuracy when using data sampled at 10 Hz for both sensor configurations, which also contributed to the framework accuracy gain with 10 Hz.

It is interesting to note that the framework performance improved when using a lower sampling rate than the original one, especially when using 10 Hz and only the accelerometer sensor, as shown in Fig. 37d. This is primarily due to the accuracy gain obtained in the static activity classifier, but it also had an important contribution of the dynamic activities classifier in 10 Hz operation. Results show that adopting a lower sampling rate and using only the accelerometer sensor can not only reduce the processed data volume and, consequently, the system computational cost, but also offer performance boost in terms of recognition rate. These findings indicate that an accurate HAR system can be built more efficiently for applications on resource-constrained devices.

Comparison with reference models. For a more incisive evaluation, the proposed HAR framework was compared with three reference models in terms of classification metrics, and the results are presented in Table 14. The framework performance was far

Table 14 – Comparison of classification metrics between the proposed framework and the reference models within the multiple sensors scenario on MHEALTH dataset.

Model	Sensor	Accuracy (%)	Precision (%)	Recall (%)	F1-score (%)
KNN	A, G	88.87	89.36	88.87	87.60
	A	92.09	92.67	92.09	90.84
Random Forest	A, G	93.70	93.63	93.70	93.22
	A	**95.00**	**95.74**	**95.00**	**94.28**
LightGBM	A, G	94.16	94.78	94.02	93.32
	A	94.65	95.62	94.51	93.96
Proposed framework	A, G	91.02	91.34	91.02	89.33
	A	90.29	91.96	90.29	88.91

A - accelerometer; G - gyroscope

below expectations, with lower performance than the other three models, when using only the accelerometer, and surpassing only the KNN model when using accelerometer and gyroscope data.

Fig. 38 shows a boxplot along with a swarm plot of the accuracy obtained by the models. The reference models had an increase in mean accuracy when using only accelerometer data, and the highest performance metrics were obtained by the Random Forest using only acceleration as input data. A possible explanation for the better performance of the reference models is the fact that they used 75 features as input. For this scenario with multiple sensors, the combination of several features extracted from sensors in different parts of the body can contribute to a more robust learning about the movements performed. However, extracting all those features can be very costly for a resource-constrained wearable device. On the other hand, merging data from multiple sensors can be a challenge for the CNN model to handle as input raw data, and therefore may have degraded the performance of the dynamic activities classifier.

When evaluating the models performance (see Fig. 39) using data with reduced sampling rate, the Random Forest maintained superior performance in the accelerometer-only use case. For the case with gyroscope data included, the LightGBM model was the winner. The best performance was observed with the Random Forest using only acceleration data and with a sampling rate of 25 Hz. Note that the most significant accuracy drop in the winner models occurred when the sampling frequency was reduced from 25 to 10 Hz. This may be an indicator that the higher sampling rate has a positive effect on the features extracted from the signal, which were used as input by these models.

6.1.1.2 Wrist-Worn Sensor Scenario

In the next scenario presented, only the wrist-worn sensor was considered. This is a much more challenging task, since it collects information from the sensor placed on a

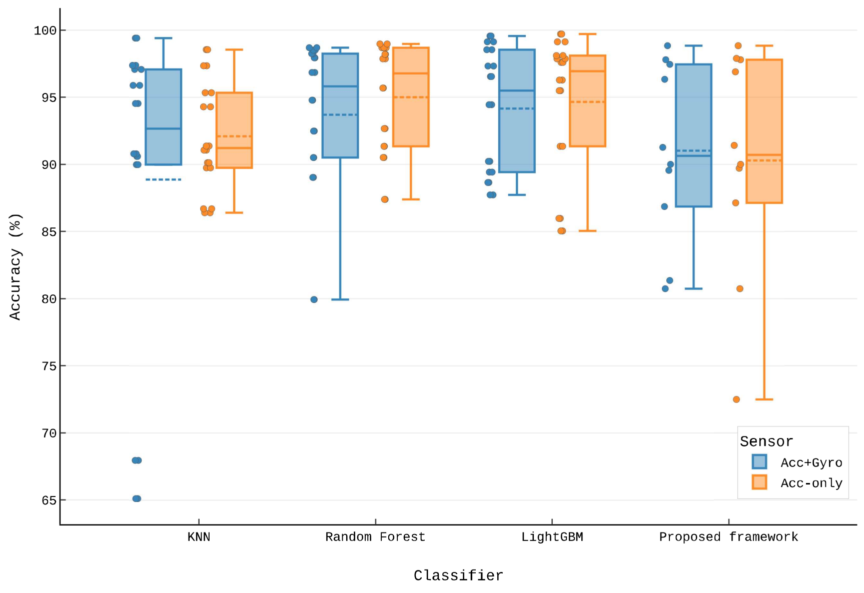


Figure 38 – Boxplot charts of accuracy of the proposed framework and the reference models on MHEALTH dataset within the multiple sensors scenario. Comparison between results using accelerometer and gyroscope data and using accelerometer-only data

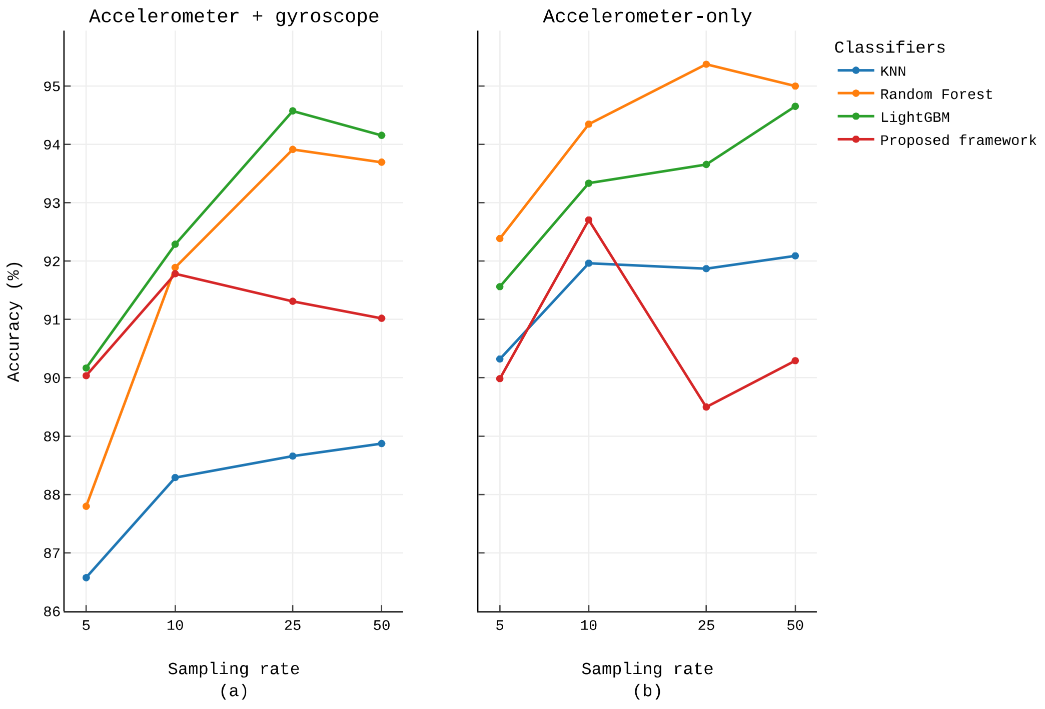


Figure 39 – Accuracy of the proposed framework and the reference models on MHEALTH dataset for different sampling rates within the multiple sensors scenario. Visualization for the accelerometer and gyroscope use case and accelerometer-only use case.

Table 15 – Framework performance on MHEALTH dataset using accelerometer and gyroscope data from the wrist-worn device.

Model	Accuracy (%)	Precision (%)	Recall (%)	F1-score (%)
First level classifier	99.30	99.31	99.30	99.30
Static activities classifier	82.61	82.29	82.61	77.93
Dynamic activities classifier	77.82	78.25	77.82	75.79
Framework	78.56	78.79	78.56	75.84

single body part, and the wrist is quite susceptible to movement artifacts. However, it is the most practical solution in usability and commercial aspects, as it can be embedded in smartwatches or smart bands, which are already part of people daily lives and are becoming more and more popular.

Accelerometer and gyroscope data. Firstly, the framework was evaluated using accelerometer and gyroscope sensors from the wrist-worn device. Table 15 shows the performance metrics for this case. The first level classifier was the only one to maintain a high level of performance, achieving 99.30% of accuracy.

The static activities classifier reached only 82.61% of accuracy, however, this is an expected result, since identifying activities related to posture (standing, sitting, lying down) with only the wrist motion sensor is extremely complex and may be even technically unfeasible. The confusion matrix (Fig. 40) shows the model accuracy for "sitting and relaxing" decreased, and the misclassification between this activity and "lying down" increased. The figure also shows the confusion matrices of three subjects which most contributed to the deterioration in performance.

The dynamic activities classifier achieved an accuracy of 77.82%, which, despite the performance drop in relation to the first scenario, is a satisfactory result since it is classifying 9 different activities using only data from a single device on the wrist. Comparing with the previous scenario, the confusion matrix in Fig. 41 shows the misclassification between "cycling" and "running" is more evident, and the accuracy of the activities "walking" and "climbing stairs" has dropped considerably, when using only the wrist-worn device.

In this use case, the first level Decision Tree mostly used entropy as splitting criterion, a mean max depth of 4.2, and the mean number of features of 2.3, being the y-axis acceleration minimum value and z-axis acceleration standard deviation value the most selected as best features. The standard deviation of one of the acceleration axes was again one of the features selected for this classifier, as in the previous cases, corroborating it is quite related to user movement actions.

For the static activities Decision Tree, the Gini splitting criterion was the most used, with a mean max depth of 2.1, mainly using 2 features: mean and maximum value both

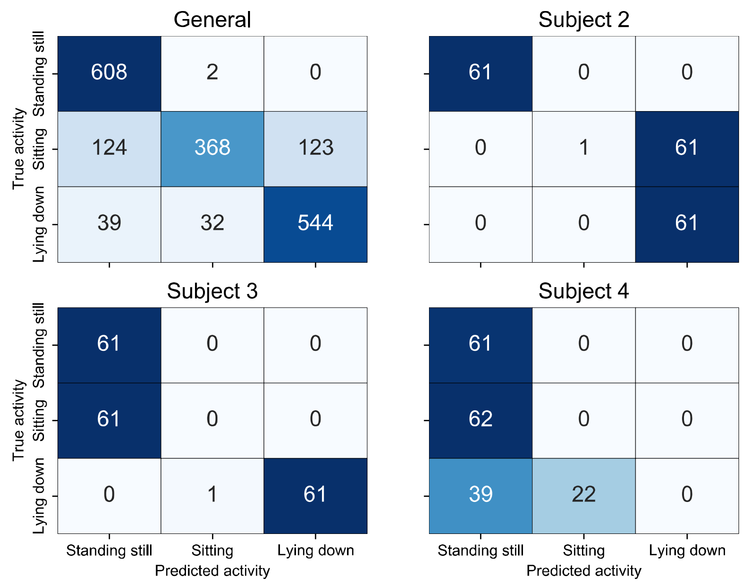


Figure 40 – Confusion matrices of static activities classifier performance, in general and for 3 subjects with the worst results, on MHEALTH dataset within the wrist-worn sensor scenario, using acceleration and gyroscope data.

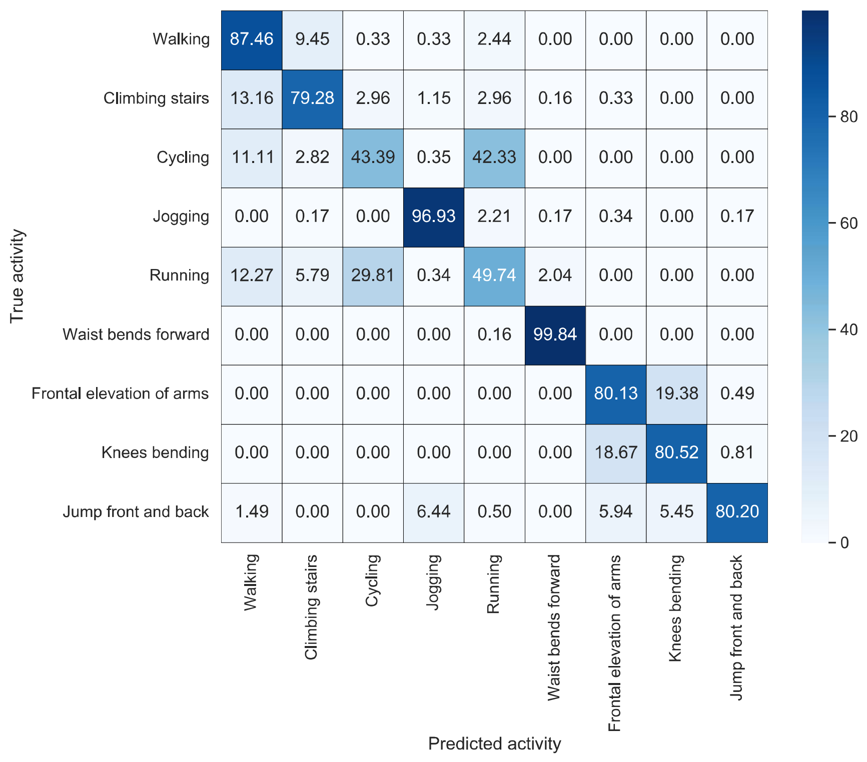


Figure 41 – Confusion matrix of dynamic activities classifier performance on MHEALTH dataset within the wrist-worn sensor scenario, using acceleration and gyroscope data.

Table 16 – Framework performance on MHEALTH dataset using only accelerometer data from the wrist-worn device.

Model	Accuracy (%)	Precision (%)	Recall (%)	F1-score (%)
First level classifier	99.29	99.29	99.29	99.29
Static activities classifier	79.24	79.34	79.24	76.09
Dynamic activities classifier	78.39	78.60	78.39	76.29
Framework	78.08	78.26	78.08	75.71

from y-axis acceleration. In static activities, the information of interest is in the individual's posture, which can be understood from the inclination of body parts in relation to the axis of gravity acceleration. In this context, the mean value of acceleration is often a significant feature, since it carries information related to the sensor inclination. The maximum value of the y-axis of acceleration was probably selected due to the value remaining within a certain range according to the posture, not reaching higher values for that position.

The optimized CNN for dynamic activities classification used only acceleration data from wrist sensor, and its architecture had 32 filters of size 5x1 in the two convolutional layers, stride of 1 in both layers, and 30 neurons in the fully connected layer. Once again, as in the prior scenario, the accelerometer has been shown to carry more relevant information than the angular velocity data.

Accelerometer-only data. When using only acceleration data, the framework recognition performance has only a slight decrease in classification metrics when compared to the prior case, as shown in Table 16, which indicates that the angular velocity does not contribute significantly to the model. The first level classifier achieved 99.29% of accuracy. The static activities classifier had a decrease in performance, reaching only 79.24% of accuracy. The dynamic activities classifier achieved an accuracy marginally higher than the previous case, 78.39%. Finally, the framework performance was almost the same as when using both accelerometer and gyroscope data, decreasing from 75.84% to 75.71% of F1-score. Nevertheless, it was much more efficient as it uses less data.

The first level Decision Tree mostly used Gini as splitting criterion, and a mean max depth of 4.1, while using 2.3 features, being the best features the same as in prior case: minimum value of y-axis acceleration, and standard deviation from z-axis acceleration. The static activities Decision Tree mostly used the Gini splitting criterion, and a mean max depth of 2.2, using only 2.2 features on average, mainly the y-axis acceleration mean, and y-axis acceleration SMA. When dealing with static activities, and consequently stationary acceleration signal, both mean and SMA features carry information about the sensor inclination. The CNN model had on average the same architecture as in the prior case.

Fig. 42 shows a comparison within this scenario between the use case with accelerometer and gyroscope data, and with accelerometer-only data. The findings are very similar to the first scenario analyzed, with data from multiple wearable sensors. The performance of the static activities model decreases when using only acceleration data, however the other models had the same or slightly better performance when using only acceleration. For that reason, the framework performance was basically the same. Note that the decrease in the accuracy of the static classifier is due to an extremely low accuracy, less than 40%, for one of the subjects.

Concluding, the results found show the framework can be much more efficient when using only the accelerometer, reducing the volume of data to be processed, and maintaining practically the same performance in terms of recognition rate.

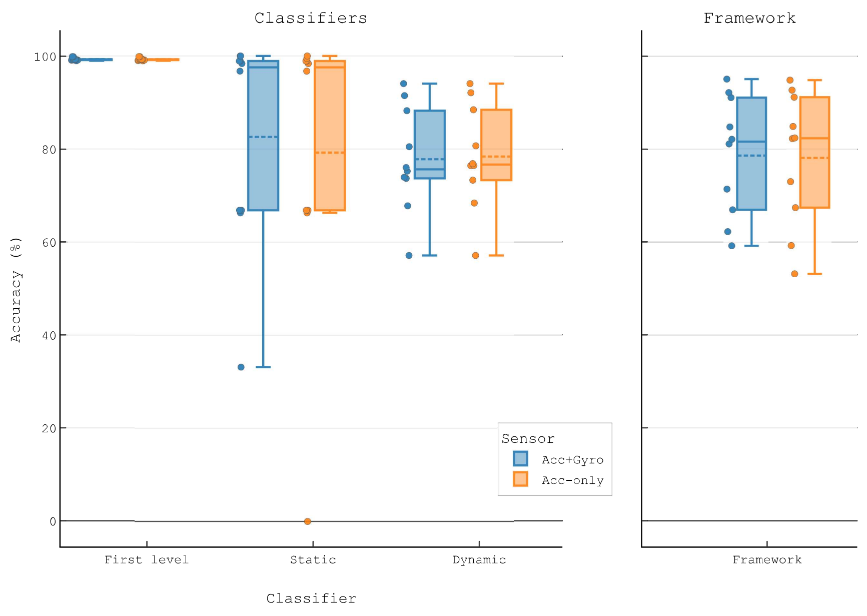


Figure 42 – Boxplot charts of models and framework accuracy on MHEALTH dataset within the wrist-worn sensor scenario. Comparison between results using accelerometer and gyroscope data and using accelerometer-only data.

Lower sampling rates. The next results show the framework performance for the wrist-worn device use case with lower sampling rates (see Fig. 43). The framework achieved the best accuracy when using only 10 Hz as sampling rate for both sensor configuration, reaching a slight higher score when the gyroscope data is included (80.08% against 79.42% of accuracy). The main contributor was the static activities classifier increase in this configuration, however the dynamic activities classifier also had a significant contribution.

As in the first scenario, the results show that a system with lower sampling rate, and consequently more efficient, can achieve even greater accuracy than a system with higher sampling rate. In this case, the inclusion of the gyroscope sensor implied a slight increase in accuracy in the best configuration (10 Hz), however, using only the accelerometer sensor can reduce the input data by half, ensuring greater efficiency.

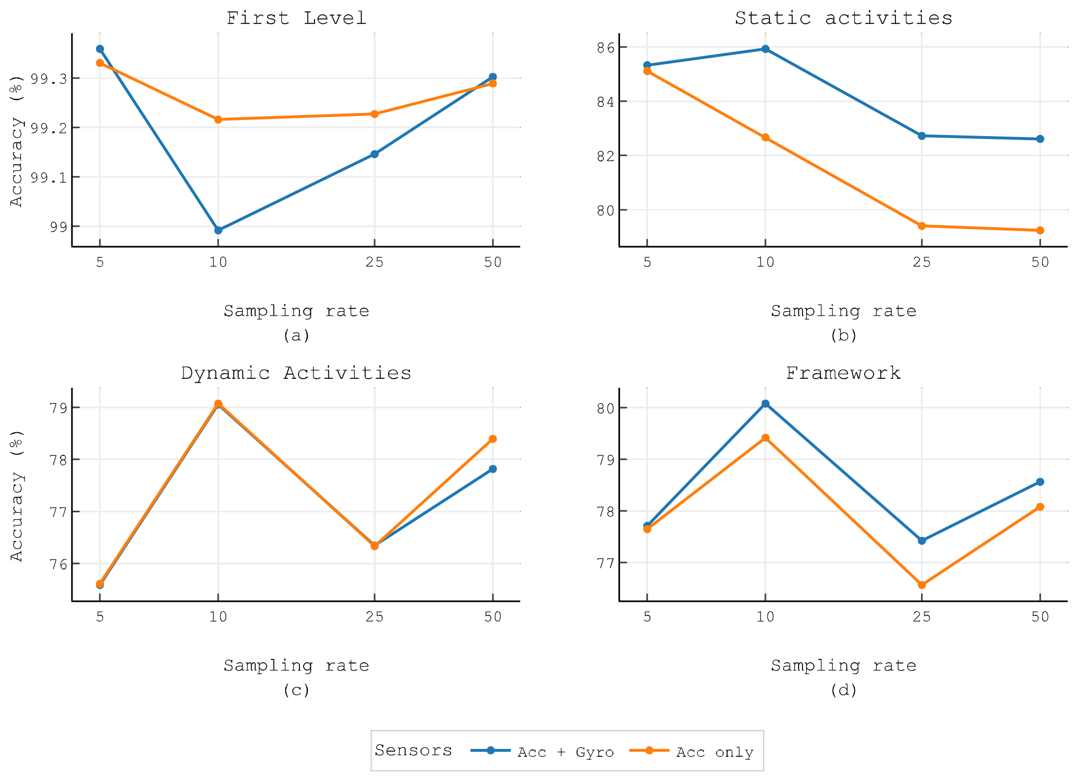


Figure 43 – Classifiers and framework accuracy on MHEALTH dataset for different sampling rates within the wrist-worn sensor scenario. Comparison between using accelerometer and gyroscope data and using accelerometer-only data.

Table 17 – Comparison of classification metrics between the proposed framework and the reference models within the wrist-worn sensor scenario on MHEALTH dataset.

Model	Sensor	Accuracy (%)	Precision (%)	Recall (%)	F1-score (%)
KNN	A, G	74.47	73.35	74.47	71.74
KNN	A	69.58	71.08	69.58	67.64
Random Forest	A, G	71.52	70.98	71.52	68.03
Random Forest	A	68.80	69.58	68.80	65.38
LightGBM	A, G	74.29	75.75	73.33	70.76
LightGBM	A	72.47	75.80	71.60	69.62
Proposed framework	A, G	**78.56**	**78.79**	**78.56**	**75.84**
Proposed framework	A	78.08	78.26	78.08	75.71

Comparison with reference models. For the wrist-worn sensor scenario, the proposed HAR framework achieved the best performance among the models evaluated, as presented in Table 17. For both the case with accelerometer and gyroscope data, and accelerometer-only data, the framework obtained classification metrics far greater than those achieved by the reference models.

Fig. 44 shows the boxplot charts and swarm plots of the results. In this case, unlike the previous scenario, the reference models achieved the best performance when the angular velocity is included in the input data. This may be due to the difficulty in recognizing activities with only one type of sensor when using data from only one device on the wrist. In contrast, the framework had nearly the same performance with or without the gyroscope data included.

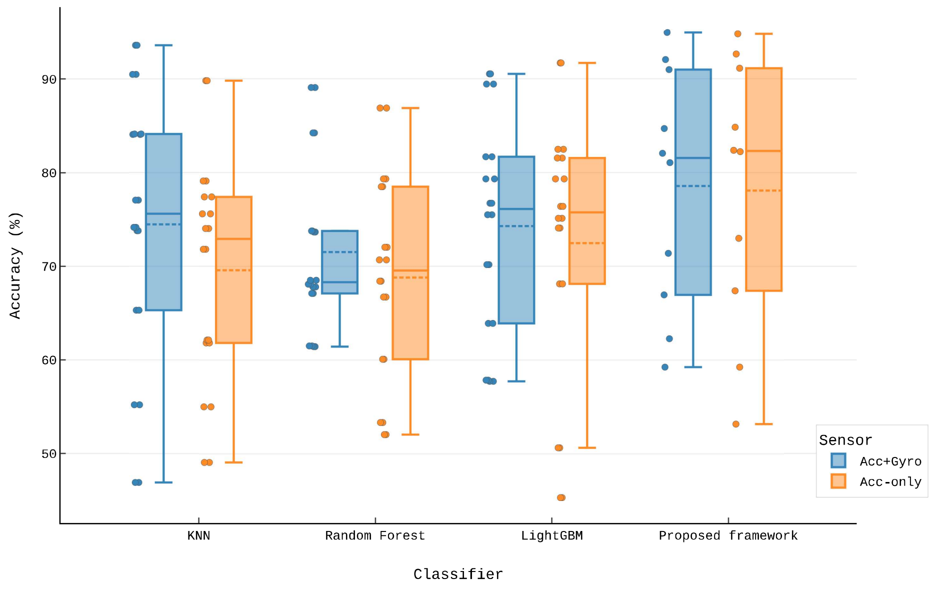


Figure 44 – Boxplot charts of accuracy of the proposed framework and the reference models on MHEALTH dataset within the wrist-worn sensor scenario. Comparison between results using accelerometer and gyroscope data and using accelerometer-only data

The findings indicate that the proposed framework is more suitable and efficient to be embedded in wrist-worn devices such as smartwatches and smart bands. Possibly, the features automatically extracted by the CNN model from the wrist sensors are more significant and representative than handcrafted features used in the other models.

In addition, when evaluating lower sampling rates (see Fig. 45), it can be noted that with only 5 Hz the framework achieved the same level of accuracy, and with 10 Hz it achieves the best performance, both for the case using accelerometer and gyroscopy and using only accelerometer. When the gyroscope data is included, the accuracy at 10 Hz is just slightly higher (around 0.5%), however, the system has to deal with greater volume of data. Results also show that the proposed framework performs even better than the reference models with reduced sampling frequency.

In conclusion, the findings show that the framework can be quite efficient when using 10 Hz as sampling rate and only the acceleration data, and still obtain much more satisfactory results than the competing reference models.

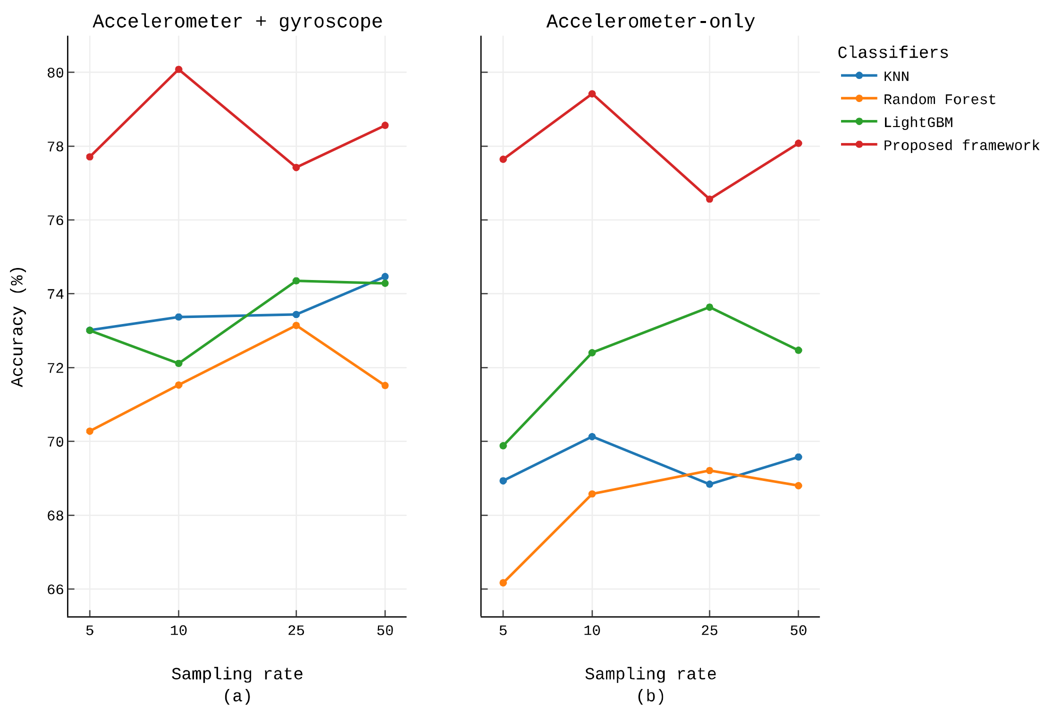


Figure 45 – Accuracy of the proposed framework and the reference models on MHEALTH dataset for different sampling rates within the wrist-worn sensor scenario. Visualization for the accelerometer and gyroscope use case and accelerometer-only use case.

6.1.2 Results on PAMAP2 Dataset

This subsection presents the results obtained on the PAMAP2 dataset. Initially, in the first scenario, the performance achieved when using multiple sensors are presented. Next, the results obtained by using only the sensor on the wrist are shown and discussed, in the second scenario evaluated.

6.1.2.1 Multiple Sensors Scenario

Accelerometer and gyroscope data. Firstly, the accelerometer and gyroscope data from chest, wrist, and ankle-worn devices were used to develop and test the classifiers and the framework. The results achieved are presented in Table 18.

The first level classifier performed far below expectations, considering that it executes a relatively simple task. The accuracy obtained was only 93.96%, much lower than the accuracy achieved on the MHEALTH dataset. One of the reasons may be some of the activities included in this dataset, which, despite being labeled as dynamic, can be performed with few movements, such as ironing and vacuuming, and therefore cause confusion in the model. The static activity and dynamic activity classifiers also performed worse with this dataset than with the MHEALTH dataset, indicating that the PAMAP2 dataset is more challenging than the previous one.

Table 18 – Framework performance on PAMAP2 dataset using accelerometer and gyroscope data from multiple wearable devices.

Model	Accuracy (%)	Precision (%)	Recall (%)	F1-score (%)
First level classifier	93.96	94.23	93.96	93.91
Static activities classifier	82.68	83.54	82.68	82.35
Dynamic activities classifier	74.00	79.19	74.00	73.46
Framework	71.99	75.83	71.99	71.49

Fig. 46 presents the general confusion matrix for the static activities classifier, and other three for specific subjects with the poorest results. Note that the main misclassification occurred between “sitting” and “standing”, as in the case of MHEALTH dataset. This result reinforces the idea that in some cases it can be difficult to understand whether the user is standing or sitting even with three devices.

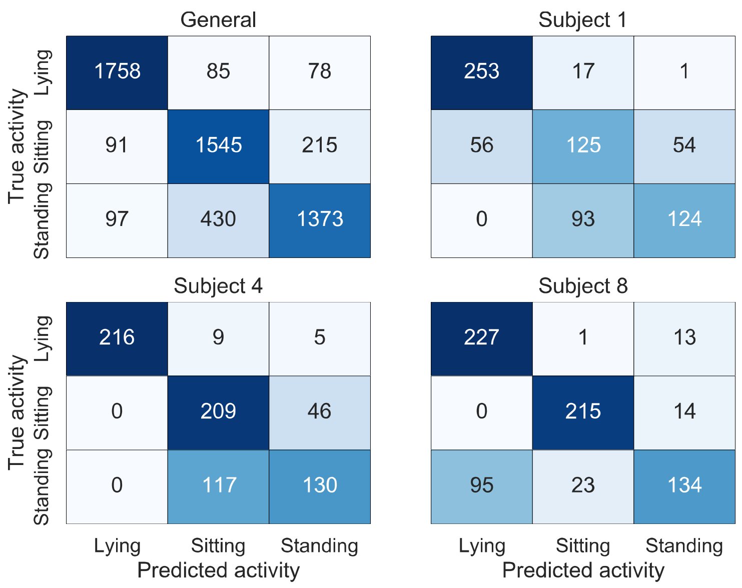
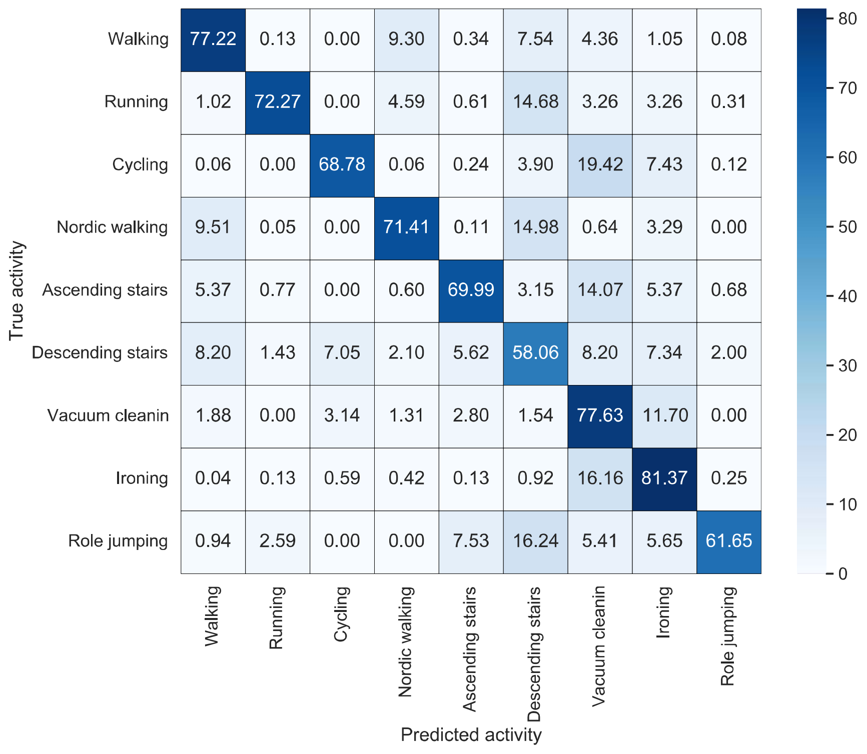


Figure 46 – Confusion matrices of static activities classifier performance, in general and for 3 subjects with the worst results, on PAMAP2 dataset within the multiples sensors scenario, using acceleration and gyroscope data.

The confusion matrix for the CNN of the dynamic activities classifier is shown in Fig. 47. The results show a distribution of confusion between activities greater than that observed with the MHEALTH dataset. In this case, the low precision of “ascending stairs” stands out, as well as for “vacuum cleaning”. On the other hand, the activities “running” and “role jumping” achieved a low rate of false positives, reaching a high precision, despite not so high accuracy.

In general, the results show that the model’s difficulties were not limited to specific activities, indicating that PAMAP2 is a more challenging dataset. One of the reasons may be the absence of standard in the construction of the dataset, in which most subjects



True activity \ Predicted activity	Walking	Running	Cycling	Nordic walking	Ascending stairs	Descending stairs	Vacuum cleanin	Ironing	Role jumping
Walking	77.22	0.13	0.00	9.30	0.34	7.54	4.36	1.05	0.08
Running	1.02	72.27	0.00	4.59	0.61	14.68	3.26	3.26	0.31
Cycling	0.06	0.00	68.78	0.06	0.24	3.90	19.42	7.43	0.12
Nordic walking	9.51	0.05	0.00	71.41	0.11	14.98	0.64	3.29	0.00
Ascending stairs	5.37	0.77	0.00	0.60	69.99	3.15	14.07	5.37	0.68
Descending stairs	8.20	1.43	7.05	2.10	5.62	58.06	8.20	7.34	2.00
Vacuum cleanin	1.88	0.00	3.14	1.31	2.80	1.54	77.63	11.70	0.00
Ironing	0.04	0.13	0.59	0.42	0.13	0.92	16.16	81.37	0.25
Role jumping	0.94	2.59	0.00	0.00	7.53	16.24	5.41	5.65	61.65

Figure 47 – Confusion matrix of dynamic activities classifier performance on PAMAP2 dataset within the multiples sensors scenario, using acceleration and gyroscope data.

performed only some of the activities.

In average of the 10 subjects tested, the first level Decision Tree mostly used Gini as the splitting criterion, a mean max depth of 3.25, and a mean of 2.25 features, mainly z-axis acceleration minimum value from chest-worn device, and x-axis acceleration standard deviation from wrist-worn device. As for the MHEALTH dataset, even with angular velocity available, the best features to classify between static and dynamic activities were derived from acceleration data.

The Decision Tree for static activities classification used the Gini splitting criterion for most subjects, and a mean max depth of 2.88, using 2.63 features in average, being x-axis acceleration mean from wrist sensor, x-axis acceleration SMA from wrist sensor, and y-axis acceleration mean from chest sensor the most selected ones. In the case of static activities, mean and SMA carries information about the position of the body part in relation to the direction of gravity acceleration. Therefore, in this case, the positioning of the user's chest and wrist demonstrated to be the most important attributes for the activity classification.

The optimized CNN for dynamic activities classification used acceleration and gyroscope data from wrist and ankle sensors, and its architecture was defined by 2 convolutional layers, with 64 and 32 filters of size 5x1, stride of 1 in both layers, and 30 neurons in the fully connected layer. Once again, the use of wrist and ankle sensors

Table 19 – Framework performance on PAMAP2 dataset using only accelerometer data from multiple wearable devices.

Model	Accuracy (%)	Precision (%)	Recall (%)	F1-score (%)
First level classifier	92.50	92.94	92.50	92.25
Static activities classifier	82.68	83.54	82.68	82.35
Dynamic activities classifier	71.86	75.91	71.86	70.26
Framework	69.44	72.62	69.44	68.14

together demonstrated to be more efficient than using all the devices.

Accelerometer-only data. In a second evaluation with multiple wearable sensors, the framework was developed and tested using only the accelerometer data, and Table 19 presents the results. The performances of first level and static activities classifiers were very similar to the prior case, when the gyroscope data were included, since the main features in that case were derived from the acceleration data. The main performance drop occurred with the dynamic activities classifier, which makes sense, given that CNN's best architecture included gyroscope data.

In this case, the first level Decision Tree used the same hyperparameters configuration and features as the previous case, suggesting that all the subjects always used only acceleration data for this classifier, even with the angular velocity available. The same situation occurred with the static activities classifier. For the dynamic activities classifier, when using only acceleration data the optimized CNN used data from wrist and ankle sensors, and its architecture was defined by 2 convolutional layers, with 64 and 32 filters of size 3x1 and 5x1, respectively, stride of 1 in both layers, and 20 neurons in the fully connected layer.

Fig. 48 presents a comparison between the two cases tested in this scenario with multiple sensors. The graph shows a boxplot chart of accuracy for each model and the framework, with the mean value represented by a dashed line. It can be observed that using only accelerometer data caused a decrease in accuracy only in the dynamic activities classifier, which causes a drop in the performance of the framework.

In summary, for this dataset and considering this scenario with multiple wearable sensors, the use of gyroscope data proved to be important for the CNN of the dynamic activities classifier, since the F1-score achieved by the framework dropped by 3.35% when using only acceleration data.

Lower sampling rates. Next, the framework was analyzed with lower sampling rates. Once again, the downsampling was performed as described in Subsection 2.2.3, and the classifiers were trained and tested with the resampled dataset for different rates. Fig. 49 shows the performance achieved for different sampling rates and sensors, in the multiple wearable devices scenario.

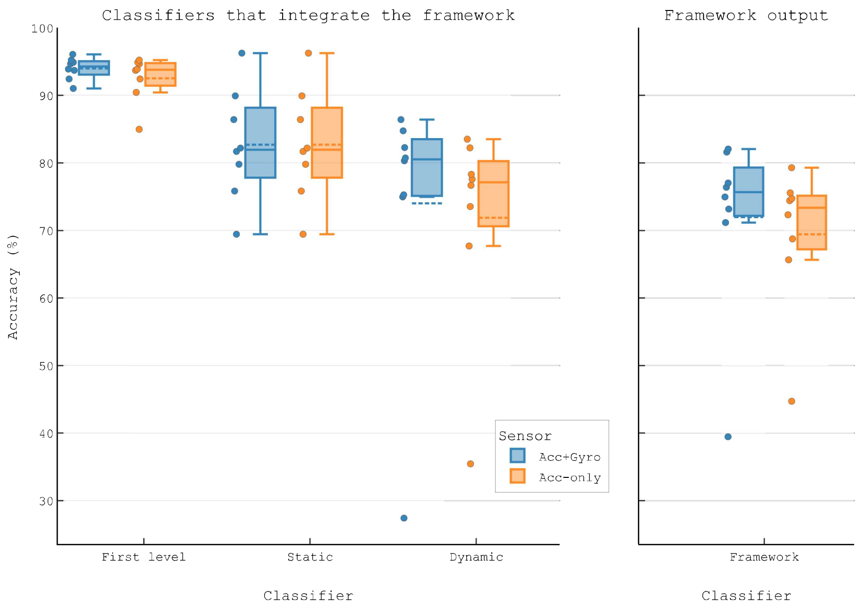


Figure 48 – Boxplot charts of models and framework accuracy on PAMAP2 dataset within the multiple sensors scenario. Comparison between results using accelerometer and gyroscope data and using accelerometer-only data.

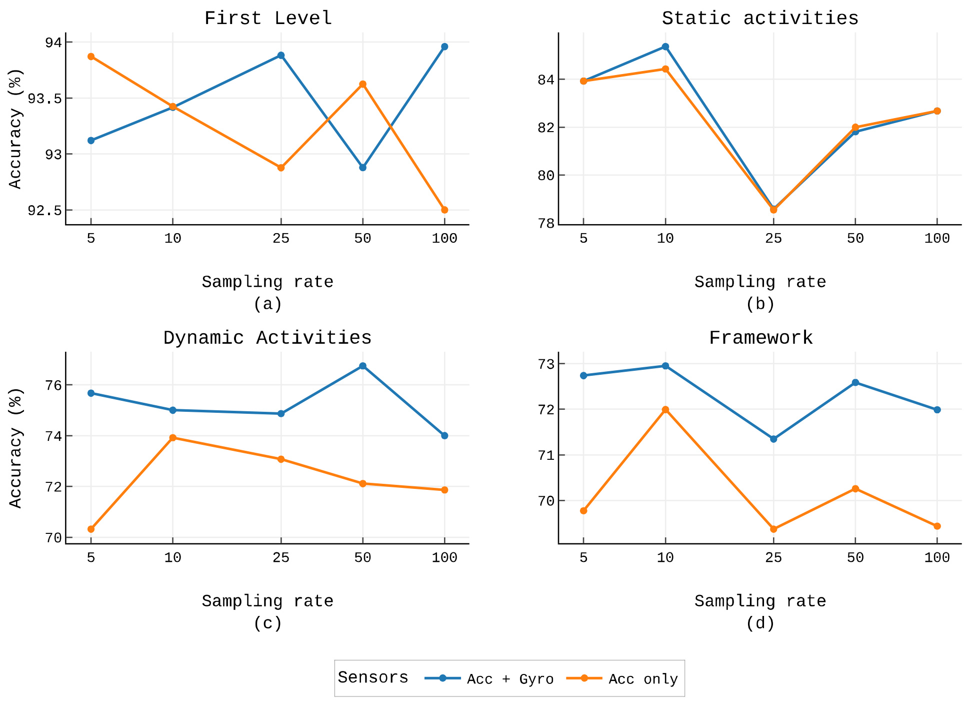


Figure 49 – Classifiers and framework accuracy on PAMAP2 dataset for different sampling rates within the multiple sensors scenario. Comparison between accelerometer and gyroscope data and accelerometer-only data.

Table 20 – Comparison of classification metrics between the proposed framework and the reference models within the multiple sensors scenario on PAMAP2 dataset.

Model	Sensor	Accuracy (%)	Precision (%)	Recall (%)	F1-score (%)
KNN	A, G	83.95	86.23	83.95	83.73
	A	80.45	83.23	80.45	80.18
Random Forest	A, G	**86.15**	**87.86**	**86.15**	**86.15**
	A	85.17	87.13	85.17	85.08
LightGBM	A, G	85.45	80.93	79.07	79.22
	A	82.11	81.29	78.04	77.83
Proposed framework	A, G	71.99	75.83	71.99	71.49
	A	69.44	72.62	69.44	68.14

A - accelerometer; G - gyroscope

The first level classifier oscillated within a range of 1.5%, maintaining high accuracy when using only acceleration and 5 Hz as sampling rate. The static activities classifier achieved the highest performances with lower sampling rates, being the best case with 10 Hz. As highlighted in the analysis with the MHEALTH dataset, the reduction of the sampling rate causes a smoothing of the signal that can be beneficial for the classification of static activities, since in this case the mean value of acceleration is more important than rapid oscillations. The dynamic activities classifier reached the highest accuracy with 50 Hz as sampling rate when using accelerometer and gyroscope data, although the decrease for lower rates was small. On the other hand, when using only acceleration data, the best accuracy was accomplished with a sampling rate of 5 Hz. Finally, the best performance of the framework was observed when using 10 Hz as sampling rate for both data use cases.

The framework performance improved when using a lower sampling rate than the original one, mainly when using 10 Hz, as shown in Fig. 49d. This is mostly due to the accuracy gain achieved with the static activity classifier, however, this result was also supported by the other classifiers. Once more, results show that adopting a lower sampling rate can contribute to better performance on classification rate, in addition to the expected reduction of computational cost, memory footprint, and energy consumption. These results contribute to show that a more efficient system can be developed without loss of classification accuracy.

Comparison with reference models. Here the results of the comparison with the reference models are presented. Table 20 shows the classification metrics achieved by the models. The performance of the proposed framework was much lower than the result obtained by the reference models. As with the MHEALTH dataset, the classifiers that used several handcrafted features as input performed better than the framework.

Fig. 50 shows a boxplot along with a swarm plot of the accuracy obtained by the models. The best results were achieved by the Random Forest model, both for the accelerometer and gyroscope use case, and for the accelerometer-only use case. For this model, discarding angular velocity did not result in a significant loss of performance. As stated before, the use of several features (75) as input for the models may be a more efficient strategy when dealing with activity classification with multiple sensors data.

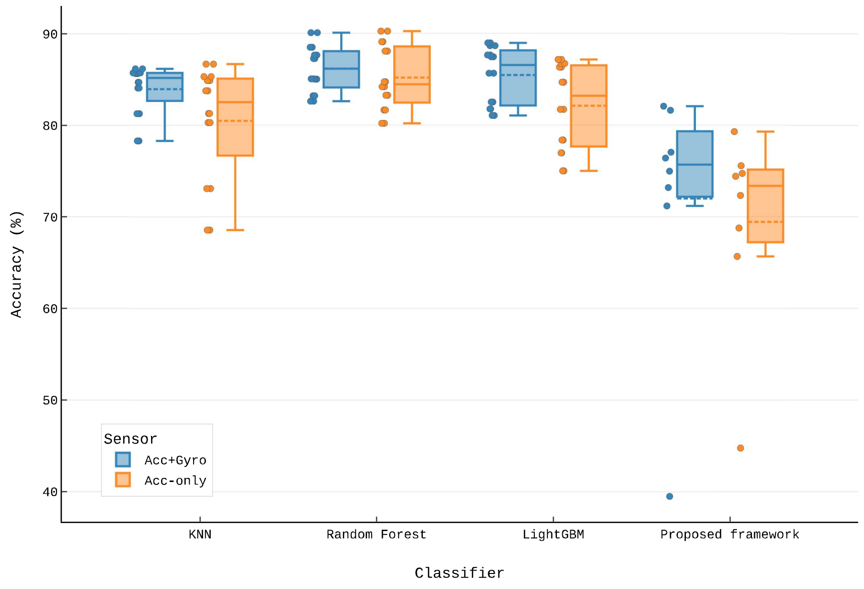


Figure 50 – Boxplot charts of accuracy of the proposed framework and the reference models on PAMAP2 dataset within the multiple sensors scenario. Comparison between results using accelerometer and gyroscope data and using accelerometer-only data

Using lower sample rates did not have a major impact on the accuracy of the models in the case of use with accelerometer and gyroscope data, as shown in Fig. 51. On the contrary, when using only acceleration data, the accuracy decrease was quite evident, with the exception of LightGBM model, which performed better at 25 Hz. One more time, these findings contributed to the understanding that a higher sampling rate has a positive impact on the extraction of handcrafted features used by the reference models.

6.1.2.2 Wrist-Worn Sensor Scenario

In this subsection, the analysis of the framework performance using only wrist-worn sensor data are presented. It is important to emphasize again that this is a more challenging scenario, however more motivating, since the implementation of the framework on smartwatches in a real world scenario is more viable.

Accelerometer and gyroscope data. As in the first scenario, the first tests were performed with accelerometer and gyroscope data, and Table 21 presents the classification metrics for this case. The first level classifier maintained a similar performance to the first

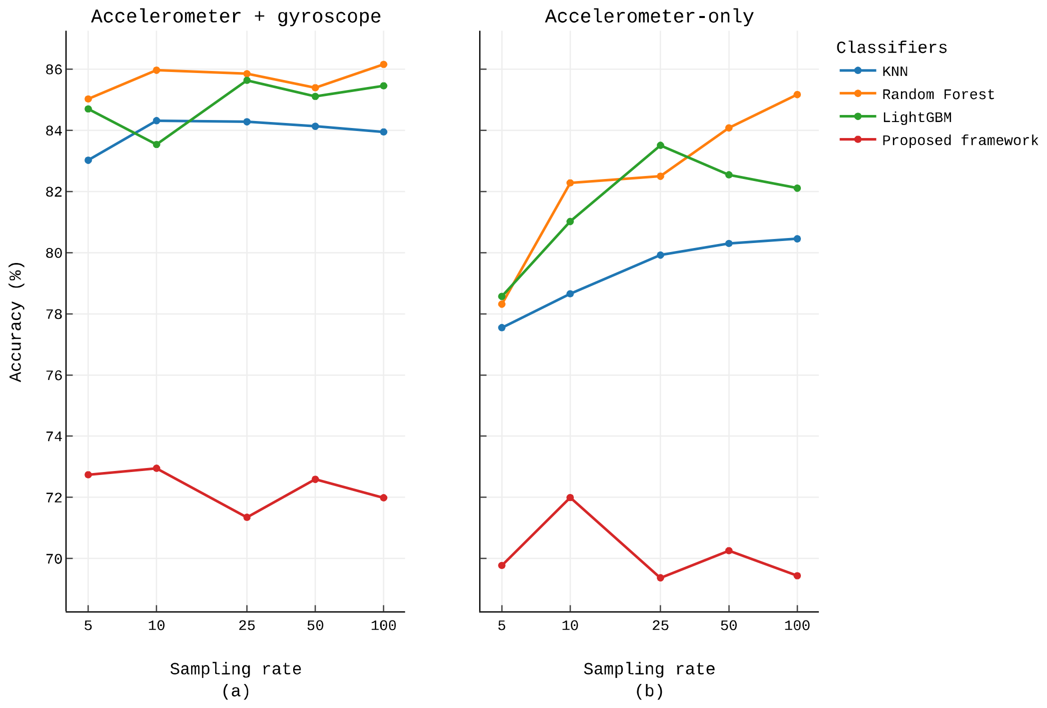


Figure 51 – Accuracy of the proposed framework and the reference models on PAMAP2 dataset for different sampling rates within the multiple sensors scenario. Visualization for the accelerometer and gyroscope use case and accelerometer-only use case.

Table 21 – Framework performance on PAMAP2 dataset using accelerometer and gyroscope data from the wrist-worn device.

Model	Accuracy (%)	Precision (%)	Recall (%)	F1-score (%)
First level classifier	93.28	93.33	93.28	93.28
Static activities classifier	78.39	78.04	78.39	77.29
Dynamic activities classifier	69.03	72.38	69.03	67.92
Framework	66.93	69.08	66.93	65.89

scenario evaluated, while the other classifiers of the framework had a significant decrease in the classification metrics, resulting in a 5.60% drop in the F1-score of the framework.

In this scenario, the confusion matrix of the static activities classifier (see Fig. 52) shows the misclassification is higher between “lying” and “sitting” than in the multi-sensor scenario. This variation is probably due to the particularities of body parts positioning of each subject, as the position of the wrist can vary greatly regardless of posture. Again, it is important to point out that the classification of static activities with only a single device on the wrist is quite challenging, and it is also not highly required for activity tracking applications.

The confusion matrix presented in Fig. 53 shows the misclassification in general is higher in this scenario, except for “running” and “role jumping”, for which the model

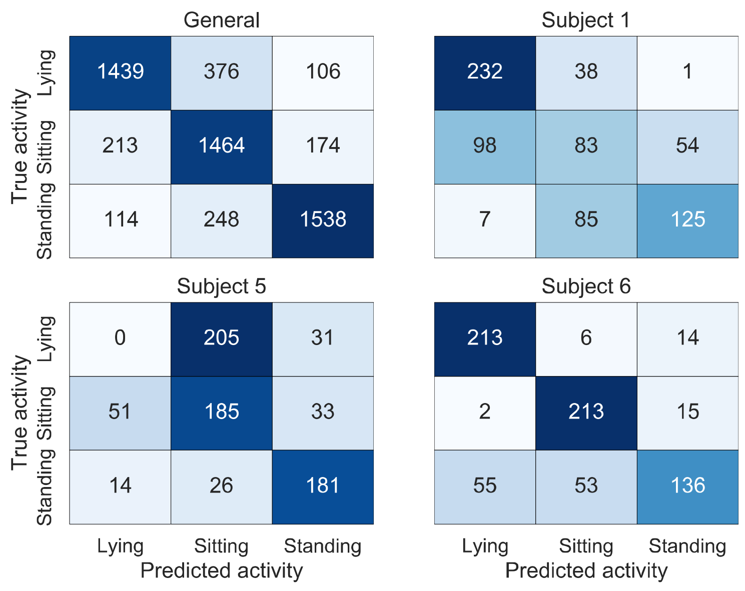


Figure 52 – Confusion matrices of static activities classifier performance, in general and for 3 subjects with the worst results, on PAMAP2 dataset within the wrist-worn sensor scenario, using acceleration and gyroscope data.

achieved higher accuracy.

When using accelerometer and gyroscope data in this scenario, the first level Decision Tree mainly used Gini as splitting criterion, a mean max depth of 3.75, and the mean number of features of 2.25, being the x-axis acceleration standard deviation and x-axis acceleration minimum value the most selected features. Once more, the standard deviation and minimum value appeared as the most important features for separating static and dynamic activities.

For the static activities Decision Tree, the entropy splitting criterion was the most used, with a mean max depth of 4, using in average 2.5 features, being the minimum value and SMA both from x-axis acceleration, and SMA from z-axis acceleration the most used.

The optimized CNN for dynamic activities classification used acceleration and gyroscope data from wrist sensor, and its architecture had 32 filters of size 5x1 in the two convolutional layers, stride of 1 in both layers, and 30 neurons in the fully connected layer.

Accelerometer-only data. When using only acceleration data, the first level and static activities classifiers achieved a recognition performance very similar to the case when the gyroscope data was included. However, the dynamic activities classifier metrics were better than in the prior case, which contributed to the better results accomplished by the framework, as show in Table 22. In addition to improving the performance, the data usage is reduced, thus it is a considerably more efficient configuration.

The first level Decision Tree mostly used Gini as splitting criterion, and a mean max depth of 3.75, while using 2.25 features, being the best features the same as in previous

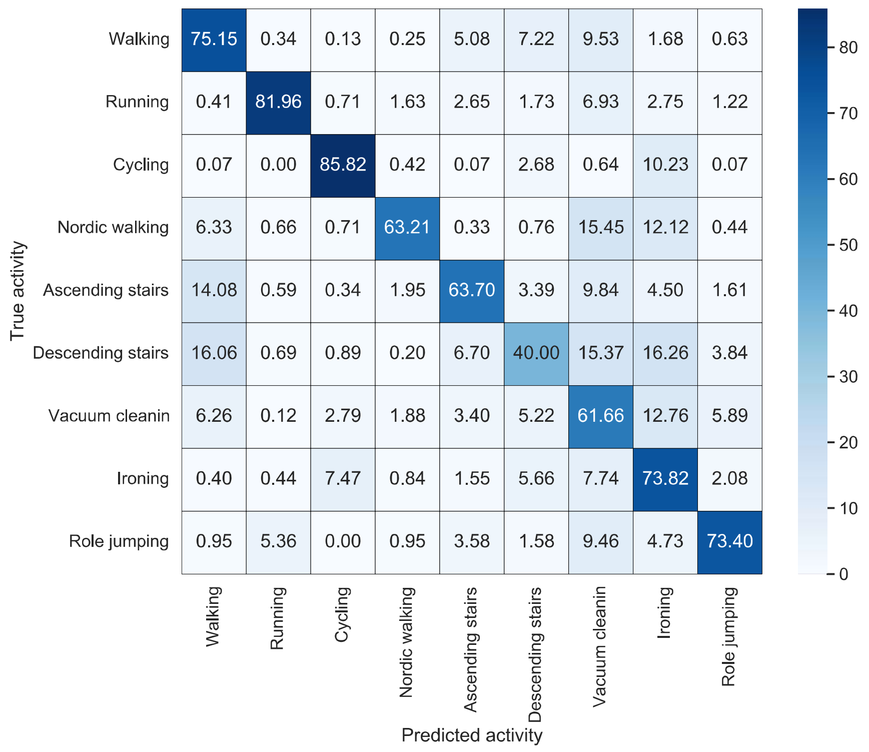


Figure 53 – Confusion matrix of dynamic activities classifier performance on PAMAP2 dataset within the wrist-worn sensor scenario, using acceleration and gyroscope data.

Table 22 – Framework performance on PAMAP2 dataset using only accelerometer data from the wrist-worn device.

Model	Accuracy (%)	Precision (%)	Recall (%)	F1-score (%)
First level classifier	92.93	92.95	92.93	92.87
Static activities classifier	78.44	78.05	78.44	77.34
Dynamic activities classifier	72.18	76.01	72.18	71.43
Framework	68.80	71.24	68.80	67.98

case: standard deviation and minimum value both from x-axis acceleration. The static activities Decision Tree mostly used the entropy splitting criterion, and a mean max depth of 4, using only 2.5 features on average, mainly the minimum value and SMA from x-axis acceleration, and SMA from y-axis acceleration. The CNN model had on average the same architecture as in the prior case.

Fig. 54 presents a comparison within the wrist-worn scenario between the use case with accelerometer and gyroscope data, and with accelerometer-only data. As stated before, the framework accuracy was slightly higher when using only acceleration data, mainly due to the better performance of the dynamic activities classifier.

Lower sampling rates. The results from testes realized with lower sampling rates, are shown in Fig. 55. It can be noted that the accuracy when using acceleration and

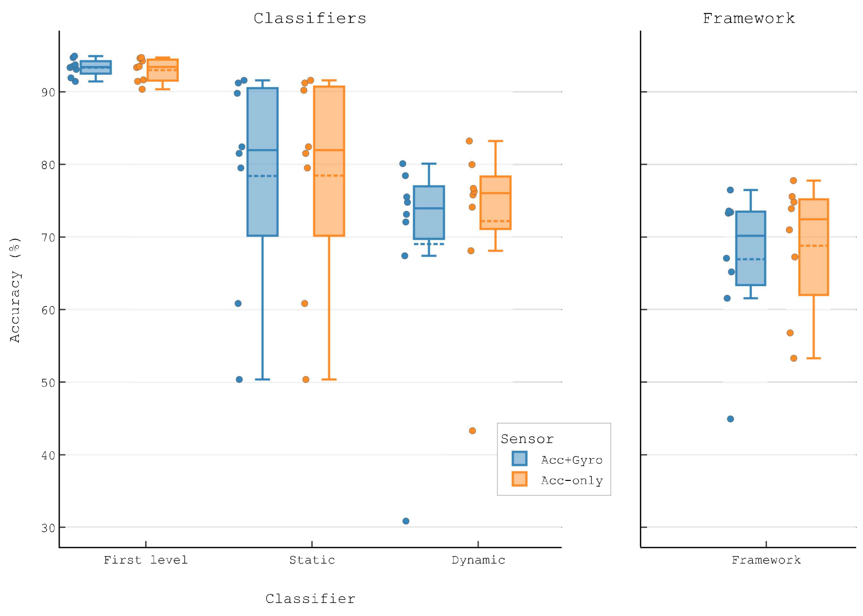


Figure 54 – Boxplot charts of models and framework accuracy on PAMAP2 dataset within the wrist-worn sensor scenario. Comparison between results using accelerometer and gyroscope data and using accelerometer-only data.

angular velocity remained within a narrow range, while the accuracy for the use case of acceleration had a significant reduction when decreasing the sampling rate, achieving better performance with the original rate of 100 Hz. In this case, when using only acceleration and only a single device on the wrist, the findings indicate the framework needs a higher sampling rate for better understanding signal patterns, especially for the dynamic activities classifier.

Comparison with reference models. For the wrist-worn sensor scenario, the best performance was achieved by the Random Forest model using accelerometer and gyroscope data (F1-score of 68.94%), followed by the proposed framework using only acceleration data (F1-score of 67.98%), as shown in Table 23. The proposed framework proves to be more efficient, since it uses less information to achieve practically the same result. In addition, the Random Forest model needs to extract several features to achieve this performance, making it very computationally expensive.

Fig. 56 shows how the reference models lose performance when they use only the accelerometer data, unlike the proposed framework, which even has superior performance. The framework guarantees performance maintenance even using less information, which indicates that it has a greater ability to extract relevant features from a single device in the wrist. The findings, once again, show the proposed energy-efficient HAR framework is more appropriate for wrist-worn devices such as smartwatches and smart bands.

In relation to the evaluation of input data with lower sampling rates for the

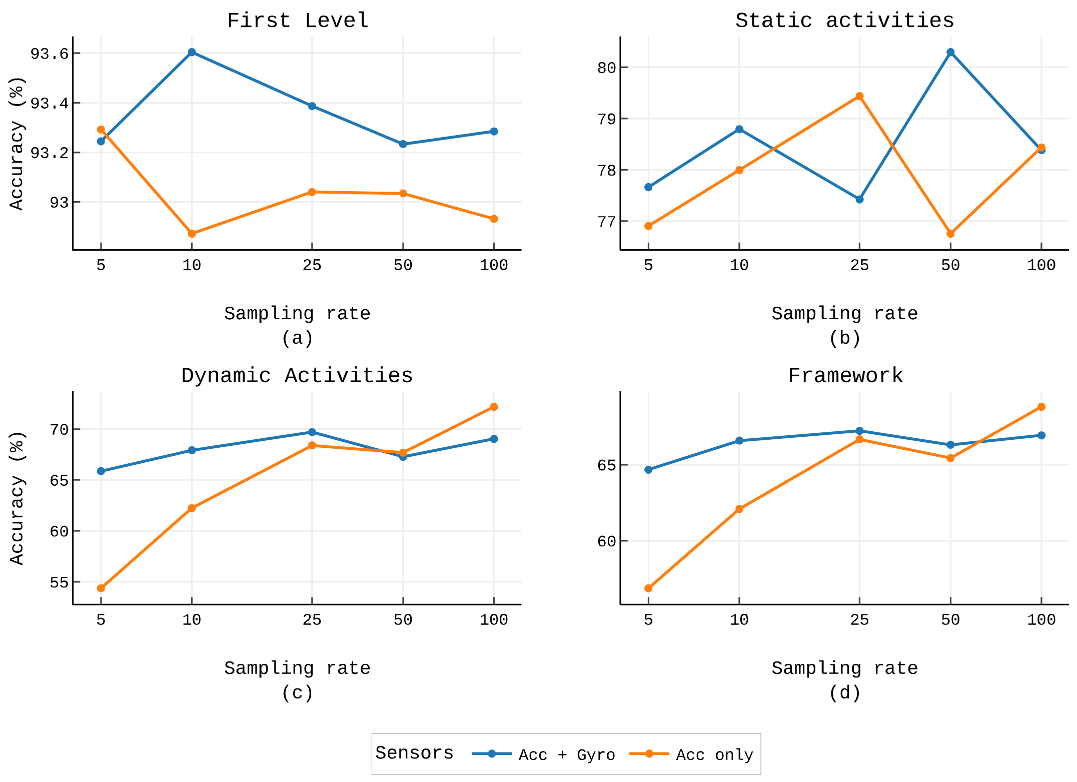


Figure 55 – Classifiers and framework accuracy on PAMAP2 dataset for different sampling rates within the wrist-worn sensor scenario. Comparison between accelerometer and gyroscope data and accelerometer-only data.

Table 23 – Comparison of classification metrics between the proposed framework and the reference models within the wrist-worn sensor scenario on PAMAP2 dataset.

Model	Sensor	Accuracy (%)	Precision (%)	Recall (%)	F1-score (%)
KNN	A, G	67.52	69.96	67.52	66.48
	A	57.39	61.06	57.39	56.41
Random Forest	A, G	70.46	**72.46**	**70.46**	**68.94**
	A	65.28	67.29	65.28	63.34
LightGBM	A, G	**72.47**	68.69	65.04	64.63
	A	66.01	61.67	58.60	57.65
Proposed framework	A, G	66.93	69.08	66.93	65.89
	A	68.80	71.24	68.80	67.98

A - accelerometer; G - gyroscope

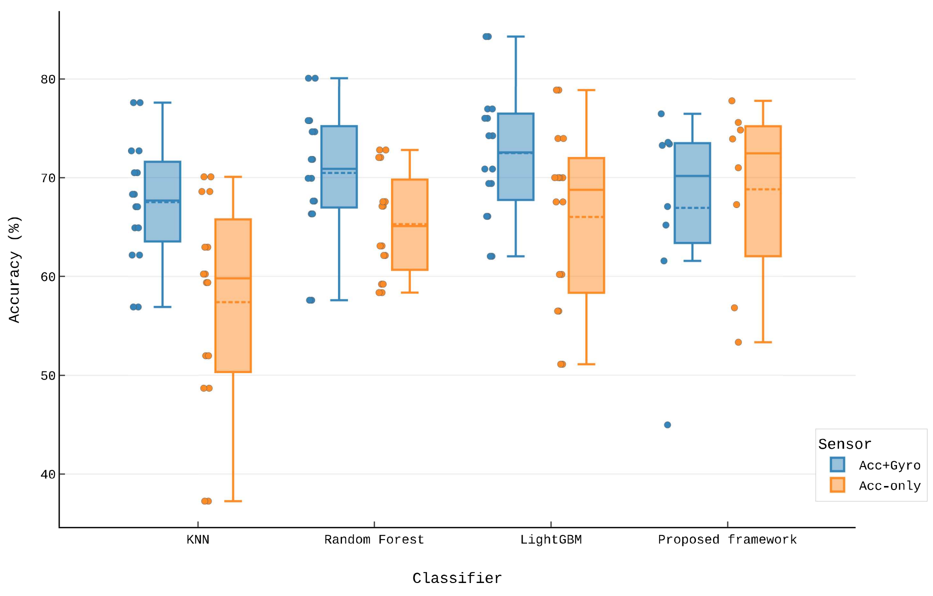


Figure 56 – Boxplot charts of accuracy of the proposed framework and the reference models on PAMAP2 dataset within the wrist-worn sensor scenario. Comparison between results using accelerometer and gyroscope data and using accelerometer-only data. The dashed line refers to the mean value.

PAMAP2 dataset (see Fig. 57), in general terms, it is noticed that there is a loss of performance of all the evaluated models when the sampling rate is reduced. One reason for this may be that this dataset has more complex activities and needs a higher sampling rate to guarantee the extraction of the patterns that better characterize the activities.

6.1.3 Results on WISDM-2019 Dataset

Finally, this subsection presents the results obtained on the WISDM-2019 dataset. As a result of the inconsistency between the timestamps of the smartwatch and the smartphone samples, synchronization was not technically feasible. Therefore, only the wrist-worn device was considered for the tests with this dataset, as it is a more viable use case to implement commercially. In the sequence, the results are presented using accelerometer and gyroscope data and only accelerometer data, in addition to the analysis with lower sampling rates, and the comparison with reference models.

Accelerometer and gyroscope data. Initially, the framework was tested considering accelerometer and gyroscope data. Table 24 shows the results for this case. Fourteen activities were used in this dataset, and among the dynamic ones there are some which can be easily confused with static activities, such as typing, and writing, for example. With this in mind, it can be said that the performance of the first level classifier (F1-score of 91.04%) was quite reasonable. However, other strategies may work better when activities of daily living with little movement are included in the target group.

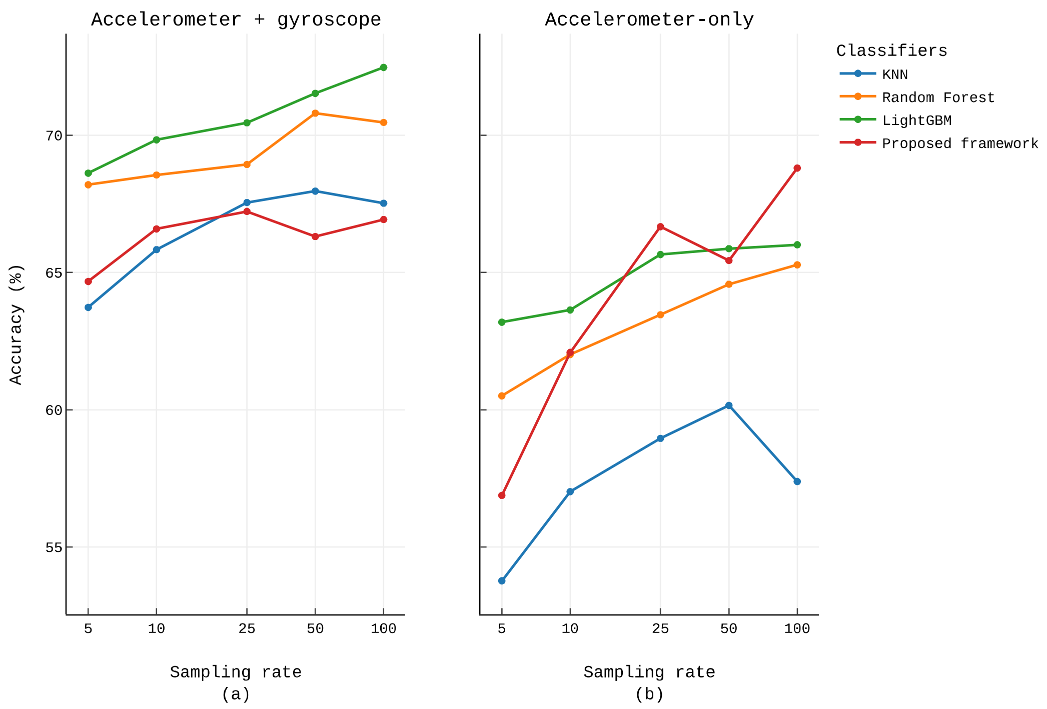


Figure 57 – Accuracy of the proposed framework and the reference models on PAMAP2 dataset for different sampling rates within the wrist-worn sensor scenario. Visualization for the accelerometer and gyroscope use case and accelerometer-only use case.

Table 24 – Framework performance on WISDM-2019 dataset using accelerometer and gyroscope data from the wrist-worn device.

Model	Accuracy (%)	Precision (%)	Recall (%)	F1-score (%)
First level classifier	91.96	90.76	91.96	91.04
Static activities classifier	82.05	83.50	82.05	81.75
Dynamic activities classifier	79.90	80.76	79.90	79.68
Framework	73.82	73.69	73.82	72.88

In respect to the static activities classifier, the accuracy achieved was apparently low (82.61%). However, once again it should be highlighted that classifying static activities with only the wrist motion sensor is highly complex and is generally not a primary requirement in commercial applications. Fig. 58 shows the confusion matrices for the static activities classifier, and for three specific subjects which most negatively impacted the results. Once more, the distinction between "sitting" and "standing" in this case is highly dependent on the particularity of the individual's wrist posture, and susceptible to various artifacts during the data collection.

The dynamic activities classifier had a quite interesting performance, considering it was dealing with the classification of 12 different activities using a single device on the wrist. Fig. 59 presents the confusion matrix. Note that the activity "running", as with

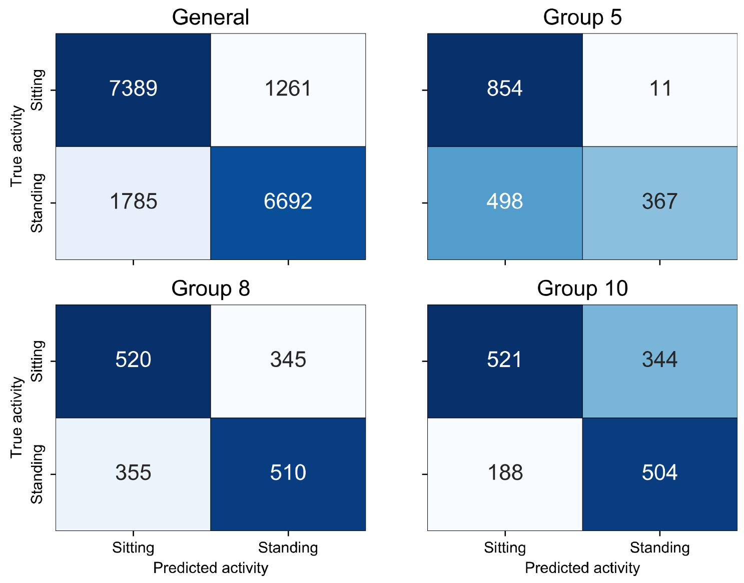


Figure 58 – Confusion matrices of static activities classifier performance, in general and for 3 subjects with the worst results, on WISDM dataset within the wrist-worn sensor scenario, using acceleration and gyroscope data.

other datasets, achieved high accuracy and precision. This is most likely explained by the presence of a periodic motion signal with high frequency and energy. The activities "using stairs" and "kicking (soccerball)" presented low accuracy, and high misclassification between the two. The latter may be highly subject-dependent, due to the particularities involved in the movements to kick a ball. The activity "eating or drinking" presented a high rate of true positives, however, the false positive rate was also very high, presenting misclassification mostly with other hand activities, such as "typing", "brushing teeth", "writing", and "folding clothes".

In this use case, the first level Decision Tree mostly used Gini as splitting criterion, a mean max depth of 4.1, and the mean number of features of 2.6, being the z-axis acceleration standard deviation, x-axis acceleration mean, and z-axis acceleration minimum value the most selected as best features. For the static activities Decision Tree, the entropy splitting criterion was the most used, with a mean max depth of 4.3, mainly using 2.4 features, being SMA from x-axis acceleration, and minimum value and SMA from z-axis acceleration the most selected ones. The optimized CNN for dynamic activities classification used acceleration and gyroscope data, and its architecture had 64 and 32 filters of size 5x1 in the convolutional layers, stride of 1 in both layers, and 30 neurons in the fully connected layer.

Accelerometer-only data. When using only acceleration data, the framework recognition performance had a decrease of 2.82% on accuracy, which was mainly due to the drop in the dynamic activities classifier performance, as show in Table 25. This result is an indication that the use of gyroscope data contributes to the classification of dynamic

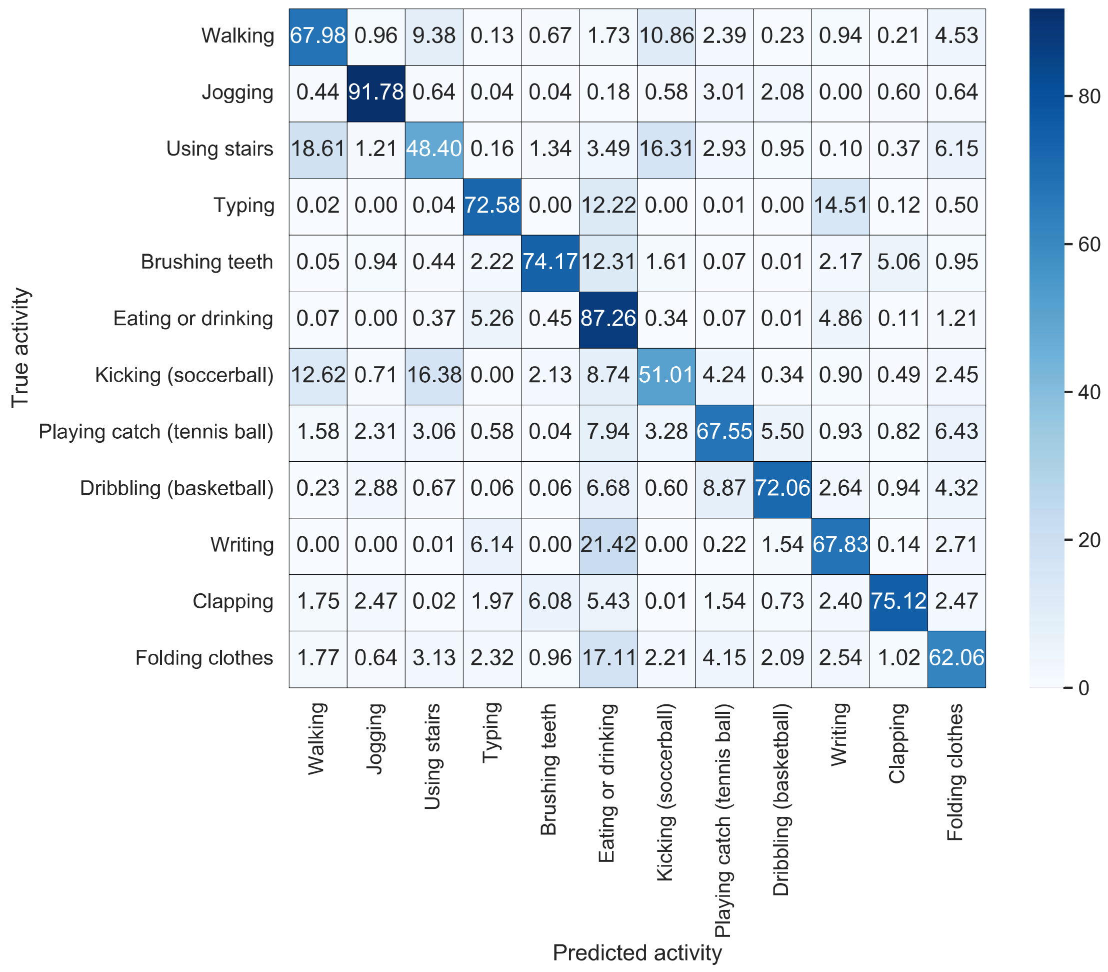


Figure 59 – Confusion matrix of dynamic activities classifier performance on WISDM-2019 dataset within the wrist-worn sensor scenario, using acceleration and gyroscope data.

Table 25 – Framework performance on WISDM-2019 dataset using only accelerometer data from the wrist-worn device.

Model	Accuracy (%)	Precision (%)	Recall (%)	F1-score (%)
First level classifier	91.83	90.80	91.83	91.03
Static activities classifier	82.44	83.17	82.44	82.35
Dynamic activities classifier	76.49	77.42	76.49	76.31
Framework	71.00	70.99	71.00	70.22

activities. However, the performance drop was small, thus the impact on computational cost should be analyzed to define the best strategy according to the application.

The first level Decision Tree mostly used Gini as splitting criterion, and a mean max depth of 4.2, while using 2.7 features, being the best features the same as in prior case. The static activities Decision Tree mostly used the Gini splitting criterion, and a mean max depth of 4.9, using only 2.7 features on average, mainly the x-axis acceleration SMA, z-axis acceleration SMA, and x-axis acceleration mean. The CNN model had on average the same architecture as in the previous case.

Fig. 60 shows a comparison within this scenario between the use case with accelerometer and gyroscope data, and with accelerometer-only data. In conclusion, the findings show the use of gyroscope data has no positive impact on the first level and static activities classifiers, and in addition, the framework keeps the performance nearly at the same level when using only acceleration data, being much more efficient in this case.

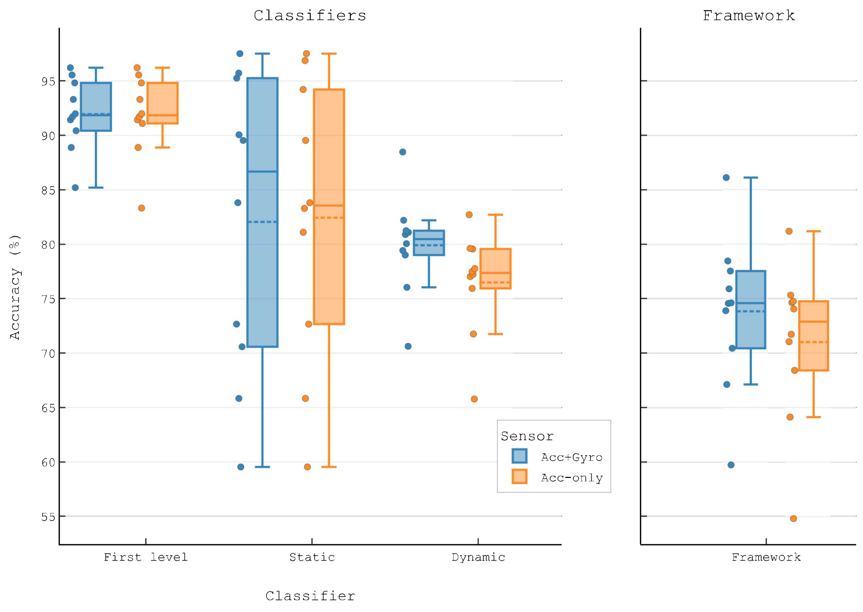


Figure 60 – Boxplot charts of models and framework accuracy on WISDM-2019 dataset. Comparison between results using accelerometer and gyroscope data and using accelerometer-only data. The dashed line refers to the mean value.

Lower sampling rates. The analysis of the impact of lower sampling rates (see Fig. 61) demonstrated that, despite the better results of the static activities classifier with lower rates, the performance deterioration of the dynamic activities classifier was more significant, causing a reasonable drop in the performance of the framework when reducing the sampling rate. In the case of acceleration-only use case, the decrease was slight, around 1%, nevertheless, halving the sampling rate can provide more benefits than a slightly higher accuracy, depending on the application requirements.

Comparison with reference models. When the performance of the proposed HAR framework was compared with the reference models, it once again had superior metrics in the wrist-worn device scenario for both the case with accelerometer and gyroscope data, and accelerometer-only data, as presented in Table 26. This result contributes to demonstrate how the framework proposed in this work is an efficient solution for wrist-worn devices.

Fig. 62 shows the boxplot charts and swarm plots of the results. It is noticeable

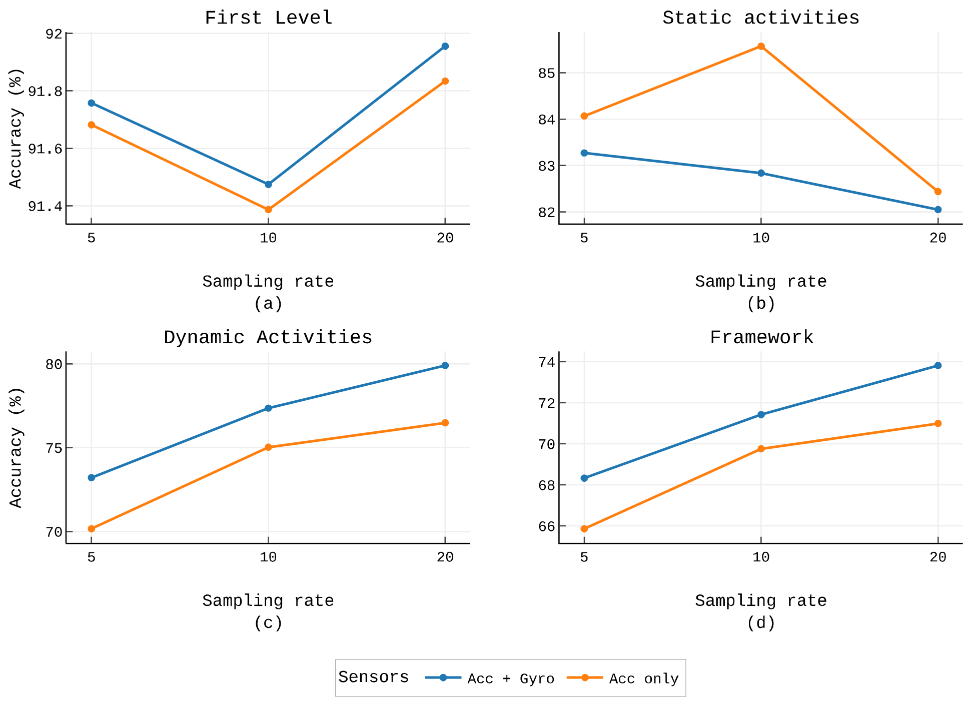


Figure 61 – Classifiers and framework accuracy on WISDM-2019 dataset for different sampling rates. Comparison between accelerometer and gyroscope data and accelerometer-only data.

Table 26 – Comparison of classification metrics between the proposed framework and the reference models within the wrist-worn sensor scenario on WISDM-2019 dataset.

Model	Sensor	Accuracy (%)	Precision (%)	Recall (%)	F1-score (%)
KNN	A, G	68.63	69.88	68.63	68.22
	A	65.08	66.26	65.08	64.61
Random Forest	A, G	69.19	72.13	69.19	68.52
	A	66.61	68.71	66.61	65.84
LightGBM	A, G	73.06	**75.62**	68.95	70.90
	A	68.19	70.33	63.50	65.44
Proposed framework	A, G	**73.82**	73.69	**73.82**	**72.88**
	A	71.00	70.99	71.00	70.22

A - accelerometer; G - gyroscope

how the models had a drop in performance when removing the gyroscope data from the input data, however the accuracy of the framework remained above the others in both cases.

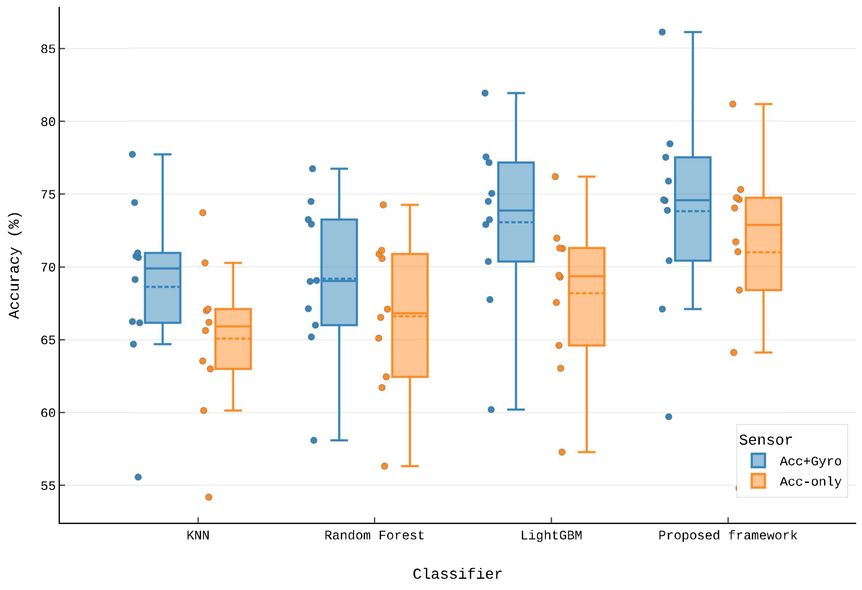


Figure 62 – Boxplot charts of accuracy of the proposed framework and the reference models on WISDM-2019 dataset. Comparison between results using accelerometer and gyroscope data and using accelerometer-only data. The dashed line refers to the mean value.

When evaluating the reduction in the sampling rate (see Fig. 63), as with the proposed framework, the other models also had a significant drop in performance. However, it is noteworthy that the framework in the case of using acceleration only, has a superior result compared to competitors even when using 10 Hz against 20 Hz of the others.

Therefore, it is suggested that, considering the characteristics of this dataset, 20 Hz proved to be the most suitable sampling rate.

6.1.4 Summary of Results and Discussion

This section sums up the results obtained from tests performed with the three datasets used in this work, and presents a discussion of the most remarkable achievements. The summarized results are shown in Fig. 64, which presents the F1-score achieved by the proposed HAR framework against the highest F1-score among the reference models, in each case and scenario analyzed, and for every dataset evaluated.

The first important understanding is that the performance of the models was better when using multiple wearable sensors compared to the wrist-worn sensor scenario. However, in cases of continuous use to monitor ADLs in real world, applications with

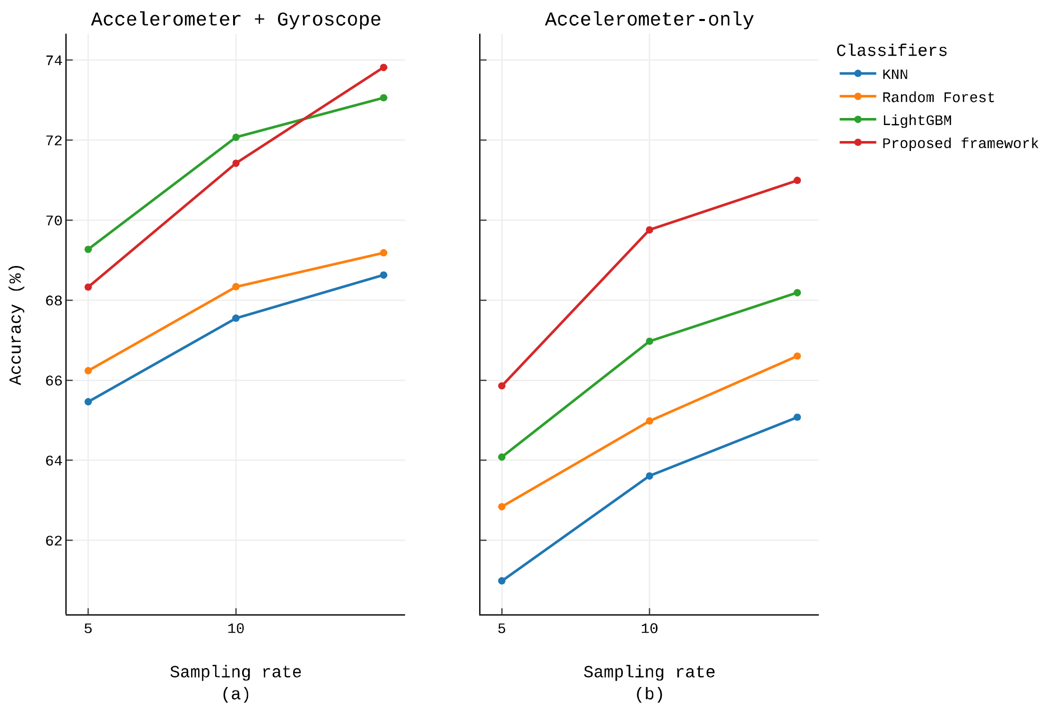


Figure 63 – Accuracy of the proposed framework and the reference models on WISDM-2019 dataset for different sampling rates. Visualization for the accelerometer and gyroscope use case and accelerometer-only use case.

multiple sensors are not viable, as their use is more intrusive, more expensive, and less adaptable to clothing. Therefore, the results achieved with the wrist-worn device corroborate more for the implementation of HAR models in smartwatches and smart bands, already commercially available. Anyhow, tests with multiple sensors are important for comparative study, since most of the works published in this area use all available sensors in the dataset.

When observing the results in the scenario of multiple sensors, the proposed framework achieved worse classification metrics than the competitors with the three datasets analyzed. On the other hand, as regards to results with the wrist-worn device, the framework performed significantly better than the reference models in MHEALTH and WISDM-2019 datasets, while in the PAMAP2 dataset it was on the same level as the competitors, being slightly surpassed by the Random Forest model. It is worth emphasizing that using only acceleration data, the framework performed better, even on the PAMAP2 dataset.

In the comparison between the use cases of acceleration and gyroscope and acceleration only, with the exception of the MHEALTH dataset, which had similar results in both cases, in the other datasets the use of the gyroscope contributed to improve the performance of the dynamic activities classifier, especially in the scenario with a single device in the wrist. However, given that the gain is not quite expressive, using only the

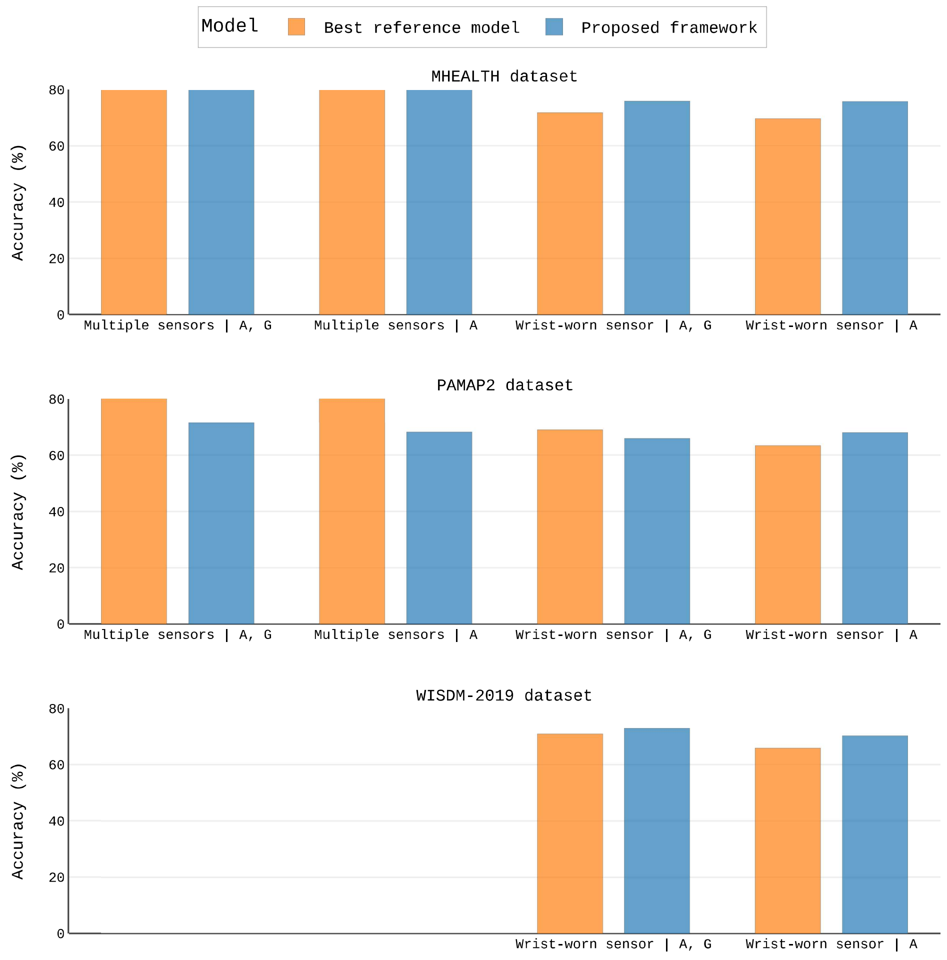


Figure 64 – Performance comparison in terms of F1-score between the best reference model and the proposed HAR framework. A and G stand for accelerometer data and gyroscope data, respectively.

accelerometer data can be much more efficient for resource-constrained devices, which aim for large battery life.

Still on the use of the gyroscope data, it is noteworthy that for the first level and static activities classifiers, parameters extracted from the gyroscope did not pass the feature selection process in most cases, as shown in the previous sections. Therefore, the gyroscope proved to be important only for the classification of dynamic activities, which makes perfect sense, since the angular velocity is influenced by the body part movements.

Also on first level and static activities classifiers, the Decision Trees had small depth, and used few features to perform their classification tasks in all the cases analyzed. These characteristics make the classifiers extremely efficient and almost invisible to the proposed HAR system, which has its computational cost concentrated in the CNN of the

Table 27 – CNN model complexity and accuracy for different sampling rates with the MHEALTH dataset.

Sampling rate	#MAC operations	Flash (kB)	RAM (kB)	Accuracy (%)
50 Hz	570,778	195.83	13.71	78.39
25 Hz	182,234	100.83	7.12	76.33
10 Hz	64,154	44.58	3.02	79.07
5 Hz	24,794	25.83	1.65	75.61

dynamic activities classifier.

In the analysis of lower sampling rates in the wrist-worn sensor scenario, the framework showed to maintain performance even with small rates, making it even more efficient than the reference models. In some cases, the framework with a reduced sample rate obtained superior results to the competitors, even with the original sample rate.

In summary, the proposed energy-efficient HAR framework proved to be more efficient than the reference models, achieving the best performance with the three datasets in the most efficient scenario, with the wrist-worn device and acceleration only. Section 6.2 assesses computational cost efficiency in order to validate the framework's ability to be embedded in resource-constrained wearable devices.

6.2 Evaluation of Computational Performance on Microcontroller

The results presented thus far have shown that, in many cases, reducing the sampling rate does not degrade the performance of the classifier, and may even improve the outcomes. In this section, the performance of the proposed framework in the microcontroller is evaluated in terms of model complexity, computational cost, and energy consumption.

6.2.1 Model Complexity

This subsection presents the analysis of the CNN model used in the dynamic activities classifier with different sampling rates and for the three datasets used in this work. The model complexity was evaluated in terms of number of parameters and MAC operations, and usage of Flash memory and RAM.

Table 27 shows the findings for the MHEALTH dataset. In this case, the CNN model achieved the highest accuracy with 10 Hz as sampling rate, using around nine times fewer MAC operations, and more than four times less Flash memory and RAM than the model with data sampled at 50 Hz. The reduction of MAC operations contributes to decrease execution time and energy consumption. Reducing memory usage, in addition to decrease energy consumption, also allows the deploy on microcontrollers with more restricted resources, since the memory is limited by cost and size.

Table 28 – CNN model complexity and accuracy for different sampling rates with the PAMAP2 dataset.

Sampling rate	#MAC operations	Flash (kB)	RAM (kB)	Accuracy (%)
100 Hz	2,459,962	409.08	54.22	72.18
50 Hz	1,091,002	217.83	25.71	67.68
25 Hz	500,602	124.08	12.62	68.39
10 Hz	64,154	44.58	3.02	62.24
5 Hz	28,282	49.08	2.03	54.37

Table 29 – CNN model complexity and accuracy for different sampling rates with the WISDM-2019 dataset.

Sampling rate	#MAC operations	Flash (kB)	RAM (kB)	Accuracy (%)
20 Hz	264,918	93.32	10.01	76.49
10 Hz	110,678	55.82	4.78	75.03
5 Hz	33,558	37.07	2.16	70.16

For the PAMAP2 dataset, the best result from CNN was with the original sampling rate of 100 Hz. However, this configuration uses more than 400 kB of Flash memory and 54 kB of RAM, as shown in Table 28, which makes it impractical to be implemented in several families of microcontrollers. As an example, the model with the best accuracy would occupy 80% of the program memory of the microcontroller used in this work, most likely compromising the development of other features necessary for a commercial application. In this case, a viable alternative would be to adopt the use of a sampling rate of 25 Hz, reducing the accuracy by 3.79%, nevertheless making the integration of the model in a real application much more achievable.

Finally, the results with the WISDM-2019 dataset (see Table 29) showed that the performance in terms of classification accuracy was very similar between the models that used 20 and 10 Hz as the data sampling rate. Therefore, the second one proves to be much more efficient, since executes less than half the number of MAC operations, in addition to reducing the usage of Flash memory and RAM.

6.2.2 Processing Time

Processing time is a fundamental metric to evaluate the efficiency of performing tasks in microcontrollers. The less time used to perform a task, the longer the processor can remain in a sleep or idle mode, reducing power consumption. In this context, the tests were realized with different sampling rates. In addition to reducing energy consumption by reducing the number of samples being read, the reduction of the sampling rating also implies less memory usage, and the simplification of operations performed in feature extraction, normalization and classification.

Table 30 – Processing time of tasks performed in each module of the framework for different sampling rates.

Module	Task	50 Hz	25 Hz	10 Hz	5 Hz
First level	Feature extraction	91.85 µs	105.58 µs	28.71 µs	20.77 µs
	Classification	1.71 µs	1.77 µs	1.71 µs	1.65 µs
Static activities	Feature extraction	56.94 µs	46.29 µs	20.04 µs	10.12 µs
	Classification	2.60 µs	1.90 µs	2.71 µs	1.75 µs
Dynamic activities	Concatenation	65.40 µs	33.08 µs	13.67 µs	7.12 µs
	Normalization	280.00 µs	212 µs	92.00 µs	52.00 µs
	Classification	64.04 ms	22.01 ms	7.85 ms	3.13 ms

In this work, the MHEALTH dataset was selected for a more detailed analysis of the processing time of the human activities classifier using the proposed HAR framework. For this evaluation, the processor clock frequency was 96 MHz. Table 30 presents the execution time measured for each task performed by each module of the framework. It can be noted that the tasks executed in shortest time were the classification task of the first level and the static activities modules, as expected, as they are based on simple Decision Trees built with few if-else operations.

The feature extraction task was much more expensive than the classification for the first level and static activities modules. It is important to note that, for the first level module, the feature extraction time was longer with data sampled at 25 Hz compared to 50 Hz since the former used three features while the latter needed only two. This difference in the number of features occurred, in this case, because at 25 Hz, the model, on average, needed to use more features to achieve the best result.

As expected, the module responsible for classifying dynamic activities was the most costly in terms of processing time. The data normalization task had higher latency than all those performed in first level and static activities modules added together. However, the most costly task with a considerably large difference was the CNN model, responsible for classifying dynamic activities. This result shows the importance of the 2-level classifier, which only uses CNN when necessary, reducing processing time compared to a system entirely based on a CNN model. The impact of reducing the sampling rate is also greater for this task, even proportionately. Therefore, opting for a lower sampling rate may be essential to make practical the implementation of a CNN in some microcontrollers.

Fig. 65 shows the graphs of processing time and accuracy by sampling rate for each classifier that compose the proposed HAR framework and for the framework. It is important to note that the scale is different on each chart. The processing time variation on the first level classifier is quite low, having little impact on the outcomes. In the case of the static activities classifier, the variation is more significant, but still much smaller in relation to the impact caused by the variations in the dynamic activities classifier.

Nevertheless, in the case of the MHEALTH dataset, one can adopt 5 Hz and at the same time obtain the lowest processing time and better accuracy for static activities.

For the CNN model of the dynamic activities classifier, reducing the sampling rate can result in significant gains in several aspects: response time, computational cost, memory usage, and energy consumption. A substantial drop in processing time can be observed in the graph when reducing the sampling rate. And, in the case of the MHEALTH dataset, the accuracy remains at the same level, reaching even a higher value at 10 Hz, making the model quite efficient. For the framework analysis, the assumed processing time was the sum of the time required to run the first level and the dynamic activities classifier, which is the most costly for the system.

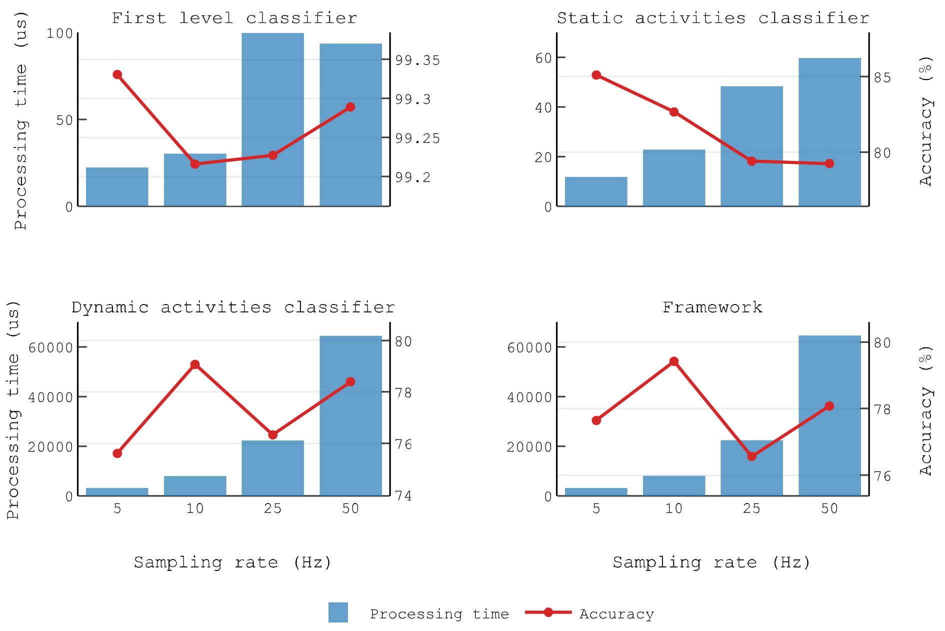


Figure 65 – Processing time and accuracy of the framework and its classifiers for different sampling rates evaluated.

As the dynamic activities classifier has the greatest impact on the HAR system processing time, the CNN execution time was evaluated, along with the accuracy, for different sampling rates for the three datasets. Table 31 shows the results for the MHEALTH dataset. It can be noted in more detail that the reduction in processing time with 10 Hz as sampling rate compared to 50 Hz was reduced by a factor of eight, and achieved a higher accuracy.

For the PAMAP2 dataset, the reduction of the sampling rate and, consequently, of the processing time, implies a decrease in the classification accuracy, as shown in Table 32. For some applications, reducing accuracy around 4% to decrease processing time by

Table 31 – Processing time and accuracy of CNN for different sampling rates with the MHEALTH dataset.

Sampling rate	Processing time (ms)	Accuracy (%)
50 Hz	64.39	78.39
25 Hz	22.26	76.33
10 Hz	7.96	79.07
5 Hz	3.19	75.61

Table 32 – Processing time and accuracy of CNN for different sampling rates with the PAMAP2 dataset.

Sampling rate	Processing time (ms)	Accuracy (%)
100 Hz	240.44	72.18
50 Hz	115.80	67.68
25 Hz	53.70	68.39
10 Hz	16.22	62.24
5 Hz	3.74	54.37

Table 33 – Processing time and accuracy of CNN for different sampling rates with the WISDM-2019 dataset.

Sampling rate	Processing time (ms)	Accuracy (%)
20 Hz	31.31	76.49
10 Hz	13.30	75.03
5 Hz	4.29	70.16

almost 5 times can be acceptable and even essential. On the other hand, when the highest classification rate is imperative, it would be necessary to adopt a system capable of dealing with a higher computational cost.

Table 33 presents the results for the WISDM-2019 dataset. In this case, the most efficient model was accomplished by the CNN with data sampled at 10 Hz, since the reduction by a factor of 2.35 in the processing time implies only a slight decrease in accuracy. This processing time reduction can contribute to the system achieving lower energy consumption and, consequently, longer battery life, as will be discussed in the next subsection.

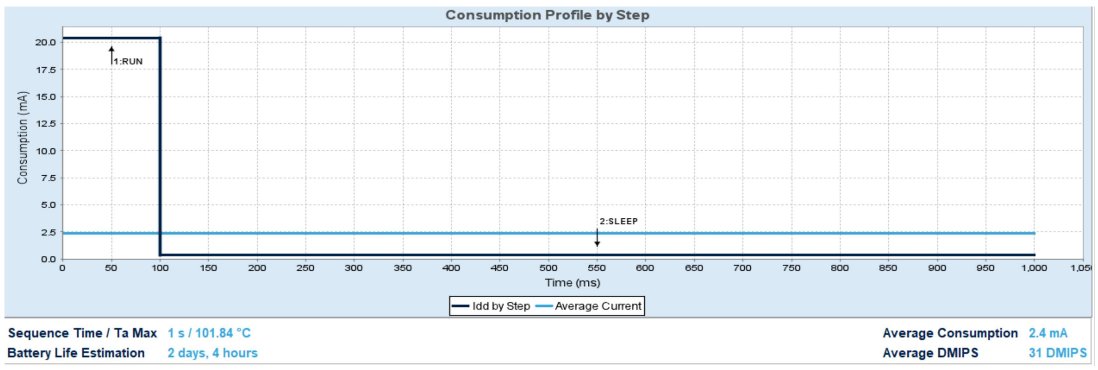
6.2.3 Energy Consumption

This section presents the energy consumption and battery life estimation based on the Power Consumption Calculator tool from STM32CubeIDE. By using this tool, the average consumption is calculated based on task execution time, sleep time, and sequence time. The sequence time is the duration to complete a sequence of steps (tasks) performed in a microcontroller operating cycle. Regarding the battery life estimation, one can select among different types, capacity and other characteristics as reference battery.

Here, it is fundamental to highlight that the analysis was performed considering only the tasks related to the classification modules of the framework. In a real application, several other functions are performed and use device resources that consume energy and time, however in this analysis one can evaluate the impact of each classifier and each configuration on the energy consumption, and realize a comparative study.

For the energy consumption analysis, an application that returns the model result every 1 second was assumed, and the processing time of the tasks presented in the previous subsection was used. The reference battery for autonomy estimation was based on the Xiaomi Mi Smart Band 6's lithium-ion polymer battery characteristics: capacity of 125 mAh, and nominal voltage of 3.7 V. In run mode, the microcontroller was configured with a clock frequency of 100 MHz and with access to Flash memory, while in low power sleep mode the frequency was 16 MHz. The supply voltage was 3.6V. Given the defined settings, the current consumed was 20.4 mA in active mode, and 400 µA in sleep mode.

In the Power Consumption Calculator settings, a step was set on run mode with duration of the respective task processing time, and the next step was set in sleep mode for the rest of the time in the cycle, which was set to 1 second, according to the application response time assumed. An example of the tool analysis results is presented in Fig. 66. The graph shows the current consumption in each step and the average value, while additional information are shown below the graph, including the battery life estimation.



Sequence Time / Ta Max 1 s / 101.84 °C
Battery Life Estimation 2 days, 4 hours
Average Consumption 2.4 mA
Average DMIPS 31 DMIPS

Figure 66 – Sequence of steps configured in the power consumption tool.

The estimation of energy consumption was performed in three different ways. First, the HAR system embedded in the device was assumed classifying only static activities, i.e., in static activity mode. Next, the HAR system was assumed to be fully running in dynamic activity mode. The processing time was included in the results for better interpretation. Finally, the HAR system was simulated by switching between static and dynamic activity modes. In this last scenario, two cycles were considered, where in the first one the system was in static activity mode, and in the second one in dynamic activity mode.

Table 34 – Estimated average consumption and battery life when running on static activity mode for different sampling rates.

Sampling rate	Processing time (µs)	Consumption (µA)	Battery life (days)
50 Hz	153.10	403.06	12.92
25 Hz	155.54	403.11	12.92
10 Hz	53.17	401.06	12.99
5 Hz	34.29	400.69	13.00

Table 35 – Estimated average consumption and battery life when running on dynamic activity mode for different sampling rates.

Sampling rate	Processing time (ms)	Consumption (µA)	Battery life (days)
50 Hz	64.48	1,689.58	3.08
25 Hz	22.36	847.29	6.15
10 Hz	7.99	559.76	9.30
5 Hz	3.21	464.19	11.22

Table 34 shows the results of the energy consumption analysis for the static activity mode. The estimated current consumption was around 400 µA for all sampling rates, while the battery life was 13 days. Since the execution time is very short, for roughly the entire cycle duration the microcontroller can remain in sleep mode. Therefore, the impact of reducing the sampling rate on the reduction of computational cost and energy consumption for the static activity module is quite low. Nevertheless, the reduction of the sampling rate has other positive impacts in the reduction of energy consumption, as it reduces the number of sensor data readings and memory access.

On the other hand, the energy consumption by the dynamic activities module has a major impact on battery life, as shown in Table 35. Reducing the sampling rate in this case greatly contributes to achieving a more efficient application. When reducing from 50 to 10 Hz, battery life increases by 3 times, from 3 to 9 days. Also, as mentioned above, lowering the sample rate has other positive impacts.

On many occasions, it is not practicable to reduce the sampling rate without degrading the accuracy, as observed with the PAMAP2 dataset. However, the system designer needs to meet application requirements and, therefore, knowing the impact of each configuration is critical to developing the most efficient solution.

Lastly, the HAR system energy consumption was analyzed with the activity mode switching between static and dynamic. In this case, the energy consumption was also greatly influenced by the sampling rate, as shown in Table 36, since the weight of the dynamic activities module stands out. The results confirm the importance of reducing the sampling rate for the dynamic activities classifier in a resource-constrained wearable device. Once again, the importance of adopting a 2-level classifier should be highlighted.

Table 36 – Estimated average consumption and battery life when switching between static and dynamic activity mode for different sampling rates.

Sampling rate	Consumption (µA)	Battery life (days)
50 Hz	1,046.32	4.98
25 Hz	625.20	8.33
10 Hz	480.41	10.84
5 Hz	432.44	12.04

The proposed HAR framework with data sampled at 50 Hz consumes 60% of current when switching equally between the activity modes compared to full dynamic activity mode.

6.3 Comparison with State-of-the-Art Techniques

This section presents a comparative study between the framework proposed in this research, and the main techniques found in the literature considered state-of-the-art. The main objective of this work was to develop a framework for designing efficient HAR systems for wearable devices, mainly wrist-worn devices. However, there are only a few works that have proposed models for HAR using only wrist-worn sensors data from public datasets, which makes it difficult to carry out a comparative study. Therefore, on some occasions comparisons were only possible with the multi-sensor scenario.

Prior to the presentation and discussion of the results, some important issues observed in literature must be highlighted. The absence of a standard method for testing HAR models creates difficulties to compare results fairly. Moreover, several researches lack a proper description of the steps of training, validation and testing process, preventing the verification of method. Furthermore, important characteristics of the model are often omitted, such as number of parameters, and processing time, making a complete comparison impractical.

Additionally, many studies found in literature present data leakage in model training process. In HAR, the most commonly used segmentation is the semi non-overlapping temporal window (SNOW), mainly with 50% overlap. However, in data split, when applying a simple random holdout technique, for example, overlapping samples can be allocated in training and test subsets, contributing to a biased model. Another common issue observed was the combined hyperparameter tuning and model selection, which results in optimistic model performance that does not generalize to unseen data.

Despite the aforementioned challenges to carry out an adequate comparative study, the results presented in this section contribute to confirm that the proposed framework is a lightweight solution with performance close to state-of-the-art results. The following subsections present and discuss the comparative results with each dataset used in this

Table 37 – Comparison with state-of-the-art results on MHEALTH dataset.

Method	#Parameters	Acc (%)	Protocol
2-D CNN (HA; YUN; CHOI, 2016)	8,433,900	98.29	Holdout
Bi-LSTM+LSTM (ALJARRAH; ALI, 2019)	483,768	97.64	Holdout
Deep CNN ensemble (SENA et al., 2021)	12,000,000	96.27	LOSO
CNN+LSTM with SA (SINGH et al., 2021)	8,221	94.86	LOTO
CNN+LSTM (SINGH et al., 2021)	426,892	93.80	LOTO
Proposed with accelerometer and gyroscope	50,869	91.02	LOSO
Proposed with accelerometer-only	50,869	90.29	LOSO

Acc. - Accuracy; SF - Self-Attention; LOSO - Leave-one-subject-out; LOTO - Leave-one-trial-out

work.

6.3.1 Results on MHEALTH dataset

First, the works evaluated with the MHEALTH dataset are discussed. In this scenario, the highest accuracy obtained by the HAR framework proposed in this work was 91.02%. Table 37 presents the results achieved by state-of-the-art researches with this dataset, using data from all the wearable sensors available. The best accuracy was reached by a 2-D CNN method, proposed by Ha, Yun and Choi (2016), followed by the proposal of Bi-LSTM and LSTM (ALJARRAH; ALI, 2019). However, both used a simple random holdout method for data splitting, which allows the allocation of examples from the same trial in both training and testing subsets, contributing for optimistic results, not achievable in real world.

The deep CNN ensemble, proposed by Sena et al. (2021) achieved an impressive accuracy of 96.27%, employing the LOSO protocol for data splitting. To achieve this result a powerful, yet complex, model was used. Even with the authors describing their model as feasible for wearable devices, it has around 12 million parameters and consumed 183 MB of RAM on a Raspberry Pi board. While the proposed energy-efficient framework has 235 times fewer parameters and consumed only 15 kB of RAM.

The other two works, which used CNN and LSTM (SINGH et al., 2021), achieved high accuracy with a low number of parameters using the Leave-one-trial-out (LOTO) protocol. This strategy partly avoids bias in the model, as it does not include the same trial in training and testing. However, it still uses data from the same individual in training and testing, favoring model performance. According to the authors, the LOSO protocol was discarded because it presented high variance between subjects.

Regarding HAR systems for wrist-worn devices, the only proposal observed in the literature using the MHEALTH dataset was the work of Kongsil, Suksawatchon and Suksawatchon (2020). They employed LOSO protocol and achieved a F1-score of 90.62%,

Table 38 – Comparison with state-of-the-art results on PAMAP2 dataset.

Method	#Parameters	Acc (%)	Protocol
Deep CNN (JAFARI et al., 2019)	175,000	93.00	Holdout
Deep CNN (WAN et al., 2020)	-	91.00	Holdout
Deep CNN ensemble (SENA et al., 2021)	12,000,000	87.59	LOSO
Lego CNN (TANG et al., 2021)	-	93.50	Holdout
Proposed with accelerometer and gyroscope	107,029	71.99	LOSO
Proposed with accelerometer-only	105,109	69.44	LOSO

Acc. - Accuracy; LOSO - Leave-one-subject-out

while the framework proposed here reached 75.84%. However, they only included 8 of 12 activities from the dataset, excluding the following more complex ones: bending waist forward, frontal elevation of arms, bending knees, and jumping front and back.

6.3.2 Results on PAMAP2 dataset

The PAMAP2 dataset is more challenging than the previous one, and it proved to be quite problematic for the proposed framework. As shown in Table 38, the best results found in the literature used the simple random holdout method for data splitting. Problems with these strategies were highlighted in the previous subsection. As for the MHEALTH dataset, the model proposed by Sena et al. (2021) proved to be quite robust, since it obtained a high accuracy by applying the LOSO method. However, once again, a complex and computationally expensive model was required.

6.3.3 Results on WISDM-2019 dataset

The WISDM-2019 is the most challenging dataset among the three used in this work, and the one that contains data closer to a real application, since it used a commercial smartwatch to collect the dataset. Table 39 presets the results found in literature using only the smartwatch data from this dataset. The highest accuracy (96.20%) was achieved by a hybrid model (CNN+LSTM) (MEKRUKSAVANICH; JITPATTANAKUL, 2020), however the simple random holdout method was used in this case.

The framework proposed by Kongsil, Suksawatchon and Suksawatchon (2020) used the LOSO method for data splitting and achieved accuracy of 84.38% and 87.41%, when using a data window of 2 and 10 seconds, respectively. Nonetheless, they used only 5 of 18 activities from the dataset. The framework proposed in the present work did not perform at the same level, however, 14 activities from the dataset were considered, including the more complex ones such as brushing teeth, writing, typing, and folding clothes.

Table 39 – Comparison with state-of-the-art results on WISDM-2019 dataset.

Method	#Parameters	Acc (%)	Protocol
Fusion learning framework[a]	-	84.38	LOSO
Fusion learning framework[b]	-	87.41	LOSO
Deep CNN+LSTM[c]	5,926,863	96.20	Holdout
Proposed with accelerometer and gyroscope	10,835	73.82	LOSO
Proposed with accelerometer-only	9,875	71.00	LOSO

Acc. - Accuracy; LOSO - Leave-one-subject-out

a) Kongsil, Suksawatchon, Suksawatchon (2020) with 2-second window

b) Kongsil, Suksawatchon, Suksawatchon (2020) with 10-second window

c) Mekruksavanich, Jitpattanakul (2020)

7 Conclusion and Recommendations

This chapter concludes the work by summarizing the key findings in relation to the research objectives and hypotheses, and discusses the main contributions provided by this research. It also describes the limitations of the study and proposes recommendations for future work.

7.1 Major Findings

This study aimed to develop an energy-efficient HAR framework to deploy and test it on the microcontroller, and to extensively analyze the framework in different configurations, comparing with reference and state-of-the-art models. The results indicate that an efficient HAR system, which takes advantage of deep learning, yet meets the requirements of resource constrained devices, is achievable.

The comprehensive analysis show that using only acceleration as input data, instead of including angular velocity, in most cases does not deteriorate the model performance, and may even improve it. In addition, acceleration-only models may be a more efficient solution for real-world applications since it requires a smaller volume of data.

Further analysis demonstrated that, mainly in multi-sensor scenarios, there may be data redundancy, which can be neglected, providing a simpler and lighter solution. In this scenario, for both MHEALTH and PAMAP2 datasets, the CNN developed for classifying dynamic activities performed better when using only the wrist and ankle devices, discarding the device placed on the chest. Also, with the MHEALTH dataset, using only the acceleration data from these two devices demonstrated to be the best configuration.

In the evaluation of wrist-worn sensor scenarios, the proposed framework achieved better classification performance in five out of the six cases analyzed in the comparison against the best reference model. This result indicates that CNN's ability to automatically extract more representative features is fundamental for this scenario, in which it is more challenging to relate the individual's movements to data from a single sensor on the wrist.

The findings also show that the 2-level classifier strategy adopted by the proposed framework works better to classify groups of activities totally at rest (standing still, sitting, lying down) from dynamic activities, as in the MHEALTH dataset. The results on this dataset were quite satisfactory even using a much simpler model than the reference in the literature.

In this research, the proposed model was also developed for a microcontroller, which was deployed and tested in practice on the device, contributing to fill an important gap in the literature. The results allow evaluating the classification performance along with the computational cost, in different configurations. The analyzes provide important insight into the impact of model complexity on processing time, memory usage, and power consumption. This contribution is considered the most relevant in this research since there are still very few works in this context that present practical results of computational cost and model complexity in terms of memory usage and number of operations.

It is worth mentioning that reducing the sampling rate contributes to the construction of simpler models with more efficient data, which have lower latency, consume less energy, and are much more suitable for wearable devices. In some scenarios, the reduction in sampling rate also helped to improve the classification performance.

The findings of this research can contribute to the development of HAR systems for real-world applications. In fact, the proposed energy-efficient HAR framework provides a basis structure that can be used to develop lighter and more efficient applications, still with the capacity of CNN for effective feature extraction. The favorable results on activity classification with lower sampling rate can encourage the development of more research and applications in this direction.

7.2 Limitations

This research has some potential limitations. The proposed framework was compared with state-of-the-art models using the same metrics, however, the methods adopted for data splitting and model evaluation are not standardized in literature. In addition, HAR systems are subject to several variables that may have significant impact on model performance, such as input data window size, sampling rate, and activity labeling, which varies greatly among proposals, making a fair comparison even more difficult. Still on the limitations in the comparison of results, many studies adopt inappropriate methods for data splitting, as mentioned previously in Section 6.3, causing data leakage. Therefore, these models are subject to bias, which makes the result unrealistic.

The most appropriate strategy to avoid bias and reduce the variability would be to implement the models to be compared, and apply the same evaluation method. However, in general the code is not available, and important information for model implementation is missing. Moreover, most of these models are complex and require a lot of time and effort for implementation and training. To mitigate these limitations, in the present study, competitive models were developed and hyper-tuned to be compared with the proposed framework. As well as the proposed framework, the reference models were optimized, tested and evaluated according to the LOSO nested cross-validation method, to ensure a fair comparison, and without data leakage.

7.3 Future Work

Additional strategies can be adopted to make the framework even more efficient. As a first recommendation for future work, the analysis of the window size of the input data may provide important information to enrich the evaluation of data volume versus performance. Furthermore, strategies such as activity change detection may greatly contribute to computational cost reduction, by maintaining the device in low-power mode for longer period.

Regarding the CNN, techniques for pruning and quantizing the model can be adopted before converting the model to C code. Pruning can make the model faster and lighter, while quantizing the parameters in 8-bit integer simplifies operations, also contributing to reduced latency.

A relevant advance of this research is the improvement of the analysis of computational performance by measuring energy consumption with current sensors instead of estimating it. Also, the deployment and testing on customizable smart bands or smartwatches is highly recommended as future work, since it would allow the analysis in a real-world scenario.

Bibliography

ALJARRAH, A. A.; ALI, A. H. Human Activity Recognition using PCA and BiLSTM Recurrent Neural Networks. *2019 2nd International Conference on Engineering Technology and its Applications, IICETA 2019*, IEEE, p. 156–160, 2019. 68, 147

AMAZFIT. *Amazfit Band 5*. 2022. Disponível em: <https://www.amazfit.com/en/band5>. 38

ANGUITA, D. et al. A public domain dataset for human activity recognition using smartphones. *ESANN 2013 proceedings, 21st European Symposium on Artificial Neural Networks, Computational Intelligence and Machine Learning*, n. April, p. 437–442, 2013. 61

APPLE. *Healthcare - Apple Watch*. 2022. Disponível em: <https://www.apple.com/healthcare/apple-watch/>. 38

APPLE. *Use fall detection with Apple Watch*. 2022. Disponível em: <https://support.apple.com/en-us/HT208944>. 38

APPLE. *Use the Workout app on your Apple Watch*. 2022. Disponível em: <https://support.apple.com/en-us/HT204523>. 38, 39

ATTAL, F. et al. Physical human activity recognition using wearable sensors. *Sensors*, v. 15, n. 12, p. 31314–31338, 2015. ISSN 1424-8220. Disponível em: <https://www.mdpi.com/1424-8220/15/12/29858>. 33, 45, 64

AVCI, A. et al. Activity Recognition Using Inertial Sensing for Healthcare, Wellbeing and Sports Applications: A Survey. *Architecture of computing systems (ARCS), 2010 23rd international conference on*, p. 1–10, 2010. ISSN 15781550. Disponível em: <http://ieeexplore.ieee.org/articleDetails.jsp?arnumber=5759000>. 42, 44

AWAD, M.; KHANNA, R. *Efficient Learning Machines*. Apress, 2015. Disponível em: <https://doi.org/10.1007/978-1-4302-5990-9>. 42, 45, 46, 47, 48

AYUMI, V. Pose-based human action recognition with extreme gradient boosting. In: *2016 IEEE Student Conference on Research and Development (SCOReD)*. [S.l.: s.n.], 2016. p. 1–5. 62

BALLI, S.; SAğBAş, E. A.; PEKER, M. Human activity recognition from smart watch sensor data using a hybrid of principal component analysis and random forest algorithm. *Measurement and Control (United Kingdom)*, v. 52, n. 1-2, p. 37–45, 2019. ISSN 00202940. 64

BANOS, O. et al. Window size impact in human activity recognition. *Sensors*, v. 14, n. 4, p. 6474–6499, 2014. ISSN 1424-8220. Disponível em: <https://www.mdpi.com/1424-8220/14/4/6474>. 43

BANOS, O. et al. mhealthdroid: A novel framework for agile development of mobile health applications. In: PECCHIA, L. et al. (Ed.). *Ambient Assisted Living and Daily Activities*.

Cham: Springer International Publishing, 2014. p. 91–98. ISBN 978-3-319-13105-4. 62, 67, 82

BATCHULUUN, G. et al. Fuzzy system based human behavior recognition by combining behavior prediction and recognition. *Expert Systems with Applications*, v. 81, p. 108–133, 2017. ISSN 0957-4174. Disponível em: <https://www.sciencedirect.com/science/article/pii/S0957417417302051>. 26

BHAT, G. et al. An ultra-low energy human activity recognition accelerator for wearable health applications. *ACM Trans. Embed. Comput. Syst.*, Association for Computing Machinery, New York, NY, USA, v. 18, n. 5s, out. 2019. ISSN 1539-9087. Disponível em: <https://doi.org/10.1145/3358175>. 71

BREIMAN, L. *Machine Learning*, Springer Science and Business Media LLC, v. 45, n. 1, p. 5–32, 2001. Disponível em: <https://doi.org/10.1023/a:1010933404324>. 48

BREIMAN, L. et al. *Classification and Regression Trees.* Taylor & Francis, 1984. ISBN 9780412048418. Disponível em: <https://books.google.com.br/books?id=JwQx-WOmSyQC>. 47

CAPELA, N. A.; LEMAIRE, E. D.; BADDOUR, N. Feature selection for wearable smartphone-based human activity recognition with able bodied, elderly, and stroke patients. *PLOS ONE*, Public Library of Science, v. 10, n. 4, p. 1–18, 04 2015. Disponível em: <https://doi.org/10.1371/journal.pone.0124414>. 26

CHAVARRIAGA, R. et al. The opportunity challenge: A benchmark database for on-body sensor-based activity recognition. *Pattern Recognition Letters*, v. 34, n. 15, p. 2033–2042, 2013. ISSN 0167-8655. Smart Approaches for Human Action Recognition. Disponível em: <https://www.sciencedirect.com/science/article/pii/S0167865512004205>. 62

CHEN, K. et al. Deep learning for sensor-based human activity recognition: Overview, challenges and opportunities. *arXiv*, v. 37, n. 4, 2020. ISSN 23318422. 31, 34, 35, 36, 40, 53, 56, 65, 69

CHEN, L.; NUGENT, C. D. *Human Activity Recognition and Behaviour Analysis: For Cyber-Physical Systems in Smart Environments.* 1st. ed. [S.l.]: Springer Publishing Company, Incorporated, 2019. ISBN 3030194078. 34, 40, 41

CHEN, Z. et al. Robust human activity recognition using smartphone sensors via ct-pca and online svm. *IEEE Transactions on Industrial Informatics*, v. 13, n. 6, p. 3070–3080, 2017. 62

COELHO, Y. et al. Human activity recognition based on convolutional neural network. *IFMBE Proceedings of XXVI Brazilian Congress on Biomedical Engineering*, Springer, v. 2, p. 247–252, Oct 2018. 73

COELHO, Y. et al. Evaluation of low-level quantization of acceleration data on human activity recognition system. In: *Proceedings of International Workshop on Assisitive Technology 2019.* [S.l.: s.n.], 2019. 73

COELHO, Y. L. et al. A lightweight model for human activity recognition based on two-level classifier and compact cnn model. In: BASTOS-FILHO, T. F.; CALDEIRA, E. M. de O.; FRIZERA-NETO, A. (Ed.). *XXVII Brazilian Congress on Biomedical*

Engineering. Cham: Springer International Publishing, 2022. p. 1895–1901. ISBN 978-3-030-70601-2. 73

COELHO, Y. L. et al. A lightweight framework for human activity recognition on wearable devices. *IEEE Sensors Journal*, v. 21, n. 21, p. 24471–24481, 2021. 73

COURBARIAUX, M. et al. *Binarized Neural Networks: Training Deep Neural Networks with Weights and Activations Constrained to +1 or -1*. arXiv, 2016. Disponível em: <https://arxiv.org/abs/1602.02830>. 70

CREAN, C.; MCGEOUGE, C.; O'KENNEDY, R. 11 - wearable biosensors for medical applications. In: HIGSON, S. (Ed.). *Biosensors for Medical Applications*. Woodhead Publishing, 2012, (Woodhead Publishing Series in Biomaterials). p. 301–330. ISBN 978-1-84569-935-2. Disponível em: <https://www.sciencedirect.com/science/article/pii/B9781845699352500115>. 40

CRUCIANI, F. et al. Comparing CNN and human crafted features for human activity recognition. *Proceedings - 2019 IEEE SmartWorld, Ubiquitous Intelligence and Computing, Advanced and Trusted Computing, Scalable Computing and Communications, Internet of People and Smart City Innovation, SmartWorld/UIC/ATC/SCALCOM/IOP/SCI 2019*, IEEE, p. 960–967, 2019. 37, 62, 66

DANG, L. M. et al. Sensor-based and vision-based human activity recognition: A comprehensive survey. *Pattern Recognition*, Elsevier Ltd, v. 108, 2020. ISSN 00313203. 26, 35, 44, 45, 65

DEMROZI, F. et al. Human activity recognition using inertial, physiological and environmental sensors: A comprehensive survey. *IEEE Access*, Institute of Electrical and Electronics Engineers (IEEE), v. 8, p. 210816–210836, 2020. ISSN 2169-3536. Disponível em: <http://dx.doi.org/10.1109/ACCESS.2020.3037715>. 31, 37

EDEL, M.; KöPPE, E. Binarized-blstm-rnn based human activity recognition. In: *2016 International Conference on Indoor Positioning and Indoor Navigation (IPIN)*. [S.l.: s.n.], 2016. p. 1–7. 68, 70

EHATISHAM-UL-HAQ, M. et al. C2FHAR: Coarse-to-Fine Human Activity Recognition with Behavioral Context Modeling Using Smart Inertial Sensors. *IEEE Access*, v. 8, p. 7731–7747, 2020. ISSN 21693536. 43

FAN, L.; WANG, Z.; WANG, H. Human activity recognition model based on decision tree. *Proceedings - 2013 International Conference on Advanced Cloud and Big Data, CBD 2013*, IEEE, p. 64–68, 2013. 62, 63

FERRI, F. et al. Comparative study of techniques for large-scale feature selection. In: GELSEMA, E. S.; KANAL, L. S. (Ed.). *Pattern Recognition in Practice IV*. North-Holland, 1994, (Machine Intelligence and Pattern Recognition, v. 16). p. 403–413. Disponível em: <https://www.sciencedirect.com/science/article/pii/B9780444818928500407>. 45

FITBIT. *Fitbit*. 2022. Disponível em: <https://www.fitbit.com/global/us/home>. 38

FITBIT. *Fitbit Versa 3*. 2022. Disponível em: <https://www.fitbit.com/global/us/products/smartwatches/versa3>. 39

FITBIT. *How do I track my workouts with my Fitbit device?* 2022. Disponível em: <https://help.fitbit.com/articles/en_US/Help_article/1935.htm>. 38

FOERSTER, F.; SMEJA, M.; FAHRENBERG, J. Detection of posture and motion by accelerometry: a validation study in ambulatory monitoring. *Computers in Human Behavior*, v. 15, n. 5, p. 571–583, 1999. ISSN 0747-5632. Disponível em: <https://www.sciencedirect.com/science/article/pii/S0747563299000370>. 62

FU, T. chung. A review on time series data mining. *Engineering Applications of Artificial Intelligence*, v. 24, n. 1, p. 164–181, 2011. ISSN 0952-1976. Disponível em: <https://www.sciencedirect.com/science/article/pii/S0952197610001727>. 43

GAO, L.; BOURKE, A.; NELSON, J. Evaluation of accelerometer based multi-sensor versus single-sensor activity recognition systems. *Medical Engineering & Physics*, v. 36, n. 6, p. 779–785, 2014. ISSN 1350-4533. Disponível em: <https://www.sciencedirect.com/science/article/pii/S1350453314000344>. 64

GARMIN. *Garmin Vivoactive 4*. 2022. Disponível em: <https://www.garmin.com/pt-PT/p/643382>. 39

GARMIN. *vívoactive 4/4S - Activity Tracking*. 2022. Disponível em: <https://www8.garmin.com/manuals/webhelp/vivoactive4_4S/EN-US/GUID-7A6BAD22-ACC7-451C-B40A-7D92B22A10AF.html>. 38

GARZóN-CASTRO, R. V. C. L. Fundamentals and paradigms in the internet of things. In: CARDONA VIJENDER KUMAR SOLANKI, C. E. G. C. M. (Ed.). *Internet of Medical Things: Paradigms of Wearable Devices*. [S.l.]: CRC Press, 2021. cap. 1, p. 1–18. 32

GOODFELLOW, I.; BENGIO, Y.; COURVILLE, A. *Deep Learning*. [S.l.]: MIT Press, 2016. <http://www.deeplearningbook.org>. 51, 52, 53, 54, 55, 56

GU, F. et al. Locomotion activity recognition using stacked denoising autoencoders. *IEEE Internet of Things Journal*, v. 5, n. 3, p. 2085–2093, 2018. 44

GUAN, Y.; PLöTZ, T. Ensembles of deep lstm learners for activity recognition using wearables. *Proc. ACM Interact. Mob. Wearable Ubiquitous Technol.*, Association for Computing Machinery, New York, NY, USA, v. 1, n. 2, jun. 2017. Disponível em: <https://doi.org/10.1145/3090076>. 68

GUYON, I.; ELISSEEFF, A. An introduction to feature extraction. In: ______. *Feature Extraction: Foundations and Applications*. Berlin, Heidelberg: Springer Berlin Heidelberg, 2006. p. 1–25. ISBN 978-3-540-35488-8. Disponível em: <https://doi.org/10.1007/978-3-540-35488-8_1>. 44, 45

HA, S.; YUN, J. M.; CHOI, S. Multi-modal Convolutional Neural Networks for Activity Recognition. *Proceedings - 2015 IEEE International Conference on Systems, Man, and Cybernetics, SMC 2015*, IEEE, p. 3017–3022, 2016. 67, 147

HAMMERLA, N. Y.; HALLORAN, S.; PLöTZ, T. Deep, convolutional, and recurrent models for human activity recognition using wearables. In: *Proceedings of the Twenty-Fifth International Joint Conference on Artificial Intelligence*. [S.l.]: AAAI Press, 2016. (IJCAI'16), p. 1533–1540. ISBN 9781577357704. 65

HAYKIN, S. *Neural Networks and Learning Machines*. Prentice Hall, 2009. (Neural networks and learning machines, v. 10). ISBN 9780131471399. Disponível em: <https://books.google.com.br/books?id=K7P36lKzI_QC>. 49, 50, 51

HOROWITZ, M. 1.1 computing's energy problem (and what we can do about it). In: *2014 IEEE International Solid-State Circuits Conference Digest of Technical Papers (ISSCC)*. [S.l.: s.n.], 2014. p. 10–14. 57

HU, C. et al. A novel random forests based class incremental learning method for activity recognition. *Pattern Recognition*, v. 78, p. 277–290, 2018. ISSN 0031-3203. Disponível em: <https://www.sciencedirect.com/science/article/pii/S0031320318300360>. 62

HUAWEI. *Huawei Band 4*. 2022. Disponível em: <https://consumer.huawei.com/br/wearables/band4/>. 38

IGNATOV, A. Real-time human activity recognition from accelerometer data using Convolutional Neural Networks. *Applied Soft Computing Journal*, Elsevier B.V., v. 62, p. 915–922, 2018. ISSN 1568-4946. Disponível em: <https://doi.org/10.1016/j.asoc.2017.09.027>. 66

IJJINA, E. P.; CHALAVADI, K. M. Human action recognition in rgb-d videos using motion sequence information and deep learning. *Pattern Recognition*, v. 72, p. 504–516, 2017. ISSN 0031-3203. Disponível em: <https://www.sciencedirect.com/science/article/pii/S0031320317302844>. 65

JABBAR, M. A. Internet of medical things (iomt): a systematic review of applications, trends, challenges, and future directions. In: PANI, S. et al. (Ed.). *The Internet of Medical Things: Enabling Technologies and Emerging Applications*. Institution of Engineering and Technology, 2021, (Healthcare Technologies). cap. 1, p. 1–18. Disponível em: <https://books.google.com.br/books?id=qGGYzgEACAAJ>. 32

JAFARI, A. et al. SensorNet: A Scalable and Low-Power Deep Convolutional Neural Network for Multimodal Data Classification. *IEEE Transactions on Circuits and Systems I: Regular Papers*, IEEE, v. 66, n. 1, p. 274–287, 2019. ISSN 15580806. 70, 148

JAKICIC, J. M. et al. Strategies for physical activity interventions in the treatment of obesity. *Endocrinology and Metabolism Clinics of North America*, v. 49, n. 2, p. 289–301, 2020. ISSN 0889-8529. Obesity. Disponível em: <https://www.sciencedirect.com/science/article/pii/S0889852920300062>. 25

JALAL, A.; KAMAL, S.; KIM, D. A depth video sensor-based life-logging human activity recognition system for elderly care in smart indoor environments. *Sensors*, v. 14, n. 7, p. 11735–11759, 2014. ISSN 1424-8220. Disponível em: <https://www.mdpi.com/1424-8220/14/7/11735>. 34, 35

JANIDARMIAN, M. et al. A comprehensive analysis on wearable acceleration sensors in human activity recognition. *Sensors*, v. 17, n. 3, 2017. ISSN 1424-8220. Disponível em: <https://www.mdpi.com/1424-8220/17/3/529>. 62

JEONG, C. Y.; KIM, M. An energy-efficient method for human activity recognition with segment-level change detection and deep learning. *Sensors (Switzerland)*, v. 19, n. 17, p. 4–11, 2019. ISSN 14248220. 71

JIANG, W.; YIN, Z. Human activity recognition using wearable sensors by deep convolutional neural networks. In: *Proceedings of the 23rd ACM International Conference on Multimedia*. New York, NY, USA: Association for Computing Machinery, 2015. (MM ’15), p. 1307–1310. ISBN 9781450334594. Disponível em: <https://doi.org/10.1145/2733373.2806333>. 67

JOINER, I. A. Chapter 8 - wearable technologies from a to z. In: JOINER, I. A. (Ed.). *Emerging Library Technologies*. Chandos Publishing, 2018, (Chandos Information Professional Series). p. 155–167. ISBN 978-0-08-102253-5. Disponível em: <https://www.sciencedirect.com/science/article/pii/B9780081022535000083>. 36, 38

JORDAO, A. et al. *Human Activity Recognition Based on Wearable Sensor Data: A Standardization of the State-of-the-Art*. arXiv, 2018. Disponível em: <https://arxiv.org/abs/1806.05226>. 43

KE, G. et al. Lightgbm: A highly efficient gradient boosting decision tree. In: GUYON, I. et al. (Ed.). *Advances in Neural Information Processing Systems*. Curran Associates, Inc., 2017. v. 30. Disponível em: <https://proceedings.neurips.cc/paper/2017/file/6449f44a102fde848669bdd9eb6b76fa-Paper.pdf>. 48

KEMPE, V. Gyroscopes. In: _____. *Inertial MEMS: Principles and Practice*. [S.l.]: Cambridge University Press, 2011. p. 364–459. 40

KEMPE, V. Introduction. In: _____. *Inertial MEMS: Principles and Practice*. [S.l.]: Cambridge University Press, 2011. p. 1–12. 39

KERAS. *Keras documentation: Layer activation functions*. 2022. Disponível em: <https://keras.io/api/layers/activations/>. 51

KIM, E.; HELAL, S.; COOK, D. Human activity recognition and pattern discovery. *IEEE Pervasive Computing*, Institute of Electrical and Electronics Engineers (IEEE), v. 9, n. 1, p. 48–53, jan. 2010. Disponível em: <https://doi.org/10.1109/mprv.2010.7>. 33

KIRANYAZ, S. et al. 1d convolutional neural networks and applications: A survey. *Mechanical Systems and Signal Processing*, v. 151, p. 107398, 2021. ISSN 0888-3270. Disponível em: <https://www.sciencedirect.com/science/article/pii/S0888327020307846>. 54, 55

KONGSIL, K.; SUKSAWATCHON, J.; SUKSAWATCHON, U. Wrist-worn Physical Activity Recognition: A Fusion Learning Approach. *InCIT 2020 - 5th International Conference on Information Technology*, p. 116–121, 2020. 64, 147, 148, 149

KUMARI, P.; MATHEW, L.; SYAL, P. Increasing trend of wearables and multimodal interface for human activity monitoring: A review. *Biosensors and Bioelectronics*, Elsevier, v. 90, p. 298–307, 2017. ISSN 18734235. Disponível em: <http://dx.doi.org/10.1016/j.bios.2016.12.001>. 37

KWAPISZ, J. R.; WEISS, G. M.; MOORE, S. A. Activity recognition using cell phone accelerometers. In: *Proceedings of the Fourth International Workshop on Knowledge Discovery from Sensor Data*. [S.l.: s.n.], 2010. p. 10–18. 61

LABRADOR, M. A.; YEJAS, O. D. L. *Human Activity Recognition*. Chapman and Hall/CRC, 2013. Disponível em: <https://doi.org/10.1201/b16098>. 26, 31, 33, 34

LAERHOVEN, K. V.; BERLIN, E.; SCHIELE, B. Enabling efficient time series analysis for wearable activity data. *8th International Conference on Machine Learning and Applications, ICMLA 2009*, p. 392–397, 2009. 44

LARA, O. D.; LABRADOR, M. A. A survey on human activity recognition using wearable sensors. *IEEE Communications Surveys Tutorials*, v. 15, n. 3, p. 1192–1209, 2013. 64

LE-HONG, P.; LE, A.-C. A comparative study of neural network models for sentence classification. In: *2018 5th NAFOSTED Conference on Information and Computer Science (NICS)*. [S.l.: s.n.], 2018. p. 360–365. 65

LEE, J.; KIM, J. Energy-Efficient Real-Time Human Activity Recognition on Smart Mobile Devices. v. 2016, 2016. 71

LEE, Y.-S.; CHO, S.-B. Activity recognition using hierarchical hidden markov models on a smartphone with 3d accelerometer. In: CORCHADO, E.; KURZYŃSKI, M.; WOŹNIAK, M. (Ed.). *Hybrid Artificial Intelligent Systems*. Berlin, Heidelberg: Springer Berlin Heidelberg, 2011. p. 460–467. ISBN 978-3-642-21219-2. 62, 63

LIASHCHYNSKYI, P.; LIASHCHYNSKYI, P. Grid Search, Random Search, Genetic Algorithm: A Big Comparison for NAS. n. 2017, p. 1–11, 2019. Disponível em: <http://arxiv.org/abs/1912.06059>. 58

LIGHTGBM. *LightGBM Features*. 2022. Disponível em: <https://lightgbm.readthedocs.io/en/latest/Features.html>. 48

LONG, N. et al. A scoping review on monitoring mental health using smart wearable devices. *Mathematical Biosciences and Engineering*, v. 19, n. 8, p. 7899–7919, 2022. ISSN 1551-0018. Disponível em: <https://www.aimspress.com/article/doi/10.3934/mbe.2022369>. 25

LONG, X. L. X.; YIN, B. Y. B.; AARTS, R. Single-accelerometer-based daily physical activity classification. *2009 Annual International Conference of the IEEE Engineering in Medicine and Biology Society*, p. 6107–6110, 2009. ISSN 1557-170X. 62, 63

LU, L. et al. Wearable health devices in health care: Narrative systematic review. *JMIR Mhealth Uhealth*, v. 8, n. 11, p. e18907, Nov 2020. ISSN 2291-5222. Disponível em: <http://mhealth.jmir.org/2020/11/e18907/>. 25

LUCZAK, T. et al. State-of-the-art review of athletic wearable technology: What 113 strength and conditioning coaches and athletic trainers from the USA said about technology in sports. *International Journal of Sports Science &amp Coaching*, SAGE Publications, v. 15, n. 1, p. 26–40, nov. 2019. Disponível em: <https://doi.org/10.1177/1747954119885244>. 36

MAHMUD, T. et al. Human Activity Recognition from Multi-modal Wearable Sensor Data Using Deep Multi-stage LSTM Architecture Based on Temporal Feature Aggregation. *Midwest Symposium on Circuits and Systems*, v. 2020-Augus, p. 249–252, 2020. ISSN 15483746. 65

MAJUMDER, S.; MONDAL, T.; DEEN, M. J. Wearable sensors for remote health monitoring. *Sensors*, v. 17, n. 1, 2017. ISSN 1424-8220. Disponível em: <https://www.mdpi.com/1424-8220/17/1/130>. 36

MATHIE, M. J. et al. Classification of basic daily movements using a triaxial accelerometer. *Medical &amp Biological Engineering &amp Computing*, Springer Science and Business Media LLC, v. 42, n. 5, p. 679–687, set. 2004. Disponível em: <https://doi.org/10.1007/bf02347551>. 40

MAXIM INTEGRATED. *Accelerometer and gyroscopes sensors: Operation, sensing, and applications*. 2022. Disponível em: <https://www.maximintegrated.com/en/design/technical-documents/app-notes/5/5830.html>. 39

MEKRUKSAVANICH, S.; JITPATTANAKUL, A. Smartwatch-based Human Activity Recognition Using Hybrid LSTM Network. *Proceedings of IEEE Sensors*, v. 2020-Octob, p. 8–11, 2020. ISSN 21689229. 68, 148, 149

MICUCCI, D.; MOBILIO, M.; NAPOLETANO, P. Unimib shar: A dataset for human activity recognition using acceleration data from smartphones. *Applied Sciences*, v. 7, n. 10, 2017. ISSN 2076-3417. Disponível em: <https://www.mdpi.com/2076-3417/7/10/1101>. 64

MITCHELL, T. M. *Machine Learning*. New York: McGraw-Hill, 1997. ISBN 978-0-07-042807-2. 46, 47

MORALES, J.; AKOPIAN, D. Physical activity recognition by smartphones, a survey. *Biocybernetics and Biomedical Engineering*, v. 37, n. 3, p. 388–400, 2017. ISSN 0208-5216. Disponível em: <https://www.sciencedirect.com/science/article/pii/S020852161630314X>. 62

MOZAFFARI, M. H.; TAY, L.-L. *A Review of 1D Convolutional Neural Networks toward Unknown Substance Identification in Portable Raman Spectrometer*. arXiv, 2020. Disponível em: <https://arxiv.org/abs/2006.10575>. 55

MUKHOPADHYAY, S. C. Wearable sensors for human activity monitoring: A review. *IEEE Sensors Journal*, v. 15, n. 3, p. 1321–1330, 2015. ISSN 1530437X. 26

MUNOZ-ORGANERO, M. Outlier Detection in Wearable Sensor Data for Human Activity Recognition (HAR) Based on DRNNs. *IEEE Access*, IEEE, v. 7, p. 74422–74436, 2019. ISSN 21693536. 31

MURAD, A.; PYUN, J.-Y. Deep recurrent neural networks for human activity recognition. *Sensors*, v. 17, n. 11, 2017. ISSN 1424-8220. Disponível em: <https://www.mdpi.com/1424-8220/17/11/2556>. 34, 67

NGUYEN, T. N. et al. A novel analysis-prediction approach for geometrically nonlinear problems using group method of data handling. *Computer Methods in Applied Mechanics and Engineering*, v. 354, p. 506–526, 2019. ISSN 0045-7825. Disponível em: <https://www.sciencedirect.com/science/article/pii/S0045782519303317>. 65

NOOR, M. H. M.; SALCIC, Z.; WANG, K. I.-K. Adaptive sliding window segmentation for physical activity recognition using a single tri-axial accelerometer. *Pervasive and Mobile Computing*, v. 38, p. 41–59, 2017. ISSN 1574-1192. Disponível em: <https://www.sciencedirect.com/science/article/pii/S1574119216302280>. 43

NWEKE, H. F. et al. *Deep learning algorithms for human activity recognition using mobile and wearable sensor networks: State of the art and research challenges*. 2018. 55, 56

OMETOV, A. et al. A Survey on Wearable Technology: History, State-of-the-Art and Current Challenges. *Computer Networks*, Elsevier B.V., v. 193, n. December 2020, p. 108074, 2021. ISSN 13891286. Disponível em: <https://doi.org/10.1016/j.comnet.2021.108074>. 36, 37

ORDóñEZ, F. J.; ROGGEN, D. Deep convolutional and lstm recurrent neural networks for multimodal wearable activity recognition. *Sensors*, v. 16, n. 1, 2016. ISSN 1424-8220. Disponível em: <https://www.mdpi.com/1424-8220/16/1/115>. 68

OYEDOTUN, O. K.; KHASHMAN, A. Deep learning in vision-based static hand gesture recognition. *Neural Computing and Applications*, v. 28, n. 12, p. 3941–3951, Dec 2017. ISSN 1433-3058. Disponível em: <https://doi.org/10.1007/s00521-016-2294-8>. 26

PASSARO, V. M. et al. Gyroscope technology and applications: A review in the industrial perspective. *Sensors (Switzerland)*, v. 17, n. 10, 2017. ISSN 14248220. 40

QI, J. et al. A hybrid hierarchical framework for gym physical activity recognition and measurement using wearable sensors. *IEEE Internet of Things Journal*, v. 6, n. 2, p. 1384–1393, 2019. 26

RASCHKA, S. *Model Evaluation, Model Selection, and Algorithm Selection in Machine Learning.* arXiv, 2018. Disponível em: <https://arxiv.org/abs/1811.12808>. 58

RASHID, N.; DEMIREL, B. U.; FARUQUE, M. A. A. Ahar: Adaptive cnn for energy-efficient human activity recognition in low-power edge devices. *ArXiv*, abs/2102.01875, 2022. 70

RAVI, N. et al. Activity recognition from accelerometer data. *Proceedings of the National Conference on Artificial Intelligence*, v. 3, n. January, p. 1541–1546, 2005. 62

REISS, A.; STRICKER, D. Creating and benchmarking a new dataset for physical activity monitoring. In: *PETRA '12*. [S.l.: s.n.], 2012. 61, 82

REISS, A.; STRICKER, D. Introducing a new benchmarked dataset for activity monitoring. In: *2012 16th International Symposium on Wearable Computers*. [S.l.: s.n.], 2012. p. 108–109. 61, 82

RODRIGUES, J. J. et al. Enabling Technologies for the Internet of Health Things. *IEEE Access*, v. 6, p. 13129–13141, 2018. ISSN 21693536. 26, 32, 33

ROGGEN, D. et al. Collecting complex activity datasets in highly rich networked sensor environments. In: *2010 Seventh International Conference on Networked Sensing Systems (INSS)*. [S.l.: s.n.], 2010. p. 233–240. 62

RONAO, C. A.; CHO, S.-B. Recognizing human activities from smartphone sensors using hierarchical continuous hidden markov models. *International Journal of Distributed Sensor Networks*, SAGE Publications Sage UK: London, England, v. 13, n. 1, p. 1550147716683687, 2017. 62

RUDER, S. An overview of gradient descent optimization algorithms. *CoRR*, abs/1609.04747, 2016. Disponível em: <http://arxiv.org/abs/1609.04747>. 51, 52

SAMSUNG. *Do Galaxy smatwatches have Fall Detection?* 2022. Disponível em: <https://www.samsung.com/global/galaxy/what-is/fall-detection/>. 38

SAMSUNG. *Samsung Galaxy Watch4*. 2022. Disponível em: <https://www.samsung.com/br/watches/galaxy-watch/galaxy-watch4-black-bluetooth-sm-r870nzkpzto/>. 39

SAMSUNG. *Use automatic workout detection on your Samsung smart watch*. 2022. Disponível em: <https://www.samsung.com/us/support/answer/ANS00083510/>. 38

SCIKIT-LEARN. *Feature importances with a forest of trees*. 2022. Disponível em: <https://scikit-learn.org/stable/auto_examples/ensemble/plot_forest_importances.html#feature-importance-based-on-mean-decrease-in-impurity>. 45

SEGAL, M. R. Regression trees for censored data. *Biometrics*, [Wiley, International Biometric Society], v. 44, n. 1, p. 35–47, 1988. ISSN 0006341X, 15410420. Disponível em: <http://www.jstor.org/stable/2531894>. 46

SENA, J. et al. Human activity recognition based on smartphone and wearable sensors using multiscale DCNN ensemble. *Neurocomputing*, Elsevier B.V., v. 444, p. 226–243, 2021. ISSN 18728286. Disponível em: <https://doi.org/10.1016/j.neucom.2020.04.151>. 147, 148

SEYFIOğLU, M. S.; ÖZBAYOğLU, A. M.; GüRBüZ, S. Z. Deep convolutional autoencoder for radar-based classification of similar aided and unaided human activities. *IEEE Transactions on Aerospace and Electronic Systems*, v. 54, n. 4, p. 1709–1723, 2018. 65

SHI, W. et al. Edge computing: Vision and challenges. *IEEE Internet of Things Journal*, v. 3, n. 5, p. 637–646, 2016. 71

SHOAIB, M. et al. Complex human activity recognition using smartphone and wrist-worn motion sensors. *Sensors*, v. 16, n. 4, 2016. ISSN 1424-8220. Disponível em: <https://www.mdpi.com/1424-8220/16/4/426>. 43

SINGH, S. P. et al. Deep ConvLSTM with Self-Attention for Human Activity Decoding Using Wearable Sensors. *IEEE Sensors Journal*, v. 21, n. 6, p. 8575–8582, 2021. ISSN 15581748. 68, 69, 147

SOK, P. et al. Activity recognition for incomplete spinal cord injury subjects using hidden markov models. *IEEE Sensors Journal*, v. 18, n. 15, p. 6369–6374, 2018. 62

STMICROELECTRONICS. *STM32 32-bit Arm Cortex MCUs*. 2021. Accessed on July 17, 2021. 74

STMICROELECTRONICS. *STM32F4DISCOVERY*. 2021. Accessed on March 31, 2021. Disponível em: <https://www.st.com/en/evaluation-tools/32f411ediscovery.html#overview>. 93, 94

STMICROELECTRONICS. *Getting started with X-CUBE-AI Expansion Package for Artificial Intelligence (AI)*. [S.l.], 2022. Disponível em: <https://www.st.com/resource/en/data_brief/x-cube-ai.pdf>. 95

STMICROELECTRONICS. *STM32CubeIDE - Integrated Development Environment for STM32 - STMicroelectronics*. 2022. Disponível em: <https://www.st.com/en/development-tools/stm32cubeide.html>. 94

STMICROELECTRONICS. *STM32CubeMX - STM32Cube initialization code generator - STMicroelectronics*. 2022. Disponível em: <https://www.st.com/en/development-tools/stm32cubemx.html>. 94

STMICROELECTRONICS. *X-CUBE-AI - AI expansion pack for STM32CubeMX - STMicroelectronics*. 2022. Disponível em: <https://www.st.com/en/embedded-software/x-cube-ai.html>. 94

STRIK, M. et al. Smartwatch-based detection of cardiac arrhythmias: Beyond the differentiation between sinus rhythm and atrial fibrillation. *Heart Rhythm*, v. 18, n. 9, p. 1524–1532, 2021. ISSN 1547-5271. Focus Issue: Atrial Fibrillation. Disponível em: <https://www.sciencedirect.com/science/article/pii/S1547527121017318>. 38

SU, T. et al. HDL: Hierarchical Deep Learning Model based Human Activity Recognition using Smartphone Sensors. *Proceedings of the International Joint Conference on Neural Networks*, IEEE, v. 2019-July, n. July, p. 1–8, 2019. 68

SUH, M. 12 - wearable sensors for athletes. In: DIAS, T. (Ed.). *Electronic Textiles*. Oxford: Woodhead Publishing, 2015. p. 257–273. ISBN 978-0-08-100201-8. Disponível em: <https://www.sciencedirect.com/science/article/pii/B9780081002018000138>. 39

SUN, B. et al. Attention-based LSTM Network for Wearable Human Activity Recognition. *2019 Chinese Control Conference (CCC)*, Technical Committee on Control Theory, Chinese Association of Automation, n. 1, p. 8677–8682, 2019. 68

SZE, V. et al. Efficient Processing of Deep Neural Networks : A Tutorial and Survey. p. 1–32, 2017. 26, 53, 57

TANG, Y. et al. Layer-Wise Training Convolutional Neural Networks with Smaller Filters for Human Activity Recognition Using Wearable Sensors. *IEEE Sensors Journal*, IEEE, v. 21, n. 1, p. 581–592, 2021. ISSN 15581748. 70, 148

THEODORIDIS, S.; KOUTROUMBAS, K. Chapter 1 - introduction. In: THEODORIDIS, S.; KOUTROUMBAS, K. (Ed.). *Pattern Recognition (Fourth Edition)*. Fourth edition. Boston: Academic Press, 2009. p. 1–12. ISBN 978-1-59749-272-0. Disponível em: <https://www.sciencedirect.com/science/article/pii/B9781597492720500037>. 42

TUFEK, N. et al. Human action recognition using deep learning methods on limited sensory data. *IEEE Sensors Journal*, v. 20, n. 6, p. 3101–3112, 2020. 68

UDDIN, M. T.; BILLAH, M. M.; HOSSAIN, M. F. Random forests based recognition of human activities and postural transitions on smartphone. In: *2016 5th International Conference on Informatics, Electronics and Vision (ICIEV)*. [S.l.: s.n.], 2016. p. 250–255. 62, 64

UNITED-NATIONS. *World Population Prospects: The 2017 Revision*. 2017. Disponível em: <https://www.un.org/development/desa/publications/world-population-prospects-the-2017-revision.html>. 25

VITA, A. D. et al. Low-Power HWAccelerator for AI Edge-Computing in Human Activity Recognition Systems. *Proceedings - 2020 IEEE International Conference on Artificial Intelligence Circuits and Systems, AICAS 2020*, v. 6, n. 1, p. 291–295, 2020. 70

VOGHOEI, S. et al. Deep learning at the edge. *CoRR*, abs/1910.10231, 2019. Disponível em: <http://arxiv.org/abs/1910.10231>. 41

VONG, C. et al. Comparison of feature selection and classification for human activity and fall recognition using smartphone sensors. In: *2021 Joint International Conference on Digital Arts, Media and Technology with ECTI Northern Section Conference on Electrical, Electronics, Computer and Telecommunication Engineering*. [S.l.: s.n.], 2021. p. 170–173. 62, 64

WAN, S. et al. Deep learning models for real-time human activity recognition with smartphones. *Mobile Networks and Applications*, v. 25, n. 2, p. 743–755, Apr 2020. ISSN 1572-8153. Disponível em: <https://doi.org/10.1007/s11036-019-01445-x>. 66, 148

WANG, A. et al. A Comparative Study on Human Activity Recognition Using Inertial Sensors in a Smartphone. *IEEE Sensors Journal*, v. 16, n. 11, p. 4566–4578, 2016. ISSN 1530437X. 34

WANG, J. et al. Deep learning for sensor-based activity recognition : A Survey. *Pattern Recognition Letters*, Elsevier B.V., v. 0, p. 1–9, 2018. ISSN 0167-8655. Disponível em: <https://doi.org/10.1016/j.patrec.2018.02.010>. 34, 38, 65

WANG, K.; HE, J.; ZHANG, L. Attention-based convolutional neural network for weakly labeled human activities' recognition with wearable sensors. *IEEE Sensors Journal*, v. 19, n. 17, p. 7598–7604, 2019. 33, 36, 43, 67

WEI, X.-S. et al. Mask-cnn: Localizing parts and selecting descriptors for fine-grained bird species categorization. *Pattern Recognition*, v. 76, p. 704–714, 2018. ISSN 0031-3203. Disponível em: <https://www.sciencedirect.com/science/article/pii/S0031320317303990>. 65

WEISS, G. M.; YONEDA, K.; HAYAJNEH, T. Smartphone and smartwatch-based biometrics using activities of daily living. *IEEE Access*, v. 7, p. 133190–133202, 2019. 26, 61, 83

XIAOMI. *Xiaomi Mi Band 6*. 2022. Disponível em: <https://www.mi.com/global/product/mi-smart-band-6>. 38

YADAV, S. K. et al. Skeleton-based human activity recognition using ConvLSTM and guided feature learning. *Soft Computing*, Springer Science and Business Media LLC, v. 26, n. 2, p. 877–890, out. 2021. Disponível em: <https://doi.org/10.1007/s00500-021-06238-7>. 34

YANG, G.-Z. et al. Multi-sensor fusion. In: _____. *Body Sensor Networks*. London: Springer London, 2014. p. 301–354. ISBN 978-1-4471-6374-9. Disponível em: <https://doi.org/10.1007/978-1-4471-6374-9_8>. 44

YANG, J. B. et al. Deep convolutional neural networks on multichannel time series for human activity recognition. In: *Proceedings of the 24th International Conference on Artificial Intelligence*. [S.l.]: AAAI Press, 2015. (IJCAI'15), p. 3995–4001. ISBN 9781577357384. 26, 64

YANG, Q. Activity recognition: Linking low-level sensors to high-level intelligence. In: *Proceedings of the 21st International Joint Conference on Artificial Intelligence*. San Francisco, CA, USA: Morgan Kaufmann Publishers Inc., 2009. (IJCAI'09), p. 20–25. 26, 44, 64

YANG, T.-j. et al. A Method to Estimate the Energy Consumption of Deep Neural Networks. p. 1916–1920, 2017. 57

YEN, H.-Y.; HUANG, H.-Y. Comparisons of physical activity and sedentary behavior between owners and non-owners of commercial wearable devices. *Perspectives in Public Health*, v. 141, n. 2, p. 89–96, 2021. PMID: 33733947. Disponível em: <https://doi.org/10.1177/1757913921989389>. 25

ZAPPI, P. et al. Activity recognition from on-body sensors: Accuracy-power trade-off by dynamic sensor selection. In: VERDONE, R. (Ed.). *Wireless Sensor Networks*. Berlin, Heidelberg: Springer Berlin Heidelberg, 2008. p. 17–33. ISBN 978-3-540-77690-1. 62

ZEBIN, T.; SCULLY, P. J.; OZANYAN, K. B. Human activity recognition with inertial sensors using a deep learning approach. In: *2016 IEEE SENSORS*. IEEE, 2016. p. 1–3. ISBN 978-1-4799-8287-5. ISSN 21689229. Disponível em: <http://ieeexplore.ieee.org/document/7808590/>. 66

Zebin, T. et al. Design and implementation of a convolutional neural network on an edge computing smartphone for human activity recognition. *IEEE Access*, v. 7, p. 133509–133520, 2019. 71

ZENG, M. et al. Convolutional neural networks for human activity recognition using mobile sensors. In: *6th International Conference on Mobile Computing, Applications and Services*. [S.l.: s.n.], 2014. p. 197–205. 66

ZHENG, A.; CASARI, A. *Feature Engineering for Machine Learning: Principles and Techniques for Data Scientists*. 1st. ed. [S.l.]: O'Reilly Media, Inc., 2018. ISBN 1491953241. 44, 45, 51

ZHENG, L. et al. A novel energy-efficient approach for human activity recognition. *Sensors (Switzerland)*, v. 17, n. 9, p. 1–21, 2017. ISSN 14248220. 70

ZHOU, L. et al. Machine learning on big data: Opportunities and challenges. *Neurocomputing*, v. 237, p. 350–361, 2017. ISSN 0925-2312. Disponível em: <https://www.sciencedirect.com/science/article/pii/S0925231217300577>. 42

ZHU, J.; SAN-SEGUNDO, R.; PARDO, J. M. Feature extraction for robust physical activity recognition. *Human-centric Computing and Information Sciences*, v. 7, n. 1, p. 16, Jun 2017. ISSN 2192-1962. Disponível em: <https://doi.org/10.1186/s13673-017-0097-2>. 70

ZHU, J. et al. Electric vehicle charging load forecasting: A comparative study of deep learning approaches. *Energies*, v. 12, n. 14, 2019. ISSN 1996-1073. Disponível em: <https://www.mdpi.com/1996-1073/12/14/2692>. 26, 56

ZHU, R. A. N. et al. Efficient Human Activity Recognition Solving the Confusing Activities Via Deep Ensemble Learning. *IEEE Access*, IEEE, v. 7, p. 75490–75499, 2019. 38, 65