

Date of publication xxxx 00, 0000, date of current version xxxx 00, 0000.

Digital Object Identifier 00.0000/ACCESS.2021.DOI

Human Activity Recognition based on acceleration data from smartphones using HMMs

SYLVAIN ILOGA^{1,2,3} , ALEXANDRE BORDAT^{2,4} , JULIEN LE KERNEC^{2,5} , **(Senior member, IEEE)**, and OLIVIER ROMAIN² , **(Member, IEEE)**

¹University of Maroua, Higher Teachers' Training College, Department of Computer Science, P.O.box 55 Maroua, Cameroon

²CY Cergy Paris University, ENSEA, CNRS, ETIS UMR 8051, F-95000, Cergy, France

³University of Sorbonne, ENSEA, CIRD, UMMISCO, F-93143, Bondy, France

⁴Bluelinea, 6 Rue Blaise Pascal, 78990, Élancourt, France

⁵University of Glasgow, James Watt School of Engineering, Glasgow, United Kingdom

Corresponding author: Sylvain Iloga (e-mail: sylvain.iologa@gmail.com).

This work was supported by the U.K. Engineering and Physical Sciences Research Council (EPSRC) under Grant EP/R041679/1 (INSHEP)

ABSTRACT Smartphones are among the most popular wearable devices to monitor human activities. Several existing methods for Human Activity Recognition (HAR) using data from smartphones are based on conventional pattern recognition techniques, but they generate handcrafted feature vectors. This drawback is overcome by deep learning techniques which unfortunately require lots of computing resources, while generating less interpretable feature vectors. The current paper addresses these limitations through the proposal of a Hidden Markov Model (HMM)-based technique for HAR. More formally, the sequential variations of spatial locations within the raw data vectors are initially captured in Markov chains, which are later used for the initialization and the training of HMMs. Meta-data extracted from these models are then saved as the components of the feature vectors. The meta-data are related to the overall time spent by the model observing every symbol for a long time span, irrespective of the state from which this symbol is observed. Classification experiments involving four classification tasks have been carried out on the recently constructed *UniMiB SHAR* database which contains 17 classes, including 9 types of activities of daily living and 8 types of falls. As a result, the proposed approach has shown best accuracies between 92% and 98.85% for all the classification tasks. This performance is more than 10% better than prior work for 2 out of 4 classification tasks.

INDEX TERMS Human activity recognition, Activities of daily living, Fall detection, Hidden Markov Models, Smartphone sensors

I. INTRODUCTION

Human Activity Recognition (HAR) has gained in importance for many decades for its capability to learn meaningful and high-level knowledge about various types of human activities including (but not limited to):

- 1) *Ambulation*: walking, running, climbing stairs, etc.
- 2) *Activities of daily living (ADL)*: eating, drinking, reading, etc.
- 3) *Falls*: fall forward, fall backward, syncope, etc.

More detailed descriptions of the existing types of human

activities are available in [1]¹ and [2]². A review of state-of-the-art techniques for abnormal HAR is also proposed in [3]. There are two main categories of HAR. These are *video-based* HAR and *sensor-based* HAR. Video-based HAR performs high-level analysis of videos or images containing human motions from cameras. No further details related to this category of HAR are provided in the current paper which rather focuses on the second category. Nevertheless, relevant surveys on video-based HAR are available in [4]–[6]. Sensor-based HAR is more popular and widely used because it better

¹See Table 1 of [1]

²See Table 3 of [2]

preserves privacy than video-based HAR. It relies on motion data from several types of smart sensors including:

- 1) *Body-worn sensors*: These sensors are worn by the user to describe the body movements. They are generally embedded in smartphones, watches, standalone devices which include sensors such as accelerometers, gyroscopes, etc.
- 2) *Object sensors*: They are attached to objects to capture object movements. Radio frequency identifiers (RFID) deployed in smart home environment or accelerometer fixed on objects (e.g: glass, cup) are generally used for this purpose.
- 3) *Ambient sensors*: They are used to capture the interaction between humans and the environment in a smart environment. There are many kinds of ambient sensors such as radars, microphones, pressure sensors, temperature sensors, WiFi, Bluetooth, etc.
- 4) *Hybrid sensors*: Here, the three former types of sensors are combined. Further details are available in [7].

Beside HAR, the aforementioned sensors are also adapted for several other topics including indoor positioning methods [8] and pedestrian dead reckoning [9], [10]. Detailed presentations of these types of sensors and related papers are provided in [11]³. When the experiments are performed in smart homes with a variety of sensors, the HAR data processing needs to be distributed over a group of heterogeneous, autonomous and interacting entities in order to be more efficient. An efficient multi-agent approach for HAR in this context has recently been proposed in [12].

HAR can be treated as a typical pattern recognition problem and existing papers in HAR can be organized into two main categories:

- 1) *Conventional pattern recognition techniques* [13]–[46] where the feature extraction and the model building steps are separated.
- 2) *Deep learning techniques* [43], [44], [47]–[88] where the features extraction and model building processes are performed simultaneously in the deep learning models.

Conventional pattern recognition techniques embed the following drawbacks analyzed in [11]⁴:

- 1) The features are extracted via a handcrafted process that heavily relies on human experience and domain knowledge.
- 2) Only shallow features can be learned according to human expertise. Such shallow features can only be used for recognizing low-level activities (*walking*, *running*, etc.), but they can hardly enable to accurately infer complex activities like *having a coffee* for example.
- 3) These techniques often require a large amount of well-labeled data to train the model. However, most of the activity data are remaining unlabeled in real applications.

³See Section 3 of [11]

⁴See Section 2.2 of [11]

The aforementioned drawbacks are overcome by deep learning sensor-based solutions. However deep learning techniques embed the following limitations analyzed in [2]⁵:

- 1) They require lots of computing resources.
- 2) The parameters of the resulting models are difficult to adjust.
- 3) The components of the resulting feature vectors are less interpretable. More details related to this limitation are available in [89] where an overview of explainable artificial intelligence for deep neural networks is proposed.

The current paper addresses these limitations through the proposal of a Hidden Markov Model (HMM)-based technique for HAR which derives interpretable feature vectors from the model's meta-data. The parameters of the resulting HMMs are understandable and can therefore be easily adjusted. Furthermore, the models' training time is reasonable compared to deep models. Raw data from triaxial smartphones sensors are preferred here because studies demonstrated that samples from smartphones sensors (e.g., accelerometer and gyroscope) are accurate enough to be used in the clinical domain, such as ADLs recognition [23].

More precisely, given a signal window w , we first represent w as a sequence $w = w_1 \dots w_T$ of 3-dimensional vectors. The sequential variations of spatial locations within these 3-dimensional vectors are then captured to transform w into the Markov chain δ_w whose content later serves to fix the parameters of an initial HMM associated with w . These parameters are then iteratively adjusted at each iteration of the *Baum-Welch* algorithm to obtain the final HMM λ_w . Thereafter, meta-data derived from λ_w are saved as the components of the descriptor vector \vec{w} associated with w . The performances of the proposed approach are evaluated through flat classification experiments on *UniMiB SHAR* [23] which is a database containing acceleration patterns captured by smartphones and constructed in 2017 for the objective evaluation of ADLs recognition and fall detection techniques.

The rest of this paper is organized as follows: The state of the art is presented in Section II, followed by a summarized presentation of HMMs in Section III. A detailed description of the approach proposed in this paper is given in Section IV. Experimental results are presented in Section V and the last section is devoted to the conclusion.

II. STATE OF THE ART

A. RELATED WORK

1) *Conventional pattern recognition techniques*
Conventional pattern recognition solutions for HAR generally rely on the following process depicted in Figure 1:

- 1) **Signal preprocessing**: Here, sensing devices capture human motions and save them in raw data vectors. Small-time windows (a couple of seconds) of the input signal h are sequentially captured and sampled at a specific sampling frequency depending on the targeted application.

⁵See Table 5 of [2]

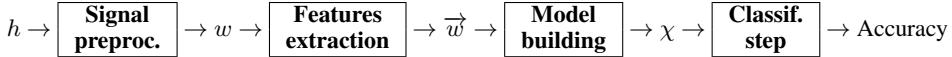


Figure 1: HAR using conventional pattern recognition techniques



Figure 2: HAR using deep learning techniques

- 2) **Features extraction:** Handcrafted features are extracted from each raw data vector w associated with a given time window through the computation of informative statistics. More precisely, *time-domain features* (variance, mean, root mean square, zero-crossing rate, etc.), *frequency domain features* (Fast Fourier Transform, Discrete Cosine Transform, etc.) and *other features* (Principal Component Analysis, Linear Discriminant Analysis, etc.) are manually computed and saved as the components of the features vector \vec{w} of each data vector w . More detailed lists of typical handcrafted features used in HAR are available in [1]⁶ and [2]⁷.
- 3) **Model building:** Here, a conventional machine learning model (classifier) χ is trained to learn the characteristics of the human activities associated with the training feature vectors. The following models are most often used for this purpose:
 - *k Nearest Neighbor* (KNN) [13]–[25].
 - *Support Vector Machines* (SVMs) [16]–[20], [22], [23], [25]–[28].
 - *Decision trees* (DT) [13], [17], [19]–[22], [25], [29]–[34].
 - *Random Forest* (RF) [19], [22], [23].
 - *Multilayer perceptron* (MLP) [17], [21], [33], [35].
 - *Artificial Neural Networks* (ANN) [22], [23].
 - *Naive Bayes and Bayesian Networks* [13], [14], [17], [19], [22], [25], [31], [36].
 - *Fuzzy Inference System* (FIS) [14], [37]–[39].
 - *Boosting and Bagging* [40], [41].
 - *Hidden Markov models* (HMMs) [36], [42]–[46].
- 4) **Classification step:** The previously trained model (classifier) is now used for inferring the corresponding human activity. The *accuracy* is the most used metric for evaluating the performances of these classifiers. Other metrics like the *F1-measure*, the *Precision*, the *Recall* are also rarely used.

2) Deep learning techniques

With deep learning techniques for HAR, the features extraction and model building processes are performed simultaneously in the deep learning models as shown in Figure 2. Here, the feature vectors are automatically learned through

⁶See Table 2 of [1]

⁷See Table 4 of [2]

the network χ instead of being manually designed. A detailed study of deep neural networks for HAR is available in [90]. An evaluation framework allowing a rigorous comparison between handcrafted features and features generated by several deep models is proposed in [91]. The following deep models are most often used in HAR:

- *Deep Neural Networks* (DNN) [47]–[50].
- *Convolutional Neural Networks* (CNN) [49], [51]–[76].
- *Recurrent Neural Networks* (RNN) [49], [72], [76]–[80].
- *Deep belief network* (DBN) and *Restricted Boltzmann Machine* (RBM) [43], [44], [60], [81]–[87].
- *Stacked autoencoder* (SAE) [67], [88].
- *Hybrid models* [7], [49], [60], [67], [72].

Although handcrafted features are considered a drawback in HAR, these features can nevertheless enhance the performances of CNN in some face related problems including age/gender estimation, face detection and emotion recognition [92]. Handcrafted features have also been combined with CNN generated features for HAR [73].

3) Performances of existing work

The implementation of a typical HAR system requires the design of a suitable dataset containing the raw data from which the signal windows will be derived. The datasets utilized in existing papers contain data related to human activities performed by various participants whose characteristics (gender, age, weight, height, etc.) vary from one dataset to the other. Some authors prefer private custom datasets [13]–[16], [26], [27], [29]–[32], [34], [37], [39]–[42], [45]–[48], [50], [53], [58], [60], [65], [66], [68]–[71], [77], [82], [84]–[86], but it is difficult to perform comparisons with these work in such conditions. Other authors adopt the use of publicly available datasets to enable further comparisons [17]–[25], [28], [33], [35], [36], [43], [49], [51], [52], [54]–[59], [61], [62], [64], [67], [72], [74]–[81], [83], [86]–[88]. A deep survey of the evolution of modern datasets for HAR is available in [93]. The performances of a given HAR systems depend on several parameters including: the type of sensors, the number of participants, the protocol of data collection, the selected features and the selected models. Tables 1 and 2 present the performances of relevant existing work with experiments on publicly available and custom datasets. In these tables, the column entitled '#HA' contains the number of human activities involved in the considered work and the last col-

Table 1: Performances of relevant existing work where publicly available datasets are experimented. The values of the *accuracy* and the *F1-measure* are in (%)

Category	N°	Authors[Ref]	Year	Classifier or Deep model	Dataset	# HA	Acc. or F1
Conventional pattern recognition techniques	1	Kwapisz et al. [33]	2011	DT, MLP	WISDM	6	91.7
	2	Anguita et al. [28]	2013	SVM	UCI-HAD	6	96
	3	Shoaib et al. [17]	2013	NB, SVM, MLP, KNN, DT	Shoaib PA	6	99
	4	Medrano et al. [18]	2014	KNN, SVM	tFall	8	95
	5	Shoaib et al. [19]	2014	NB, SVM, DT, RF, KNN	Shoaib SA	7	99.5
	6	Kabir et al. [36]	2016	HMM, NB	Kasteren-A	10	F1:70
					Kasteren-B	13	F1:69
					Kasteren-C	16	F1:69
	7	Reyes et al. [20]	2016	DT, SVM, KNN	SBHAR	7	95.8
					PAMAP2	12	94.33
					REALDISP	33	99.52
	8	Sztyler & Heiner [22]	2016	NB, SVM, DT RF, KNN, ANN	RealWorld(HAR)	8	89
	9	Vavoulas et al. [21]	2016	KNN, DT, MLP	MobiAct	13	99.88
	10	Walser et al. [35]	2016	MLP	UCI-HAD	6	98.11
	11	Micucci et al. [23]	2017	SVM, KNN RF, ANN	UniMiB SHAR	17	98.71
	12	Vavoulas et al. [24]	2017	KNN	MobiFall	13	94.65
	13	Santoyo et al. [25]	2018	NB, SVM, KNN, DT	UMA-Fall	14	85.83
Deep learning techniques	14	Plötz et al. [81]	2011	RBM	OPPORTUNITY	11	75.9
	15	Zeng et al. [51]	2014	CNN	OPPORTUNITY	11	76.83
					Actitracker	6	96.88
	16	Zheng et al. [52]	2014	CNN	PAMAP2	4	93.36
	17	Alsheikh et al. [43]	2015	RBM & HMM	WISDM	6	98.23
	18	Ha et al. [54]	2015		MHEALTH	12	98.29
	19	Hayashi et al. [83]	2015	RBM	HASC	13	91.7
					SHO	7	99.93
	20	Jiang & Yin [55]	2015		USC-HAD	11	97.01
					UCI-HAD	6	95.18
	21	Yang et al. [56]	2015	CNN	ActRecTul	12	96.0
					OPPORTUNITY	18	87.7
	22	Bhattacharya & Lane [86]	2016	RBM	OPPORTUNITY	4	72.1
	23	Chen et al. [57]	2016	CNN	WISDM	6	92.1
	24	Edel & Köppe [77]	2016	RNN	OPPORTUNITY	18	78
					PAMAP2	18	93
	25	Gjoreski et al. [58]	2016	CNN	OPPORTUNITY	4	67
	26	Ha & Choi [59]	2016	CNN	MHEALTH	12	99.66
	27	Hammerla et al. [49]	2016	DNN, CNN, RNN	OPPORTUNITY	18	F1: 92.7
	28	Morales & Roggen [61]	2016		PAMAP2	12	F1: 93.7
	29	Radu et al. [87]	2016		HHAR	6	60
	30	Ravi et al. [62]	2016	CNN	ActiveMiles	7	95.1
					WISDM	6	98.2
	31	Ronao et al. [64]	2016	CNN	UCI-HAD	6	95.75
	32	Zheng et al. [67]	2016	CNN, SAE	PAMAP2	4	93.43
	33	Almaslukh et al. [88]	2017		UCI-HAD	6	97.5
	34	Guan & Plötz [79]	2017	RNN	OPPORTUNITY	18	F1: 72.6
					PAMAP2	12	F1: 85.4
	35	Murad & Pyun [78]	2017	RNN	OPPORTUNITY	18	F1: 92
					UCI-HAD	6	96.7
					USC-HAD	11	97.8
	36	Yao et al. [72]	2017	CNN, RNN	HHAR	6	94.2
	37	Dong et al. [73]	2018	CNN	UCI-HAD	6	96.9
	38	Moya et al. [74]	2018	CNN	OPPORTUNITY	18	92.44
					PAMAP2	12	92.55
	39	Xu et al. [80]	2019	RNN	OPPORTUNITY	18	94.6
					PAMAP2	18	93.5
					UCI-HAD	6	94.5
	40	Wan et al. [75]	2020	CNN	UCI-HAD	6	92.71
					PAMAP2	12	91.00
	41	Xia et al. [76]	2020		OPPORTUNITY	18	92.63
					UCI-HAD	6	95.78
					WISDM	6	95.85

Table 2: Performances of relevant existing work where custom datasets are experimented. Accuracies are in (%).

Category	N°	Authors[Ref]	Year	Classifier or Deep model	# HA	Acc.
Conventional pattern recognition techniques	1	Bao & Intille [13]	2004	NB, DT, KNN	20	84.26
	2	Maurer et al. [29]	2006	DT	6	92.8
	3	Parkka et al. [30]	2006	DT	7	86
	4	Minnen et al. [40]	2007	Boosting	14	90.3
	5	Tapia et al. [31]	2007	NB, DT	30	97.6
	6	Chen et al. [37]	2008	FIS	8	92.86
	7	Ermes & Parkka [32]	2008	DT	6	94
	8	He & Jin [26]	2008	SVM	4	92.25
	9	Jatoba et al. [14]	2008	NB, KNN, FIS	6	86.6
	10	Brezmes et al. [15]	2009	KNN	6	80
	11	He et al. [27]	2009	SVM	4	97.51
	12	Zhu & Sheng [42]	2009	HMM	4	89.74
	13	Altun & Barshan [16]	2010	SVM, KNN	19	98.8
	14	Berchtold et al. [39]	2010	FIS	10	97
	15	Lara et al. [41]	2012	Bagging & Boosting	5	95.7
	16	Lara et al. [34]	2012	DT	3	92.74
	17	Fallmann & Kropf [45]	2016	HMM	14	88.75
	18	Padar et al. [46]	2016	HMM	4	90
Deep learning techniques	19	Fang & Hu [82]	2014	RBM	3	93
	20	Chen & Xue [53]	2015	CNN	8	93.8
	21	Lane & Georgiev [84]	2015	RBM	14	73
	22	Vepakomma et al. [47]	2015	DNN	22	90
	23	Zhang et al. [48]	2015	DNN + HMM	5	93.52
	24	Zhang et al. [85]	2015	RBM	7	99.98
	25	Bhattacharya & Lane [86]	2016	RBM	3	93
	26	Edel & Köppe [77]	2016	RNN	14	83
	27	Gjoreski et al. [58]	2016	CNN	8	75.5
	28	Liu et al. [60]	2016	CNN, RBM	11	98.9
	29	Wang et al. [65]	2016	CNN	4	96.67
	30	Zebin et al. [66]	2016	CNN	6	97.01
	31	Kim & Li [68]	2017	CNN	6	98.8
	32	Lee et al. [69]	2017	CNN	3	92.71
	33	Mohammed & Tashev [70]	2017	CNN	18	90.48
	34	Panwar et al. [71]	2017	CNN	20	99.8
	35	Zhang et al. [50]	2017	DNN	11	98.86

umn contains the *accuracy* of each reviewed work (except for [36], [49], [61], [79], [88] where the *F1-measure* is rather provided).

Among the datasets experimented by existing deep learning techniques listed in Table 1, *OPPORTUNITY*, *PAMAP2* and *UCI-HAD* are the most experimented datasets. However these datasets have not been considered in the current work for several reasons. *OPPORTUNITY* and *PAMAP2* were respectively designed with 4 and 9 participants, which are low values. Additionally, the subset of human activities selected by the authors for these two datasets varies from one work to the other. The dataset *UCI-HAD* was design with enough participants (30 participants) but it only enables us to identify 6 human activities. The advantages of the dataset *UniMiB SHAR* (selected here) compared to the other publicly available datasets listed in Table 1 are thoroughly analyzed in [23]⁸. A summarized description of the UniMiB SHAR datatset is given in Section V-A.

B. PROBLEM STATEMENT

Existing approaches for HAR rely on conventional pattern recognition techniques or deep learning techniques for de-

riving feature vectors from the raw data acquired by diverse sensors. These feature vectors are used for classification purposes. Conventional pattern recognition techniques only enable to learn shallow features extracted via handcrafted processes and require a large amount of well-labeled data. Deep learning techniques for HAR overcome these drawbacks, but they require lots of computing resources and they generate less interpretable feature vectors. Additionally, it is challenging to adjust the parameters of the resulting deep models.

The current paper attempts to provide a solution to these limitations. This solution is based on the following observation: *Every human activity is a sequential process; hence it embeds a natural temporality that is meaningful for its characterization*. For this reason, human activity is generally captured at a precise sampling frequency by dedicated sensors which sequentially record several values in a raw data vector. Unfortunately, the natural temporality embedded in the raw data vectors is actually ignored during the computation of existing feature vectors. Our opinion is that, the sequential variations of spatial locations within each raw data vector enables the derivation of relevant feature vectors which may provide a better characterization of the considered human activity.

⁸Cf. Sections 1 and 2 of [23]



Figure 3: HAR using the proposed HMM-based learning technique.

The current paper follows the principle presented in Figure 3 to analyze these sequential variations in order to generate one new feature vector \vec{w} from each raw data vector w through a machine learning process. HMMs have been selected in this paper on the one hand because they are suitable for sequential data and their training time is reasonable compared to deep models. On the other hand, these models are managed by algorithms whose robustness and efficiency are widespread. HMMs have already been used in HAR systems as classifiers [36], [42], [45], [46]. They have also been combined with deep learning techniques [43], [44]. But they are used here in a different way and for a very different purpose (i.e: feature vectors are extracted from their meta-data).

III. PRESENTATION OF HMMS

A. HMM DEFINITION

A HMM $\lambda = (A, B, \pi)$ is fully characterized by [94]:

- 1) The number N of states of the model. The set of states is $S = \{s_1, s_2, \dots, s_N\}$. The state of the model at time x is generally noted $q_x \in S$.
- 2) The number M of symbols. The set of symbols is $\vartheta = \{v_1, v_2, \dots, v_M\}$. The symbol observed at time x is generally noted $o_x \in \vartheta$.
- 3) The state transition probability distribution $A = \{A[s_i, s_j]\}$ where $A[s_i, s_j] = Prob(q_{x+1} = s_j | q_x = s_i)$ with $1 \leq i, j \leq N$.
- 4) The symbols probabilities distributions $B = \{B[s_i, v_k]\}$ where $B[s_i, v_k] = Prob(v_k \text{ at time } x | q_x = s_i)$ with $1 \leq i \leq N$ and $1 \leq k \leq M$.
- 5) The initial state probability distribution $\pi = \{\pi[s_i]\}$ where $\pi[s_i] = Prob(q_1 = s_i)$ with $1 \leq i \leq N$.

B. HMM USED AS SEQUENCE GENERATOR

A HMM $\lambda = (A, B, \pi)$ can be used to generate a sequence $O = o_1 o_2 \dots o_X$ composed of X symbols observed by the sequence of states $q = q_1 q_2 \dots q_X$ as described in the *Markov chain* (MC) shown in Figure 4. In order to obtain the MC presented in Figure 4, the following algorithm is executed:

- 1) Select the initial state $s_j \in S$ according to the distribution π and set $x = 0$.
- 2) Set $x = x + 1$ and change the current state to $q_x = s_j$
- 3) Select the symbol $o_x \in \vartheta$ to be observed at state q_x according to the distributions in B .
- 4) If ($x < X$) **go to** step 5, else **terminate**.
- 5) Select the state transition to be realized from the current state q_x to another state $s_j \in S$ according to the distribution A , then **go to** step 2.

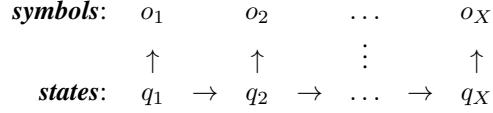


Figure 4: HMM used as sequence generator

C. MANIPULATION OF A HMMS

Consider a sequence of symbols $O = o_1 o_2 \dots o_X$ and a HMM $\lambda = (A, B, \pi)$. The probability $Prob(O|\lambda)$ to observe O given λ is efficiently calculated by the *Forward-Backward* algorithm [94] which runs in $\theta(X.N^2)$. Given a sequence of symbols $O = o_1 o_2 \dots o_X$, it is possible to iteratively re-estimate the parameters of a HMM $\lambda = (A, B, \pi)$ in order to maximize the value of $Prob(O|\bar{\lambda})$, where $\bar{\lambda} = (\bar{A}, \bar{B}, \bar{\pi})$ is the re-estimated model. The *Baum-Welch* algorithm [94] is generally used to perform this re-estimation. This algorithm runs in $\theta(\gamma.X.N^2)$ where γ is the user-defined maximum number of iterations. In this paper, the value $\gamma = 100$ is selected following [95].

D. STATIONARY DISTRIBUTION OF A HMM

A vector $\varphi = (\varphi[s_1], \dots, \varphi[s_N])$ is a stationary distribution of a HMM $\lambda = \{A, B, \pi\}$ if:

- 1) $\sum_j \varphi[s_j] = 1$
- 2) $\forall j, \varphi[s_j] \geq 0$
- 3) $\varphi = \varphi.A \Leftrightarrow (\varphi[s_j] = \sum_i \varphi[s_i] \times A[s_i, s_j], \forall j)$

$\varphi[s_j]$ estimates the overall proportion of time spent by λ in state s_j over a long time span. φ can be extracted from any line of the matrix $A^r = A \times A \times \dots \times A$ (r times) when $r \rightarrow +\infty$. Therefore, the computation of φ requires $\theta(r.N^3)$ arithmetic operations.

IV. THE PROPOSED APPROACH

A. MAIN IDEA

Given a human activity, we assume that the experimental data are recorded by a triaxial smartphone accelerometer which generates 3 vectors $x = (x_1, \dots, x_T)$, $y = (y_1, \dots, y_T)$ and $z = (z_1, \dots, z_T)$ for each signal window w . These vectors are respectively obtained after sampling the signal on each Cartesian axis at a unique sampling frequency. Figure 5 depicts this process for fall detection where the smartphone is located inside subject's waist pocket. In that figure, a signal window extracted from the raw data generated by the smartphone triaxial accelerometer is sampled on each Cartesian axis to derive the 3 data vectors x , y and z .

Existing techniques for HAR generally concatenate these 3 raw data vectors to obtain a unique vector $w = (x_1, \dots, x_T, y_1, \dots, y_T, z_1, \dots, z_T)$. All the $(3T)$ -

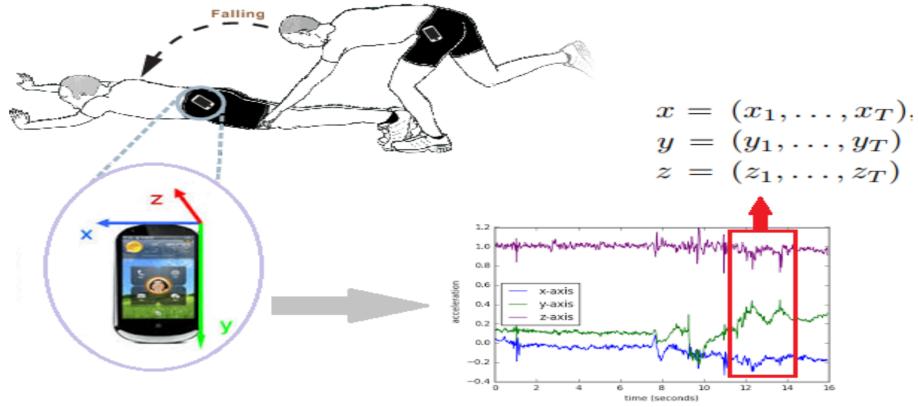


Figure 5: Extraction of the 3 data vectors x , y and z from the raw data generated by the triaxial accelerometer of a smartphone located inside subject's waist pocket during a fall.

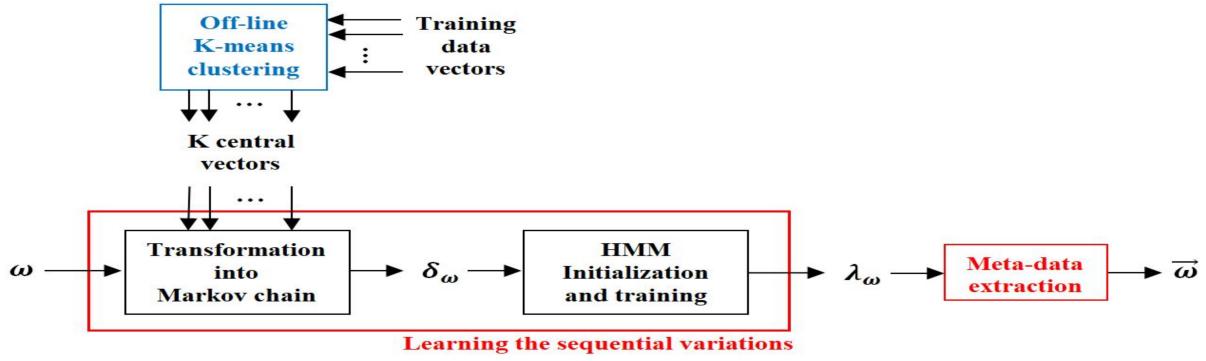


Figure 6: Methodology for deriving the features vector \bar{w} associated with the signal windows w

dimensional raw data vectors resulting from the application of this principle on all the signal windows of the database are then used for handcrafted feature extraction or for deep feature extraction.

The main idea of this work is related to the fact that the former raw data vectors x , y and z can also be viewed as one sequence $w = w_1 w_2 \dots w_T$ composed of 3-dimensional data vectors as depicted in Figure 7 where each $w_i = (x_i, y_i, z_i)$ with $(1 \leq i \leq T)$. Hence, the *sequential variations of the spatial locations* within the T vectors composing w can be captured into a MC δ_w which can later be used to initialize and train a dedicated HMM λ_w . Thereafter, one single feature vector \bar{w} derived from the model's meta-data can finally be associated with w for classification purposes.

$$\begin{aligned} x &= ([x_1], [x_2], \dots, [x_T]) \\ y &= ([y_1], [y_2], \dots, [y_T]) \\ z &= ([z_1], [z_2], \dots, [z_T]) \\ w &= w_1 \quad w_2 \quad \dots \quad w_T \end{aligned}$$

Figure 7: Proposed representation of a signal window $w = w_1 \dots w_T$ derived from the raw data vectors.

B. METHODOLOGY

As it is summarized in Figure 6, the proposed methodology for deriving the feature vector \bar{w} associated with the sequence $w = w_1 w_2 \dots w_T$ is composed of the three following steps:

- 1) The spatial locations of the T data vectors composing the input sequence $w = w_1 w_2 \dots w_T$ are captured by transforming w into the MC δ_w through a calculation involving each w_i and the K central vectors derived from the off-line *K-means* clustering of the training data vectors. More explanations about this step are provided in Section IV-C.
- 2) The content of δ_w is used for initializing a HMM which is then trained using the *Baum-Welch* algorithm to learn the sequential variations occurring inside δ_w (and consequently, inside w). The resulting model is λ_w . Section IV-D is devoted to the presentation of this step.
- 3) Meta-data are finally extracted from λ_w to derive the features vector \bar{w} . More precisely, \bar{w} has M components, where M is the number of symbols of λ_w . The k^{th} component \bar{w}_k of \bar{w} being the overall proportion of time spent by λ_w observing the symbol $v_k \in \vartheta$ over a long time span, irrespective of the state from

which v_k is observed. This step is fully presented in Section IV-E.

C. TRANSFORMATION INTO MARKOV CHAIN

As shown in Figure 4, a MC is composed of symbols and states, both belonging to finite sets. In order to transform a signal window into a MC, these two finite sets must be defined.

To determine the set of symbols, we are initially going to cluster the data vectors derived from all the signal windows found in the training database. More formally, let $H = \{H_1, \dots, H_n\}$ be the set of activities found in the training database, each activity being represented by $|H_j|$ signal windows, with $(1 \leq j \leq n)$. Given that each signal window w is now considered as a sequence $w_1 \dots w_T$ of 3-dimensional data vectors, the experimental database becomes a collection composed of $T \times (\sum_{j=1}^n |H_j|)$ data vectors. The *k-means* clustering algorithm [96] is then executed off-line for organizing this collection of training data vectors into K clusters, where K is a positive user-defined integer. The resulting set $\vartheta = \{v_1, \dots, v_K\}$ of clusters is finally considered as the set of symbols of the model in such a way that all the vectors found inside a given cluster are associated with the same symbol. If we note $v(w_i)$ the cluster containing the data vector w_i , then the signal window $w = w_1 \dots w_T$ is associated with the sequence of symbols $v(w_1) \dots v(w_T)$. The *k-means* clustering algorithm is preferred in this work due to its simplicity of implementation and the quality of its resulting clusters.

To determine the set of states, we focus on the spatial locations of the data vectors inside each cluster. More formally, consider a data vector w_i and let $\tilde{v}(w_i)$ be the center vector of cluster $v(w_i)$. We first evaluate the distance between w_i and $\tilde{v}(w_i)$, then we compare the resulting distance to the highest distance between any data vector of cluster $v(w_i)$ and $\tilde{v}(w_i)$. This comparison leads to the computation of a percentage $\alpha(w_i)$ which spatially characterizes each data vector w_i inside its cluster. Given a selected distance measure $dist$ between vectors, the computation scheme of $\alpha(w_i)$ is shown in (1).

$$\begin{aligned} \alpha(w_i) &= 100 \times (U/V) \text{ in } (\%) \quad \text{where} \\ U &= dist(w_i, \tilde{v}(w_i)) \quad \text{and} \\ V &= \max\{dist(w_j, \tilde{v}(w_i)), \forall w_j \in v(w_i)\} \end{aligned} \quad (1)$$

Our objective is to consider all the possible values of $\alpha(w_i)$ as the set of states. In these conditions, Figure 8 depicts the resulting 'pseudo' MC $\bar{\delta}_w$ associated with $w = w_1 w_2 \dots w_T$.

clusters:	$v(w_1)$	$v(w_2)$	\dots	$v(w_T)$			
	\uparrow	\uparrow	\vdots	\uparrow			
locations:	$\alpha(w_1)$	\rightarrow	$\alpha(w_2)$	\rightarrow	\dots	\rightarrow	$\alpha(w_T)$

Figure 8: Pseudo MC $\bar{\delta}_w$ associated with w

$\bar{\delta}_w$ is not a valid MC because the spatial locations can take any value in the continuous interval $[0, 100]$. However, the states of a MC must always belong to a finite set. To overcome this limitation, the interval $[0, 100]$ is first split into $(m + 1)$ slices $\{s_0, s_1, \dots, s_m\}$ following [97]⁹ as shown in (2), where m is a user-defined integer.

$$s_0 = \{0\} \quad \text{and} \quad s_j = \left] \frac{100}{m} \times (j - 1), \frac{100}{m} \times j \right], \quad (1 \leq j \leq m) \quad (2)$$

If the value of m is very high, the width of each slice s_j becomes tiny and all the elements in s_j converge to one unique value which is $\frac{100}{m} \times j$. In that case, the elements of s_j can be approximated by this single value which we identify here by the index j of slice s_j . In these conditions, the finite set $\{s_0, s_1, \dots, s_m\}$ of slices can be considered here as the set of states. This reasoning enables defining the valid MC δ_w associated with $w = w_1 w_2 \dots w_T$ by replacing every $\alpha(w_i)$ appearing in $\bar{\delta}_w$ by the value $\beta(w_i)$ which is the index j of the slice s_j containing $\alpha(w_i)$ as shown in (3). Figure 9 shows the resulting MC.

$$(\beta(w_i) = j) \Leftrightarrow (\alpha(w_i) \in s_j), \quad (0 \leq j \leq m) \quad (3)$$

symbols:	$v(w_1)$	$v(w_2)$	\dots	$v(w_T)$			
	\uparrow	\uparrow	\vdots	\uparrow			
states:	$\beta(w_1)$	\rightarrow	$\beta(w_2)$	\rightarrow	\dots	\rightarrow	$\beta(w_T)$

Figure 9: MC δ_w associated with w

Proceeding this way, δ_w effectively embeds information related to the sequential variations of spatial locations within w because:

- 1) The sequence $v(w_1) \dots v(w_T)$ of symbols embeds information related to the sequential variations of clusters within w .
- 2) The corresponding sequence $\beta(w_1) \dots \beta(w_T)$ of states embeds information related to the sequential variations of spatial locations inside the various clusters within w .

Consequently, if a HMM λ_w is initialized and trained according to the content of δ_w , this model will learn all the information related to these sequential variations.

D. HMM INITIALIZATION AND TRAINING

- 1) Design of the initial HMM

Given a positive user-defined constant ε , the parameters of each the initial HMM λ_w^0 associated with w , are set to statistically capture the state transitions and the symbols probability distributions from the content of δ_w as follows:

- 1) The set of symbols is the set $\vartheta = \{v_1, \dots, v_K\}$ of clusters generated by the *k-means* clustering algorithm, the number $K = M$ of clusters (symbols) being user-defined.

⁹See Equation 7 of [97]

- 2) The set of states is $S = \{s_0, s_1, \dots, s_m\}$ whose content is computed in (2) where m is the user-defined number of slices used to split the interval $[0, 100]$. Consequently, the number of states is $N = m + 1$.
- 3) The probability of transiting from state s_j to state s_k is calculated in (4), where $\text{transit}(s_j, s_k, \delta_w)$ is the number of transitions from state s_j to state s_k in δ_w and $\text{transit}(s_j, -, \delta_w)$ is the number of transitions from state s_j to any destination in δ_w .

$$A_w^0[s_j, s_k] = \frac{\text{transit}(s_j, s_k, \delta_w)}{\text{transit}(s_j, -, \delta_w) + \varepsilon} \quad (4)$$

- 4) The probability to observe symbol v_k at state s_j is calculated in (5), where $\text{observe}(v_k, s_j, \delta_w)$ is the number of times symbol v_k is observed at state s_j in δ_w , and $\text{observe}(-, s_j, \delta_w)$ is the number of occurrences of state s_j in δ_w .

$$B_w^0[s_j, v_k] = \frac{\text{observe}(v_k, s_j, \delta_w)}{\text{observe}(-, s_j, \delta_w) + \varepsilon} \quad (5)$$

- 5) The probability that the observation starts with state s_j is calculated in (6), where $\text{start}(s_j, \delta_w) = 1$ if δ_w starts with state s_j , 0 otherwise.

$$\pi_w^0[s_j] = \frac{\text{start}(s_j, \delta_w)}{1 + \varepsilon} \quad (6)$$

2) Readjustment of the initial HMM

The parameters of λ_w^0 are not probability distributions. This inconvenience is intentionally introduced by adding ε to the denominators of its various components in order to avoid eventual divisions by zero and zero probabilities. In this work, we experimentally fixed $\varepsilon = 1$. An equitable redistribution of the missing quantity is applied to each element of each line in $\lambda_w^0 = (A_w^0, B_w^0, \pi_w^0)$ to obtain the readjusted initial model $\lambda_w^1 = (A_w^1, B_w^1, \pi_w^1)$ whose parameters are:

- 1) $A_w^1[s_j, s_k] = A_w^0[s_j, s_k] + \frac{1}{m+1} (1 - \sum_{l=0}^m A_w^0[s_j, s_l])$
- 2) $B_w^1[s_j, v_k] = B_w^0[s_j, v_k] + \frac{1}{K} (1 - \sum_{l=1}^K B_w^0[s_j, v_l])$
- 3) $\pi_w^1[s_j] = \pi_w^0[s_j] + \frac{1}{m+1} (1 - \sum_{l=0}^m \pi_w^0[s_l])$

3) Training of the HMM

The readjusted initial HMM λ_w^1 is trained to learn the sequential variations occurring inside δ_w using the Baum-Welch algorithm. The resulting HMM λ_w is the final model associated with w . During this training phase, the training sequences are exclusively composed of symbols appearing in δ_w .

E. FEATURES VECTOR COMPUTATION

The features vector $\vec{w} = (\bar{w}_0, \bar{w}_1, \dots, \bar{w}_m)$ associated with w is finally derived from $\lambda_w = (A_w, B_w, \pi_w)$ by analyzing the behavior of λ_w regarding each symbol v_k . More precisely, we propose to consider \bar{w}_k as the overall proportion of time spent by λ_w at observing symbol v_k over the long term, irrespective of the state from which this observation

is realized. In order to compute \bar{w}_k , one must first evaluate the overall proportion of time spent by λ_w at observing v_k in each state s_i over the long term as follows:

- 1) Evaluate the overall proportion of time spent by λ_w in state s_i over the long term. This proportion is given by the i^{th} component $\varphi_w[s_i]$ of the stationary distribution of λ_w .
- 2) Multiply the result obtained at step 1 by the probability of observing v_k in state s_i which is $B_w[s_i, v_k]$.

The value of \bar{w}_k is finally obtained by repeating this process for every state s_i and summing the resulting proportions (7).

$$\begin{aligned} \vec{w} &= (\bar{w}_0, \bar{w}_1, \dots, \bar{w}_m) && \text{where} \\ \bar{w}_k &= \sum_{i=1}^K (\varphi_w[s_i] \times B_w[s_i, v_k]) \text{ with } (0 \leq k \leq m) \end{aligned} \quad (7)$$

V. EXPERIMENTAL RESULTS

A. EXPERIMENTAL DATASET

Among the publicly available databases recorded with smartphones listed in Table 1, the dataset *UniMiB SHAR* has been selected in this work [23]. It is database of acceleration patterns measured by smartphones to be used as a common benchmark for the objective evaluation of both, ADLs recognition and fall detection techniques. This dataset contains 17 human activities including 9 different types of ADLs and 8 different types of falls. Table 3 presents the description of each ADL/fall.

During the construction of the selected database, human activities were performed by 30 subjects (including 24 females) between 18 and 60 years of age. Each ADL/fall type was performed twice by each subject: the first time with the smartphone in the right pocket and the second time with the smartphone in the left pocket. Signal windows of 3s each were saved during every experimental trial of a given ADL/fall performed by each subject. For each signal window w , the accelerometer recorded three data vectors (samples) x , y and z , each having $T = 151$ components. The database contains a total of 11.771 samples not equally distributed across activity types: 7.759 samples describing ADLs and 4.192 samples describing falls. Further details about the data acquisition, the experimental protocols, the characteristics of the subjects, the signal segmentation and the signal processing are available in [23].

B. EXPERIMENTAL SETTINGS

The classification experiments performed in this work were realized on a personal computer having 16 GB of main memory and the following processor: *Intel(R) Core(TM) i7-8665U CPU @ 1.90 GHz 2.11 GHz*. We evaluated four different classification tasks, following [23]:

- 1) **AF-17** which contains 17 classes (9 ADL classes and 8 FALL classes).
- 2) **A-9** which contains 9 ADL classes.
- 3) **F-8** contains 8 FALL classes.

Table 3: Descriptions of the 17 human activities of *UniMiB SHAR*

Category	Label	Description
ADLs	StandingUpFL	From laying on the bed to standing
	LyingDownFS	From standing to lying on a bed
	StandingUpFS	From standing to sitting on a chair
	Running	Moderate running
	SittingDown	From standing to sitting on a chair
	GoingDownS	Climb the stairs moderately
	GoingUpS	Down the stairs moderately
	Walking	Normal walking
	Jumping	Continuous jumping
Falls	FallingBackSC	Fall backward while trying to sit on a chair
	FallingBack	Generic fall backward from standing
	FallingWithPS	Falls using compensation strategies to prevent the impact
	FallingForw	Fall forward from standing, use of hands to dampen fall
	FallingLeft	Fall left from standing
	FallingRight	Fall right from standing
	HittingObstacle	Falls with contact to an obstacle before hitting the ground
	Syncope	Getting unconscious

- 4) **AF-2** which contains 2 classes obtained by considering all the ADLs as one class and all the FALLs as one class.

The *Euclidean* distance was selected in this work as the distance $dist$ between two vectors required in (1). (2) was used to split the interval $[0, 100]$ into 51 slices (i.e: we fixed $m = 50$) following [97] where the authors used analog reasoning to split the same interval to perform the comparison of finite sets of histograms using HMMs. Hence, the number of states of the HMMs designed in the current work is 51.

To analyze the impact of the user-defined number K of clusters discovered by the *k-means* clustering algorithm on the performances of the proposed approach, we experimented with the 5 following values of K : 20, 40, 60, 80 and 100. Consequently, the number of symbols of the HMMs designed in this work also varies accordingly. The first step of the machine learning process presented in Figure 6 and corresponding to the transformation of every signal w into the MC δ_w was entirely developed in Matlab. This choice was dictated by the fact that the database files were available as Matlab tables. Therefore, we executed a Matlab version of the *k-means* clustering algorithm during this step. Depending on the classification task and on the selected number of clusters, the off-line clustering step could sometimes take over an hour.

The two remaining steps of the proposed machine learning process were both developed in C language. Given a signal window w and its associated HMM $\lambda_w = (A_w, B_w, \pi_w)$, the stationary distribution φ_w of λ_w is obtained in this work by extracting the first line of the matrix $(A_w)^r$ with $r = 100$. After discovering the K clusters for each classification task, the computation of each feature vector \vec{w} associated with w took between 50 and 3500 ms, depending on the content of the signal window. The overall time taken for the computation of the feature vectors associated with all the signal windows varied from one classification task to another depending on the considered number of signal windows. For

each classification task in $\{\text{AF-2}, \text{F-8}, \text{A-9}, \text{AF-17}\}$ and for each number of clusters in $\{20, 40, 60, 80, 100\}$, the resulting descriptor vectors have been saved into one online available '*arff*' files ¹⁰ which are taken as inputs by WEKA. The Matlab items (codes and tables) required to perform the off-line *k-means* clustering and the transformation into of Markov chains are also available through this same URL.

C. CLASSIFICATION PERFORMANCES

Classification experiments were realized with the soft WEKA [98] through 5-fold cross-validation following [23]. The following classifiers have been selected in this paper and their corresponding names in WEKA are shown in brackets:

- 1) *k-NN* (IBk) with $k = 1$, was used for the *Euclidean* and the *Manhattan* distances.
- 2) *SVMs* with polynomial kernel (SMO).
- 3) *Multilayer Perceptron* (MLP).
- 4) *Decision trees* (J48).
- 5) *Random Forest* (RF), these are bootstrap-aggregated decision trees with 300 bagged classification trees.

Table 4a presents the best classification accuracies for each classification task using the proposed feature vectors, irrespective of the number of clusters generating these accuracies. According to Table 4a, the selected classifiers can be organized in descending order of performances for all the classification tasks as follows: RF, IBk (*Manhattan*), IBk (*Euclidean*), J48, MLP and SMO. The content of Table 4a demonstrates the high quality of the proposed descriptor vectors derived from the proposed HMM-based learning process with best accuracies always above 75%, 84% and 92% for the J48, the IBk and the RF classifiers respectively. This table also reveals the unsatisfactory performances exhibited by the SMO and the MLP classifiers for the AF-17 classification task and the very poor performances of these same classifiers for the F-8 classification task. Detailed classification performances for each classification task are presented in

¹⁰<http://perso-etis.ensea.fr/sylvain.iloga/index.html>

Table 4: Classification results. Accuracies are in (%). The best accuracies are in bold.

(a)- Best performances						
	IBk		SMO	MLP	J48	RF
	Eucli	Manha				
AF-2	97.17	97.55	89.21	93.97	95.79	98.85
F-8	84.35	87.71	45.06	49.59	75.07	92.00
A-9	91.81	92.57	72.56	82.20	84.91	95.28
AF-17	87.05	89.61	53.58	62.88	79.39	93.22

(c)- 40 clusters						
	IBk		SMO	MLP	J48	RF
	Eucli	Manha				
AF-2	97.17	97.55	89.18	93.37	95.79	98.62
F-8	81.15	84.78	34.06	44.53	74.54	90.67
A-9	91.81	92.57	72.56	82.20	84.91	95.28
AF-17	87.05	89.61	50.92	61.86	79.39	93.22

(d)- 60 clusters						
	IBk		SMO	MLP	J48	RF
	Eucli	Manha				
AF-2	96.85	97.14	88.44	87.51	95.62	98.70
F-8	82.56	85.28	38.14	49.59	75.07	91.53
A-9	85.88	87.08	69.60	76.90	77.59	93.45
AF-17	85.10	87.26	53.58	62.88	77.02	92.77

(e)- 80 clusters						
	IBk		SMO	MLP	J48	RF
	Eucli	Manha				
AF-2	96.61	97.12	88.11	89.00	94.10	98.85
F-8	83.42	87.11	40.17	34.42	74.57	92.00
A-9	81.39	82.99	64.34	61.24	68.53	92.71
AF-17	80.87	83.67	52.35	51.53	71.92	92.09

(f)- 100 clusters						
	IBk		SMO	MLP	J48	RF
	Eucli	Manha				
AF-2	96.23	96.53	89.21	77.66	93.60	98.63
F-8	84.35	87.71	45.06	29.46	74.83	91.81
A-9	76.54	78.79	56.73	42.68	58.20	91.66
AF-17	78.03	81.69	49.97	33.26	65.82	91.09

Tables 4b to 4f respectively for 20, 40, 60, 80 and 100 clusters. According to these tables, the variations of the user-defined number K of clusters do not significantly influence the performances of the proposed technique. Indeed, the gaps between the classification accuracies for the 5 experimental values of K are low. Consequently, it is not worth selecting a high value of K . We therefore recommend low values between 20 and 40.

D. COMPARISONS WITH RELATED WORK

We have compared the best classification performances obtained in this paper with those obtained in [23] where the authors conducted classification experiments on the same database and for the same classification tasks, using the following classifiers:

- 1) *k-NN* with $k = 1$.
- 2) *SVMs* with a radial basis kernel.
- 3) *Artificial Neural Networks*.
- 4) *RF* with 300 bagged classification trees.

Comparison results presented in Table 5 reveal that the approach proposed in this paper always outperforms [23] with positive accuracy gains reaching +13.45% and +10.36% respectively for F-8 and AF-17.

Table 5: Comparisons with [23]. Accuracies are in (%). The best accuracies are in bold.

	Micucci et al. 2017	This work	Accuracy gains
AF-2	98.71	98.85	+0.14
F-8	78.55	92.00	+13.45
A-9	88.41	95.28	+6.87
AF-17	82.86	93.22	+10.36

- 2) γ is the user-defined maximum number of iterations of the algorithm. In this paper $\gamma = 100$.
- 3) T is the number of vectors in the sequence w . In this paper $T = 151$.
- 4) m is the user-defined number of slices used to split the interval $[0, 100]$. In this paper $m = 50$.

This time cost can be further reduced by gradually reducing the values of parameters like r or γ without negatively impacting the classification results. If the stationary distribution is discovered after r_0 iterations with ($r_0 < r$), this stationary distribution will not change during ($r - r_0$) remaining iterations. Similarly, if the *Baum-Welch* algorithm reaches its local optimum after γ_0 iterations with ($\gamma_0 < \gamma$), this local optimum will not change during the ($\gamma - \gamma_0$) remaining iterations. The experimental value $T = 151$ cannot be modified because it was fixed during the design of the experimental database. Similarly, the experimental value $m = 50$ cannot be changed here because it was fixed after several experiments performed in [97].

2) Experimental time cost

In order to measure the speed-up of the two most time-consuming stages of the processing chain (i.e. the stages of HMM training and the meta-data extraction), we executed and benchmarked the program on two different architectures: a desktop and the *Nvidia JETSON TX2* [99] comparable in terms of hardware to an embedded platform.

The main characteristics of the experimental desktop are the following:

- *CPU: AMD Ryzen 5 5600X (6 Cores) with dedicated core frequency when used at 4649.877 MHz*
- *RAM: 32GB DDR4 3600MHz G SKill Trident Z Neo*
- *GPU: RTX 3080 Gaming X Trio MSI*
- *Motherboard: Asus ROG STRIX B550-F GAMING WiFi*
- *Power supply: RMX 850W Corsair (Certified 89% of efficiency)*

The System-On-a-Chip (SOC) used on the *Nvidia JETSON TX2* has in common 4 cores used on the SOC of the *Exynos 7 Octa* (5433) [100] clocked at 1.9 GHz which is used on the *Galaxy Note 4* and the *Galaxy Tab S2*. The experiment presented in this section has been performed using the 11.771 MCs obtained when the k -means algorithm is executed with $k = 20$ (our smallest experimental value of k).

The *Nvidia Jetson TX2* has 8 GB L128 bit DDR4 of main memory. It runs on L4T (Linux for Tegra, i.e. Linux Kernel 4.9) and has a Parker SOC consisting of:

- 1) A Pascal GPU at 256 Cuda cores (not used in this experiment).
- 2) One HMP (6 cores) including 2 Denver cores (custom core designed by Nvidia to run the ARMv8 ISA) and 4 Arm Cortex A57 cores (to run the ARMv8 ISA). All the cores are compatible with ISA ARMv8 which is the 64 bits architecture of Arm.

The *Nvidia Jetson TX2* has several power supply modes that influence the frequency of the different cores. The modes

used during our tests were the '*Max-N*' mode which allows to reach 2 GHz on each core and the '*Max-P Core-All*' mode allowing each core to be used at 1.4 GHz (i.e. a tradeoff between performance and energy consumption). During the current experiment, the program was executed on cores excluded from the Linux scheduler allowing to dedicate the core entirely to the program and thus, not to distort the performance measurements. The following units were selected to measure the performance of our program using the *Performance Monitoring Unit* and *Syscall*:

- 1) Milliseconds (ms)
- 2) Instructions per cycle (IPC)
- 3) Instructions per second (IPS)
- 4) Floating point operations per second (FLOPS)

Tables 6 and 7 present the performances of our program on the experimental devices. According to these tables, the best performance is obtained by the desktop with a mean time cost of less than 2 seconds. Nevertheless, the performance on *Nvidia Jetson TX2* is also very interesting with a mean time cost of around 12 seconds when the *Core Denver 2 GHz* is used.

Table 6: Time cost in (ms) on the experimental devices

Device	Worst case	Best case	Mean
<i>Core ARM 1.4 GHz</i>	40749.07	882.29	23837.94
<i>Core ARM 2 GHz</i>	28424.88	612.08	16647.94
<i>Core Denver 1.4 GHz</i>	26586.49	457.24	16565.96
<i>Core Denver 2 GHz</i>	18494.03	300.81	11490.03
<i>Desktop 4.6 GHz</i>	1408.53	50.81	1228.7

Table 7: Means of IPC, IPS and FLOPS on the experimental devices

Device	IPC	IPS	FLOPS
<i>Core ARM 1.4 GHz</i>	1.52	2150141967.96	3051739.08
<i>Core ARM 2 GHz</i>	1.52	3079014192.13	4358501.28
<i>Core Denver 1.4 GHz</i>	2.22	3122548664.66	3295845.56
<i>Core Denver 2 GHz</i>	2.23	4509673864.60	4751731.28
<i>Desktop 4.6 GHz</i>	2.15	9995826825.13	98253721.03

A wattmeter was additionally used for measuring the energy consumption in order to deduce the *Average Power Consumption* (APC) of each experimental device. This enabled us to calculate the *Figure Of Merit* (FOM) of each device by multiplying the *Average Execution Time* (AET) by the APC as shown in Equation (8). According to the FOM presented in Table 8, the *Nvidia Jetson TX2* is 2.138 times more efficient than the *Desktop 4.6 GHz* for the execution of our program.

$$\text{Efficiency}(\text{Time}, \text{Power}) = (\text{AET}) \times (\text{APC}) \quad (8)$$

Table 8: APC and FOM

Device	APC	FOM
<i>Desktop 4.6 GHz</i>	120 W	147.444 J
<i>Nvidia Jetson TX2</i>	6 W	68.94018 J

F. MAIN ASSETS

The technique proposed in this paper :

- 1) Considers the sequential variations of spatial locations inside the raw data vectors unlike existing techniques.
- 2) Uses HMMs for the extraction of feature vectors, unlike existing techniques which only use these same models during the classification step.
- 3) Generates feature vectors whose components are interpretable, unlike existing techniques generating hand-crafted or less interpretable feature vectors.
- 4) Performs the feature vectors extraction in reasonable time compared to deep learning techniques.
- 5) Efficiently performs HAR and outperforms prior work for the selected database.
- 6) The algorithm has been demonstrated viable on embedded platform processors from 2014 mobile phones and tablets, namely, *ARM* and *Denver* cores clocked at 1.4GHz and 2GHz. This shows that current mobile phones and tablets would be much closer to the performances of a PC.

VI. CONCLUSION

This paper addresses the problem of HAR based on acceleration data from Smartphones. Existing approaches for this purpose either rely on conventional pattern recognition techniques or on deep learning techniques. Conventional pattern recognition techniques generate shallow handcrafted feature vectors which heavily rely on the human experience/expertise. Deep learning techniques are preferable, but they require lots of computing resources while generating less interpretable feature vectors. The current paper attempts to overcome these limitations by proposing an efficient HMM-based technique that generates interpretable feature vectors while requiring a reasonable time cost with a demonstrated feasibility of implementation on embedded processors, namely, *Denver* and *ARM* cores. More precisely, the sequential variations of spatial locations within each raw data vector are initially captured in Markov chains. These Markov chains are later used for the initialization and the training of HMMs whose meta-data later enable to derive the feature vectors that serve as inputs during the classification phase. The k^{th} component of each feature vector is the overall proportion of time spent by the considered HMM observing the k^{th} symbol over a long time span, irrespective of the state from which this symbol is observed. Four different classification tasks have been tested on the *UniMiB SHAR* dataset containing 17 human activities including 9 types of ADLs and 8 types of falls. Classification results have demonstrated the efficiency of the proposed approach with the best accuracies between 92% and 98.85% for all the classification tasks. This performance is more than 10% better than state of the art for two classification tasks.

The main contribution of the current work is the HMM-based sequential learning of the sample (raw data vector) w associated with a human activity. Meta-data extracted from the resulting HMM λ_w are then used for deriving the

corresponding feature vector \vec{w} . Consequently, the number of samples in the experimental database does not impact the components of \vec{w} since each sample in the database is handled individually, irrespective of the other samples in the database. For this reason, the proposed approach will still exhibit good classification results even for large scale databases. Only the overall computation time for all the samples in the database will increase in these conditions. Parallel computation of all the feature vectors in the data base can also be implemented to reduce this overall computation time.

The current work has a dual impact on further researches in HAR. Firstly, it has been theoretically and experimentally demonstrated that learning a human activity as a sequential process enhances the quality of the resulting feature vectors and consequently, induces better classification results. A lot of the research in HAR only considers discrete activities as opposed to activities in a continuum. The proposed method enables extracting salient features from a stream of data using HMM. Secondly, the proposed method is an advance towards real-time implementation since this method can be efficiently ported on embedded platforms. Indeed, the current work uses HMMs for generating the feature vectors and the resulting feature vectors exhibit good classification performance, even with a basic classifier like the k -NN. Given that efficient hardware implementations of the *Baum-Welch* [101] and the k -NN [102] algorithms on Field-Programmable Gate Array (FPGA) chips are available, this method can therefore be deployed using hardware platforms with a lower footprint than GPUs in terms of energy and using fewer resources on a FPGA due to the simplicity of the implementation compared to CNN or DNN approaches.

References

- [1] Oscar D Lara and Miguel A Labrador. A survey on human activity recognition using wearable sensors. *IEEE communications surveys & tutorials*, 15(3):1192–1209, 2012.
- [2] Yan Wang, Shuang Cang, and Hongnian Yu. A survey on wearable sensor modality centred human activity recognition in health care. *Expert Systems with Applications*, 137:167–190, 2019.
- [3] Chhavi Dhiman and Dinesh Kumar Vishwakarma. A review of state-of-the-art techniques for abnormal human activity recognition. *Engineering Applications of Artificial Intelligence*, 77:21–45, 2019.
- [4] Sarvesh Vishwakarma and Anupam Agrawal. A survey on activity recognition and behavior understanding in video surveillance. *The Visual Computer*, 29(10):983–1009, 2013.
- [5] Leonardo Onofri, Paolo Soda, Mykola Pechenizkiy, and Giulio Iannello. A survey on using domain and contextual knowledge for human activity recognition in video streams. *Expert Systems with Applications*, 63:97–111, 2016.
- [6] Djamilia Romaissa Beddiar, Brahim Nini, Mohammad Sabokrou, and Abdeneur Hadid. Vision-based human activity recognition: a survey. *Multimedia Tools and Applications*, 79(41):30509–30555, 2020.
- [7] Vasundhara Ghate et al. Hybrid deep learning approaches for smartphone sensor-based human activity recognition. *Multimedia Tools and Applications*, pages 1–20, 2021.
- [8] Shihao Xu, Ruizhi Chen, Yue Yu, Guangyi Guo, and Lixiong Huang. Locating smartphones indoors using built-in sensors and wi-fi ranging with an enhanced particle filter. *IEEE Access*, 7:95140–95153, 2019.
- [9] Hojin Ju, So Young Park, and Chan Gook Park. A smartphone-based pedestrian dead reckoning system with multiple virtual tracking for indoor navigation. *IEEE Sensors Journal*, 18(16):6756–6764, 2018.

- [10] Sen Qiu, Zhelong Wang, Hongyu Zhao, Kairong Qin, Zhenglin Li, and Huosheng Hu. Inertial/magnetic sensors based pedestrian dead reckoning by means of multi-sensor fusion. *Information Fusion*, 39:108–119, 2018.
- [11] Jindong Wang, Yiqiang Chen, Shuji Hao, Xiaohui Peng, and Lisha Hu. Deep learning for sensor-based activity recognition: A survey. *Pattern Recognition Letters*, 119:3–11, 2019.
- [12] Amina Jarraya, Amel Bouzeghoub, Amel Borgi, and Khedija Arour. Dcr: A new distributed model for human activity recognition in smart homes. *Expert Systems with Applications*, 140:112849, 2020.
- [13] Ling Bao and Stephen S Intille. Activity recognition from user-annotated acceleration data. In *International conference on pervasive computing*, pages 1–17. Springer, 2004.
- [14] Luciana C Jatoba, Ulrich Grossmann, Christophe Kunze, Jorg Ottenbacher, and Wilhelm Stork. Context-aware mobile health monitoring: Evaluation of different pattern recognition methods for classification of physical activity. In *2008 30th annual international conference of the ieee engineering in medicine and biology society*, pages 5250–5253. IEEE, 2008.
- [15] Tomas Brezmes, Juan-Luis Gorracho, and Josep Cotrina. Activity recognition from accelerometer data on a mobile phone. In *International workshop conference on artificial neural networks*, pages 796–799. Springer, 2009.
- [16] Kerem Altun and Billur Barshan. Human activity recognition using inertial/magnetic sensor units. In *International workshop on human behavior understanding*, pages 38–51. Springer, 2010.
- [17] Muhammad Shoaib, Hans Scholten, and Paul JM Havinga. Towards physical activity recognition using smartphone sensors. In *2013 IEEE 10th international conference on ubiquitous intelligence and computing and 2013 IEEE 10th international conference on autonomic and trusted computing*, pages 80–87. IEEE, 2013.
- [18] Carlos Medrano, Raul Igual, Inmaculada Plaza, and Manuel Castro. Detecting falls as novelties in acceleration patterns acquired with smartphones. *PloS one*, 9(4):e94811, 2014.
- [19] Muhammad Shoaib, Stephan Bosch, Ozlem Durmaz Incel, Hans Scholten, and Paul JM Havinga. Fusion of smartphone motion sensors for physical activity recognition. *Sensors*, 14(6):10146–10176, 2014.
- [20] Jorge-L Reyes-Ortiz, Luca Oneto, Albert Samà, Xavier Parra, and Davide Anguita. Transition-aware human activity recognition using smartphones. *Neurocomputing*, 171:754–767, 2016.
- [21] George Vavoulas, Charikleia Chatzaki, Thodoris Malliotakis, Matthew Pediaditis, and Manolis Tsiknakis. The mobiact dataset: Recognition of activities of daily living using smartphones. In *ICT4AgeingWell*, pages 143–151, 2016.
- [22] Timo Sztyler and Heiner Stuckenschmidt. On-body localization of wearable devices: An investigation of position-aware activity recognition. In *2016 IEEE International Conference on Pervasive Computing and Communications (PerCom)*, pages 1–9. IEEE, 2016.
- [23] Daniela Micucci, Marco Mobilio, and Paolo Napoletano. Unimib shar: a new dataset for human activity recognition using acceleration data from smartphones. *arXiv preprint arXiv:1611.07688v2*, 2017.
- [24] George Vavoulas, Matthew Pediaditis, Charikleia Chatzaki, Emmanouil G Spanakis, and Manolis Tsiknakis. The mobifall dataset: Fall detection and classification with a smartphone. In *Artificial intelligence: Concepts, methodologies, tools, and applications*, pages 1218–1231. IGI Global, 2017.
- [25] José Antonio Santoyo-Ramón, Eduardo Casilar, and José Manuel Cano-García. Analysis of a smartphone-based architecture with multiple mobility sensors for fall detection with supervised learning. *Sensors*, 18(4):1155, 2018.
- [26] Zhen-Yu He and Lian-Wen Jin. Activity recognition from acceleration data using ar model representation and svm. In *2008 international conference on machine learning and cybernetics*, volume 4, pages 2245–2250. IEEE, 2008.
- [27] Zhenyu He and Lianwen Jin. Activity recognition from acceleration data based on discrete cosine transform and svm. In *2009 IEEE International Conference on Systems, Man and Cybernetics*, pages 5041–5044. IEEE, 2009.
- [28] Davide Anguita, Alessandro Ghio, Luca Oneto, Xavier Parra, and Jorge Luis Reyes-Ortiz. A public domain dataset for human activity recognition using smartphones. In *Esann*, volume 3, page 3, 2013.
- [29] Uwe Maurer, Asim Smailagic, Daniel P Siewiorek, and Michael Deisher. Activity recognition and monitoring using multiple sensors on different body positions. In *International Workshop on Wearable and Implantable Body Sensor Networks (BSN'06)*, pages 4–pp. IEEE, 2006.
- [30] Juha Parkka, Miikka Ermes, Panu Korpiapaa, Jani Mantyjarvi, Johannes Peltola, and Ilkka Korhonen. Activity classification using realistic data from wearable sensors. *IEEE Transactions on information technology in biomedicine*, 10(1):119–128, 2006.
- [31] Emmanuel Munguia Tapia, Stephen S Intille, William Haskell, Kent Larson, Julie Wright, Abby King, and Robert Friedman. Real-time recognition of physical activities and their intensities using wireless accelerometers and a heart rate monitor. In *2007 11th IEEE international symposium on wearable computers*, pages 37–40. IEEE, 2007.
- [32] Miikka Ermes, Juha Parkka, and Luc Cluitmans. Advancing from offline to online activity recognition with wearable sensors. In *2008 30th annual international conference of the ieee engineering in medicine and biology society*, pages 4451–4454. IEEE, 2008.
- [33] Jennifer R Kwapisz, Gary M Weiss, and Samuel A Moore. Activity recognition using cell phone accelerometers. *ACM SigKDD Explorations Newsletter*, 12(2):74–82, 2011.
- [34] Oscar D Lara and Miguel A Labrador. A mobile platform for real-time human activity recognition. In *2012 IEEE consumer communications and networking conference (CCNC)*, pages 667–671. IEEE, 2012.
- [35] Kishor H Walse, Rajiv V Dharaskar, and Vilas M Thakare. Pca based optimal ann classifiers for human activity recognition using mobile sensors data. In *Proceedings of First International Conference on Information and Communication Technology for Intelligent Systems: Volume 1*, pages 429–436. Springer, 2016.
- [36] M Humayun Kabir, M Robiul Hoque, Keshav Thapa, and Sung-Hyun Yang. Two-layer hidden markov model for human activity recognition in home environments. *International Journal of Distributed Sensor Networks*, 12(1):4560365, 2016.
- [37] Yen-Ping Chen, Jhun-Ying Yang, Shun-Nan Liou, Gwo-Yun Lee, and Jeen-Shing Wang. Online classifier construction algorithm for human activity detection using a tri-axial accelerometer. *Applied Mathematics and Computation*, 205(2):849–860, 2008.
- [38] Tzu-Ping Kao, Che-Wei Lin, and Jeen-Shing Wang. Development of a portable activity detector for daily activity recognition. In *2009 ieee international symposium on industrial electronics*, pages 115–120. IEEE, 2009.
- [39] Martin Berchtold, Matthias Budde, Dawud Gordon, Hedda R Schmidtke, and Michael Beigl. Actiserv: Activity recognition service for mobile phones. In *International symposium on wearable computers (ISWC) 2010*, pages 1–8. IEEE, 2010.
- [40] David Minnen, Tracy Westeyn, Daniel Ashbrook, Peter Presti, and Thad Starner. Recognizing soldier activities in the field. In *4th International workshop on wearable and implantable body sensor networks (BSN 2007)*, pages 236–241. Springer, 2007.
- [41] Oscar D Lara, Alfredo J Pérez, Miguel A Labrador, and José D Posada. Centinela: A human activity recognition system based on acceleration and vital sign data. *Pervasive and mobile computing*, 8(5):717–729, 2012.
- [42] Chun Zhu and Weihua Sheng. Human daily activity recognition in robot-assisted living using multi-sensor fusion. In *2009 ieee international conference on robotics and automation*, pages 2154–2159. IEEE, 2009.
- [43] Mohammad Abu Alsheikh, Ahmed Selim, Dusit Niyato, Linda Doyle, Shaowei Lin, and Hwee-Pink Tan. Deep activity recognition models with triaxial accelerometers. *arXiv preprint arXiv:1511.04664*, 2015.
- [44] Licheng Zhang, Xihong Wu, and Dingsheng Luo. Recognizing human activities from raw accelerometer data using deep neural networks. In *2015 IEEE 14th International Conference on Machine Learning and Applications (ICMLA)*, pages 865–870. IEEE, 2015.
- [45] Sarah Fallmann and Johannes Kropf. Human activity recognition of continuous data using hidden markov models and the aspect of including discrete data. In *2016 Intl IEEE Conferences on Ubiquitous Intelligence & Computing, Advanced and Trusted Computing, Scalable Computing and Communications, Cloud and Big Data Computing, Internet of People, and Smart World Congress (UIC/ATC/ScalCom/CBDCom/IoP/SmartWorld)*, pages 121–126. IEEE, 2016.
- [46] Mehmet Onur Padar, Ali Erdem Ertan, and Çağatay Candan. Classification of human motion using radar micro-doppler signatures with hidden markov models. In *2016 IEEE Radar Conference (RadarConf)*, pages 1–6. IEEE, 2016.
- [47] Praneeth Vepakomma, Debraj De, Sajal K Das, and Shekhar Bhansali. A-wristocracy: Deep learning on wrist-worn sensing for recognition of user complex activities. In *2015 IEEE 12th International conference on*

- wearable and implantable body sensor networks (BSN), pages 1–6. IEEE, 2015.
- [48] Licheng Zhang, Xihong Wu, and Dingsheng Luo. Human activity recognition with hmm-dnn model. In 2015 IEEE 14th International Conference on Cognitive Informatics & Cognitive Computing (ICCI*CC), pages 192–197. IEEE, 2015.
 - [49] Nils Y Hammerla, Shane Halloran, and Thomas Plötz. Deep, convolutional, and recurrent models for human activity recognition using wearables. arXiv preprint arXiv:1604.08880, 2016.
 - [50] Shuai Zhang, Wing WY Ng, Jianjun Zhang, and Chris D Nugent. Human activity recognition using radial basis function neural network trained via a minimization of localized generalization error. In International Conference on Ubiquitous Computing and Ambient Intelligence, pages 498–507. Springer, 2017.
 - [51] Ming Zeng, Le T Nguyen, Bo Yu, Ole J Mengshoel, Jiang Zhu, Pang Wu, and Joy Zhang. Convolutional neural networks for human activity recognition using mobile sensors. In 6th International Conference on Mobile Computing, Applications and Services, pages 197–205. IEEE, 2014.
 - [52] Yi Zheng, Qi Liu, Enhong Chen, Yong Ge, and J Leon Zhao. Time series classification using multi-channels deep convolutional neural networks. In International conference on web-age information management, pages 298–310. Springer, 2014.
 - [53] Yuqing Chen and Yang Xue. A deep learning approach to human activity recognition based on single accelerometer. In 2015 ieee international conference on systems, man, and cybernetics, pages 1488–1492. IEEE, 2015.
 - [54] Sojeong Ha, Jeong-Min Yun, and Seungjin Choi. Multi-modal convolutional neural networks for activity recognition. In 2015 IEEE International conference on systems, man, and cybernetics, pages 3017–3022. IEEE, 2015.
 - [55] Wenchao Jiang and Zhaozheng Yin. Human activity recognition using wearable sensors by deep convolutional neural networks. In Proceedings of the 23rd ACM international conference on Multimedia, pages 1307–1310, 2015.
 - [56] Jianbo Yang, Minh Nhut Nguyen, Phyto Phyto San, Xiaoli Li, and Shonali Krishnaswamy. Deep convolutional neural networks on multichannel time series for human activity recognition. In Ijcai, volume 15, pages 3995–4001. Buenos Aires, Argentina, 2015.
 - [57] Yuwen Chen, Kunhua Zhong, Ju Zhang, Qilong Sun, and Xueliang Zhao. Lstm networks for mobile human activity recognition. In 2016 International conference on artificial intelligence: technologies and applications, pages 50–53. Atlantis Press, 2016.
 - [58] Hristijan Gjoreski, Jani Bizjak, Martin Gjoreski, and Matjaž Gams. Comparing deep and classical machine learning methods for human activity recognition using wrist accelerometer. In Proceedings of the IJCAI 2016 Workshop on Deep Learning for Artificial Intelligence, New York, NY, USA, volume 10, page 970, 2016.
 - [59] Sojeong Ha and Seungjin Choi. Convolutional neural networks for human activity recognition using multiple accelerometer and gyroscope sensors. In 2016 International Joint Conference on Neural Networks (IJCNN), pages 381–388. IEEE, 2016.
 - [60] Cihang Liu, Lan Zhang, Zongqian Liu, Kebin Liu, Xiangyang Li, and Yunhao Liu. Lasagna: towards deep hierarchical understanding and searching over mobile sensing data. In Proceedings of the 22nd Annual International Conference on Mobile Computing and Networking, pages 334–347, 2016.
 - [61] Francisco Javier Ordóñez Morales and Daniel Roggen. Deep convolutional feature transfer across mobile activity recognition domains, sensor modalities and locations. In Proceedings of the 2016 ACM International Symposium on Wearable Computers, pages 92–99, 2016.
 - [62] Daniele Ravi, Charence Wong, Benny Lo, and Guang-Zhong Yang. Deep learning for human activity recognition: A resource efficient implementation on low-power devices. In 2016 IEEE 13th international conference on wearable and implantable body sensor networks (BSN), pages 71–76. IEEE, 2016.
 - [63] Daniele Ravi, Charence Wong, Benny Lo, and Guang-Zhong Yang. A deep learning approach to on-node sensor data analytics for mobile or wearable devices. IEEE journal of biomedical and health informatics, 21(1):56–64, 2016.
 - [64] Charissa Ann Ronao and Sung-Bae Cho. Human activity recognition with smartphone sensors using deep learning neural networks. Expert systems with applications, 59:235–244, 2016.
 - [65] Jie Wang, Xiao Zhang, Qinhuai Gao, Hao Yue, and Hongyu Wang. Device-free wireless localization and activity recognition: A deep learning approach. IEEE Transactions on Vehicular Technology, 66(7):6258–6267, 2016.
 - [66] Tahmina Zebin, Patricia J Scully, and Krikor B Ozanyan. Human activity recognition with inertial sensors using a deep learning approach. In 2016 IEEE SENSORS, pages 1–3. IEEE, 2016.
 - [67] Yi Zheng, Qi Liu, Enhong Chen, Yong Ge, and J Leon Zhao. Exploiting multi-channels deep convolutional neural networks for multivariate time series classification. Frontiers of Computer Science, 10(1):96–112, 2016.
 - [68] Youngwook Kim and Yang Li. Human activity classification with transmission and reflection coefficients of on-body antennas through deep convolutional neural networks. IEEE Transactions on Antennas and Propagation, 65(5):2764–2768, 2017.
 - [69] Song-Mi Lee, Sang Min Yoon, and Heeryon Cho. Human activity recognition from accelerometer data using convolutional neural network. In 2017 ieee international conference on big data and smart computing (bigcomp), pages 131–134. IEEE, 2017.
 - [70] Shoail Mohammed and Ivan Tashev. Unsupervised deep representation learning to remove motion artifacts in free-mode body sensor networks. In 2017 IEEE 14th International Conference on Wearable and Implantable Body Sensor Networks (BSN), pages 183–188. IEEE, 2017.
 - [71] Madhuri Panwar, S Ram Dyuthi, K Chandra Prakash, Dwaipayan Biswas, Amit Acharyya, Koushik Maharana, Arvind Gautam, and Ganesh R Naik. Cnn based approach for activity recognition using a wrist-worn accelerometer. In 2017 39th Annual International Conference of the IEEE Engineering in Medicine and Biology Society (EMBC), pages 2438–2441. IEEE, 2017.
 - [72] Shuochao Yao, Shaohan Hu, Yiran Zhao, Aston Zhang, and Tarek Abdelzaher. Deepsense: A unified deep learning framework for time-series mobile sensing data processing. In Proceedings of the 26th International Conference on World Wide Web, pages 351–360, 2017.
 - [73] Mingtao Dong, Jindong Han, Yuan He, and Xiaojun Jing. Har-net: Fusing deep representation and hand-crafted features for human activity recognition. In International Conference On Signal And Information Processing, Networking And Computers, pages 32–40. Springer, 2018.
 - [74] Fernando Moya Rueda, René Grzesick, Gernot A Fink, Sascha Feldhorst, and Michael Ten Hompel. Convolutional neural networks for human activity recognition using body-worn sensors. In Informatics, volume 5, page 26. Multidisciplinary Digital Publishing Institute, 2018.
 - [75] Shaohua Wan, Liyanqong Qi, Xiaolong Xu, Chao Tong, and Zonghua Gu. Deep learning models for real-time human activity recognition with smartphones. Mobile Networks and Applications, 25(2):743–755, 2020.
 - [76] Kun Xia, Jianguang Huang, and Hanyu Wang. Lstm-cnn architecture for human activity recognition. IEEE Access, 8:56855–56866, 2020.
 - [77] Marcus Edel and Enrico Köppe. Binarized-blstm-rnn based human activity recognition. In 2016 International conference on indoor positioning and indoor navigation (IPIN), pages 1–7. IEEE, 2016.
 - [78] Abdulmajid Murad and Jae-Young Pyun. Deep recurrent neural networks for human activity recognition. Sensors, 17(11):2556, 2017.
 - [79] Yu Guan and Thomas Plötz. Ensembles of deep lstm learners for activity recognition using wearables. Proceedings of the ACM on Interactive, Mobile, Wearable and Ubiquitous Technologies, 1(2):1–28, 2017.
 - [80] Cheng Xu, Duo Chai, Jie He, Xiaotong Zhang, and Shihong Duan. Innohar: A deep neural network for complex human activity recognition. Ieee Access, 7:9893–9902, 2019.
 - [81] Thomas Plötz, Nils Y Hammerla, and Patrick L Olivier. Feature learning for activity recognition in ubiquitous computing. In Twenty-second international joint conference on artificial intelligence, 2011.
 - [82] Hongqing Fang and Chen Hu. Recognizing human activity in smart home using deep learning algorithm. In Proceedings of the 33rd chinese control conference, pages 4716–4720. IEEE, 2014.
 - [83] Tomoki Hayashi, Masafumi Nishida, Norihide Kitaoka, and Kazuya Takeda. Daily activity recognition based on dnn using environmental sound and acceleration signals. In 2015 23rd European Signal Processing Conference (EUSIPCO), pages 2306–2310. IEEE, 2015.
 - [84] Nicholas D Lane and Petko Georgiev. Can deep learning revolutionize mobile sensing? In Proceedings of the 16th International Workshop on Mobile Computing Systems and Applications, pages 117–122, 2015.
 - [85] Licheng Zhang, Xihong Wu, and Dingsheng Luo. Real-time activity recognition on smartphones using deep neural networks. In 2015 IEEE 12th Intl Conf on Ubiquitous Intelligence and Computing and 2015 IEEE 12th Intl Conf on Autonomic and Trusted Computing and 2015 IEEE 15th Intl Conf on Scalable Computing and Communications and Its

- Associated Workshops (UIC-ATC-ScalCom), pages 1236–1242. IEEE, 2015.
- [86] Sourav Bhattacharya and Nicholas D Lane. From smart to deep: Robust activity recognition on smartwatches using deep learning. In 2016 IEEE International conference on pervasive computing and communication workshops (PerCom Workshops), pages 1–6. IEEE, 2016.
- [87] Valentin Radu, Nicholas D Lane, Sourav Bhattacharya, Cecilia Mascolo, Mahesh K Marina, and Fahim Kawsar. Towards multimodal deep learning for activity recognition on mobile devices. In Proceedings of the 2016 ACM International Joint Conference on Pervasive and Ubiquitous Computing: Adjunct, pages 185–188, 2016.
- [88] Bandar Almaslukh, Jalal AlMuhtadi, and Abdelmanim Artoli. An effective deep autoencoder approach for online smartphone-based human activity recognition. *Int. J. Comput. Sci. Netw. Secur.*, 17(4):160–165, 2017.
- [89] Wojciech Samek, Grégoire Montavon, Sebastian Lapuschkin, Christopher J Anders, and Klaus-Robert Müller. Explaining deep neural networks and beyond: A review of methods and applications. *Proceedings of the IEEE*, 109(3):247–278, 2021.
- [90] Emilio Sansano, Raúl Montoliu, and Óscar Belmonte Fernández. A study of deep neural networks for human activity recognition. *Computational Intelligence*, 36(3):1113–1139, 2020.
- [91] Frédéric Li, Kimiaki Shirahama, Muhammad Adeel Nisar, Lukas Köping, and Marcin Grzegorzek. Comparison of feature learning methods for human activity recognition using wearable sensors. *Sensors*, 18(2):679, 2018.
- [92] Sepidehsadat Hosseini, Seok Hee Lee, and Nam Ik Cho. Feeding hand-crafted features for enhancing the performance of convolutional neural networks. *arXiv preprint arXiv:1801.07848*, 2018.
- [93] Roshan Singh, Ankur Sonawane, and Rajeev Srivastava. Recent evolution of modern datasets for human activity recognition: a deep survey. *Multimedia Systems*, pages 1–24, 2019.
- [94] Lawrence R Rabiner. A tutorial on hidden markov models and selected applications in speech recognition. *Proceedings of the IEEE*, 77(2):257–286, 1989.
- [95] Sylvain Illoga. Customizable hmm-based measures to accurately compare tree sets. *Pattern Analysis and Applications*, pages 1–22, 2021.
- [96] Aristidis Likas, Nikos Vlassis, and Jakob J Verbeek. The global k-means clustering algorithm. *Pattern recognition*, 36(2):451–461, 2003.
- [97] Sylvain Illoga, Olivier Romain, and Maurice Tchuenté. An accurate hmm-based similarity measure between finite sets of histograms. *Pattern Analysis and Applications*, 22(3):1079–1104, 2018.
- [98] Witten Ian H. and Frank Eibe. Data mining: Practical machine learning tools and techniques. <http://weka.sourceforge.net/>, 2005.
- [99] Nvidia jetson tx2 user manual. <https://www.manualslib.com/manual/1634693/Nvidia-Jetson-Tx2.html>.
- [100] Mobile processor exynos 7 octa (5433). <https://www.samsung.com/semiconductor/minisite/exynos/products/mobileprocessor/exynos-7-octa-5433/>.
- [101] A. López-López A. Espinosa-Manzo and M. O. Arias-Estrada. Implementing hidden markov models in a hardware architecture. In Proceedings of the International Meeting of Computer Science (ENC'01), Aguascalientes, Mexico, volume II, pages 1007–1016, 2001.
- [102] Alec Lu, Zhenman Fang, Nazanin Farahpour, and Lesley Shannon. Chip-knn: A configurable and high-performance k-nearest neighbors accelerator on cloud fpgas. In 2020 International Conference on Field-Programmable Technology (ICFPT), pages 139–147. IEEE, 2020.



SYLVAIN ILOGA obtained his PhD in Computer Science in January 2018 at the University of Yaoundé 1 in Cameroon. Since January 2010, he has been a teacher in the department of Computer Science of the High Teachers' Training College of Maroua in Cameroon. Between September 2017 and August 2019, he completed a research and teaching internship at the Department of Electronic Engineering and Industrial Computing of the IUT of Cergy-Pontoise at Neuville University, in France.

In May 2018, he was promoted to the rank of Lecturer in Cameroon. Subsequently in January 2019, he obtained his qualification for the functions of Lecturer in France, section 27 (Computer Science). His research focuses on the design of taxonomies for hierarchical classification, sequential data mining, machine learning using hidden Markov models, and the implementation of reconfigurable architectures based on the FPGA technology.



ALEXANDRE BORDAT is an Engineer of Embedded System graduated from ENSEA (École nationale supérieure de l'électronique et de ses applications) in France. He is currently preparing a PhD thesis in embedded system in this same school.



JULIEN LE KERNEC received the B.Eng. and M.Eng. degrees in electronic engineering from the Cork Institute of Technology, Ireland, in 2004 and 2006, respectively, and the Ph.D. degree in electronic engineering from University Pierre and Marie Curie, France, in 2011. He is currently a Senior Lecturer with the School of Engineering, University of Glasgow. He is also a Senior Lecturer with the University of Electronic Science and Technology of China and an Adjunct Associate Professor with the ETIS Laboratory, University of Cergy-Pontoise, France. His research interests include radar system design, software-defined radio/radar, signal processing, and health applications.



OLIVIER ROMAIN obtained his engineering degree from ENS Cachan, then his Masters from Louis Pasteur University, and his PhD from Pierre and Marie Curie University in Paris. All these diplomas were obtained in Electronics. Between 2012 and 2019 he was the head of the department of Architecture of the ETIS-UMR8051 laboratory and since January 2020 he is the director of this laboratory. In addition, he is University Professor in Electrical Engineering at CY Cergy Paris University. His research focuses mainly on systems on chips for diffusion and on biomedical applications.