UNIVERSIDADE FEDERAL DO CEARÁ

CENTRO DE CIÊNCIAS

DEPARTAMENTO DE COMPUTAÇÃO

PROGRAMA DE PÓS-GRADUAÇÃO EM CIÊNCIA DA COMPUTAÇÃO

DOUTORADO EM CIÊNCIA DA COMPUTAÇÃO

LUCAS PERES GASPAR

AETHER: AUGMENTATION AND EPISODIC TASK HARNESSING FOR EFFICIENT RECOGNITION

FORTALEZA

2026

LUCAS PERES GASPAR

AETHER: AUGMENTATION AND EPISODIC TASK HARNESSING FOR EFFICIENT RECOGNITION

Tese apresentada ao Programa de Pós-Graduação em Ciência da Computação do Centro de Ciências da Universidade Federal do Ceará, como requisito parcial à obtenção do título de doutor em Ciência da Computação. Área de Concentração: Banco de Dados.

Orientador: Prof. Dr. Jose Antonio Fernandes de Macedo.
Coorientadora: Profa. Dra. Lívia Almada Cruz.

FORTALEZA

2026

LUCAS PERES GASPAR

AETHER: AUGMENTATION AND EPISODIC TASK HARNESSING FOR EFFICIENT RECOGNITION

Tese apresentada ao Programa de Pós-Graduação em Ciência da Computação do Centro de Ciências da Universidade Federal do Ceará, como requisito parcial à obtenção do título de doutor em Ciência da Computação. Área de Concentração: Banco de Dados.

Aprovada em: 26/02/2025.

BANCA EXAMINADORA

Prof. Dr. Jose Antonio Fernandes de Macedo (Orientador)
Universidade Federal do Ceará (UFC)

Prof. Dr. João Paulo do Vale Madeiro
Universidade Federal do Ceará (UFC)

Dra. Chiara Renso
Consiglio Nazionale delle Ricerche (CNR-Italy)

Prof. Dr. Regis Pires Magalhães
Universidade Federal do Ceará (UFC)

Prof. Dr. Luis Gustavo Coutinho do Rêgo
Instituto Federal de Educação, Ciência e Tecnologia do Estado do Ceará (IFCE)

A Deus, pois foi Tua graça que me capacitou.À minha esposa, que se manteve ao meu lado durante todos os momentos. À minha família, por sua capacidade de acreditar em mim e investir em mim. A meus amigos, que me apoiaram em todos os momentos.

ACKNOWLEDGEMENTS

A Deus, por me sustentar todos os dias, me dando forças para lutar por todos os meus objetivos. À minha esposa, que sempre me motivou e nunca desistiu de mim, mesmo quando eu o havia. Que me acompanha dia após dia, seja fácil ou difícil, nunca me deixando desistir ou aceitar menos do que meus sonhos. À minha mãe, que me serviu de exemplo para alcançar o título de doutor. A meu pai, que sempre me incentivou a estudar a área em que hoje atuo.

Aos membros das minhas bancas de doutorado, cujos comentários contribuíram grandemente para os resultados alcançados. Em especial, para meu orientador, José Macedo, que me abriu portas no começo de minha carreira, permitindo que eu seguisse essa jornada, e minha co-orientadora Lívia Almada, que foi fundamental para que todas nossas ideias tomassem a forma deste trabalho.

A meus amigos e professores que são ou já foram do Insight Data Science Lab, onde vivi anos muito importantes para minha vida, tanto pessoal quanto acadêmica. A meus colegas de graduação e pós, que lutaram comigo muitas batalhas.

A todos os meus professores, desde os do início dos meus estudos no Colégio Universo, no encerramento do meu ensino médio no Colégio Master, e na minha vida acadêmica na Universidade Federal do Ceará. Obrigado por cada ensinamento, cada desafio, cada crítica e repreensão. Jamais esquecerei do papel fundamental de vocês em minha vida.

Aos meus irmãos da Igreja Batista Regular Maanaim, em especial aos do polo Caucaia, que tanto me colocaram em suas orações e desejaram para que este dia chegasse. Aos meus companheiros de mesa de RPG, cujas aventuras me ajudaram a desopilar durante muitos anos.

Por último, mas jamais menos importante, à minha filha, que mesmo ainda não tendo nascido, já me motiva a ser um bom exemplo de estudo e perseverança.

”Always pass on what you have learned.”

(Master Yoda (Frank Oz), in *Star Wars Episode VI: Return of the Jedi*, 1983)

ABSTRACT

O Reconhecimento de Atividades Humanas (RAH) tem se tornado uma significante área de pesquisa para análise de comportamento humano. Pesquisas da metade da década passada provam que modelos de aprendizagem profunda são adequados para identificar padrões sobre series temporais coletadas de dispositivos inteligentes (*smartphones* e *smartwatches*) e realizar um reconhecimento preciso de atividades sob um conjunto fixo de atividades observadas. Entretanto, métodos de aprendizagem profunda encaram alguns desafios para dados de séries temporais, como a falta de dados suficientes para treinar um modelo eficiente. Outro desafio que acompanha RAH são as particularidades na forma como usuários realizam uma mesma atividade, ou até mesmo na forma como diferentes sensores coletam os dados, gerando algumas condições individuais. Esta tese de doutorado apresenta um algoritmo de meta-aprendizagem que supera a limitação das condições individuais provendo uma estratégia de treinamento que facilita a generalização através de diferentes tarefas, permitindo que o modelo se adapte rapidamente a novos usuários, sensores, e atividades. Ele também supera o problema de escassez de dados usando uma etapa de aumento de dados para aumentar o número de observações usadas durante o meta-treinamento. O algoritmo é comparado com a literatura usando conjuntos de dados reais e públicos, obtendo bons resultados, superando competidores da literatura em 20%. Os meta-modelos trainados são aplicados sobre outros conjuntos de dados públicos, permitindo avaliá-los em cenários completamente novos, onde o algoritmo proposto foi capaz de superar, em alguns casos, competidores em mais de 40%.

Keywords: meta-learning; reconhecimento de atividade humana; aumentação de dados; aprendizado profundo.

ABSTRACT

Human Activity Recognition (HAR) has become a significant research area for human behavior analysis. Researches from the middle from the last decade prove that deep learning based models are suitable to identify patterns over time series data collected from smart devices (smartphones, smartwatches) and perform accurate activity recognition over a fixed set of observed activities. However, deep learning approaches face some challenges for time series data, like the lack of sufficient data to train an efficient model. Another challenge that comes with HAR is the particularities in the way that users perform the same activity or how the sensor collects the data, generating some individual conditions. This Ph.D. thesis presents a meta-learning algorithm that overcomes the individual condition limitation by providing a training strategy that facilitates the generalization across different tasks, allowing the model to adapt to unseen users, sensors, and activities. It also overcomes the labeled data scarcity limitation by proposing a data augmentation stage to increase the number of observations to be used during the meta-training. The algorithm is compared against the literature using real-world public datasets and obtains encouraging results. The algorithm is compared against the literature using real-world public datasets and obtains good results, surpassing some literature baselines by 20%. Furthermore, the trained meta-models are applied against other public datasets, allowing us to evaluate the meta-models in completely new scenarios, where the proposed algorithm was able to overcome, in some cases, the baselines by over than 40%.

Keywords: meta-learning; human activity recognition; data augmentation; deep learning.

LIST OF FIGURES

LIST OF TABLES

LIST OF ALGORITHMS

LIST OF ACRONYMS AND ABBREVIATIONS

AETHER	Augmentation and Episodic Task Harnessing for Efficient Recognition
Bi-LSTM	Bidirectional LSTM
CNN	Convolution Neural Network
GRU	Gated Recurrent Units
HAR	Human Activity Recognition
HHAR	Heterogeneity Human Activity Recognition
ICD	Individual Condition Dataset
IoT	Internet of Things
KNN	K-Nearest-Neighbors
KU-HAR	Khulna University Human Activity Recognition
LSTM	Long Short Term Memory
MAML	Model Agnostic Meta-Learning
MLP	Multi Layer Perceptrons
RNN	Recurrent Neural Network
SNAIL	Simple neural attentive meta-learner
SVM	Support Vector Machine
UCI-HAR	University of California Irvine Human Activity Recognition
WISDM	Wireless Sensor Data Mining Dataset

LIST OF SYMBOLS

$\mathbf{X}$	Set of observed samples
Y	Set of known outputs
θ	Parameters from a model (coefficients, correlation matrix, weights, etc.)
f_θ	Function representation of a supervised model with parameters θ
D	Dataset of observed samples $\mathbf{X}$ and their outputs Y
L	Loss function
$L_D(\theta)$	Loss function for the parameters θ over the dataset D
ω	Meta-knowledge
g_ω	Function representation of the meta-model with parameters ω
T	Task containing a set of observations and their outputs
$p(T)$	Distribution of tasks
T^S	Support set of a task
T^Q	Query set of a task
N	Number of classes per task
k	Number of observations per class inside a task
α	Learning rate to be used inside tasks optimizations
β	Learning rate to be used for meta-knowledge optimizations
D_{icd}	Dataset where all observations are from a same ICD
(x, y)	Pair of observed sample and its output
Y_T	Set of distinct classes present into a task
$\lvert Y_T \rvert$	Number of distinct classes present into a task
x^m	New observation obtained applying the data augmentation method m over x
M	Set of modifications used to generate the augmented dataset
$\lvert M \rvert$	Number of modifications used to generate the augmented dataset
μ	Mean
σ	Standard deviation
$T^{S^{aug}}$	Augmented support set from a task

TABLE OF CONTENTS

1 INTRODUCTION

Having been more than a decade since the creation of smartphones, smart wearable devices contribute to our lives through their increasing computing resources and sensing capabilities. Usually empowered by deep learning, mobile sensing has been developed for several applications, which were not possible in the last decade (Gong *et al.*, 2022). Some examples from the literature include Human Activity Recognition (HAR) (Gong *et al.*, 2022; Challa *et al.*, 2022; Vettoruzzo *et al.*, 2025), device-free authentication (Chauhan *et al.*, 2018), sign language recognition (Dai *et al.*, 2017), and prediction of depressive states (Mehrotra; Musolesi, 2018), among others.

Human Activity Recognition (HAR) has become a significant research area for human behavior analysis. Its application has broadened beyond simply recognizing which basic activity a user is performing (walking, running, jumping, etc.), including indoor localization (Xu *et al.*, 2016), smart hospitals (Sánchez *et al.*, 2008), smart homes (Bouchabou *et al.*, 2021), and healthcare (Woznowski *et al.*, 2016). Usually, they rely on deep learning-based methods, which have been demonstrated to be a successful strategy for pattern recognition over sensor time series (Dang *et al.*, 2020).

Despite the recent advances in HAR, there are still significant challenges to overcome. One of the challenges is to have a system capable of recognizing a wide range of different activities (Vettoruzzo *et al.*, 2025). Additionally, due to the complexity of the HAR data, the datasets may be highly imbalanced, scarce in annotated samples, and contain diverse activity styles among users (Alawneh *et al.*, 2021). Moreover, not only can different people perform the same activity in different ways, but also each smart device may have its own particularity on the signal collected (frequency, sensitivity, axis orientations, etc.).

Taking HAR with smartwatches as an example, users may use the watch on different sides (left, right); they can have different walking speeds or strides; they may have some movement particularity due to some activity quirk, like a boxer fighter who flinches his right fist before punching with the left one. The combination of these particularities and the different smart device brands and models available on the market may generate dissimilar data for the same activity being performed. These *individual conditions* are the main cause of performance degradation when trying to deploy HAR applications on smart devices (Gong *et al.*, 2022).

One naive approach to overcome this problem is to create individual models for each user and device, collect sufficient data, and manually label it. Modern deep learning models,

however, require a massive amount of data to train them without overfitting the model to the training data (Huang *et al.*, 2017). This has an expensive cost and tedious user effort for the labeling process, which may be impractical for most applications.

Data augmentation has emerged as a technique to address unbalanced and annotation-scarce datasets. Despite its widespread use for image and language data, this technique presents distinct challenges in the context of time series data (Wen *et al.*, 2020). The time series data are usually high-dimensional complex data, since they are a collection of ordered multi-dimensional points, characterized by intricate and, sometimes, inexplicable patterns (Vettoruzzo *et al.*, 2025). Aggressive augmentation techniques risk destroying these vital patterns, decreasing the model's capability to identify them. On the other hand, augmentations that produce signals very similar to the original fail to provide new, useful information for the model training. Therefore, selecting the most adequate data augmentation techniques is an important challenge to overcome.

Another strategy to enable the training of deep learning models using small datasets is *few-shot learning* (Wang *et al.*, 2020). Some of these methods try to use some prior knowledge to improve the model training over the few labeled samples. Being one of these methods, meta-learning tries to ensure model generalization by learning from different tasks. Meta-learning learns *how to learn* characteristic empowers systems to rapidly adapt to new unseen scenarios using only a few samples of data (Rêgo *et al.*, 2022). Using such a strategy allows HAR meta-models to be applied to scenarios where new activities need to be identified.

The goal of this Ph.D. thesis is to develop a method to perform human activity recognition over smart device sensors' time series, allowing it to overcome the limitations mentioned before. To achieve this, the following research questions are raised:

Q1 - Does the use of a meta-learning strategy significantly improve the training efficiency and adaptation performance of deep learning models for Human Activity Recognition when compared to conventional supervised learning approaches?

Q2 - Does the proposed task-based meta-learning strategy improve cross-task generalization in Human Activity Recognition when compared to standard episodic training approaches?

Q3 - Can the proposed method reduce performance degradation caused by inter-user and inter-device variability when compared to existing Human Activity Recognition approaches?

Q4 - Does leveraging public Human Activity Recognition datasets during the meta-training stage improve the model's ability to adapt to new activities, users, and devices in unseen datasets?

This thesis investigates the synergy between task generation strategies and data augmentation methods within a meta-learning framework. Specifically, it evaluates their effectiveness in addressing few-shot HAR challenges across datasets characterized by varying users and devices. The specific objectives are listed below:

O1 - Provide a meta-learning algorithm for training a HAR model that can rapidly adapt to new tasks.

O2 - Create a heuristic that improves the generalization of the individual conditions across different tasks.

O3 - Apply data augmentation techniques to improve the adaptation of the model to new tasks.

O4 - Validate the proposed algorithm by comparing it against HAR solutions using public datasets.

O5 - Validate the trained meta-model by evaluating its performance over new datasets.

The culmination of this research is AETHER (**A**ugmentation and **E**pisodic **T**ask **H**arnessing for **E**fficient **R**ecognition), a meta-learning algorithm that combines tasks generation heuristics and data-augmentation techniques for learning HAR, achieving objectives **01** to **03**. Several experiments were conducted to evaluate AETHER against other literature approaches, achieving the objective **O4**, and to evaluate how the trained meta-model performs on other datasets, which will contain new activities, subjects, and smart devices, achieving the objective **O5**.

The remainder of this document is organized as follows. Chapter 2 presents the theoretical foundation of this thesis; Chapter 3 brings an overview of the literature for the HAR problem. Chapter 4 presents AETHER's heuristics and algorithms. Chapter 5 performs an extensive experimental evaluation and results analysis of the proposed method, comparing it with some models from the literature using public datasets. Chapter 6 concludes this Ph.D. thesis.

2 THEORETICAL FOUNDATION

2.1 Introduction

This chapter presents some of the theoretical foundations relevant to this thesis, focusing on Human Activity Recognition (HAR) and meta-learning as the central concepts guiding the proposed approach.

2.2 Human Activity Recognition (HAR)

Sensor technology has achieved great developments in computational power, size, accuracy, and manufacturing cost (Liu *et al.*, 2016). Such advancements give the opportunity for a great variety of sensors to be integrated into several portable devices (smartphones, smartwatches, and other wearable devices). Using these sensor data, machines can identify which activities are being performed by humans. Not only can these activities be identified by using wearable sensor data, but also by using video and image data. Dang *et al.* (2020) provides a very comprehensive survey on sensor-based and video-based HAR. For this thesis, we focus only on sensor-based scenarios.

Dang *et al.* (2020) provides a taxonomy to divide the human activity into 5 levels:

1. **Gestures:** simple movements to convey an idea or meaning, like hand-waving, head shaking, and thumbs up.
2. **Action:** simple activities that involve several gestures, like running, walking, and jumping.
3. **Human-object interaction:** where the human performs the activity by interacting with some object, like eating, riding a bike, writing, and typing.
4. **Human-Human interaction:** where two humans interact with each other to perform the activity, like wrestling, hugging, and shaking hands.
5. **Group activities:** performed by more than two people and may involve, or not, object interactions, like a soccer match, a theater play, and choreography.

Assuming a person performs a set of n distinct activities A, denoted as $A = A_1, A_2, ..., A_n$, and is carrying (or wearing) some device that contains a set of sensors collecting some features $S \in R^N$ from the observed sensors over time, a time series $\{S_1, S_2, ..., S_k\}$ of k points is collected. For instance, by monitoring a triaxial accelerometer and gyroscope, each point S_i will be a vector of length 6 ($S_i \in R^6$). For each point, an activity can be assigned to it, often generating datasets

structured like $D = \{(S_1, a_1), (S_2, a_1), ..., (S_k, a_k)\}$, where each of the k points has the activity that was being performed when the data was generated.

Suppose a person is using a wearable triaxial accelerometer device ($S_i \in R^3$), and for each $S_i = (x, y, z)$, y is the vertical axis. The person will perform a jump, and then a squash. The time series would look like $\{$ $([0,0,0], jump)$, $([0,1,0], jump)$, $([0,0,0], jump)$, $([0,0,0], squash)$, $([0,-1,0], squash)$, $([0,1,0], squash)$, $([0,0,0], squash)$ $\}$

Usually, the public datasets are structured as such, providing the sensors collected features and the activity being performed. It is also common to have the timestamp of the collection, which is important to analyze the sampling frequency of the sensors. Since different devices sample data at different frequencies, it is important to consider resampling the data when comparing model performance on different datasets. For instance, suppose a dataset where the features were sampled at 20Hz. If we split the time series into 120 points subsequences, we would have a representation of 6 seconds of activities. Furthermore, another dataset sampled at 200Hz, the same 120 points would represent only 0.6 seconds. The patterns present on these subsequences could be very different for activities with complex and long patterns. Therefore, it is important to model the problem based not only on the number of points to observe on the time series, but as well as the time window they represent. Also, to perform a fair evaluation across different datasets, one may perform a resample of the time series into a common frequency, allowing all the experiments to use the same time series length and representing the same time window.

2.3 Meta-Learning

The literature contains several works based on deep learning networks for a diverse set of tasks, like natural language processing (Bharadiya, 2023), computer vision (Liu; Jin, 2023), and, of course, HAR (Jobanputra *et al.*, 2019). These networks are capable of identifying hidden and complex patterns, assuming they have enough data, time, and computational resources (Huisman *et al.*, 2021).

Data availability became an issue for deep learning approaches: although there are several public and huge datasets available on the internet, we lack data for very specific domains. Furthermore, some domains may present some evolution, like a new type of disease, human activities, sleep patterns, etc. To overcome that obstacle, techniques based on few-shot learning started to be studied (Wang *et al.*, 2020), where the learning process try to generalize the patterns

over just a few data-samples. Allied to these techniques, are the meta-learning algorithms.

Meta-learning is one of the pillars of this thesis, so it is important to understand the fundamentals around it. This section is based on our previous work (Rêgo *et al.*, 2022), which provides a comprehensive overview of meta-learning, presenting some of the algorithms. For this thesis, we focus on explaining meta-learning for supervised problems. In the following sections, we are going to give an overview of it (Subsection 2.3.1), explaining how to perform training and tests using these algorithms (Subsection 2.3.2), and present the types of algorithms that exists at the literature (Subsection 2.3.3), giving more focus on the one used at this thesis (Subsection 2.3.3.1).

2.3.1 What is meta-learning?

Meta-learning models were created to mimic the human mind's learning ability for different tasks. The knowledge of a human is a result of several tasks that he/she developed through their life. This may result in an ability to adapt to new tasks rapidly, easing the learning process.

In **supervised learning**, the goal is to obtain a function f_θ which maps a set of observations $\mathbf{X}$ to a known set of outputs Y. Here, θ are the model parameters (the coefficients of a Linear Regression, the correlations of a Gaussian Naive-Bayes, or the weights of a neural network, for example), which will determine how the function f will behave. These parameters are learned by applying some training algorithm over a dataset $D = \{\mathbf{X}, Y\}$, minimizing some loss function L applied over D. In short, we aim to find

$$\theta' = \min_{\theta} L_D(\theta). \tag{2.1}$$

The learned parameters θ' are specific to the data observed on D. Therefore, it may not generalize very well to examples outside of the dataset. A popular strategy to evaluate the generalization of the model is to apply a holdout cross-validation (Hastie *et al.*, 2009), where the dataset D is divided into pairs of two subsets D_{train} and D_{test}, where $D_{train} \cup D_{test} = D$ and $D_{train} \cap D_{test} = \emptyset$, where D_{train} is used to calculate θ' and D_{test} is used to calculate the metric to evaluate the model performance (accuracy, f1-score, mean-squared-error, etc.).

Finding a global parameter θ' which is optimal to all the data is, usually, unfeasible. However, it can be approximated, using some pre-defined meta-knowledge *omega* (Hospedales *et al.*, 2021), which may include components like an initial value for θ, and a best optimizer and

learning rate. Therefore, we could use a training procedure g with this meta-knowledge ω to approximate

$$\theta' \approx g_{\omega}(D, L_D). \tag{2.2}$$

In contrast, the supervised meta-learning process does not assume that the meta-knowledge ω is defined. Nevertheless, the goal is to find the best ω which allows f_θ (usually referred to as the base-learner inside the context of meta-learning) to approximate the best value for θ as quickly as possible. Furthermore, while supervised learning uses one data set D through the learning process, supervised meta-learning uses a set of different data sets, which are called *tasks*. Learning the meta-knowledge ω is to observe how f_θ is adapted for each task T. Therefore, the model is learning to learn.

More formally, given a probability distribution of tasks $p(T)$, we want to find an optimal meta-knowledge as presented by Equation 2.3 (Huisman *et al.*, 2021). Since $p(T)$ is a probability distribution, the notation $\mathbb{E}_{T_j \sim p(T)}$ represents the expectation of a random task T_j sampled from the distribution.

$$\omega' = \min_{\omega} \underbrace{\mathbb{E}_{T_j \sim p(T)}}_{outer-learning} \underbrace{[L_{T_j}(g_{\omega}(T_j, L_{T_j})]}_{inner-learning} \tag{2.3}$$

Here, the inner-learning concerns on task-specific patterns, while the outer-learning concerns on multiple-tasks. This makes meta-learning a little easier to understand: we learn some ω that allows the quick learning of new tasks T_j at the inner-level. Hence, we have learned how to learn.

This approach is very useful to generate a meta-knowledge that can leverage knowledge from one task to another. Making a parallel with the way that humans learn, a person who can play an acoustic guitar may learn fast how to play an electric one, or a bass; a person who knows how to program with Java may learn fast how to program with Scala. However, this notion of *transferring* some knowledge from one task to another may raise the question of *why not use transfer learning?*

As the name suggests, transfer learning is to transfer learned knowledge from one task to another (Pan; Yang, 2010). It is proven in the literature that, when the new task contains data that follows a different feature distribution, the learned knowledge may not be useful

(Iqbal *et al.*, 2018). Still, it may be confusing what makes meta-learning different from transfer learning. A key property is that meta-learning has this *meta-objective* of optimizing the training performance across different tasks. Meanwhile, transfer learning just does a *pre-train* and then applies some *fine-tuning* over a new task.

Another learning strategy that is similar to meta-learning is *multi-task learning*. The idea is to train a model that will be able to perform well on different tasks from a fixed set (Hospedales *et al.*, 2021). In contrast, meta-learning aims to find a model that can learn unseen tasks quickly.

2.3.2 Meta-training and testing

Every meta-learning algorithm has particularities in how the meta-knowledge ω is defined and optimized. However, the learning process follows three main steps:

- At the **internal training**, the meta-knowledge ω is used to help find the best θ which minimizes the loss function over the task T_j, using part of its data;
- Once θ is found, we do the **external training**, which will use θ to calculate the loss function for the unseen data from T_j. This is going to measure how well the model is generalized. This loss is then used to optimize the meta-knowledge ω;
- At last, when the meta-knowledge ω is optimized, we perform the **meta-test** stage. Here, another internal training will occur for new tasks. However, the value of ω is not updated. This will validate if the meta-knowledge is providing an easier adaptation for new tasks.

Coming from a supervised learning environment, the **meta-test** may seem confusing. One may ask *how can this test be valid if we are training the model using the test task data?*. The answer to this question relies on the *meta-setup*.

The meta-learning process iterates on a set of different tasks. Usually, these tasks are disjoint, meaning that all tasks have a unique set of target classes, where $T_i \in p(T), T_j \in p(T), T_i \cap T_j = \emptyset$. This set of different tasks helps the meta-knowledge to improve the generalization. However, it is important to note that all the data inside each task must come from the same type of source (images, matrices, accelerometer time series, word embeddings, etc.).

Each task T_j is divided into two subsets: a support set T_j^S, which is used to perform the **internal training**, and a query set T_j^Q, which is used at the **external-training** to compute the loss and update the meta-knowledge ω and, at the **meta-test**, to calculate the evaluation metric of that task adaptation. So, although there is a training phase inside the meta-test stage,

the training uses T_j^S, while the metrics are computed over T_j^Q. This is similar to evaluate several small supervised learning tasks, but since we are also evaluating the generalization capability of ω, we need to perform the training phase also.

Probably the most common strategy to structure a task T_j is to use the *N-way k-shot* setup (Rêgo *et al.*, 2022). All tasks $T_j \in p(T)$ will have N distinct classes and a support set T_j^S, a query set T_j^Q with k observations for each class. Assuming $N = 3$ and $k = 2$, there will be a total of 12 observations (6 for support and 6 for query). Figure 1 presents an example of a 3-way 2-shot task generation.

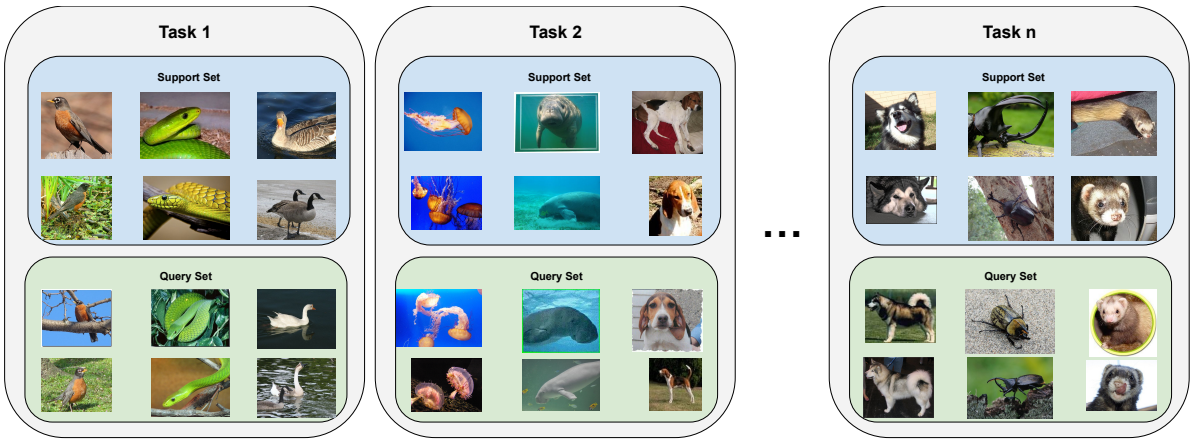


Figure 1 – Example of a 3-way 2-shot set of tasks with images from ImageNet (Fei-Fei *et al.*, 2009).

For some scenarios like *image classification*, there are several datasets with lots of classes (like *ImageNet* (Fei-Fei *et al.*, 2009) and *Omniglot* (Lake *et al.*, 2011)). These datasets allow to generate tasks that are totally disjoint (given any two tasks T_j and T_i, there is no intersection between the N classes present at them), which is great to improve the generalization across the meta-training. However, other data domains may not have such a variety of data (like sensors time series), so it is possible to have some class intersection between the tasks.

2.3.3 Algorithms

So far, we discussed what is meta-learning; how it is different from the traditional supervised-learning, transfer learning, and multi task learning; which are the steps needed to perform the meta-learning process, and how the tasks can be structured, it is time to explain about the algorithms that follows the meta-learning strategy.

Vinyals (2017) proposes a taxonomy for the meta-learning algorithms, which divides the them into three groups:

- **Metric-based:** their objective is to model the meta-knowledge ω as a good feature space, which can be used by various tasks. New tasks can be learned by comparing new inputs to already known inputs (the data in the support set) in the ω learned space. Using similarity metrics, it is possible to identify which already observed input(s) are more similar to the new one. The Prototypical Networks, proposed by Snell *et al.* (2017), is not only a metric-based meta-learning algorithm but also one of the state-of-the-art algorithms used for few-shot learning. The algorithm aims to identify a feature space ω that allows new tasks to be solved using the Nearest Neighbors algorithm.
- **Model-based:** also known as *black box*, these approaches train an internal neural network using the support data and an initial set of meta-parameters. This *black box* can have its model structure changed during the training, allowing a better representation for the learned tasks. These approaches significantly reduce the explainability of the model (Rêgo *et al.*, 2022). One of the advantages of using these methods is internal flexibility due to the model structure change. However, model-based methods are usually surpassed by metric-based methods, computationally more expensive for large datasets, and have lower generalization capability when dealing with more dissimilar tasks (like a task for recognizing animals and other to recognize numerical digits), if compared to optimization-based methods (Huisman *et al.*, 2021). One of the most famous model-based methods is the Simple neural attentive meta-learner (SNAIL), proposed by Mishra *et al.* (2017), which brings a model architecture that combines convolutions, allowing a "high bandwidth" of memory, and attention mechanisms to pinpoint specific experiences.
- **Optimization-based:** these techniques use a different perspective than the previous two. Their goal is to optimize the learning time. The meta-training is modeled like a bi-level optimization problem. The internal training will find the best θ (base-learner) for that task. Then, at the outer training, the meta-knowledge ω will be updated to optimize the performance across tasks. One of the most famous algorithms is Model Agnostic Meta-Learning (MAML), proposed by Finn *et al.* (2017). It will be detailed in Subsection 2.3.3.1.

This thesis focuses on an optimization-based approach to find a meta-knowledge to maximize the generalization across tasks and rapidly adapt the base-learner to new tasks. The MAML algorithm was chosen to be implemented due to its simplicity and the proven efficiency in some state-of-the-art works (Gong *et al.*, 2022; Vettoruzzo *et al.*, 2025). It will be detailed

in the following subsection. For a deeper overview of other algorithms, we suggest the survey written by Huisman *et al.* (2021).

2.3.3.1 Model Agnostic Meta-Learning

The main objective of this algorithm is to rapidly adapt a model for a new task using only a few samples and (internal) training epochs. The meta-knowledge ω will be the weights of a neural network g_ω, which will serve as an input for the base-learner neural network f_θ. Therefore, given a set of samples X, their predictions can be calculated as $f_\theta(g_\omega(X))$. As suggested by its name, the model is agnostic to the architecture chosen to represent the meta-knowledge function g and the base-learner function f.

Finn *et al.* (2017) models the meta-learning process into a bi-level optimization problem. Since it is model-agnostic, the original algorithm does not distinguish what is the meta-model and what is the base learner, using only a model with coefficients θ. The first optimization is for the inner-learning stage, where, given the initial θ, we want to find, for each task T_j, a θ' to minimize the loss function L over its support set T_j^S. By applying the inner-training to these tasks, using the same initial θ, we can measure its efficiency by analyzing how well θ' was adapted. So, we can optimize θ by applying the learned model, for each task, to the query set T_j^Q and calculate its loss and applying some gradient-based optimization technique.

Algorithm 1 shows the pseudo-code described by Finn *et al.* (2017). It provides a very abstract of meta-training, not differentiating what is the meta-model or the task-specific model. For this reason, we provide a rewriting for the bi-level optimization, which decomposes the agnostic model θ into the meta-model ω and the actual base-learner θ. The inner-learning stage optimization uses the same initial meta-model ω and base-learner θ to find, for each task T_j, the updated base-learner (θ') and meta-model (ω') that minimize the loss function L over its support set T_j^S. Then, we can measure the efficiency of ω by analyzing how well ω' and θ' were adapted. So, we optimize ω by applying the learned model, for each task, to the query set T_j^Q and calculate its loss, and applying some gradient-based optimization technique. By explicitly describing where and how ω and θ are used in the algorithm, it becomes clear how the meta-model is present on new tasks and how should we properly update, facilitating the algorithm implementation.

Algorithm 2 presents this rewriting of MAML. It is important to emphasize that this rewriting is not a proposal for a new algorithm, but just an artifact to help better understand

Algorithm 1: Model Agnostic-Meta-Learning from Finn *et al.* (2017)

Input: $p(\mathscr{T})$: distribution over tasks
α, β: step size hyperparameters
randomly initialize θ
while *not done* **do**
  Sample batch of tasks $\mathscr{T}_i \sim p(\mathscr{T})$
  for *all* $\mathscr{T}_i$ **do**
    Evaluate $\nabla_\theta \mathscr{L}_{\mathscr{T}_i}(f_\theta)$ with respect to K examples
    Compute adapted parameters with gradient descent: $\theta_i' = \theta - \alpha \nabla_\theta \mathscr{L}_{\mathscr{T}_i}(f_\theta)$
  end
  Update $\theta \leftarrow \theta - \beta \nabla_\theta \sum_{\mathscr{T}_i \sim p(\mathscr{T})} \mathscr{L}_{\mathscr{T}_i}(f_{\theta_i'})$
end

how one could implement MAML with more detailed steps. The algorithm requires the base-learner f_θ, which will be responsible to learn tasks-specific patterns; a meta-model g_ω, which will learn the meta-knowledge that will assists on faster tasks adaptation; The learning rates α and β to update the task-specific model and the meta-model across tasks, respectively; a task distribution $p(T)$, which we can sample random tasks to train; a number of epochs to perform the outer-training (*oe*) and inner-training (*ie*); and a loss function *L*.

The algorithm will perform the meta-training over *oe* epochs (line 1). For each epoch, it will sample a batch B of random tasks (line 2). Usually, these tasks are generated randomly. However, one could propose some heuristics for a better construction of the tasks batch, like (Gong *et al.*, 2022). We create a list to store the loss calculated after the training of each task, using its query set, so we can compare the impact of ω on learning new tasks (line 3). For each task (lines 5 to 15) we get the support set T_j^S and the query set T_j^Q (line 5) and store a copy of θ and ω (lines 6 and 7). This step is important because we ensure that all tasks will use the same weights for the meta-model and the base-learner, which can be updated during the inner-training stage (lines 8 to 11). Once the inner-training epochs *ie* ran, we use the learned weights θ' and ω' to calculate the loss for the query set (line 13). Once all the tasks from the batch are trained, we can use the query set losses to optimize the initial value for ω (line 15). That way, we are updating it using a loss that evaluates its performance across tasks.

The algorithm's implementation is straightforward and very flexible: we can choose any model architecture for both meta-model and base-learner; we can give different learning rates for both inner- and outer-learning stages; any gradient method can be used as an optimizer (we can even use different methods for inner and outer); we can easily adapt the algorithm to analyze multiple batches per meta-epoch.

Algorithm 2: Model Agnostic-Meta-Learning rewriting

Input: the base learner f and its weights θ
the learning rate for the base learner α
the meta-model g and its weights ω
the learning rate for the meta-model β
a task distribution $p(T)$
a number of epochs for outer-training *oe*
a number of epochs for inner-training *ie*
a loss function L

for $metaEpoch$ ***from*** 1 ***to*** oe ***by*** 1 **do**
  $B \leftarrow$ Sample a batch of tasks from $p(T)$
  $tasksLoss \leftarrow \emptyset$
  for $T_j \in B$ **do**
    $(T_j^S, T_j^Q) \leftarrow$ Split support and query set from T_j
    $\theta' \leftarrow \theta$
    $\omega' \leftarrow \omega$
    for $taskEpoch$ ***from*** 1 ***to*** ie ***by*** 1 **do**
      $loss \leftarrow L_{T_j^S}(g_{\omega'}, f_{\theta'})$
      $\theta' \leftarrow$ Optimize θ', using $loss$ and learning rate α
      $\omega' \leftarrow$ Optimize ω', using $loss$ and learning rate α
    end
    $tasksLoss.add(L_{T_j^Q}(g_{\omega'}, f_{\theta'}))$
  end
  $\omega \leftarrow$ Optimizes ω using $tasksLoss$ and learning rate β
end

2.4 Conclusion

This chapter presented the theoretical foundation of this thesis. It presented what is HAR, explaining different levels of activities type, and how the datasets are usually structured, presenting some of their nuances. The chapter also explained how meta-learning can be performed, making a parallel with the traditional supervised learning approach. It also discusses briefly the differences between meta-learning and other similar approaches. The overview of the meta-learning process was presented, as well as the task structure setup and its stages. Some of the most known meta-learning algorithms were presented, with more focus on MAML, which is the one used in this thesis.

3 RELATED WORKS

3.1 Introduction

The study of HAR became a trend in research topics due to the availability of several low-cost sensors (such as accelerometers and gyroscopes), time series and open video data, advances in artificial intelligence, and Internet of Things (IoT). The research field consists of identifying, as the name suggests, activities that are performed by humans like walking, running, sitting, standing, etc.; by analyzing time series data from sensors or video image frames. Among the several HAR applications, we can cite: medical diagnosis, keeping track of elderly people, crime rate control, smart home monitoring, and military actions recognition (Jobanputra *et al.*, 2019).

Examining HAR surveys, such as those by Dang *et al.* (2020) and Wang *et al.* (2019), it is notable how HAR has been extensively explored by the academy. A more recent survey written by Saha *et al.* (2024) explored the use of meta-heuristics to solve the HAR problem. However, no meta-learning works were presented. The related works of this thesis were gathered by applying a keyword search over the Google Scholar[1] platform and SciSpace AI[2] platform, using the keywords ‘Meta-Learning‘ and ‘Human Activity Recognition‘, filtering by publications from the last 10 years (2015-2025). Only six papers presented interesting contributions for sensor-based HAR with meta-learning. We also gathered some works with a high number of citations by replacing the ‘Meta-Learning‘ keyword with ‘Deep Learning‘ and using only ‘Human Activity Recognition‘ and looking for works that use traditional machine learning approaches.

This chapter presents a literature overview of HAR solutions. Section 3.2 starts by presenting traditional learning works about HAR, followed by Section 3.3, which will present works based on deep learning, and Section 3.4, which will bring the application of meta-learning for the HAR problem. Then, Section 3.5 compares the state-of-the-art with the contributions proposed with this thesis. Finally, Section 3.6 concludes this chapter.

3.2 Traditional Machine Learning for HAR

Hardware constant evolution has turned the use of deep learning models more accessible and easier to train. It is common to use such models to learn several tasks (Dong *et*

[1] https://scholar.google.com/
[2] https://scispace.com/

al., 2021). However, classic machine learning algorithms, such as K-Nearest-Neighbors (KNN), decision trees, and Support Vector Machine (SVM), remain relevant in modern research.

Mohsen *et al.* (2021) used the KNN algorithm, and some variations, to develop an accurate model for HAR, achieving an F1-score of 90%. Meanwhile, Sani *et al.* (2017) uses deep learning methods to extract features over sensor time series, which will be an input to a KNN model to perform HAR, achieving 95% f1-score.

Maswadi *et al.* (2021) used decision trees and naive Bayes algorithms to perform HAR using accelerometer data. They achieved 89.5% and 99.9% accuracy for naive Bayes and decision trees, respectively. Balli *et al.* (2019) approached the HAR problem by using a combination of principal components analysis and random forests, comparing it with KNN and SVM, obtaining over 90% accuracy for all the methods.

Hossain *et al.* (2025) made a comparison using classical machine learning and deep learning models, using 5 different datasets benchmarks. Table 1 presents the metrics calculated for the models over the first version of the WISDM dataset (Kwapisz *et al.*, 2011), which contains 6 unbalanced classes (Downstairs - 38.93%, Jogging - 30.23%, Sitting - 11.42%, Standing - 9.33%, Upstairs - 5.58%, Walking - 4.51%).

Table 1 – Models performance over WISDM dataset, adapted from (Hossain *et al.*, 2025). The traditional models are on the upper section, while deep learning models are on the bottom section.

Model Name	**Precision**	**Recall**	**F1-Score**	**Accuracy**
Decision Tree	0.86	0.85	0.85	0.86
Random Forest	0.92	0.92	0.92	0.92
Logistic Regression	0.95	0.95	0.95	0.95
SVM	0.96	0.96	0.96	0.96
KNN	0.90	0.89	0.89	0.90
Convolution Neural Network (CNN)	0.95	0.95	0.95	0.95
Recurrent Neural Network (RNN)	0.82	0.78	0.78	0.79
Long Short Term Memory (LSTM)	0.83	0.82	0.82	0.82
Bidirectional LSTM (Bi-LSTM)	0.85	0.84	0.84	0.85
Gated Recurrent Units (GRU)	0.83	0.82	0.82	0.82

The results in Table 1 show that traditional models are very competitive with deep learning models. SVM was able to outperform most of the deep learning models, and tie with CNN. These results reinforce that the traditional models deserve to be evaluated as well with modern approaches.

3.3 Deep Learning for HAR

Deep learning tries to surpass the classic learning algorithms by imitating the network of neurons in the human brain, allowing the computer to deal with complicated problems and even reach human-level performance for some of them (Dang *et al.*, 2020). Through the years, many types of neuron models have been introduced, such as Multi Layer Perceptrons (MLP), CNN, RNN, and others.

Ronao e Cho (2016) used a CNN to classify 1D time series signals collected from smartphones and achieved 94.79% accuracy over a dataset collected with 30 volunteers. Hughes e Correll (2018) used CNN to improve HAR adding the information of the sensor location (arms, body, and legs), achieving over 90% accuracy.

RNN-based models (like LSTM and GRU) are in the most common approaches for performing HAR using sensors time series (Gu *et al.*, 2021). Murad e Pyun (2017) used a deep neural network of LSTM to recognize activities from several public datasets, achieving over 90% accuracy through the experiments. Inoue *et al.* (2018) also uses a deep LSTM network, achieving 95% accuracy.

The survey written by Gu *et al.* (2021) provides a brief and comprehensive explanation of the fundamentals for the deep learning models most used in the literature, as well as an overview of several works from 2011 to 2020. There, we can notice how versatile these models are. By making small changes to the model architecture (like adding a new layer, using other activation functions, etc.), we can improve the results. Kaseris *et al.* (2024) brings a literature review up to 2023 of deep learning methods for HAR.

Another approach to improve the model performance is to combine different types of neuron models. Challa *et al.* (2022) proposes a multi-branch hybrid CNN with Bi-LSTM model to perform HAR over accelerometers and gyroscope data. The input time series is used to feed three parallel branches of a CNN network, each with different filter sizes (3, 7, and 11). The idea is for these networks to capture various temporal and local dependencies. Their output is then concatenated and used to feed a 2-layer deep Bi-LSTM network, which will allow the model to capture long-term dependencies over the data. Then, the output is fed into a MLP layer to perform the classification. The model is evaluated with some datasets, being WISDM one of them, where it achieved 96% accuracy.

The transformer architecture, proposed by Vaswani *et al.* (2017), brings an innovative way to enhance neural-networks commplex pattern detection capability with attention layers.

Patwardhan *et al.* (2023) provides a survey showing their efficiency for natural language processing. However, for the HAR scenario, there are not many works applying the use of transformers for sensor-based data. Leite *et al.* (2024) presented some initial works which apply transformers for HAR. Furthermore, one great challenge for this model to be applied to HAR scenarios is that, since the models run on edge devices, there are considerable resource constraints. Lattanzi *et al.* (2024) presents a study which proves that they are not a viable solution to use with micro devices since CNN or LSTM models have better performances when they are built to fit inside these micro architectures.

The infinity of possible deep learning architectures that can be built gives margin to endless studies and experimentation. However, all deep learning approaches require a large amount of data to train. Also, training these very complex and combined neurons with these large datasets require lots of computational resources, which may make unfeasible their applications to scenarios where the we have just started to collect the data, or even when the tasks keep having slight variations that can invalidate the use of the trained model, like new activities to be recognized, new versions of the sensors, etc.

3.4 Meta-learning for HAR

Deep learning algorithms are highly dependent on large datasets to learn complex patterns (Dang *et al.*, 2020). To overcome this challenge, meta-learning has been very explored at the literature (Huisman *et al.*, 2021). The key idea is to improve the model learning ability by learning how to learn. A more detailed explanation is provided in Chapter 2.

Meta-learning has been applied for HAR for some years now. Li *et al.* (2021) applies the meta-training intuition to improve a federated learning process by creating a federated representation learning framework, which will be generated by meta-learning processes. Wijekoon e Wiratunga (2020) proposed PersonalizedMAML, which uses the MAML(Finn *et al.*, 2017) meta-learning algorithm to produce a meta-model capable of identifying activities for a specific user using only a few labeled samples, achieving up to 97% f1-score. During the meta-model training, the tasks distribution $p(T)$ is obtained by random sampling observations from the same user, allowing the meta-model to help identify user-specific patterns to improve the activity recognition. They performed experiments over public datasets and achieved up to 97% accuracy for some experiments. However, some datasets provided data from different sensors (wrist, chest, ankle, etc.). They handle these different sensors by performing the experiment for each dataset

and sensor. For instance, no ankle data is present in the chest experiment, nor is the meta-model trained from the chest applied to wrist data.

Metasense, proposed by Gong *et al.* (2022), brings an improvement on Personalized-MAML, although it is not directly mentioned. They created a more sophisticated heuristic to generate a distribution of tasks ($p(T)$), which improves the meta-model generalization. They also explore the fact that each person may not perform the same activity in the same way twice, which will lead to different signals being generated. Furthermore, different devices may generate different time series due to hardware particularities (sensitivity, axis orientation, etc.). They propose the concept of Individual Condition Dataset (ICD), which represents the data from one specific person using one specific device. By applying the proposed heuristic, Metasense was able to outperform standard deep learning approaches, some state-of-the-art methods, and MAML. The details of how this heuristic works are going to be explained in Chapter 4, Section 4.2. However, it is worth to mention that one of these heuristics is to oblige all the tasks to have the same activities, making it not-applicable to scenarios where we need to learn a model for new activities. Their experiments show that they can achieve at least 67.2% accuracy for 1-shot scenarios.

Although meta-learning facilitates the training process of deep learning models, it does not change the fact that using larger datasets will allow the models to learn more complex patterns. Based on that, Vettoruzzo *et al.* (2025) proposed MADA, which combines meta-learning with data augmentation techniques to increase the size of the support set T_j^S to improve the model's performance. It also proposed a meta-parameter to learn which data augmentation techniques bring more improvement to the learning process. It is evaluated across 9 datasets, being WISDM 2 of them (one for phones and the other for watches). It is important to mention that, although it is a meta-learning approach, it required a few thousand epochs for training the task model. This long training process makes the adaptation of the model to new tasks long and unfeasible for low-resource devices.

Wang e Faruque (2025) proposes TACO, which uses a transformer to generalize the domain of the sensors data for HAR, achieving 94.5% accuracy. Their solution also applies some data augmentation techniques to generate more samples for the transformer to learn. TACO meta-training iterated over 500 epochs and achieved great results over public datasets. The authors do not make clear if all epochs are using the same tasks or how many epochs are necessary to fine-tune the model for new tasks.

Ganesha *et al.* (2024) was proposed to improve meta-learning by combining it with transfer learning. First, the model is trained using open large HAR datasets. Then, the model will be used as the initial meta-knowledge from an optimization-based algorithm, which will perform the meta-learning process. By comparing the meta-learning process with and without the transfer learning model, the latter was able to improve the accuracy of the final model by 5%.

3.5 Works comparison

Given the works presented in the previous sections, there are some key points to be observed for better standing out AETHER differences for the literature:

- **Training time (TT)**. Some of the works (mostly the deep learning ones) require a large number of epochs for training the models, resulting in a long training time period, which may not be adequate for scenarios where we need to train new models several times. As new users start using a device and performing some activities, they don't want their device to take several hours, or even a couple of days, to start identifying when they are performing activities.
- **Allow new activities (NA)**. It is a good feature for a model to allow new activities to be classified. This allows the model to be applied for commercial use on wearable devices.
- **Improved tasks generation (ITG)**. Meta-learning algorithms usually generate random tasks, which may not represent accurate scenarios for the new tasks, affecting their generalization for the context on which it will be applied.
- **Use of data augmentation techniques (DA)**. These techniques can be useful to scale the training dataset size, allowing the model to be trained with smaller sets of data.

Table 2 summarizes these points for the discussed works. Since all the works presented at Sections 3.2 and 3.3 share the same behavior inside their group, we put only one row for each section. However, for the works at Section 3.4, we explain them individually.

The traditional machine learning algorithms usually do not require long training stages, except for really large datasets and very specific model parameters, like Random Forest with thousands of estimators. They do not allow new activities to be learned without the need to train the model again, and, since they are not meta-learning based, they do not require task generation strategies. As for the deep learning models, they require an extensive training period, resulted of thousands of epochs of training over huge datasets. They also do not allow new activities to be learned and do not require task generation strategies.

Table 2 – Overview of the related works, analyzing the key points: Training time (TT), Allow new activities (NA), Improved tasks generation (ITG), and use of data augmentation techniques (DA). At the bottom line, the same key points are analyzed for AETHER.

Work	**TT**	**NA**	**ITG**	**DA**
Section 3.2	Usually fast training	No	Not needed	May use
Section 3.3	Thousands of epochs with lots of data, taking hours or even days	No	Not needed	May use
Section 3.4				
PersonalizedMAML (Wijekoon; Wiratunga, 2020)	100 epochs per task	Can, but choose not to	Yes	No
Metasense (Gong *et al.*, 2022)	5 epochs per task	Can, but choose not to	Yes	No
MADA (Vettoruzzo *et al.*, 2025)	5000 epochs per task	Yes	No	Yes
(Ganesha *et al.*, 2024)	Long pre-training for selecting the starting model	Yes	No	No
AETHER	**5 epochs per task**	**Yes**	**Yes**	**Yes**

The meta-learning approaches usually do not require a long training time, since they are designed to rapidly adapt for tasks with few shots. However, the work proposed by Ganesha *et al.* (2024) has an initial long training period to select the meta-model to train. They can handle learning new activities. However, PersonalizedMAML and Metasense decided not to do it in their experiments, claiming to increase performance. Also, these two works are the only ones that used improved task generation heuristics.

AETHER combines the strengths of all these works to overcome their limitations. By using meta-learning strategies, it can rapidly train a meta-model and adapt it for new tasks. Also, this allows it to handle new tasks that contains new activities to be recognized. AETHER also applies some heuristics for tasks generation and improves it by augmenting the task support set. However, AETHER does not have a meta-parameter to identify which data-augmentations are the best for the meta-learning phase, as MADA does, needing them to be selected through experimentation. This choice was made to enforce a well-designed selection of these data-augmentations, as well as to apply experimentation to choose the best selection.

3.6 Conclusion

This chapter presented an overview of the literature about HAR. It explored how previous researches solved this problem using classical machine learning algorithms to the use of deep learning and meta-learning. It also highlighted some key points of these works and how AETHER is different from what is available so far.

4 AETHER: AUGMENTATION AND EPISODIC TASK HARNESSING FOR EFFICIENT RECOGNITION

4.1 Introduction

This chapter presents AETHER, a novel approach for human activity recognition using sensors time series, applying meta-learning and data augmentation. It consists of using an efficient strategy for building the tasks distribution $p(T)$ for the meta-learning process and augmenting the task support set T_j^S by applying time series transformations over its observations.

Figure 2 presents the AETHER overview. The process starts by separating the ICDs from the dataset. As proposed by Gong *et al.* (2022), each ICD is the set of observations from a unique pair of user and device (i.e., John Doe using a Galaxy Watch 4). The ICDs will be used to generate the tasks used in the meta-training. Each task will have its support data augmented, increasing the number of observations to be used by the inner-learning stage. Then, the meta-learning will be executed, using the MAML algorithm. Finally, the meta-model will be ready to be applied to new tasks.

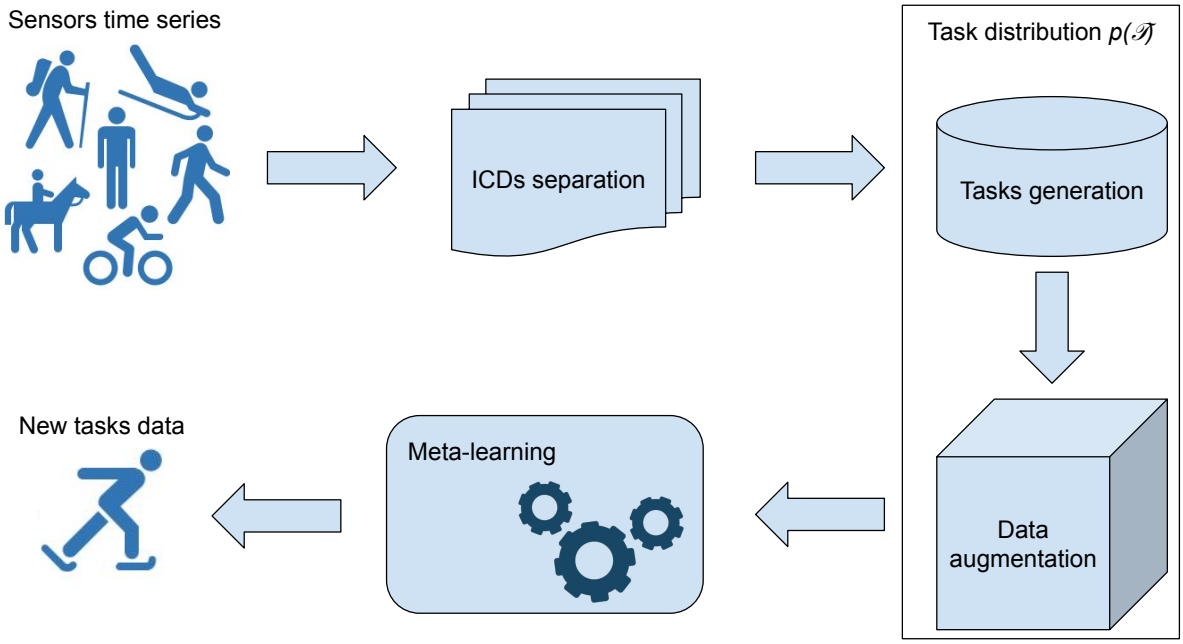


Figure 2 – AETHER overview.

The chapter is structured as follows: Section 4.2 describes how the tasks are generated; Section 4.3 shows the data augmentation process; Section 4.4 presents the algorithm for AETHER; Section 4.5 concludes this chapter by explaining the deep learning model architecture that will represent the meta-knowledge ω.

4.2 Task generation strategies

Usually, meta-learning algorithms perform the meta-training stage by generating a random task distribution $p(T)$ using *N-way k-shot* tasks. A task T_j can be differentiated from another T_i by the classes that are going to be learned. By doing this, the model will try to learn how to adapt to any possible task using that type of data. However, for scenarios where we have a limited data source, creating a large set of different tasks can be a challenge (Gong *et al.*, 2022). Let us take as an example the WISDM dataset used by many works. It contains 6 distinct classes. Suppose we use a *3-way k-shot* task generation, regardless of k, we will only be able to obtain 20 different tasks (combination of 3 items from a set of 6 possible items), which is a small number to generalize real scenarios. By having different combinations of classes, this increased number of configurations allows the meta-learning process to explore more combinations, providing a meta-model with better generalization.

Metasense (Gong *et al.*, 2022) proposes a different strategy for generating $p(T)$, by differentiating the tasks using the ICDs present at T, allowing the number of different tasks to scale with the number of people and devices as well. Their objective is to have a model that can: **(i)** perform well when meeting an unseen ICD, and **(ii)** avoid overfitting for the same ICD and allow it to generalize the patterns in a broader way, making it suitable to handle multiple ICDs at the same time.

To better understand the task generation strategy, let's first assume a set of sensors time series D_{icd}, which contains only data from a specific ICD icd. It contains the observations $(x_i, y_i) \in D_{icd}$, where x_i is the time series and y_i is the activity being performed during x_i. Also, inside D_{icd}, there are only C different classes.

The task generation strategy generates two types of tasks:

- **Per-condition.** Each task T contains observations for the same ICD, that means for any $(x_i, y_i) \in T$, $(x_i, y_i) \in D_{icd}$. With these tasks, the model will be able to achieve **(i)** by learning over tasks of the same individual. In other words, if one observation from this task is from *Jon Doe* using a *Galaxy Watch 4* ICD, all of the observations, regardless of the activitie being performed, will be of this same ICD.
- **Multi-conditioned.** In contrast to **per-condition**, these tasks may have multiple ICDs inside it. The only restriction is that, for each class c inside the task, all observations are from the same ICD. Given any observations $(x_i, y_i) \in T$, $(x_j, y_j) \in T$, if $y_i = y_j$, $(x_i, y_i) \in D_{icd}$ and $(x_j, y_j) \in D_{icd}$. In other words, if one observation o the activity *running*

is from *Jon Doe* using a *Galaxy Watch 4* ICD, all observations of *running* are from the same ICD. However, inside the same task T, we can have the activity *walking* with data from another ICD. Since we have N classes per task, we can have at most N distinct ICDs inside of T. These tasks help to achieve **(ii)**, since they will not allow an overfit over one ICD.

Figure 3 (Gong *et al.*, 2022) illustrates the task generation strategies. Metasense also has a restriction on the tasks generated to be **homogeneous**. This restriction enforces that all tasks share the same set of activities. Given the set of unique classes of a task Y_T, for all $T_i \in p(T)$ and $T_j \in p(T)$, $Y_{T_i} = Y_{T_j}$. For their application scenario, keeping labels consistent for all tasks in $p(T)$ leverages the common knowledge from each class across the tasks' learning. However, this brings some contrast with the Meta-learning strategy.

A key disadvantage of restricting all tasks to have the exact same set of classes is that the model fails to develop true generalization capabilities for novel classes. Meta-learning is specifically designed to learn how to learn, allowing it to quickly adapt to entirely new tasks with unseen classes. If the model only encounters tasks where the class labels are always identical, it learns a specific, non-transferable representation tied to those three categories. Consequently, the model's meta-knowledge becomes over-specialized, and its ability to quickly and effectively adapt to a new task will be significantly impaired. Therefore, AETHER does not apply this restriction, since we want to be able to handle new activities on new tasks.

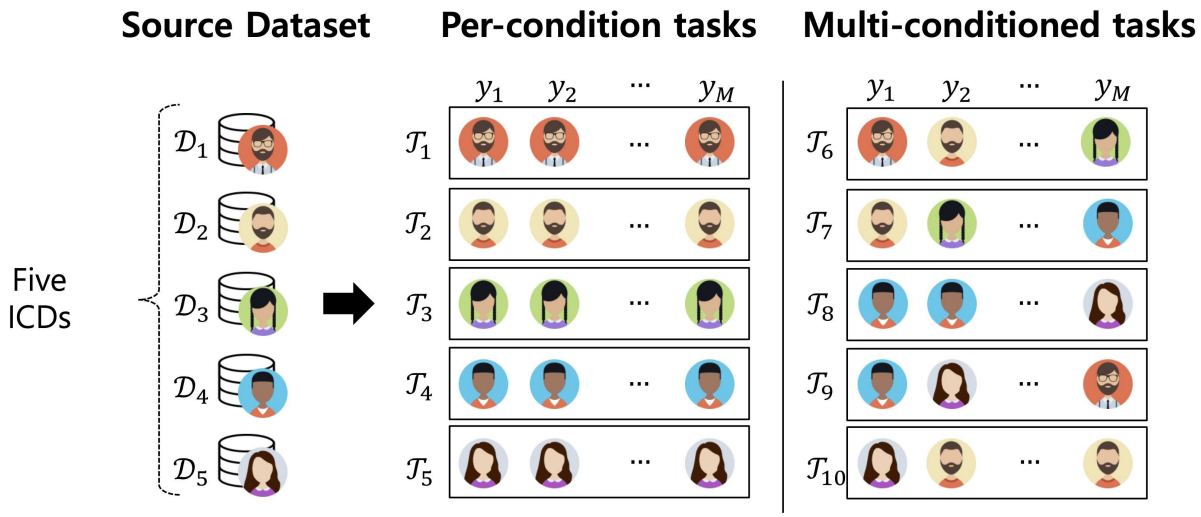


Figure 3 – Tasks generation strategies, extracted from Gong *et al.* (2022).

4.3 Data augmentation process

Using meta-learning empowers learning systems with the capability of acquiring knowledge across different tasks, allowing deep learning models to improve their performance on scenarios with few samples (Vettoruzzo *et al.*, 2025). However, it does not change the fact that these deep learning models would perform better if they had more data to train with.

To improve the learning capability of meta-learning tasks training, AETHER applies data augmentation techniques over the tasks data. Given a task $T_j \in p(T)$ with observations $(x_i, y_i) \in T_j^S$, and a set of data augmentation operations $m \in M$, we can generate a new x_i^m by applying the operation m over x_i. That way, by adding (x_i^m, y_i) to T_j^S we are increasing the support set size by $|M|$ times.

Figure 4 presents the data augmentation techniques used by AETHER. This set of techniques is based on those used by MADA (Vettoruzzo *et al.*, 2025). We opted not to use the same set because we do not have any parameter to learn which ones are the best augmentations, therefore we cannot just use a huge set and let the meta-model identify what is good and what is not. The methods were chosen based on the learned importance of the evaluated data augmentations at Vettoruzzo *et al.* (2025).

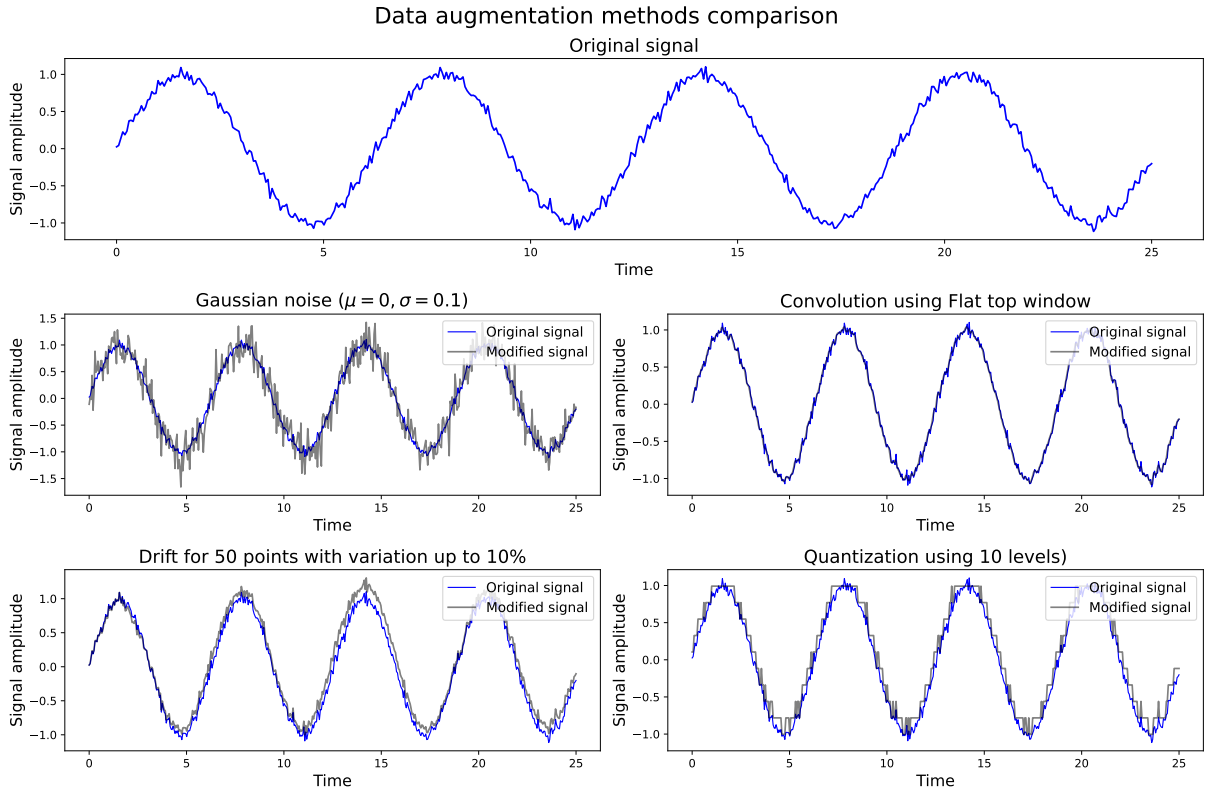


Figure 4 – Comparison of data-augmentation techniques applied to a same time series.

AETHER uses four different data augmentation techniques, which are visible at Figure 4:

- **Random Noise**, which applies a random noise signal following a normal distribution with mean μ and some standard deviation σ. The noise can be added through addition, where each point of the series is added to each point of the noise, or multiplication, where each point of the series is multiplied by the noise.
- **Convolution**, which applies a convolution kernel window over the signal. This window is applied over the signal, providing a *smoother* version of the signal. The convolution can vary on the size of the window and the kernel function used. At the figure, we are using *Flat Top* (D'Antona; Ferrero, 2006).
- **Drift**, which will displace some points of the series by a random value. At the figure, we have 50 points displaced by a random value which will move up or down the original point by at most of 10% of its value.
- **Quantization**, which quantize the values from the signal into a level set. Then, each point of the series will be rounded to the nearest level.

These four data augmentation techniques are used to create the set M. Each technique appears multiple times, varying its parameters. More details about the parameters variation for each modification are presented at Chapter 5.

4.4 Training algorithm

Algorithm 3 presents how AETHER trains the meta-model. It is an adaptation of the MAML algorithm, originally proposed by (Finn *et al.*, 2017), presented in Chapter 2, Section 2.3.3, Subsection 2.3.3.1. The main difference is to add the task distribution generation and the data augmentation.

Since it uses MAML as the meta-learning algorithm, it requires the base-learner f_θ, for learning tasks-specific patterns; the meta-model g_ω, to represent the meta-knowledge; the learning rates α and β to inner- and outer-learning, respectively; the number of tasks NT which will be used by each meta-epoch; a number of epochs to perform the outer-training (oe) and inner-training (ie); and a loss function L.

The algorithm will perform the meta-training over oe epochs (line 1). For each meta-learning epoch, it will create NT sets of per-condition (line 2) and multi-conditioned tasks (line 3). The original MAML algorithm receives $p(T)$ as input, which is usually composed

Algorithm 3: AETHER training algorithm

Input: the base learner f and its weights θ
the learning rate for the base learner α
the meta-model g and its weights ω
the learning rate for the meta-model β
the amount of tasks to be used per meta-learning epoch NT
a number of epochs for outer-training oe
a number of epochs for inner-training ie
a loss function L
a set of M data augmentation operations

1 **for** $metaEpoch$ ***from*** 1 ***to*** oe ***by*** 1 **do**
2 $pcts \leftarrow$ Create NT random per-conditon tasks
3 $mcts \leftarrow$ Create NT random multi-conditoned tasks
4 $p(T) \leftarrow$ Create a distribution of random tasks sampled from $pcts \cup mcts$
5 **for** $B \in$ *Sample NT batch of tasks from* $p(T)$ **do**
6 $tasksLoss \leftarrow \emptyset$
7 **for** $T_j \in B$ **do**
8 $(T_j^S, T_j^Q) \leftarrow$ Split support and query set from T_j
9 $T_j^{S^{aug}} \leftarrow$ Apply the M augmentations and append to the support set
10 $\theta' \leftarrow \theta$
11 $\omega' \leftarrow \omega$
12 **for** $taskEpoch$ ***from*** 1 ***to*** ie ***by*** 1 **do**
13 $loss \leftarrow L_{T_j^{S^{aug}}}(g_{\omega'}, f_{\theta'})$
14 $\theta' \leftarrow$ Optimize θ', using $loss$ and learning rate α
15 $\omega' \leftarrow$ Optimize ω', using $loss$ and learning rate α
16 **end**
17 $tasksLoss.add(L_{T_j^Q}(g_{\omega'}, f_{\theta'}))$
18 **end**
19 $\omega \leftarrow$ Optimizes ω using $tasksLoss$ and learning rate β
20 **end**
21 **end**

of random tasks generated from the source dataset. AETHER creates $p(T)$ by combining the per-condition and multi-conditioned tasks generated and random sampling from them (line 4). That way, each meta-learning epoch will train using NT and each task has the same probability of being a per-conditioned or multi-conditioned. Then, a sample batch B of random tasks (line 5) is taken, and we create a list to store the loss calculated after the training of each task to update the meta-model (line 6). For each task (lines 8 to 19) e get the support set T_j^S and the query set T_j^Q (line 8) and store a copy of θ and ω (lines 10 and 11), ensuring that all tasks will use the same weights for the meta-model and the base-learner. Now comes another difference from MAML original algorithm: line 9 augment the support set using the M data augmentation

operations, increasing its size. The remaining process is the same as MAML: lines 12 to 15 perform the inner-training, using the augmented support set $T_j^{S^{aug}}$; we use the learned weights θ' and ω' to calculate the loss for the query set (line 17); we use the query set losses to optimize the initial value for ω (line 19).

Since the algorithm is an adaptation of MAML, AETHER inherits MAML's flexibilities: we can choose any model architecture for both meta-model and base-learner; we can combine different learning rates strategies and optimizers for both inner- and outer-training. Also, one could use a different distribution for $p(T)$ by generating more samples for one of the tasks types, or even make the algorithm use $\frac{NT}{2}$ tasks of each.

4.5 Deep learning model overview

Chapter 2 showed how flexible meta-learning approaches are regarding the deep learning models that can be used. At this chapter, Section 4.4 reinforced it by showing that AETHER is agnostic to the model architecture used by focusing on the learning optimization across different tasks. This section will describe the meta-model architecture that we use in this thesis. The model the same hybrid CNN-Bi-LSTM network proposed by Challa *et al.* (2022), presented in Figure 5. However, the classification layer will not be part of the meta-model. Instead, it will be the base-learner.

Human activities usually consist of executing repeated movements, so the time series collected from sensors that monitor these activities should contain these repetitive patterns. A CNN network identifies local patterns over the sensors' time series, transforming them as features from the time series (Challa *et al.*, 2022). Meanwhile, the RNN models take the temporal information into consideration, meaning that they can leverage repeated patterns identified over the time series. Therefore, a combination of both networks was proposed.

As we can see in Figure 5, the input data is used to feed three parallel CNN networks (also called branches), which only differ in the size of the convolution filter used by the first layer. The idea to use different filter sizes is to capture different patterns with different sizes. Each branch consists of two convolution layers, a dropout layer, responsible for helping reduce overfitting during training; a pooling layer, to perform dimensionality reduction, and a flatten layer to keep the features as a vector. Once all three branches process the input data, they are concatenated as one single sequence.

Traditional RNN cannot capture dependencies on long series due to vanishing

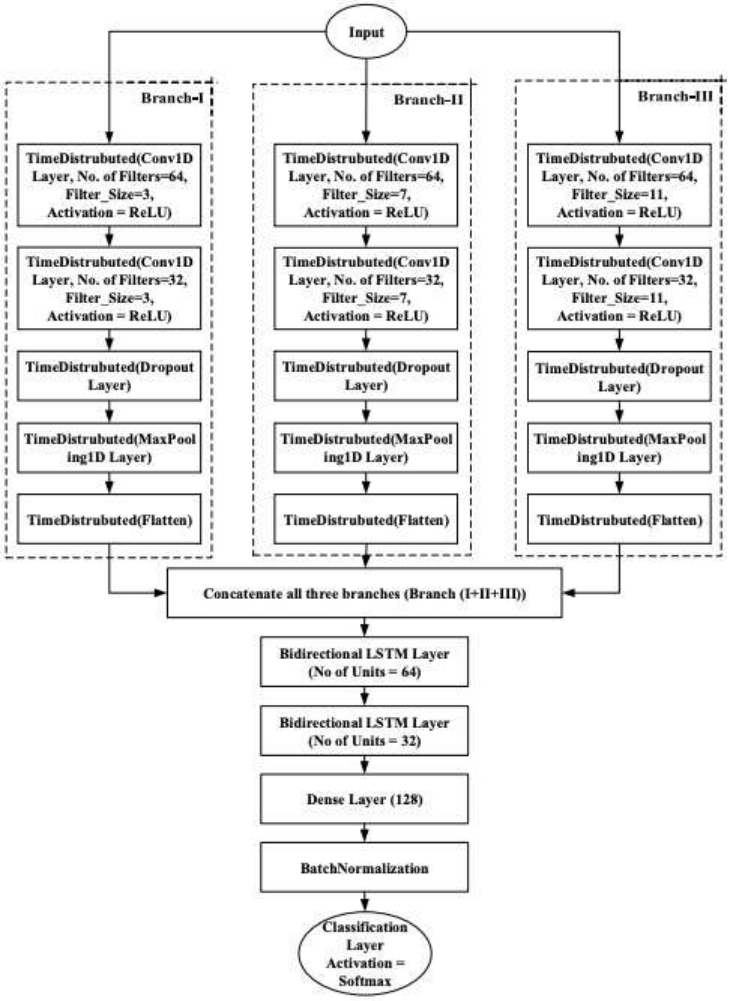


Figure 5 – Hybrid CNN-BiLSTM model architecture for HAR proposed Challa *et al.* (2022).

gradient problems (Bengio *et al.*, 1994). LSTM is a variation of the RNN which overcomes this limitation and has proven a powerful architecture to sequential data classification (Al-Selwi *et al.*, 2024). The Bi-LSTM improves the original LSTM architecture by simultaneously training two LSTM, one with the forward direction of the time series and one with the backward, allowing the network to make better predictions (Challa *et al.*, 2022). For that reason, the concatenated features from the CNN branches as used to train two Bi-LSTM layers, which will identify the temporal patterns over the local patterns identified by the branches.

Once the Bi-LSTM processes all the features, its output is then passed through a simple neural network with 128 neurons, which will project the identified local and temporal patterns into a 128-dimensional space, which will be normalized, turning the batch of features with zero-mean and unit standard deviation. Finally, a classification layer will use these normalized 128-dimensional features to perform the classification.

Challa *et al.* (2022) performed an extensive experimentation using this architecture

for some public HAR datasets, proving it to be suitable for this problem. AETHER uses this same network (Figure 5) to represent the meta-knowledge ω across all tasks, except for the classification layer, which will be the base-learner, created for each task. That way, ω will learn how to extract features from HAR time series in a way to facilitates the identification of the tasks by only a few epochs of training the task-specific classification layer.

4.6 Conclusion

This chapter presented AETHER, explaining the method overview, tasks generation process, and data augmentation techniques. Also, the pseudo-code for AETHER was presented, comparing it to the standard MAML implementation. Finally, we presented the model architecture that will be responsible for representing the meta-knowledge ω, which can be used to rapidly train models to new tasks.

5 EXPERIMENTS AND RESULTS

5.1 Introduction

In Chapter 4, we presented AETHER's algorithm, detailing its stages and the deep learning model architecture used. This chapter will present the experimentation to analyze the AETHER performance and compare it against other works.

This chapter is structured as following: Section 5.2 will explain the experimentation setup, presenting the dataset used for training the meta-model (Subsection 5.2.1), the baselines and competitors (Subsection 5.2.2), the variations of AETHER to be compared (Subsection 5.2.3), and the training an testing configurations (Subsection 5.2.4); Section 5.3 will present the test results for the modes, focusing on answer the following questions: (i) How well does AETHER performs against existing deep learning and meta-learning works for HAR (Subsection 5.3.1)? (ii) How the data augmentation stage impacts AETHER performance (Subsection 5.3.2)? (iii) How well does AETHER performs against traditional machine learning models (Subsection 5.3.3)?. Section 5.4 will evaluate the trained meta-models against other datasets. Finally, Section 5.5 concludes this chapter.

5.2 Experimental setup

5.2.1 Dataset

Kwapisz *et al.* (2011) proposed the WISDM dataset, which is often used in the literature for HAR solutions evaluation. It contains 6 activities: Downstairs, Jogging, Sitting, Standing, Upstairs, and Walking. This thesis uses a newer version of the dataset was proposed by Weiss (2019), containing 51 subjects, 3 minutes of data per activity, collected at 20Hz, and 18 activities (Walking, Jogging, Stairs, Sitting, Standing, Typing, Brushing Teeth, Eating Soup, Eating Chips, Eating Pasta, Drinking from Cup, Eating Sandwich, Kicking (Soccer Ball), Playing Catch w/Tennis Ball, Dribbling (Basketball), Writing, Clapping, and Folding Clothes).

The experiments used both the smartwatch and smartphone accelerometer and gyroscope time series. However, not all activities can be identified through the smartphone: when writing or typing, the person is usually sitting or standing with the smartphone in the pocket or over the table, producing useless data for the activity. The experiments considered only the activities Sitting, Standing Stairs, Jogging, Walking, Kicking, Catch, and Dribbling,

for the smartphone data. As for the smartwatch data, all activities were considered It is often common for people not to keep their smartphones with them during physical activities. However, the WISDM documentation assures that each subject was with the device during each activity.

To apply the task generation methods presented in Chapter 4, Section 4.2, we need to define the ICDs, which are the subject identifier and the device used at the signal collection (the smartwatch or smartphone). The per-condition and multi-conditioned tasks are generated by using a 5-way 5-shot strategy, as explained in Chapter 2, Section 2.3.2. We pre-process the time series data as proposed by Challa *et al.* (2022): first, the time series are normalized to have zero mean and unit standard deviation for each one of the sensor axes. Then, they are split using a sliding window of 128 points ($\sim$ 6 seconds) with 50% of overlapping. Finally, each segment is divided into 4 subsequences of 32 points and re-organized into a matrix 4x32x6 (4 rows of 32 points with 6 dimensions, representing the 3 axes of each sensor) to be fed into the CNN branches. Figure 6 presents an overview of the signals pre-processing.

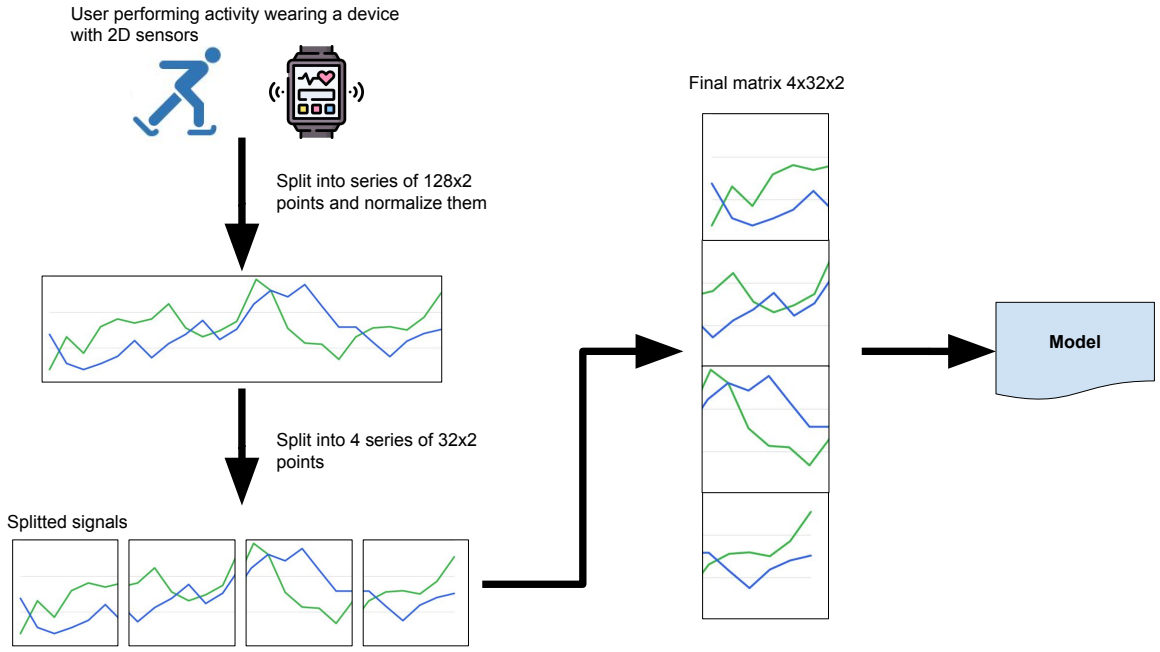


Figure 6 – Data processing flow for each user and device

The training dataset consists of 78 ICD with a total of 57180 sequences ($\sim$ 80%), while the test dataset contains 23 ICD with a total of 13371 points ($\sim$ 20%). No ICD from the test set is present in the training set. However, all 18 activities can be randomly used during the training and testing stages.

It is important to mention that we did not use a validation set to analyze the model convergence. On standard supervised learning, we have a fixed training set, which the model

will adapt over it. However, for meta-learning, the tasks sets keep changing. Although the idea is to provide a meta-model that facilitates task learning, the convergence analysis would not follow the expected monotonic decreasing behavior, due to the tasks variation.

5.2.2 Baselines and Competitors

To properly evaluate AETHER, we structured the experimental evaluation around two distinct categories of reference models: baselines and competitors. The baselines consist of fundamental approaches that provide a performance floor and verify the necessity of the proposed complexity. In contrast, the competitors comprise contemporary, state-of-the-art architectures that define the current technological frontier. By evaluating against both groups, we can demonstrate not only an incremental improvement over existing high-performance solutions but also the foundational robustness of our approach relative to established standards.

We use the following baselines:

- **Traditional DNN** is a traditional deep learning training on the data set, trying to classify the 18 classes using the entire training set and then testing it over the test set. This will allow us to compare whether the AETHER meta-learning procedure is able to improve the performance of the standard deep learning approach.
- **Task-specific DNN** will train a deep learning model specific to each task. This will allow us to evaluate if the learned meta-knowledge ω is able to improve the model performance for new tasks or if it is better to just train a new model for each new task.
- **Task-specific DNN with Data Augmentation.** is the same as the previous one, but the support set of each task will be augmented by the M data augmentation operations. This also allows us to measure the impact of adding the data augmentation process to train the model.

Inspired by (Hossain *et al.*, 2025), AETHER is also compared with some traditional machine learning algorithms, whose parameters were defined by grid-search: Logistic regression; Gaussian Naive Bayes; Decision Tree; SVM with RBF kernel, using $C = 10$, and the *scale* value for γ (which is calculated by $\frac{1}{|features|+\sigma}$); and KNN with $k = 1$. Also, three other models were trained using the autoML framework Autogluon (Erickson *et al.*, 2020): Random Forest, Light GBM, and CatBoost, which had their hyperparameters defined through internal optimization without the use of model ensemble. Since these models do not handle time series directly, they are fed with the flattened vector of the time series (vectors with length 768). This comparison

will evaluate whether AETHER brings a better performance over the traditional approaches.

As for the competitors, the following were used:

- **Personalized MAML** is the implementation proposed by (Wijekoon; Wiratunga, 2020), which is similar to have only per-condition tasks. However, during the training stage, the model can see tasks with only smartphone or only smartwatch data. This is different from what the paper proposes, but since we are going to evaluate with both devices, it is fair that both appear during training.
- **Multi-Conditioned MAML** is analogous to Personalized MAML, but it uses only multi-condition tasks. Although, as far as we know, there is no such model proposed in the literature, we decided to bring it to improve our evaluation. By having both Personalized and Multi-Conditioned MAML, we will be able to evaluate the impact of these heuristics on the generated meta-model.
- **Metasense** is the same implementation proposed by Gong *et al.* (2022). Here, we will need to use 18-way 5-shot tasks, since Metasense has the homogeneity restriction to have all the classes present at all tasks. Comparing it to AETHER will allow us to verify whether homogeneity is advantageous or not.

5.2.3 AETHER variations

AETHER is a flexible method that can be easily customized. One could evaluate different model architectures, optimizers, sets of data augmentation methods, and more. This experiment aims to evaluate if the proposed task generation strategy and data augmentation stage improve the meta-model performance. Therefore, we will analyze three versions of AETHER:

- **AETHER-** will be the approach presented in Section 4 but with no data augmentation stages. It is equivalent to implementing the Metasense algorithm without the homogeneity restriction. This allows us to evaluate if the homogeneity restriction improves the meta-model generalization. Furthermore, by not always using the same activities on the tasks, the meta-model can be used against other datasets with other activities.
- **AETHER-***D* will be the AETHER standard implementation using all the data augmentations of type *D*. For instance, *AETHER-Conv* will use all the convolution augmentations. This allows us to compare how each set of data augmentation performs individually.
- **P-AETHER-***D* will be the same as **AETHER-***D*, but using only per-condition tasks. This allows us to compare the joint impact of this task generation heuristic and the data

augmentation.

- **MC-AETHER-*D*** will be the same as **P-AETHER-*D***, but only using multi-conditioned tasks.
- **AETHER+** uses all the data augmentation for the support set during the meta-training phase.

The data augmentation set used for this experimentation is defined by the following groups:

- **Random Noise (AETHER-RN)** contains 20 augmentations created by combining $\mu \in \{0, 0.5, 1\}$ and $\sigma \in \{0.001, 0.01, 0.1, 1\}$, for the additive strategy, and $\mu \in \{0, 0.5\}$ and $\sigma \in \{0.001, 0.01, 0.1, 0.5\}$ for the multiplicative strategy.
- **Convolution (AETHER-Conv)** contains 14 augmentations, using the window size value $[2, 4, 5, 10]$ for **flattop**, $[4, 5]$ for **hann**, $[2, 3, 4, 5]$ for **hamming**, and $[2, 3]$ for **triangular** and *cosine*.
- **Drift (AETHER-Drift)** contains 20 augmentations created by combining the number of points in $[5, 10, 25, 20]$ with the maximum drift in $[0.1, 0.2, 0.3, 0.4, 0.5]$.
- **Quantization (AETHER-Qtz)** contains 6 augmentation, quantizing the series values into $[10, 20, 30, 40, 50, 100]$ levels.

5.2.4 Train and Test configuration

To ensure a fair evaluation, the experiments will use the same model proposed at Chapter 4 Section 4.5 across all the methods (except for the traditional machine learning ones), as well as the number of epochs, learning rates, and number of tasks per epoch. Based on the experiments performed at Gong *et al.* (2022), we used 5 epochs for inner-learning and 10 epochs outer-learning. The idea is to use only a few of inner-learning epochs to evaluate that the meta model will be able to learn with only a few epochs. The learning rate β for the meta-knowledge ω used was 0.01, and the learning rate α for the inner-training was 0.001. The task distribution $p(T)$ was created using 100 per-condition and 100 multi-conditioned tasks, and 100 tasks are observed per epoch. For Personalized MAML and Multi-Conditioned MAML, we used only their respective heuristics for task generation. We used the ADAM optimizer to apply the gradients over the model.

For testing, we compare the models' adaptation to new per-conditioned tasks by varying the number of observations available for training (1, 2, 5, and 10 shots), as proposed

by Gong *et al.* (2022). The intuition behind this is to see how the models are going to adapt to new users and activities. We evaluate them using the mean of the F1-score metric for each task, calculated as follows:

$$recall = \frac{|\text{true positive}|}{|\text{true positive + false negative}|} \tag{5.1}$$

$$precision = \frac{|\text{true positive}|}{|\text{true positive + false positive}|} \tag{5.2}$$

$$\text{F1-score} = 2 \times \frac{recall \times precision}{recall + precision}. \tag{5.3}$$

Since the tasks are multiclass problems, we will calculate the F1-score for each class and take their average. It is common to calculate a weighted average, but since the tasks are balanced due to the N-way k-shot strategy, the result is the same. Then, for each model, we compute the mean F1-score across the tasks and its standard deviation

5.3 Results

5.3.1 How well does AETHER performs against existing deep learning works for HAR?

Figure 7 shows the mean F1-score for the test tasks and the standard deviation. The traditional DNN had the best score for the test, which may be expected, since it was trained using all the training data. The task-specific DNNs (with and without data augmentation) had the lowest results, being almost as good as a random guess of one of the 5 classes. Metasense was able to perform better than the task-specific DNNs, which is expected due to the trained meta-model. Personalized and Multi-Conditioned MAML had performance similar to Metasense, with Personalized MAML slightly better in most the cases.

AETHER+ not only had the best F1-score among the other task-based models but also had a score that almost reached the traditional DNN, showing that the learning potential of the meta-learning, combined with the tasks generation heuristic and the data augmentation, using small sets of data (using only 1-shot it could achieve over 60%), can have a performance almost as good as the traditional deep learning approach.

An important highlight for this experimentation is that there seems to be no clear impact on which task-generation strategy performed the best. When testing Metasense, Gong *et al.* (2022) provide this same experimentation, concluding that task generation strategies

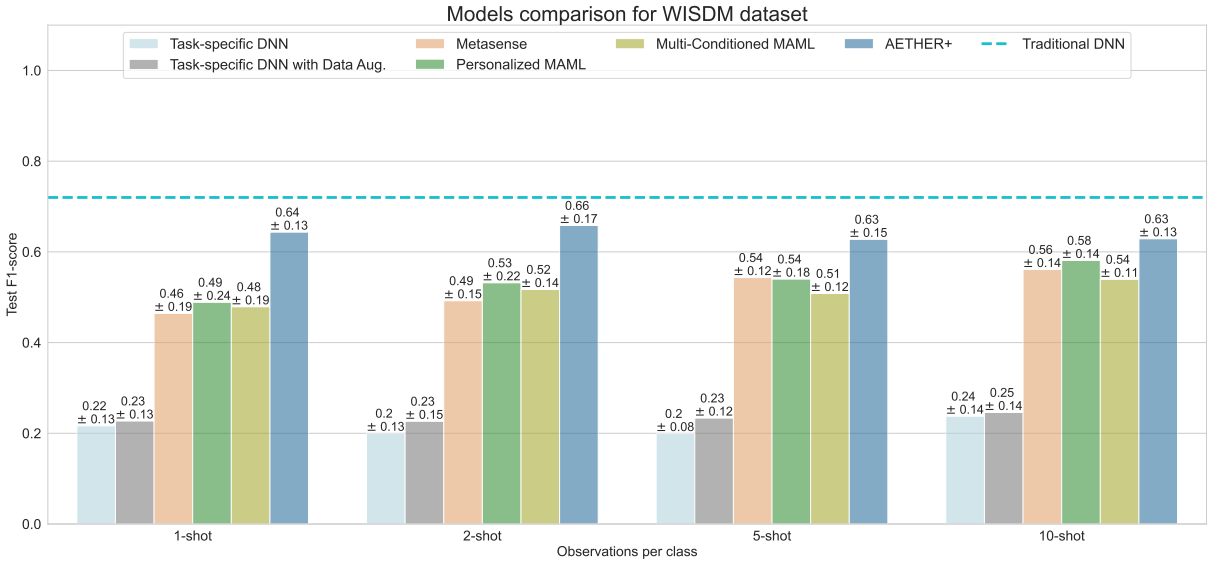


Figure 7 – F1-scores computed for the test set for AETHER, baselines, and competitors.

provide more plausible scenarios for the meta-model to learn. However, they claimed that the homogeneity restriction can help the meta-knowledge learning, but all other meta-models provided similar results or even better.

5.3.2 How the data augmentation stage impacts AETHER performance?

Figure 8 presents the F1-scores for the AETHER variations and the other meta-models. The results for AETHER-, although they are the lowest, allow the Metasense original proposal to be applied over tasks with unseen activity. By comparing the other AETHER variations with the competitors, we can see that the data augmentation provided a significant improvement on the results, being AETHER-Conv able to outperform AETHER+ and even provide results similar to the traditional deep learning baseline.

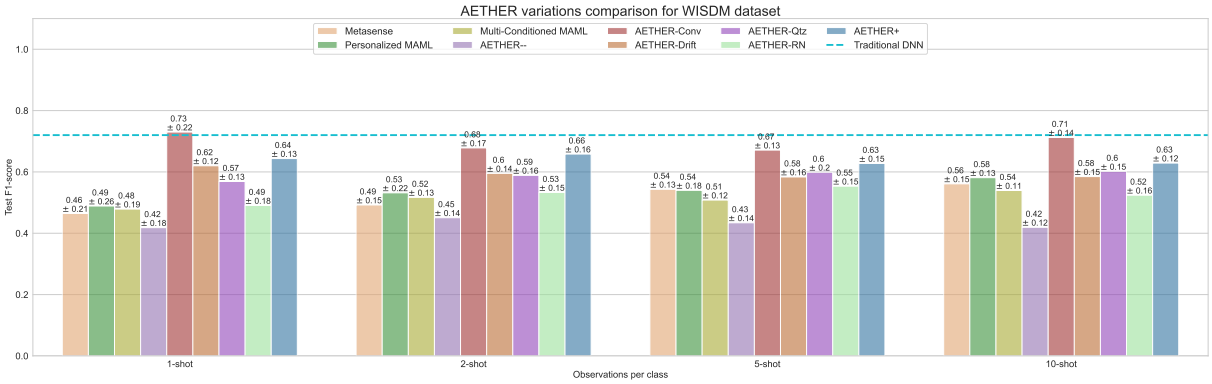


Figure 8 – F1-scores computed for AETHER variations.

The performance of AETHER-Conv may raise the question of *why AETHER+ does*

not use only the Convolutional augmentation set? The answer is that the definition and evaluation of different types of data augmentation sets must be encouraged, since different applications of this method will require different sets of data augmentation. For instance, in a scenario where the signal contains lots of electrical noise, the Random Noise (RN) method may provide signals where the patterns are harder to identify. Figure 9 shows the scores for the Task-specific DNNs using the same sets of data augmentation as AETHER. Without the previous meta-training, the Drift data augmentation set is usually the one that performs better. Also, in some cases, not using the data augmentation can be better than using a particular set.

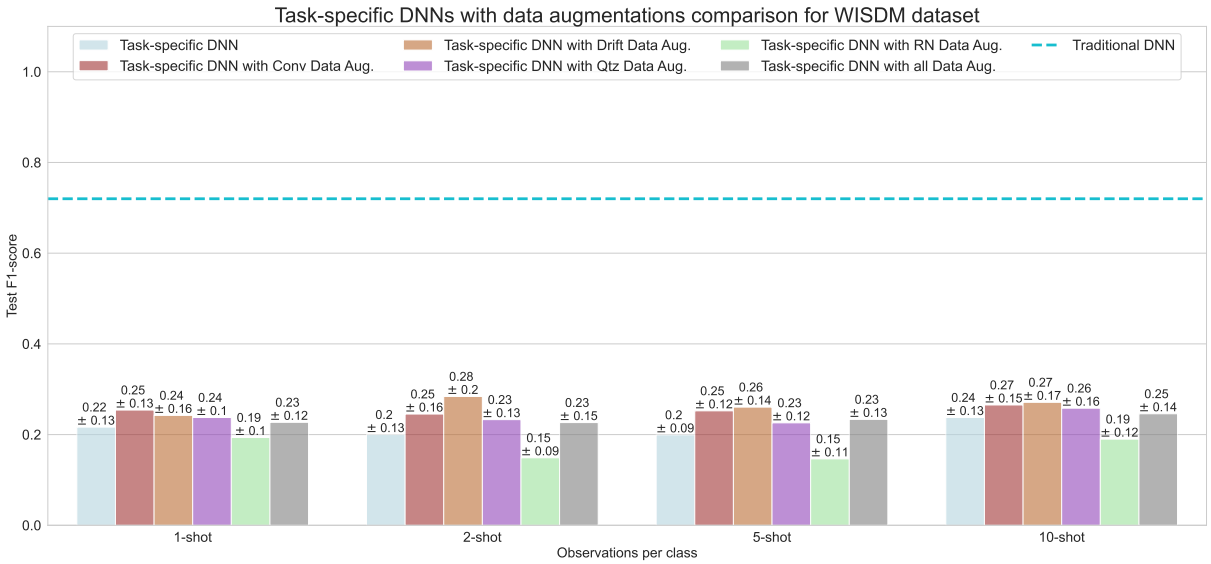


Figure 9 – F1-scores computed for the task-specific models with data augmentation set variations.

Figure 10 presents the F1-scores for the P-AETHER models, varying the data augmentation sets, as well as the baselines and competitors. This time, the quantization data augmentation set stands out from the others, being even a little better than AETHER+ for 1-shot. It's worth noticing that AETHER+ performs better than most of the other variations, and P-AETHER performs better than the competitors. However, P-AETHER+ has the worst result.

Figure 11 presents the F1-scores for the MC-AETHER models, varying the data augmentation sets, as well as the baselines and competitors. For these variations, the Drift data augmentation is the one that provided better results, and, in some cases, MC-AETHER was not able to outperform the competitors.

Lastly, Table 3 summarizes all the metrics for AETHER and its variations, organizing them by the data augmentation set used, the type of task generation strategy, and the number of observed samples per class (shots). For the empty data augmentation sets ($\emptyset$), the scores are for

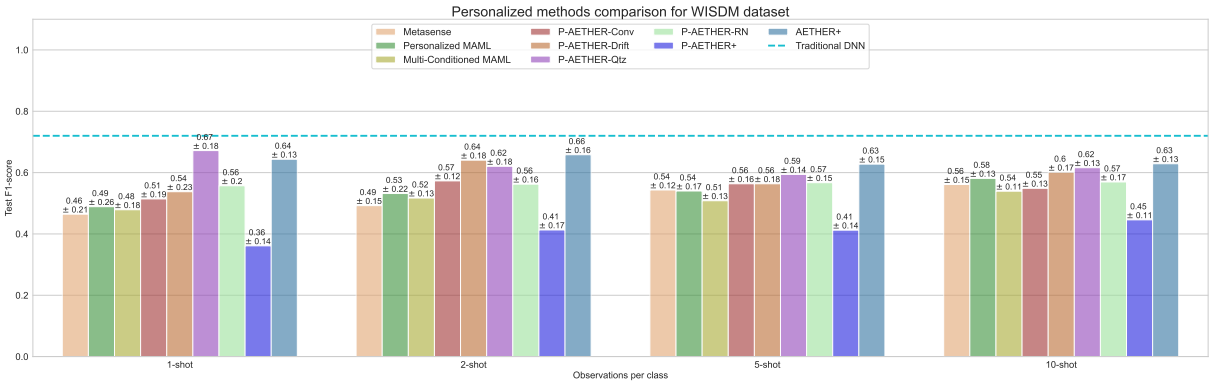


Figure 10 – F1-scores computed for P-AETHER models with data augmentation set variations.

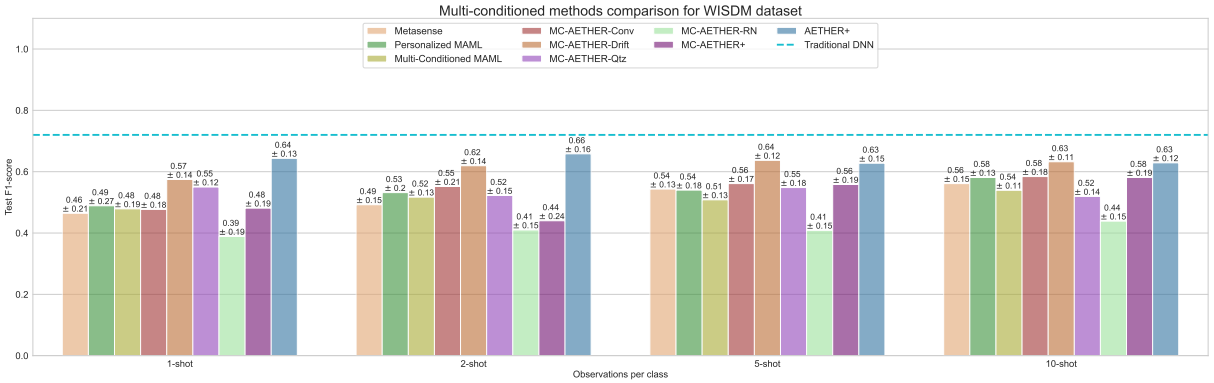


Figure 11 – F1-scores computed for MC-AETHER models with data augmentation set variations.

Personalized MAML, Multi-Conditioned MAML, and AETHER–. The best overall result was for AETHER-Conv, showing that the combination of both task generation strategies and the use of data augmentation improved the results by at least 31%.

Table 3 – WISDM F1-scores for all AETHER (P-, MC-) variations, organized by the data augmentation set used, the type of task generation strategy, and the number of observed samples per class (shots). In **bold**, we have the best **score for each shot**; In *italic*, the best for *the data augmentation set*; Underlined the best for the tasks generation heuristic

Task gen.	Personalized				Multi-Conditioned				AETHER			
Shots	1	2	5	10	1	2	5	10	1	2	5	10
∅	0.49	0.53	0.54	*0.58*	0.48	0.52	0.51	0.54	0.42	0.45	0.43	0.42
Conv	0.51	0.57	0.56	0.55	0.48	0.55	0.56	0.58	***0.73***	**0.68**	**0.67**	**0.71**
Drift	0.54	*0.64*	0.56	0.60	0.57	0.62	*0.64*	0.63	0.62	0.60	0.58	0.58
Qtz	*0.67*	0.62	0.59	0.62	0.55	0.52	0.55	0.52	0.57	0.59	0.60	0.60
RN	0.56	0.56	*0.57*	*0.57*	0.39	0.41	0.41	0.44	0.49	0.53	0.55	0.52
All	0.36	0.41	0.41	0.45	0.48	0.44	0.56	0.58	0.64	*0.66*	0.63	0.63

Table 4 – F1-scores computed for the test set for AETHER+, Metasense, Personalized, and Multi-Conditioned MAML, and traditional machine learning algorithms. The former are trained by generating tasks over the training data and evaluated using tasks generated from the test data, while the latter uses the whole training data to train the models and the whole test data to compute the evaluation metrics

Models	1-shot	2-shot	5-shot	10-shot
Personalized MAML	0.49	0.53	0.54	0.58
Multi-Conditioned MAML	0.48	0.52	0.51	0.54
Metasense	0.46	0.49	0.54	0.56
AETHER+	0.64	0.66	0.63	0.63
AETHER-Conv	**0.73**	**0.68**	**0.67**	**0.71**
Traditional DNN	**0.72**			
Light GBM	**0.71**			
CatBoost	**0.7**			
Random Forest	0.6			
SVM	0.54			
Decision Tree	0.37			
Gaussian Naive Bayes	0.41			
Knn	0.32			
Logistic Regression	0.22			

5.3.3 How well does AETHER performs against traditional machine learning models?

Table 4 shows the F1-score obtained by the traditional machine learning algorithms. The Autogluon models had the best scores (Random Forest, CatBoost, and Light GBM), achieving around 70%, being competitive with the Traditional DNN model. However, it is worth remembering that all of them have the same limitation of not being able to handle unseen activities on new tasks. Furthermore, these results endorse that the traditional machine learning still can compete with modern deep learning and even meta-learning approaches.

5.4 Evaluating AETHER trained meta-model with other datasets

The advantage of training a meta-model is its capability to be applied to unseen scenarios. For HAR, these scenarios could be a new combination of users and devices, like experimented with for the WISDM dataset in the previous section. Although the test setup divided the ICDs in such a way that the test data contains unseen ICDs, some of the used devices, users, and activities were present during the meta-training stage. This section will evaluate the trained meta-model on a more complex set of unseen data, where we use other datasets to create tasks with actual unseen users, devices, and activities. This evaluation will compare not only

AETHER and its variations, but also Personalized MAML, Multi-Conditioned MAML, and Task Specific DNNs. We did not experiment with Metasense due to its homogeneity restriction, since the datasets may bring new tasks and we do not want to map these tasks into the already known set for Metasense. It is worth remembering that, by analyzing AETHER– results, we have a glimpse of how Metasense could have performed. For each dataset, no meta-training was performed. Also, the traditional machine learning models and the traditional deep neural network weren't used since they would require to be trained againg over these datasets, so they can predict their classes. We use the whole dataset for meta-testing.

The WISDM dataset was collected by devices with a sampling rate of 20Hz. To ensure a fair evaluation of the new datasets, we resampled the signals to the same frequency. By doing that, we allow the meta-model to observe time series of the same amount of time for which they were trained. Regarding the accelerometer and gyroscope axes, each device may be different. Since the meta-model should be able to learn patterns from unseen devices, it should be able to identify the patterns regardless of which time series dimension represents which axis. The only thing we kept consistent is that, from the 6 dimensions of each point of the time series, the first three are for accelerometers and the rest for gyroscope. We used the python *pandas* library resampling algorithm to downsample the time series by aggregating the windows using their mean value.

5.4.1 UCI-HAR

The UCI-HAR dataset, presented by Reyes-Ortiz Jorge e Parra (2013), was built by collecting sensors recordings of 30 subjects who performed daily activities (walking, upstairs, downstairs, sitting, standing, and lying) while having the smartphone attached to their waist. Only one device was used, resulting in only 30 ICDs. Since we have only a few different activities, we created the tasks using 6-way 5-shot configuration.

Figure 12 presents the results for the UCI-HAR dataset. Analyzing the task-specific models, we did not get any improvement by using the data augmentations. The Personalized MAML had the worst scores in almost all scenarios, being worse than creating a model per task. The Multi-Conditioned MAML had the best result for the 1-shot scenario, while AETHER+ was the best for the others. However, Multi-Conditioned MAML and AETHER+ had competitive results against each other.

Figure 13 presents the confusion matrices created with the results of the test tasks.

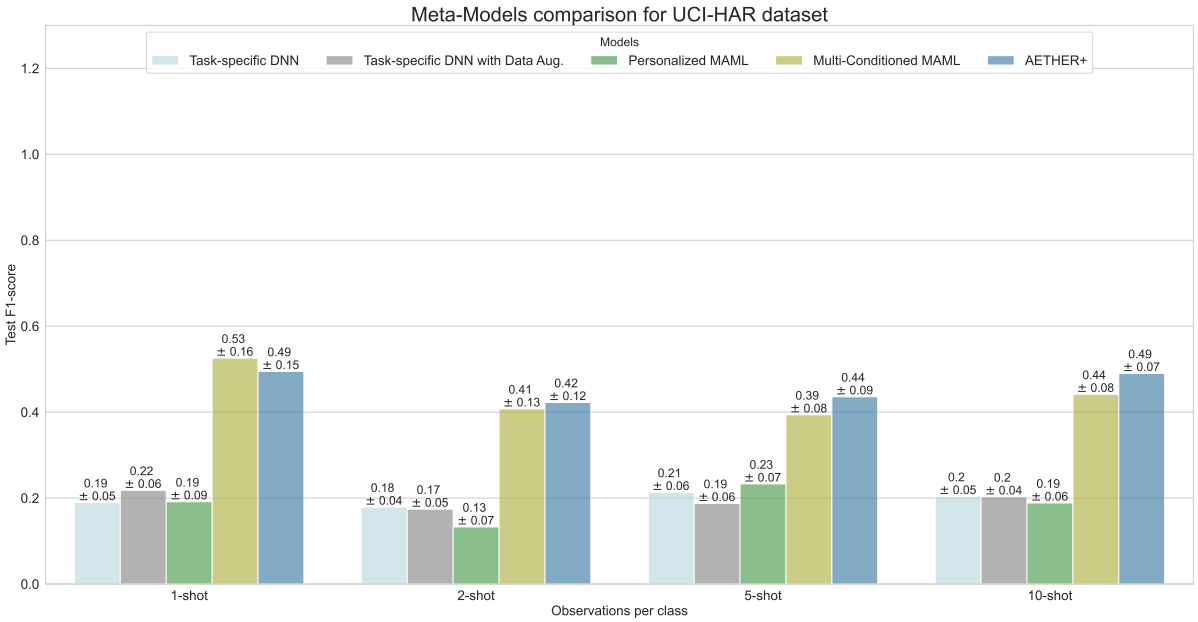


Figure 12 – F1-scores computed for AETHER+, the baselines, and competitors for UCI-HAR dataset.

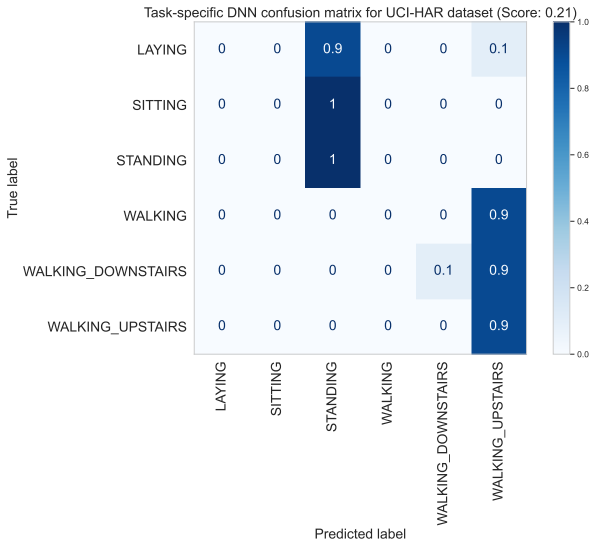
Figure 12 showed that the task-specific models and Personalized MAML had the worst performances, worse even than task-specific models. However, their confusion matrices are very different. While the task-specific models only try to predict one of two classes (having some predictions downstairs), Personalized MAML was able to provide a set of predictions for more classes, showing that, despite the low score, the model tried to learn more complex patterns. By analyzing all the matrices, we can notice that all models would be able to differentiate between stationary activities (laying, sitting, standing) and movement activities (walking, downstairs, upstairs). However, only Multi-Conditioned MAML and AETHER+ are able to identify which movement activities are being performed. Going even further, AETHER+ has a more accurate identification of the movement activities.

Analyzing Figure 14, we can see the AETHER variations based on the data augmentation sets used. AETHER– had the worst performance, being slightly better than the task-specific models. Different from the WISDM dataset, the Convolutional data augmentation set did not achieve the best results, although they are similar to the others. This time, the Random Noise set had the best results, along with the full augmentations set.

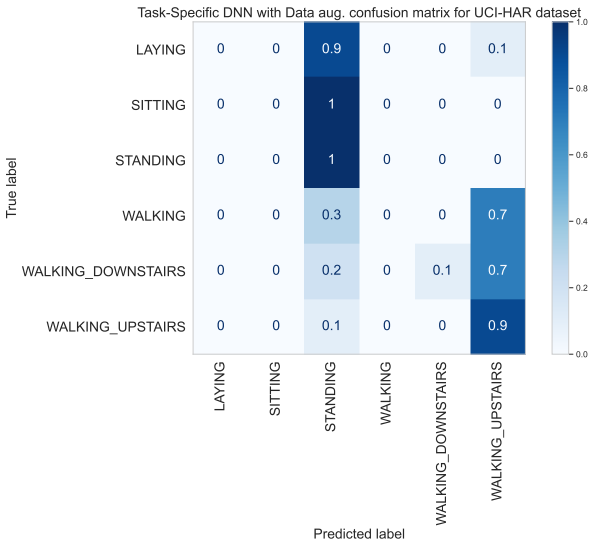
Figures 15 and 16 presents, respectively, the P-AETHER and MC-AETHER variations based on the data augmentation sets. Their results are very similar, with the Random Noise being the data augmentation set with the best results most of the time.

Table 5 shows the summary of the F1-scores obtained from this experiment, organiz-

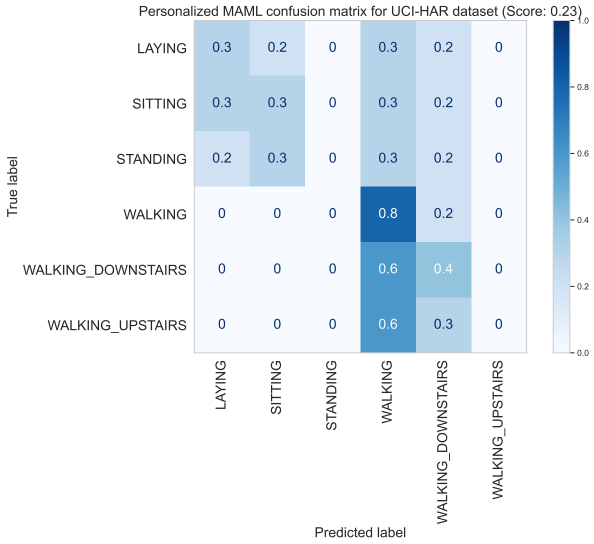
(a) Task-specific DNN



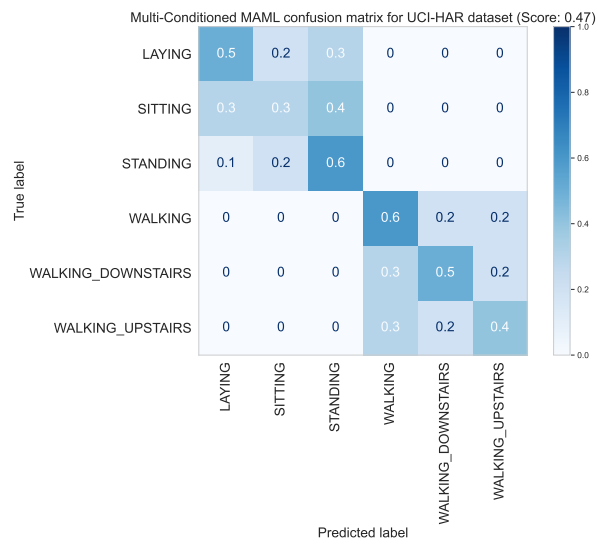
(b) Task-specific DNN with data augmentation



(c) Personalized MAML



(d) Multi-Conditioned MAML



(e) AETHER+

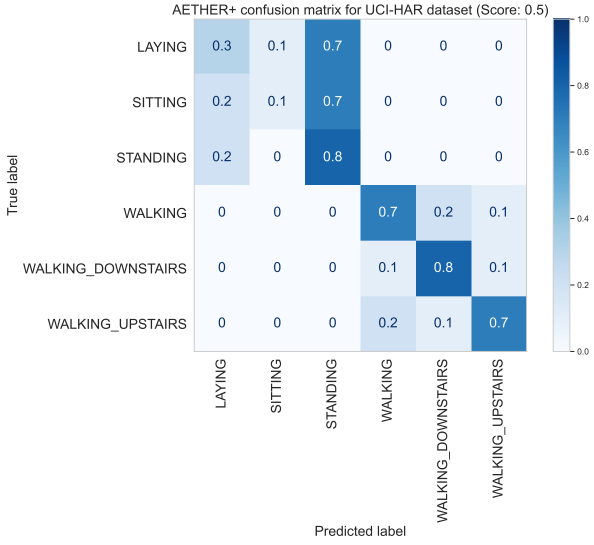


Figure 13 – Confusion Matrix for Task-specific DNN (with and without data augmentation), Personalized MAML, Multi-Conditioned MAML, and AETHER+ models using UCI-HAR dataset

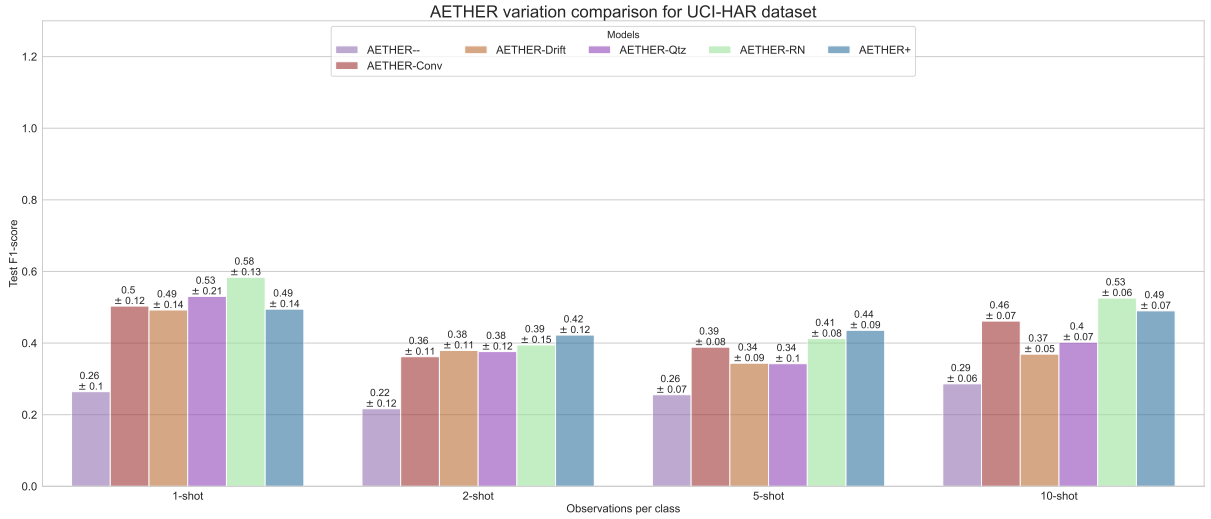


Figure 14 – F1-scores computed for AETHER variations for UCI-HAR dataset.

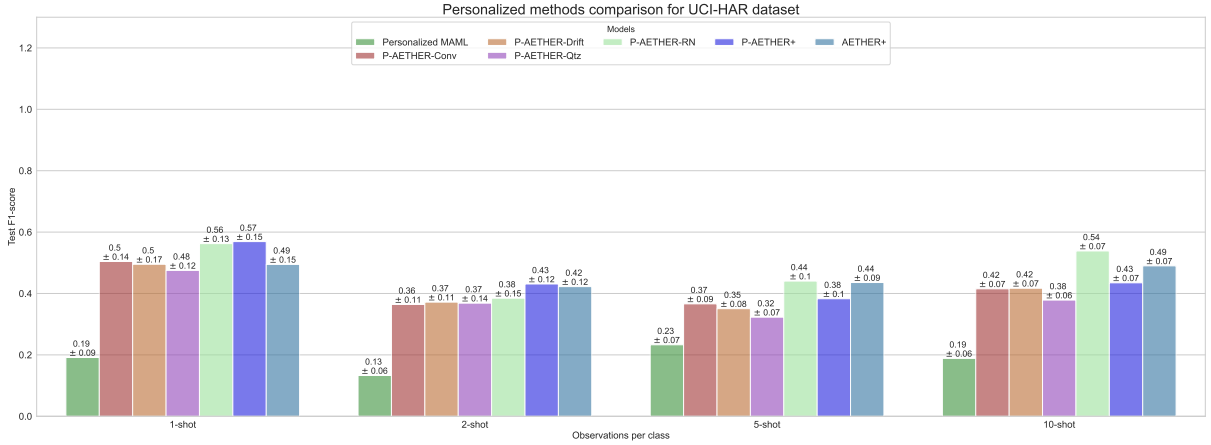


Figure 15 – F1-scores computed for P-AETHER variations for UCI-HAR dataset.

ing them by the data augmentation set used, the type of task generation strategy, and the number of observed samples per class. Again, for the empty data augmentation sets (∅), the scores are for Personalized MAML, Multi-Conditioned MAML, and AETHER–. There is no clear best model

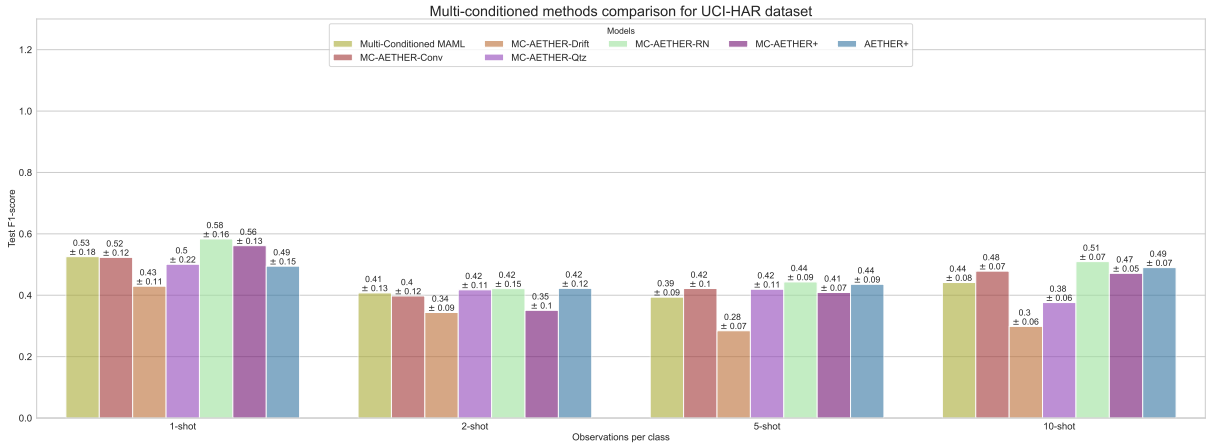


Figure 16 – F1-scores computed for MC-AETHER variations for UCI-HAR dataset.

Table 5 – UCI-HAR F1-scores for all AETHER (P-, MC-) variations, organized by the data augmentation set used, the type of task generation strategy, and the number of observed samples per class (shots). In **bold**, we have the best **score for each shot**; In *italic*, the best for *the data augmentation set*; Underlined the best for the tasks generation heuristic

Task gen.	Personalized				Multi-Conditioned				AETHER			
Shots	1	2	5	10	1	2	5	10	1	2	5	10
∅	0.19	0.13	0.23	0.19	*0.53*	0.41	0.39	0.44	0.26	0.22	0.25	0.29
Conv	0.50	0.36	0.37	0.42	*0.52*	0.40	0.42	0.48	0.50	0.36	0.39	0.46
Drift	*0.49*	0.37	0.35	0.42	0.43	0.34	0.28	0.30	*0.49*	0.38	0.34	0.37
Qtz	0.47	0.37	0.32	0.38	0.50	0.42	0.42	0.38	*0.53*	0.38	0.34	0.40
RN	0.56	0.39	**0.44**	**0.54**	***0.58***	0.42	**0.44**	0.51	***0.58***	0.40	0.41	0.52
All	*0.57*	**0.43**	0.38	0.44	0.56	0.35	0.41	0.47	0.49	0.42	0.43	0.49

for this experiment. For 1-shot, AETHER-RN and MC-AETHER-RN had the best results; For 2-shot, P-AETHER+ had the best results; 5-shot and 10-shot we have P-AETHER-RN (tying with MC-AETHER-RN on 5-shot). Although the P-AETHER variations appear most as the best model, all variations present competitive results. One important detail to notice is that the data augmentation was able to double the performance for the Personalized MAML and AETHER–, proving that it helps on the meta-model creation.

It is worth to mention that all the tasks are using the six classes, and three of them are stationary patterns. As we could see at the confusion matrices, the models often predict all the stationary samples as one class (STANDING) and then try to classify the others, which is a acceptable behavior since differentiating from sitting and standing is a hard task. If we consider the non-stationary activities, we can see that AETHER+ had over 70% of f1-score, way higher than the others.

5.4.2 HHAR

The HHAR dataset from smartphones and smartwatches is a dataset created at Blunck Henrik e Dey (2015) for benchmarking HAR models for real-world contexts. It was collected by 9 users, using 12 different devices (4 smartwatches and 8 smartphones), and performing 6 activities (biking, sitting, standing, walking, stair up, stair down). Again, since we have only a few different activities, we created the tasks using 6-way 5-shot configuration.

Figure 17 presents the results for the HHAR dataset. For this dataset, the data augmentation was able to bring a some improvement for the task-specific model. The Multi-Conditioned MAML outperformed Personalized MAML by at least 30%. This may be caused

by the grand variety of devices (therefore, ICDs) that are present in the dataset. Since the Multi-Conditioned MAML was trained using different ICDs per task, it may be more suitable than Personalized MAML to handle new device peculiarities. Finally, AETHER+ stands out with the best scores, achieving at least 87%, being twice as good as the Personalized MAML.

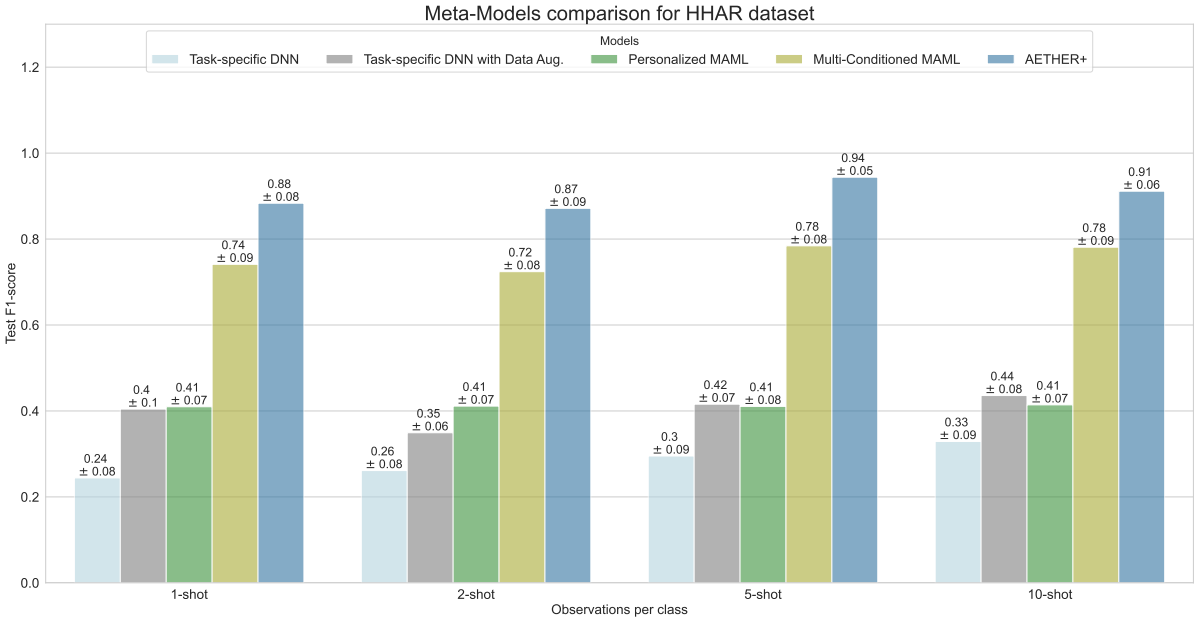


Figure 17 – F1-scores computed for AETHER+, the baselines, and competitors for HHAR dataset.

Analyzing Figure 18, we can see the AETHER variations based on the data augmentation sets used. AETHER– had the worst performance, being worse than the task-specific models with data augmentation. This sustains the hypothesis that the data augmentation allows the meta-model to better generalize the patterns across the tasks. Again, the Convolutional data augmentation set achieved great results, but they are not as different from the others, being as good as the Random Noise set or the combination of all sets.

Figures 19 and 20 presents, respectively, the P-AETHER and MC-AETHER variations based on the data augmentation sets. The data augmentation was able to improve significantly the results for the personalized methods, by at least doubling the Personalized MAML. The multi-conditioned models were improved by the data augmentations, except for the Drift set, which provided results worse than the Multi-Conditioned MAML. It is worth noticing that the P-AETHER variations had slightly better scores than the MC-AETHER ones, despite the opposite behavior when comparing the competitors.

Table 6 shows the summary of the F1-scores obtained from this experiment, organizing them by the data augmentation set used, the type of task generation strategy, and the number

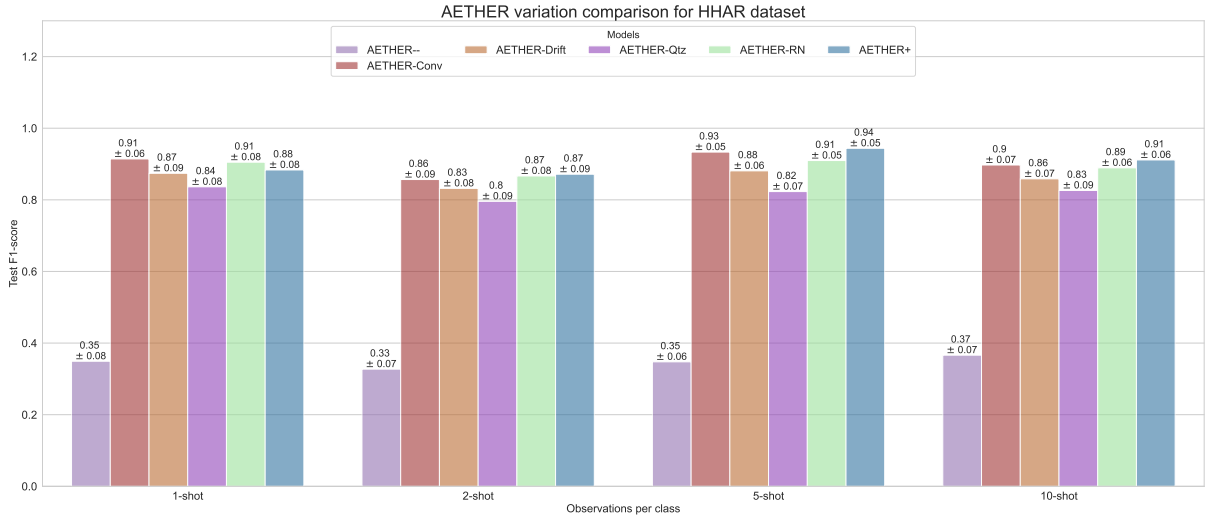


Figure 18 – F1-scores computed for AETHER variations for HHAR dataset.

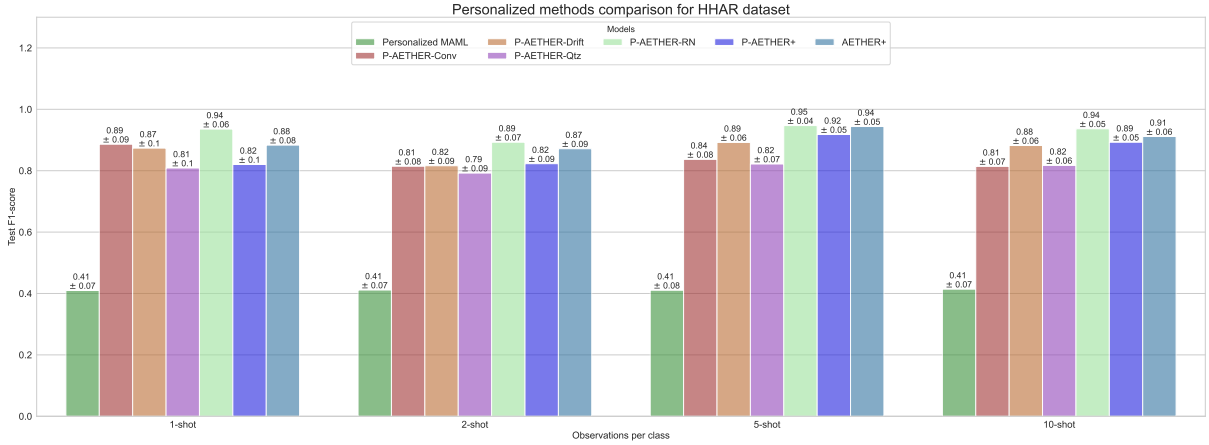


Figure 19 – F1-scores computed for P-AETHER variations for HHAR dataset.

of observed samples per class. Again, for the empty data augmentation sets (∅), the scores are for Personalized MAML, Multi-Conditioned MAML, and AETHER–. The P-AETHER-RN was the best model for all the HHAR experiments, performing over twice as well as its competitor

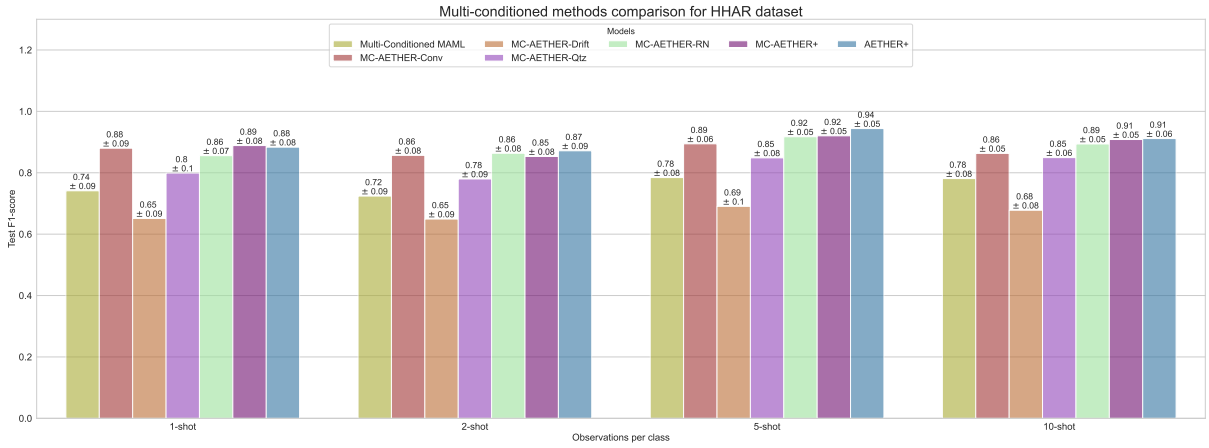


Figure 20 – F1-scores computed for MC-AETHER variations for HHAR dataset.

version. Regarding the multi-conditioned results, the Multi-Conditioned MAML had the best results for all the models without any data augmentation being applied, and, except for the Drift set, the data augmentations only improved its result, with the Random Noise and the full set being the best results. Finally, regarding AETHER, AETHER– had the worst performance, being less than half of its variations. All the data augmentation provided good results and was not far from P-AETHER-RN. Although this experimentation did not stand out AETHER from the other models, the best model used both a task generation strategy and data augmentation to generate the meta-model.

Table 6 – HHAR F1-scores for all AETHER (P-, MC-) variations, organized by the data augmentation set used, the type of task generation strategy, and the number of observed samples per class (shots). In **bold**, we have the best **score for each shot**; In *italic*, the best for *the data augmentation set*; Underlined the best for the tasks generation heuristic

Task gen.	Personalized				Multi-Conditioned				AETHER			
Shots	1	2	5	10	1	2	5	10	1	2	5	10
∅	0.41	0.41	0.41	0.42	0.74	0.72	*0.78*	*0.78*	0.35	0.33	0.35	0.37
Conv	0.89	0.81	0.84	0.81	0.88	0.86	0.89	0.86	0.91	0.86	*0.93*	0.90
Drift	0.87	0.82	*0.89*	0.88	0.65	0.65	0.69	0.68	0.87	0.83	0.88	0.86
Qtz	0.81	0.79	0.82	0.82	0.80	0.78	0.85	*0.85*	0.84	0.80	0.82	0.83
RN	**0.93**	**0.89**	***0.95***	**0.94**	0.85	0.86	0.92	0.89	0.90	0.87	0.91	0.89
All	0.82	0.82	0.92	0.89	0.89	0.85	0.92	0.91	0.88	0.87	*0.94*	0.91

5.4.3 KU-HAR

The KU-HAR dataset, presented by Sikder e Nahid (2021), was built by collecting smartphones sensors recordings of 90 participants performing 18 different activities (stand, sit, talk-sit, talk-stand, stand-sit, lay, lay-stand, pick, jump, push-up, sit-up, walk, walk-backward, walk-circle, run, stair-up, stair-down, and table-tennis). Since, for this dataset, we have a great amount of different activities, we created the tasks using 5-way 5-shot configuration, as we did for WISDM.

Figure 21 presents the results for the KU-HAR dataset. For this dataset, the data augmentation did not bring any performance improvement for the task-specific model. The Multi-Conditioned MAML outperformed Personalized MAML for the 1 and 2-shot scenarios. Finally, AETHER+ stands out with the best scores, achieving at least 46%, being twice as good as the task-specific approaches, and superior that Personalized MAML by at least 15%.

Figure 22 presents the confusion matrices created with the results of the test tasks.

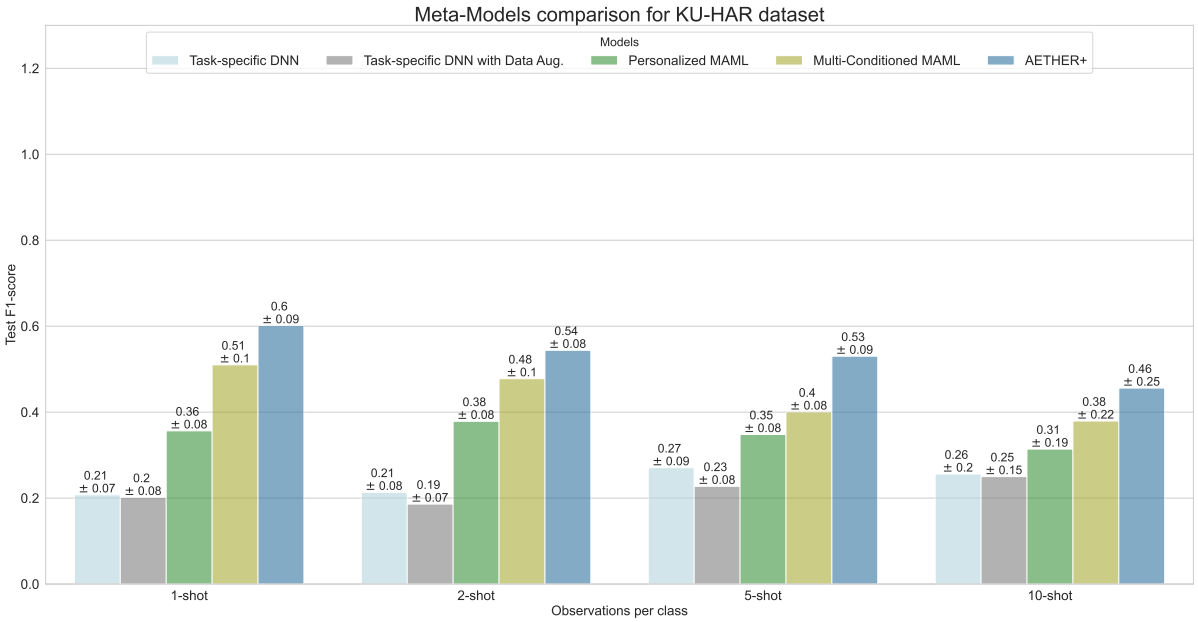


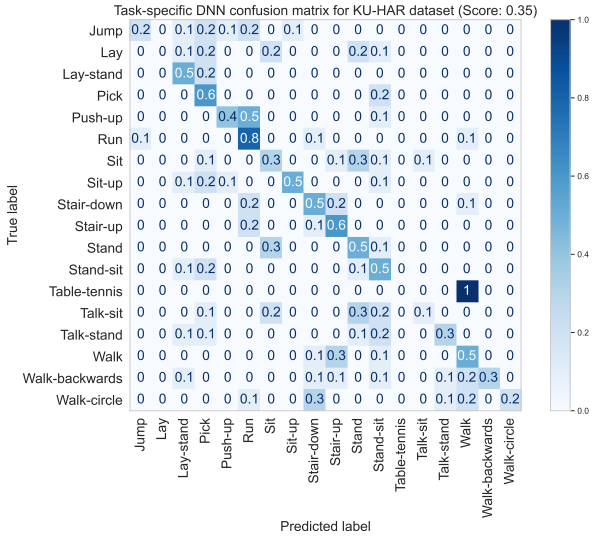
Figure 21 – F1-scores computed for AETHER+, the baselines, and competitors for KU-HAR dataset.

Figure 21 showed that the task-specific models and the task-specific models had the worst performances. By comparing their confusion matrices, we can notice that the Task-specific DNN without the data augmentation has a slightly better result, with a diagonal with higher values. Comparing the meta-learning models, we can see that AETHER+ have a matrix with with higher diagonal values, which is proven by the higher F1-score. Also, if we analyze the area bellow the main diagonal, we can notice that AETHER+ has less prediction mistakes in terms of which classes are misclassified, while the other baselines and competitors have lots of scattered misclassifications.

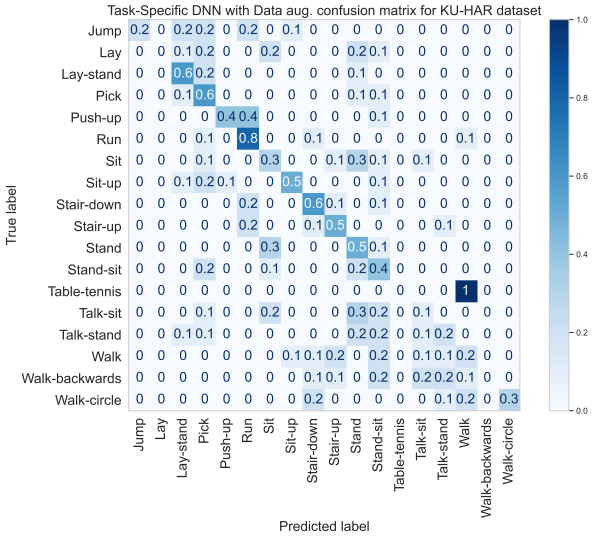
Figure 23 presents the scores of AETHER variations based on the data augmentation sets. AETHER– again had the worst performance, being better only than the task-specific models. Again, the meta-models trained with data augmentation produced better results, being the Convolutional set as good as the full set. Although they have similar scores, AETHER-Conv could be the chosen model for the final application, since, by having less data augmentations to perform than using the full set, the meta-training that could be done in the future for updating the meta-model would be faster.

Figures 24 and 25 presents, respectively, the P-AETHER and MC-AETHER variations. Except for the 10-shot experiment for MC-AETHER variations, the proposed methods have always higher values than the competitors. Also, AETHER+ always has the best result. It is interesting to notice that the MC-AETHER+ have results close to AETHER+ in most

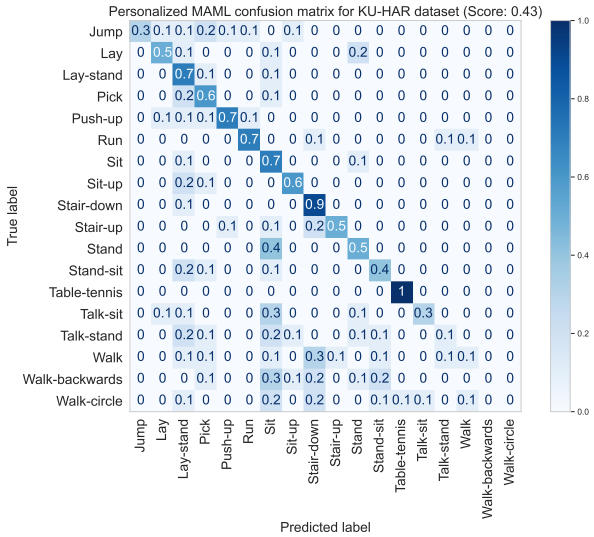
(a) Task-specific DNN



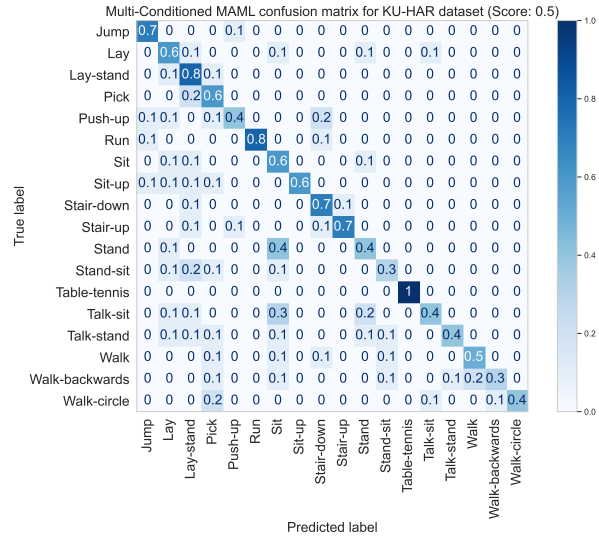
(b) Task-specific DNN with data augmentation



(c) Personalized MAML



(d) Multi-Conditioned MAML



(e) AETHER+

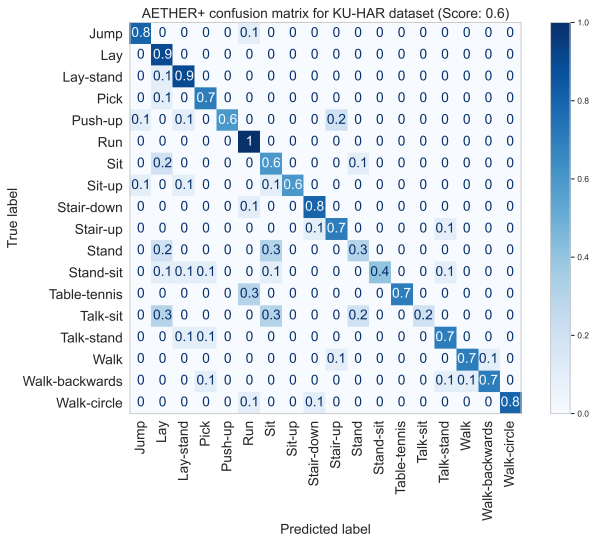


Figure 22 – Confusion Matrix for Task-specific DNN (with and without data augmentation), Personalized MAML, Multi-Conditioned MAML, and AETHER+ models using KU-HAR dataset

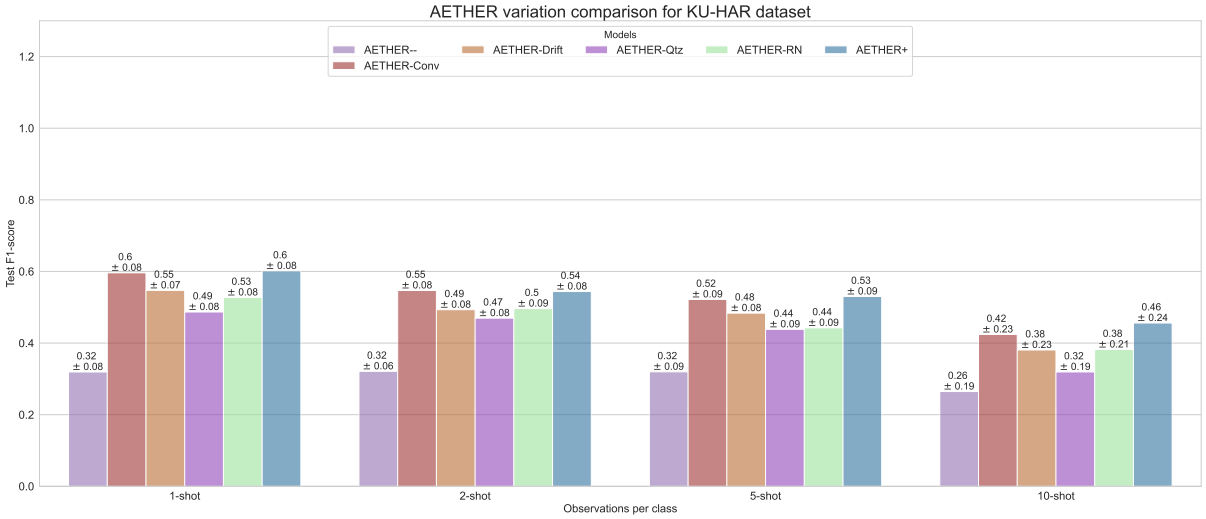


Figure 23 – F1-scores computed for AETHER variations for KU-HAR dataset.

scenarios, while P-AETHER+ has slightly lower metrics. This may be due to the variety of ICDs explored in a single task for Multi-Conditioned models, making them more adaptable to new scenarios. The fact that AETHER+ was trained using both task generation heuristics shows that, although this experiment favors the Multi-Conditioned trained models, the combination of multi-conditioned and personalized tasks provides a better model. However, for MC-AETHER, the score improvement barelly reached 13%, being usually very similar results, except for the 10-shot scenario, where, for the Convolutional and Quantization sets, the results were worse.

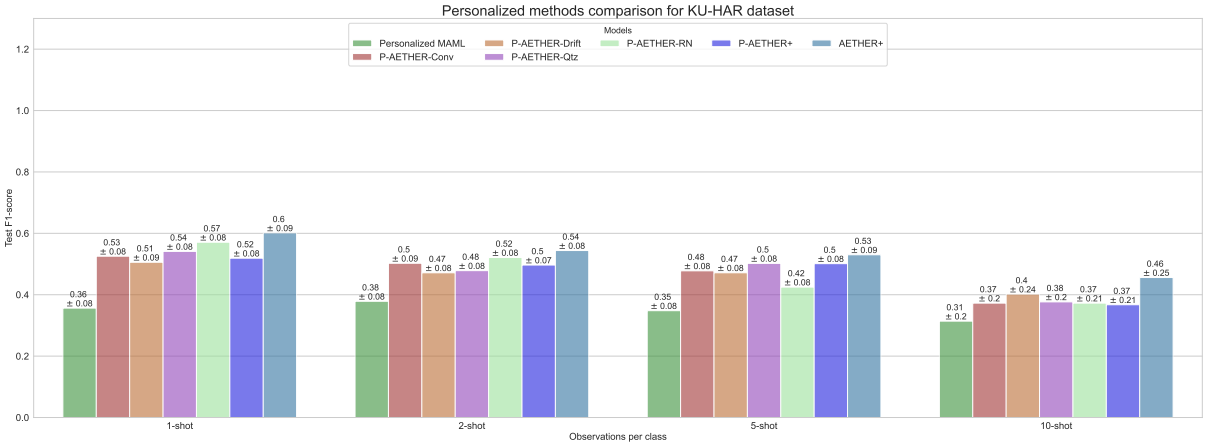


Figure 24 – F1-scores computed for P-AETHER variations for KU-HAR dataset.

Finally, Table 7 summarize all the scores for the KU-HAR dataset, organizing them by the data augmentation set used, the type of task generation strategy, and the number of observed samples per class. Again, for the empty data augmentation sets (∅), the scores are for Personalized MAML, Multi-Conditioned MAML, and AETHER–. AETHER has the best scores for all shot, being equal or superior to the competitors in all cases. Comparing the variations

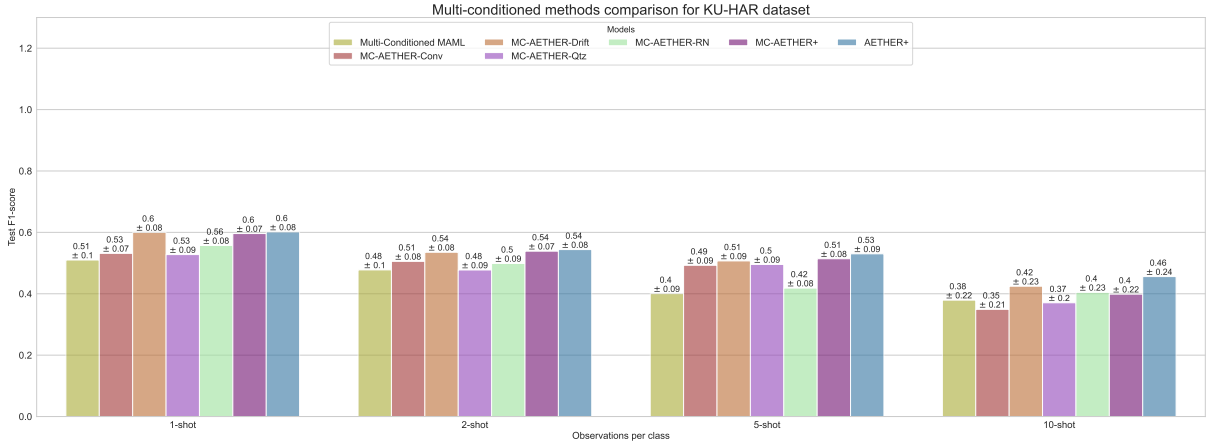


Figure 25 – F1-scores computed for MC-AETHER variations for KU-HAR dataset.

without any data augmentation, the Multi-Conditioned MAML presented the best results. As for the data augmentation sets, the full set is the one with better overall results, being AETHER+ the best model for 1, 5, and 10-shot scenarios.

Table 7 – KU-HAR F1-scores for all AETHER (P-, MC-) variations, organized by the data augmentation set used, the type of task generation strategy, and the number of observed samples per class (shots). In **bold**, we have the best **score for each shot**; In *italic*, the best for *the data augmentation set*; Underlined the best for the tasks generation heuristic

Task gen.	Personalized				Multi-Conditioned				AETHER			
Shots	1	2	5	10	1	2	5	10	1	2	5	10
∅	0.36	0.38	0.35	0.31	*0.51*	0.48	0.40	0.38	0.32	0.32	0.32	0.26
Conv	0.53	0.50	0.48	0.37	0.53	0.51	0.49	0.35	***0.60***	**0.55**	0.52	0.42
Drift	0.51	0.47	0.47	0.40	***0.60***	0.54	0.51	0.42	0.55	0.49	0.48	0.38
Qtz	*0.54*	0.48	0.50	0.38	0.53	0.48	0.50	0.37	0.49	0.47	0.44	0.32
RN	*0.57*	0.52	0.42	0.37	0.56	0.50	0.42	0.40	0.53	0.50	0.44	0.38
All	0.52	0.50	0.50	0.37	***0.60***	0.54	0.51	0.40	***0.60***	0.54	**0.53**	**0.46**

5.5 Conclusion

This chapter presents the experiments with AETHER, explaining all the steps taken and the decisions made for the setup. We compared AETHER against several baselines and competitors, and the results showed that AETHER brings an improvement over the existing meta-learning competitors, being competitive, in some cases, with the deep learning approach. Also, several variations of AETHER were explored to analyze the impact of its improvements, analyzing the data augmentation sets and the task generation strategies individually. Following, we compared AETHER against traditional machine learning algorithms, where some of them

were able to have competitive results, although they do not have the same capability of handling unseen activities.

Furthermore, AETHER and other meta-learning competitors had their meta-models trained over the WISDM dataset and applied over other datasets, allowing us to properly evaluate the meta-models under completely new scenarios. Although the evaluation results did not always produce high metric values, by comparing the baselines and competitors with the AETHER variations, we could notice that the proposed improvements provided better results.

6 CONCLUSION

This document presented the motivation and literature review for human activity recognition problems, focusing on deep learning and meta-learning strategies. We presented AETHER, an optimization-based meta-learning algorithm for HAR which improves the meta-learning process by applying heuristics to generate tasks that will improve meta-knowledge learning and augment data using data augmentation methods. This thesis also performed an extensive experimentation comparing AETHER with some baseline models, which provided good results.

AETHER achieved the general goal of this thesis by providing a model to perform HAR overcoming the individual conditions and labeled data scarcity limitations, and being able to rapidly adapt to new users, devices, and activities. Regarding the specific objectives:

O1 - Provide a meta-learning algorithm for training a HAR model that can rapidly adapt to new tasks.

- AETHER uses MAML algorithm to generate a meta-model that can rapidly adapt to new tasks

O2 - Create a heuristic that improves the generalization of the individual conditions across different tasks.

By combining the per-condition and multi-conditioned tasks to build the task distribution for the meta-learning process, the meta-model was able to improve its generalization.

O3 - Apply data augmentation techniques to improve the adaptation of the model to new tasks.

By using data augmentation techniques, AETHER can increase the size of the support set for the tasks for the inner-learning stage during the meta-learning process, giving the model more samples to identify patterns.

O4 - Validate the proposed algorithm by comparing it against HAR solutions using public datasets.

Chapter 5 presented an experimental evaluation using the WISDM dataset, which demonstrated that the task generation strategies, allied to the data augmentation techniques, were able to improve the scores of the trained meta-model.

O5 - Validate the trained meta-model by evaluating its performance over new datasets.

Chapter 5 also presented an extensive evaluation of the trained meta-models for WISDM dataset by applying them over other public datasets. The results allowed to conclude that, despite for some datasets the metric scores are not very high, the task

generation strategies and data augmentation techniques provided results better than the baselines.

The experiments performed did not only show that AETHER present a valid contribution but also proved that, due to its versatility of how to setup the meta-training process by combining per-condition and multi-conditioned tasks with data augmentation sets, extensive experimentation is needed before deciding which meta-model to use. This experimentation can be an exhaustive process, due to the number of setups one can do to the algorithm. When using AETHER, one need to make the following decisions:

1. Which meta-learning hyperparameters to use, allowing they to use different amounts of tasks to be used per epoch, as well as the number of epochs for the inner-learning stage, and the learning rates for the outer- and inner-learning stages.
2. How to construct the task distribution from where the random tasks for the meta-learning process will be sampled. Although we experimented using per-condition and multi-conditioned tasks using the proportions of 50%/50%, 0%/100% , and 100%/0%, other ratios may be experimented.
3. The definition of the data augmentation sets to be used. During the experimentation, no signal analyzes was performed over the WISDM dataset to decide which data augmentation technique was more suitable to that data context. We only experimented with different techniques and evaluated the impact of the contribution. Also, the experiments proved that there is no data augmentation set that was always the best across the datasets, endorsing the need of experimentation.
4. The architecture of the meta-model and base-learner. Since AETHER is based on a model-agnostic algorithm, any model can be used.

As a future work, one could improve AETHER with a meta-parameter to learn the most adequate data augmentation methods, removing the necessity of manual investigation of the best methods and extensive experimentation. Also, the experimental evaluation can be extended: more HAR datasets can be explored; other deep learning architectures could be investigated; more data augmentation methods, parameters, or even combinations could be evaluated. Also, the model could be applied to edge devices to see how it would behave into real applications.

Finally, despite AETHER being presented to HAR scenario, the contribution is not restricted to such type of tasks. The ICD concept can be easily adapted for other contexts, like weather prediction using sensors placed on different types of environments (top of a house, top of

a building, or in a plain field) and the sensor model; image or video classification using cameras with different image properties (regular image, night view, infrared); fall detection using different types of devices (smartphones, watches, motion units) attached on different parts of the body (legs, wrist, chest); diseases detection using different types of images (x-ray, magnetic resonance) from different parts of the body. Once the data are organized by their ICD, an adequate set of data augmentation techniques should be defined, as well as the meta-model and base-learner architectures, allowing AETHER algorithm to be applied.

BIBLIOGRAPHY

AL-SELWI, S. M.; HASSAN, M. F.; ABDULKADIR, S. J.; MUNEER, A.; SUMIEA, E. H.; ALQUSHAIBI, A.; RAGAB, M. G. Rnn-lstm: From applications to modeling techniques and beyond—systematic review. **Journal of King Saud University-Computer and Information Sciences**, Elsevier, p. 102068, 2024.

ALAWNEH, L.; ALSARHAN, T.; AL-ZINATI, M.; AL-AYYOUB, M.; JARARWEH, Y.; LU, H. Enhancing human activity recognition using deep learning and time series augmented data. **Journal of Ambient Intelligence and Humanized Computing**, Springer, p. 1–16, 2021.

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**, SAGE Publications Sage UK: London, England, v. 52, n. 1-2, p. 37–45, 2019.

BENGIO, Y.; SIMARD, P.; FRASCONI, P. Learning long-term dependencies with gradient descent is difficult. **IEEE transactions on neural networks**, IEEE, v. 5, n. 2, p. 157–166, 1994.

BHARADIYA, J. A comprehensive survey of deep learning techniques natural language processing. **European Journal of Technology**, v. 7, n. 1, p. 58–66, 2023.

BLUNCK HENRIK, B. S. P. T. K. M.; DEY, A. **Heterogeneity Activity Recognition**. 2015. UCI Machine Learning Repository. DOI: https://doi.org/10.24432/C5689X.

BOUCHABOU, D.; NGUYEN, S. M.; LOHR, C.; LEDUC, B.; KANELLOS, I. A survey of human activity recognition in smart homes based on iot sensors algorithms: Taxonomies, challenges, and opportunities with deep learning. **Sensors**, MDPI, v. 21, n. 18, p. 6037, 2021.

CHALLA, S. K.; KUMAR, A.; SEMWAL, V. B. A multibranch cnn-bilstm model for human activity recognition using wearable sensor data. **The Visual Computer**, Springer, v. 38, n. 12, p. 4095–4109, 2022.

CHAUHAN, J.; RAJASEGARAN, J.; SENEVIRATNE, S.; MISRA, A.; SENEVIRATNE, A.; LEE, Y. Performance characterization of deep learning models for breathing-based authentication on resource-constrained devices. **Proceedings of the ACM on Interactive, Mobile, Wearable and Ubiquitous Technologies**, ACM New York, NY, USA, v. 2, n. 4, p. 1–24, 2018.

DAI, Q.; HOU, J.; YANG, P.; LI, X.; WANG, F.; ZHANG, X. The sound of silence: end-to-end sign language recognition using smartwatch. In: **Proceedings of the 23rd Annual International Conference on Mobile Computing and Networking**. [*S. l.*: *s. n.*], 2017. p. 462–464.

DANG, L. M.; MIN, K.; WANG, H.; PIRAN, M. J.; LEE, C. H.; MOON, H. Sensor-based and vision-based human activity recognition: A comprehensive survey. **Pattern Recognition**, Elsevier, v. 108, p. 107561, 2020.

D'ANTONA, G.; FERRERO, A. **Digital signal processing for measurement systems: theory and applications**. [*S. l.*]: Springer Science & Business Media, 2006.

DONG, S.; WANG, P.; ABBAS, K. A survey on deep learning and its applications. **Computer Science Review**, Elsevier, v. 40, p. 100379, 2021.

ERICKSON, N.; MUELLER, J.; SHIRKOV, A.; ZHANG, H.; LARROY, P.; LI, M.; SMOLA, A. Autogluon-tabular: Robust and accurate automl for structured data. **arXiv preprint arXiv:2003.06505**, 2020.

FEI-FEI, L.; DENG, J.; LI, K. Imagenet: Constructing a large-scale image database. **Journal of vision**, The Association for Research in Vision and Ophthalmology, v. 9, n. 8, p. 1037–1037, 2009.

FINN, C.; ABBEEL, P.; LEVINE, S. Model-agnostic meta-learning for fast adaptation of deep networks. In: PMLR. **International conference on machine learning**. [*S. l.*], 2017. p. 1126–1135.

GANESHA, H.; GUPTA, R.; GUPTA, S. H.; RAJAN, S. Few-shot transfer learning for wearable imu-based human activity recognition. **Neural Computing and Applications**, Springer, v. 36, n. 18, p. 10811–10823, 2024.

GONG, T.; KIM, Y.; CHOI, R.; SHIN, J.; LEE, S.-J. Adapting to unknown conditions in learning-based mobile sensing. **IEEE Transactions on Mobile Computing**, v. 21, n. 10, p. 3470–3485, 2022.

GU, F.; CHUNG, M.-H.; CHIGNELL, M.; VALAEE, S.; ZHOU, B.; LIU, X. A survey on deep learning for human activity recognition. **ACM Computing Surveys (CSUR)**, ACM New York, NY, v. 54, n. 8, p. 1–34, 2021.

HASTIE, T.; TIBSHIRANI, R.; FRIEDMAN, J. *et al.* **The elements of statistical learning**. [*S. l.*]: Citeseer, 2009.

HOSPEDALES, T.; ANTONIOU, A.; MICAELLI, P.; STORKEY, A. Meta-learning in neural networks: A survey. **IEEE transactions on pattern analysis and machine intelligence**, IEEE, v. 44, n. 9, p. 5149–5169, 2021.

HOSSAIN, M. M.; HAN, T. A.; ARA, S. S.; SHAMSZAMAN, Z. U. Benchmarking classical, deep, and generative models for human activity recognition. **arXiv preprint arXiv:2501.08471**, 2025.

HUANG, G.; LIU, Z.; MAATEN, L. V. D.; WEINBERGER, K. Q. Densely connected convolutional networks. In: **Proceedings of the IEEE conference on computer vision and pattern recognition**. [*S. l.*: *s. n.*], 2017. p. 4700–4708.

HUGHES, D.; CORRELL, N. Distributed convolutional neural networks for human activity recognition in wearable robotics. In: SPRINGER. **Distributed Autonomous Robotic Systems: The 13th International Symposium**. [*S. l.*], 2018. p. 619–631.

HUISMAN, M.; RIJN, J. N. V.; PLAAT, A. A survey of deep meta-learning. **Artificial Intelligence Review**, Springer, v. 54, n. 6, p. 4483–4541, 2021.

INOUE, M.; INOUE, S.; NISHIDA, T. Deep recurrent neural network for mobile human activity recognition with high throughput. **Artificial Life and Robotics**, Springer, v. 23, p. 173–185, 2018.

IQBAL, M. S.; LUO, B.; KHAN, T.; MEHMOOD, R.; SADIQ, M. Heterogeneous transfer learning techniques for machine learning. **Iran Journal of Computer Science**, Springer, v. 1, p. 31–46, 2018.

JOBANPUTRA, C.; BAVISHI, J.; DOSHI, N. Human activity recognition: A survey. **Procedia Computer Science**, Elsevier, v. 155, p. 698–703, 2019.

KASERIS, M.; KOSTAVELIS, I.; MALASSIOTIS, S. A comprehensive survey on deep learning methods in human activity recognition. **Machine Learning and Knowledge Extraction**, MDPI, v. 6, n. 2, p. 842–876, 2024.

KWAPISZ, J. R.; WEISS, G. M.; MOORE, S. A. Activity recognition using cell phone accelerometers. **ACM SigKDD Explorations Newsletter**, ACM New York, NY, USA, v. 12, n. 2, p. 74–82, 2011.

LAKE, B.; SALAKHUTDINOV, R.; GROSS, J.; TENENBAUM, J. One shot learning of simple visual concepts. In: **Proceedings of the annual meeting of the cognitive science society**. [*S. l.*: *s. n.*], 2011. v. 33, n. 33.

LATTANZI, E.; CALISTI, L.; CONTOLI, C. Are transformers a useful tool for tiny devices in human activity recognition? In: **Proceedings of the 2024 8th International Conference on Advances in Artificial Intelligence**. [*S. l.*: *s. n.*], 2024. p. 339–344.

LEITE, C. S.; MAURANEN, H.; ZHANABATYROVA, A.; XIAO, Y. Transformer-based approaches for sensor-based human activity recognition: Opportunities and challenges. **arXiv preprint arXiv:2410.13605**, 2024.

LI, C.; NIU, D.; JIANG, B.; ZUO, X.; YANG, J. Meta-har: Federated representation learning for human activity recognition. In: **Proceedings of the web conference 2021**. [*S. l.*: *s. n.*], 2021. p. 912–922.

LIU, J.; JIN, Y. A comprehensive survey of robust deep learning in computer vision. **Journal of Automation and Intelligence**, Elsevier, v. 2, n. 4, p. 175–195, 2023.

LIU, Y.; NIE, L.; LIU, L.; ROSENBLUM, D. S. From action to activity: Sensor-based activity recognition. **Neurocomputing**, v. 181, p. 108–115, 2016. ISSN 0925-2312. Big Data Driven Intelligent Transportation Systems. Disponível em: https://www.sciencedirect.com/science/article/pii/S0925231215016331.

MASWADI, K.; GHANI, N. A.; HAMID, S.; RASHEED, M. B. Human activity classification using decision tree and naïve bayes classifiers. **Multimedia Tools and Applications**, Springer, v. 80, p. 21709–21726, 2021.

MEHROTRA, A.; MUSOLESI, M. Using autoencoders to automatically extract mobility features for predicting depressive states. **Proceedings of the ACM on Interactive, Mobile, Wearable and Ubiquitous Technologies**, ACM New York, NY, USA, v. 2, n. 3, p. 1–20, 2018.

MISHRA, N.; ROHANINEJAD, M.; CHEN, X.; ABBEEL, P. A simple neural attentive meta-learner. **arXiv preprint arXiv:1707.03141**, 2017.

MOHSEN, S.; ELKASEER, A.; SCHOLZ, S. G. Human activity recognition using k-nearest neighbor machine learning algorithm. In: SPRINGER. **Proceedings of the International Conference on Sustainable Design and Manufacturing**. [*S. l.*], 2021. p. 304–313.

MURAD, A.; PYUN, J.-Y. Deep recurrent neural networks for human activity recognition. **Sensors**, MDPI, v. 17, n. 11, p. 2556, 2017.

PAN, S.; YANG, Q. **A survey on transfer learning. IEEE Transaction on Knowledge Discovery and Data Engineering, 22 (10)**. [*S. l.*]: IEEE press, 2010.

PATWARDHAN, N.; MARRONE, S.; SANSONE, C. Transformers in the real world: A survey on nlp applications. **Information**, MDPI, v. 14, n. 4, p. 242, 2023.

RÊGO, L. G. C. do; OLIVEIRA, B. S. N.; GASPAR, L. P.; BRANCO, J. A. C.; MACÊDO, J. A. F. de. Uso de meta-learning em tarefas de aprendizado profundo. 2022.

REYES-ORTIZ JORGE, A. D. G. A. O. L.; PARRA, X. **Human Activity Recognition Using Smartphones**. 2013. UCI Machine Learning Repository. DOI: https://doi.org/10.24432/C54S4K.

RONAO, C. A.; CHO, S.-B. Human activity recognition with smartphone sensors using deep learning neural networks. **Expert systems with applications**, Elsevier, v. 59, p. 235–244, 2016.

SAHA, A.; RAJAK, S.; SAHA, J.; CHOWDHURY, C. A survey of machine learning and meta-heuristics approaches for sensor-based human activity recognition systems. **Journal of Ambient Intelligence and Humanized Computing**, Springer, v. 15, n. 1, p. 29–56, 2024.

SÁNCHEZ, D.; TENTORI, M.; FAVELA, J. Activity recognition for the smart hospital. **IEEE intelligent systems**, IEEE, v. 23, n. 2, p. 50–57, 2008.

SANI, S.; WIRATUNGA, N.; MASSIE, S. Learning deep features for knn-based human activity recognition. In: CEUR WORKSHOP PROCEEDINGS. [*S. l.*], 2017.

SIKDER, N.; NAHID, A.-A. Ku-har: An open dataset for heterogeneous human activity recognition. **Pattern Recognition Letters**, Elsevier, v. 146, p. 46–54, 2021.

SNELL, J.; SWERSKY, K.; ZEMEL, R. Prototypical networks for few-shot learning. **Advances in neural information processing systems**, v. 30, 2017.

VASWANI, A.; SHAZEER, N.; PARMAR, N.; USZKOREIT, J.; JONES, L.; GOMEZ, A. N.; KAISER, Ł.; POLOSUKHIN, I. Attention is all you need. **Advances in neural information processing systems**, v. 30, 2017.

VETTORUZZO, A.; BOUGUELIA, M.-R.; RÖGNVALDSSON, T. Efficient few-shot human activity recognition via meta-learning and data augmentation. **Neural Computing and Applications**, Springer, p. 1–17, 2025.

VINYALS, O. Talk: Model vs optimization meta learning. In: **Proc. Int. conf. Neural Inf. Process. Syst**. [*S. l.*: *s. n.*], 2017.

WANG, J.; FARUQUE, M. A. A. Transformer-based contrastive meta-learning for low-resource generalizable activity recognition. In: IEEE. **ICASSP 2025-2025 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)**. [*S. l.*], 2025. p. 1–5.

WANG, Y.; CANG, S.; YU, H. A survey on wearable sensor modality centred human activity recognition in health care. **Expert Systems with Applications**, Elsevier, v. 137, p. 167–190, 2019.

WANG, Y.; YAO, Q.; KWOK, J. T.; NI, L. M. Generalizing from a few examples: A survey on few-shot learning. **ACM computing surveys (csur)**, ACM New York, NY, USA, v. 53, n. 3, p. 1–34, 2020.

WEISS, G. M. Wisdm smartphone and smartwatch activity and biometrics dataset. **UCI Machine Learning Repository: WISDM Smartphone and Smartwatch Activity and Biometrics Dataset Data Set**, v. 7, n. 133190-133202, p. 5, 2019.

WEN, Q.; SUN, L.; YANG, F.; SONG, X.; GAO, J.; WANG, X.; XU, H. Time series data augmentation for deep learning: A survey. **arXiv preprint arXiv:2002.12478**, 2020.

WIJEKOON, A.; WIRATUNGA, N. Personalised meta-learning for human activity recognition with few-data. In: SPRINGER. **International Conference on Innovative Techniques and Applications of Artificial Intelligence**. [*S. l.*], 2020. p. 79–93.

WOZNOWSKI, P.; KING, R.; HARWIN, W.; CRADDOCK, I. A human activity recognition framework for healthcare applications: ontology, labelling strategies, and best practice. In: SCITEPRESS. **International Conference on Internet of Things and Big Data**. [*S. l.*], 2016. v. 2, p. 369–377.

XU, H.; YANG, Z.; ZHOU, Z.; SHANGGUAN, L.; YI, K.; LIU, Y. Indoor localization via multi-modal sensing on smartphones. In: **Proceedings of the 2016 ACM International Joint Conference on Pervasive and Ubiquitous Computing**. [*S. l.*: *s. n.*], 2016. p. 208–219.