Exploiting Local Temporal Characteristics via Multinomial Decomposition Algorithm for Real-Time Activity Recognition
Manuel Abbas, Régine Le Bouquin Jeannes

To cite this version:

HAL Id: hal-03412612
https://univ-rennes.hal.science/hal-03412612
Submitted on 3 Nov 2021

HAL is a multi-disciplinary open access archive for the deposit and dissemination of scientific research documents, whether they are published or not. The documents may come from teaching and research institutions in France or abroad, or from public or private research centers.

L’archive ouverte pluridisciplinaire HAL, est destinée au dépôt et à la diffusion de documents scientifiques de niveau recherche, publiés ou non, émanant des établissements d’enseignement et de recherche français ou étrangers, des laboratoires publics ou privés.

Distributed under a Creative Commons Attribution - NonCommercial 4.0 International License
Exploiting Local Temporal Characteristics via Multinomial Decomposition Algorithm for Real-time Activity Recognition

Manuel Abbas and Régine Le Bouquin Jeannès, Member, IEEE

Abstract—With the current expansion of Internet of Things (IoT), tracking human physical activity through wearable devices is becoming increasingly frequent. A human activity recognition (HAR) module is needed to localize and classify the patterns of acquired signals. Although this topic has been a talking point in the past decades, the trade-off between efficiency, reliability, and computational complexity is still an open research challenge. In this work, we propose a novel recognition process, based on online feature vector computation followed by a multinomial decomposition algorithm (MDA). Specifically, the temporal characteristics of performed activities are encoded over local segments. Afterwards, a low-cost algorithm divides the multinomial classification into different stages, where the encoded patterns feed a neural network at each stage to classify the corresponding segment. The proposed approach has been evaluated on five public datasets and compared with a set of state-of-the-art methods. Experimental results show that with a few hundreds of floating point operations per second (flops), our solution can match the performance in terms of accuracy. The effects of the sliding window length, the sampling frequency, the size of the training set, and overfitting have been studied. A prototype was developed to display the behavior of this solution in real world conditions.

Index Terms—Human activity recognition, wearable sensors, local temporal characteristics, online computation, multinomial decomposition, computational complexity.

I. INTRODUCTION

HUMAN activity recognition (HAR) is a wide field of research, and is currently one of the most challenging topics. The goal is to reveal and predict the successive movements of a subject, using a 4-step process, namely (a) data acquisition, (b) data segmentation, (c) feature extraction, and (d) classification. The identification of human activities serves a broad range of applications in different domains. For instance, HAR-based techniques are an instrument for assisted living approaches and smart homes [1]. Moreover, monitoring certain activities of daily living (ADLs) is the cornerstone of healthcare applications [2]. Besides, motion detection is also used in security and surveillance systems [3], in order to assist human operators and improve their work.

HAR has been a trending research area in the past decades, and continues to be an active topic [4], [5]. It can be divided into two branches, namely (i) wearable sensor-based systems [6] and (ii) ambient sensor-based systems, using cameras for data acquisition [7]–[9]. The former are light-weight devices consisting of miniaturized sensors like accelerometers, gyroscopes, magnetometers, and barometers. Furthermore, soft wearable sensors are frequently employed in the context of gesture recognition [10]. They are a solution for gait analysis [11], ensuring long-term monitoring, as well as movement tracing and rehabilitation [12]. In terms of privacy, wearable trackers do not reveal the subject’s identity, thus have an advantage over the second category. Moreover, since they are body-worn, they supply data anywhere continuously, unlike ambient sensors which suffer from visibility issues and are shortened to a certain environment. Besides, camera-based HAR has its limitations like diversity in camera views, variety in environmental conditions, and variety of human subjects [13]. Ideally, the activity is identified regardless the environment and/or the performing person. Hence, wearable sensors are considered in our study.

Now, inertial measurement units (IMUs) and magnetic angular rate and gravity (MARG) sensors are usually employed for inertial human motion tracking [14]. One important factor influencing the development of a recognition process is the power consumption of the device. It is worth noting that the current consumed by a gyroscope is huge compared to other sensors [15]. Furthermore, the reliability of a magnetometer in indoor environments could be questionable, especially with the existence of artificial fields provided by different sources like smartphones and electrical materials. Another important factor is the sampling frequency ($F_s$) of the sensors. While a lower sampling frequency deteriorates the shape of the windowed signal, a higher one increases the computational load of the system and could not provide low latency for real-time prediction. A third factor is the length $L$ of the window over which data are processed to predict the performed activity. Generally, HAR systems should automatically localize the targeted ADLs under unsupervised conditions to satisfy the aforementioned applications. Therefore, while larger windows may inaccurately delimit the targeted activity by containing unrelated movements, smaller windows may exclude informative segments for the recognition.

Considering all the aforementioned elements, we propose a low-cost and highly accurate acceleration-based HAR approach. Although the number of degrees of freedom (DOF) is reduced after excluding the gyroscope and the magnetometer from the acquisition unit, data acquired with...
Wearable sensors placement: several body placements have been considered in the literature, such as the waist, the wrist, the ankle, and so on. Moreover, multi-positional body sensing was also considered [22]. The body trunk, particularly the waist or the neck, is the best position while employing one sensing device when it comes to reliability [18], [23], [37]. Additionally, the trunk acceleration patterns are meaningful for post-processing units (like postural transitions for example), unlike the wrist or the ankle which generate complex and random movements.

Feature extraction and classification: the majority of HAR modules are based on machine learning (ML) techniques [24]. Most of these methods rely on the extraction of handcrafted features from acquired data, which feed a classifier to make a decision. In the literature, time-domain and frequency-domain features were proposed such as mean, median absolute deviation, interquartile range, entropy [25]–[27], as well as the Fast Fourier Transform (FFT) coefficients and spectral power [28], [29] to name a few. These features feed a classifier to predict the activity over a window. Some low-cost classifiers, like decision trees (DT) and support vector machines (SVM) with linear/quadratic kernel, may be preferred to memory-based ones such as \( k \)-nearest neighbors (KNN), which raise the computational complexity of the system. These features, which are computed globally over windowed signals, encode the statistical patterns of these signals but not their temporal structures, which may lead to undesirable confusion between different classes. On the other hand, deep learning techniques were also implemented to develop HAR systems, where feature extraction is done automatically from raw signals to learn and make intelligent decisions. For instance, convolutional neural networks (CNN) [30], [33] and recurrent neural networks (RNN) [31] were proposed. The long short-term memory (LSTM) network, which is a type of recurrent networks, was also developed for this purpose [32]. LSTM addresses the vanishing/exploding gradient problem that can happen with RNNs. However, its reliability is affected by the size of the training set. Therefore, a third architecture of recurrent networks has been proposed, namely gated recurrent unit (GRU) network, which (i) is faster than LSTM, and (ii) addresses the vanishing/exploding gradient problem that can happen with RNNs. However, its reliability is affected by the size of the training set. Therefore, a third architecture of recurrent networks has been proposed, namely gated recurrent unit (GRU) network, which (i) is faster than LSTM, and (ii) can outperform it [34]. More recently, Chen et al. proposed a feature fusion technique [35]. It consists of two separate modules: (a) handcrafted features followed by fully connected layers, and (b) automatic feature learning (deep learning). Both outputs are then concatenated to predict the activity.

Sampling frequency \( F_s \) and window length \( L \): these parameters received little attention in previous studies. These two quantities, which determine the number of points in time-series (acceleration signals), affect the feature computation (handcrafted features or those learned automatically), thus the accuracy of HAR systems.

Computational complexity and embeddability: computational complexity is a key-point to consider. According to the limited power resources of a micro-controller, the feature computation and the selected classifier should require a reasonable number of flops. The computation of certain mathematical operations like exponential needs a software library instead of a hardware implementation like the addition or the multiplication. This
might increase the latency of the system. The trade-off between efficiency and computational complexity remains an open research challenge.

### B. Problem Formulation

The proposed approach aims at analyzing (i) the postural transitions, (ii) cyclic/recurrent movements, and (iii) inactivity periods. Hence, our method considers basic ADLs, often encountered in daily routine of people of diverse profiles, namely (1) sitting down (walk-to-sit and stand-to-sit), (2) standing up (sit-to-stand and sit-to-walk), (3) lying down (sit-to-lying and stand-to-lying), (4) rising up (lie-to-sitting and lie-to-stand), (5) walking (same level, climbing up, climbing down), (6) jogging, and (7) rest (standing, sitting, lying). For real-time recognition, a sliding window cuts the signals to make a decision, which may result in some windowed data representing random movements occurring in free-living conditions but not targeted by the system (section II-A), like the beginning/end of a cyclic activity (i.e. walk-to-stand, stand-to-walk, start/stop jogging). These movements constitute an eighth class called (8) others to be filtered out.

The goal behind the proposed model is to regroup the targeted activities into different clusters at each level, creating a tree structure classification process. This strategy aims at increasing the discrimination power of the system, and thus reduces the confusion between different output classes. Fig. 1 displays the proposed subdivision to label the movement. Firstly, the ADLs, which constitute the output of the proposed approach, are subject to a first-level classification into (i) inactive (i.e. rest) and (ii) active. If active, the second-level category is either (ii-a) change of trunk orientation (lying-down/rising-up), (ii-b) sudden change of movements intensity (rapid change in acceleration level), and (ii-c) the remaining movements. The remaining ADLs can be separated into cyclic (i.e. recurrent) and transient movements. ADLs reaching the cyclic node are either walking or jogging. The weightlessness state oc-curring when the human body leaves the floor (for a short time) while jogging, and the variability of acceleration signals, are two useful quantities to discriminate between these two ADLs at this stage. Lastly, ADLs reaching the transient node are either sitting down, standing up, or some random non-postural transition. The up-down movements are symbolized by a sine wave in acceleration magnitude $||a||$.

This phenomenon is scientifically interpreted in Appendix A. Moreover, the acceleration component directed towards the trunk fluctuates when the subject sits/stands. This is not the case for other similar non-postural transitions. Fig. 2 illustrates the acceleration patterns of sitting-down and standing-up.

### III. MATERIALS AND METHODS

This section presents the full process of the proposed HAR mechanism. It first begins by the description of local features to be extracted from acceleration signals. Next, an online feature vector computation approach is proposed to reduce the complexity of the system. Finally, the novel multinomial decomposition technique is detailed, which proves to be a low-cost and highly accurate solution. Throughout the rest of the paper, the acceleration magnitude $||a||$ is defined as follows:

$$||a|| = \sqrt{a_X^2 + a_Y^2 + a_Z^2}$$

where $a_X$ is the vertical component (when the subject is standing), $a_Y$ is the lateral component, and $a_Z$ is pointing towards the human’s trunk.
These two variables, when computed globally over the window as a whole (from end to end), determine the statistical patterns of a signal. Now, to encode the temporal structures of the windowed data, we computed these two features over local segments. Specifically, $a_X$ and $a_Z$ are divided into equal non-overlapping sub-segments of 1-second length to calculate their mean values. Meanwhile, $||a||$ is divided into 500-ms segments to calculate its local mean values. Successive local mean values reveal, in some way, the shape/pattern of the signal. Hence, $\mu$ is computed over smaller fragments for acceleration magnitude (500 ms for $||a||$ vs 1 s for $a_X$ and $a_Z$), since it contains faster transitions, as shown in Fig. 2 (fast sine wave transition in $||a||$ vs slower deviation in $a_Z$). As for the intensity of movements, local SD values are calculated over 1-second segments of $||a||$, seeing that $\sigma$ is sensitive to random fluctuations over very small sub-segments (like 500 ms), which might raise the confusion between the output classes.

Consequently, the succession of these mean values reveals the trunk orientation, the up-down movements, and locates the weightlessness states and the deviation in a signal. The succession of SD values detects the sudden change in a signal. Consequently, the succession of these mean values reveals the trunk orientation, the up-down movements, and locates the weightlessness states and the deviation in a signal. The succession of SD values detects the sudden change in a signal.

**Algorithm 1 Online Feature Computation**

1: Define: $L$, $F_s$, $\text{counter} = 0$
2: Initialize: $\Lambda[i], \Upsilon[i]$, $\lambda$, $\rho$ are set to 0
3: while true do
4: \hspace{1em} Read acquired acceleration value $V_i$
5: \hspace{1em} $\nu \leftarrow V_i$
6: \hspace{1em} Shift values $\Lambda[i]$; $\Lambda[n] \leftarrow \nu$
7: \hspace{1em} $\lambda_i \leftarrow \lambda_i - \Lambda[i] + \Lambda[n]
8: \hspace{1em} $\nu \leftarrow V_i^2$
9: \hspace{1em} Shift values $\Upsilon[i]$; $\Upsilon[n] \leftarrow \nu$
10: \hspace{1em} $\rho_i \leftarrow \rho_i - \Upsilon[i] + \Upsilon[n]
11: \hspace{1em} $\text{counter} \leftarrow \text{counter} + 1$
12: if $\text{counter} = F_s$ then
13: \hspace{1em} Calculate $\mu_i$ and $\sigma_i$ according to Eqs. (4) and (5)
14: \hspace{1em} $\text{counter} \leftarrow 0$

making a decision. With each new acquisition, i.e. when the accelerometer provides a new data-point, a new value $\nu$ is introduced by shifting $\zeta[i]$ as indicated in Fig. 3 and the values $\chi_i$ are updated: $\chi_i \leftarrow \chi_i - \zeta[i] + \chi_i[n]$. Now, let us detail the aforementioned operations.

**Mean values:** we consider a sliding window of length $L$ and a sampling frequency equal to $F_s$. Moreover, buffers $\Lambda[i]$ are considered to store acquired acceleration data-points $V_i$. $\Lambda_i = \sum_{j=1}^{n} \Lambda[j]$ is updated with each new acquisition. In this case, we get:

$$\nu \leftarrow V_i ; \ \zeta[i] \leftarrow \Lambda[i] ; \ \chi_i \leftarrow \lambda_i$$

For x-component and z-component, $m = L$ and $n = F_s$, whereas for $||a||$, $m = 2L$ and $n = F_s^2$. Consequently, at each decision, the mean value of the $i$-th segment is computed based on $\lambda_i$ and $c = \frac{1}{n}$ (Eq. (2)):

$$\mu_i = c \times \lambda_i$$ (4)

**SD values:** in addition to $\Lambda[i]$, new buffers $\Upsilon[i]$ are considered to store $V_i^2$ values. Therefore, $\rho_i = \sum_{j=1}^{n} \Upsilon[i]$ is updated with each new acquisition. Hence, we get:

$$\nu \leftarrow V_i^2 ; \ \zeta[i] \leftarrow \Upsilon[i] ; \ \chi_i \leftarrow \rho_i$$

Here, $||a||$ is involved, with $m = L$ and $n = F_s$. Consequently, at each decision, the SD of the $i$-th segment ($i$-th second) is calculated based on $\rho_i$, $\lambda_i$, and $c$ (Eq. (3)):

$$\sigma_i = \sqrt{c(\rho_i - c\lambda_i^2)}$$ (5)

**TABLE I**

<table>
<thead>
<tr>
<th>Symbol</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>$V_i$</td>
<td>Raw acceleration value (acquired data)</td>
</tr>
<tr>
<td>$\Lambda[i]$</td>
<td>A buffer to store raw acceleration values $V_i$</td>
</tr>
<tr>
<td>$\lambda_i = \sum_{j=1}^{n} \Lambda[j]$</td>
<td>Sum of acceleration values ($i$th second)</td>
</tr>
<tr>
<td>$\rho_i = \sum_{j=1}^{n} \Upsilon[i]$</td>
<td>A buffer to store squared acceleration values $V_i^2$</td>
</tr>
<tr>
<td>$\mu_i$, $\sigma_i$</td>
<td>Mean and SD value of the $i$th second respectively</td>
</tr>
</tbody>
</table>

**Fig. 3.** The process consists of $m$ buffers $\zeta[i]$ of length $n$. The red arrows indicate the propagation of acquired data-points. For each new read value $\nu$, the $\zeta[i]$ values are shifted by one sample, $\zeta[i]$ is excluded, and the elements $\lambda_i$, $\rho_i$, $\mu_i$, $\sigma_i$ are updated as indicated in the figure.
TABLE I summarizes the mathematical symbols, and Algorithm II provides the pseudo-code of this process.

C. Multinomial Decomposition Algorithm

With online extraction, 5L local features are calculated automatically in parallel with data acquisition, then stored in an array. Suppose that \( F \) is the \( L \)-element of \( F \), the extracted features are as follows (see Fig. 4):
- \( \mu \) values of \( a_X \) over each second in the \( L \)-second sliding window (denoted by \( F_{1\to L} \), i.e. first \( L \) elements of \( F \))
- \( \sigma \) values of \( ||a|| \) over each second in the \( L \)-second sliding window (denoted by \( F_{L+1\to2L} \))
- \( \mu \) values of \( ||a|| \) over each 500 ms in the \( L \)-second sliding window (denoted by \( F_{2L+1\to4L} \))
- \( \mu \) values of \( a_Z \) over each second in the \( L \)-second sliding window (denoted by \( F_{3L+1\to5L} \))

We now introduce the MDA, which exploits the aforementioned features to predict ADLs.

ML Classifier: an artificial Neural Network (NN) is called at each level of the tree-structure classification process to make a prediction, until it reaches a leaf. This NN consists of one hidden layer of 10 neurons, with a customized transfer function \( h'(x) \) detailed in Appendix II. It is derived from the hyperbolic tangent sigmoid \( h(x) \), after introducing an expansion of exponential to reduce its computational complexity. Fig. 5 illustrates both functions, with the corresponding gap being displayed using double head arrows. Although the exponential increases faster as \( x \) increases, both curves have similar shapes and thus produce similar results (in terms of classification). It was experimentally shown (using parallel loops) that the logarithm operation is almost 25 times more complex than a multiplication. Finally, the output layer consists of a linear function.

Classification by stages: MDA follows the logic of section II-B and is illustrated, in its optimized form, in Fig. 6. The SD values of \( ||a|| \) \( (F_{L+1\to2L}) \) feed the predefined NN to assign a first-level label to the windowed data (S1), namely (I) Inactivity, (II) Cyclic, and (III) Transient. (I) is a leaf, and its corresponding output is “rest”. Next, mean values of \( a_X \) and SD values of \( ||a|| \) \( (F_{1\to2L}) \) feed the NN to assign a second-level label (S2), namely (a) Orientation, (b) Intensity, and (c) None. The ADL reaching (a) is either “lying down” or “rising up”, depending on the direction of the transition, whereas that reaching (b) is filtered out. Finally, the third-level stage (S3) depends on the previous attributes. For ADLs belonging to (II-c), a third NN, whose inputs are the mean and SD values of \( ||a|| \) \( (F_{1\to4L}) \) is applied, to see whether the subject is “walking” or “jogging”. As for (III-c), the inputs of the third NN are the SD values of \( ||a|| \) and the mean values of \( a_Z \) \( (F_{2L+1\to5L}) \), to see whether the subject is “sitting down” or “standing up”, or if the activity is to be excluded (non-postural transition).

IV. Performance Evaluation

To evaluate the performance of the proposed HAR approach, we resorted to five public datasets. Our method is compared with state-of-the-art techniques, particularly (a) a broad range of widely used handcrafted features, and (b) shallow learning (SL) and deep learning (DL) techniques to assess the efficiency
of MDA as well as that of the local temporal features. The experiments are detailed in the next sub-sections.

A. Experimental Frame

Five public datasets (see TABLE II) were used in the experiments. They employ different accelerometers with different sampling rates $F_s$. We resorted to these datasets to run extensive performance evaluations in order to study different aspects of a HAR system. Note that, for the five datasets, the components are adjusted in such a way that $a_X$ represents the vertical component (oscillating around 1 g when the subject is walking) and $a_Z$ is pointing towards the body. The corresponding data were segmented as follows.

**FallAllD:** the seven targeted classes (from (1) to (7)) are selected from this dataset. Moreover, some movements like falls, jumping, start/stop walking/jogging were selected randomly to constitute the eighth class “Others”. These movements could occur in free-living conditions, but are to be filtered out by the algorithm. For transient ADLs, acceleration signals were segmented using 7 windows: the one centered on the transition moment, and six others shifted by ±125/±250/±500 ms. Afterwards, 5 out of 7 windows were chosen randomly. As for cyclic ADLs, 5 random windows were chosen to cut raw data. These operations resulted in 9100 samples consisting of eight classes: 1080 as “sitting down”, 1080 as “standing up”, 720 as “lying down”, 720 as “rising up”, 1130 as “walking”, 520 as “jogging”, 1095 as “rest” (inactivity), and 2755 non-targeted activities (others). The classifiers were first evaluated on this dataset using 10-fold cross validation, then trained on this dataset and tested on the remaining datasets. This choice was made since (i) FallAllID covers a wide-range of scenarios for the same ADL (like stand-to-sit and walk-to-sit for “sitting down” for example), (ii) the employed accelerometer is configured with the highest $F_s$ (so data can be easily down-sampled), and (iii) this dataset is relatively larger to train most of DL models.

**The remaining datasets:** these datasets were used for validation. Here, only the seven targeted classes were considered. 3 windows were selected for each ADL. For transient activities, the centers of the 3 selected windows were randomly set between the transition moment ±500 ms. For cyclic activities, the windows were selected randomly over the period of these movements.

Now, the proposed solution was first tested to quantify its performance (section IV-B). Next, we tested the following well-known handcrafted features from time and frequency domains, which have been shown to be effective [25], [28], [29], [40]:

- **time-domain:** mean, SD, median absolute deviation, maximum, minimum, energy, interquartile range, and entropy for each of the three components; the first four autoregressive coefficients of $|a|$: correlation coefficients between each pair of two components; signal magnitude area.
- **frequency-domain:** largest frequency component, weighted average, skewness, kurtosis, and energy for each component; angle between each pair of two components.

Afterwards, the classification was done using:

- classical classification strategy with one NN using the proposed feature vectors (section IV-C1).
- 9 shallow learning classifiers, namely multilayer perceptron (MLP), support vector machine (SVM), $k$-nearest neighbors (KNN), random forest (RF), gradient boosting machine (GBM), decision tree (DT), naive Bayes (NB), linear discriminant analysis (LDA), and quadratic discriminant analysis (QDA).
- CNN using raw acceleration components.
- simple RNN, LSTM, and GRU using raw data.
- stacked autoencoder (SAE) of dense layers.
- feature fusion (FF) as in [35].

A comparative study (section IV-C) between the proposed HAR solution and the aforementioned techniques is conducted in terms of accuracy, computational load, overfitting, and training on relatively small datasets. It is worth noting that the tests were done using a set of different sampling frequencies $F_s$ and window lengths $L$ to understand their effect on the discrimination power of each classifier and ensemble of features. Finally, the embeddability of MDA is discussed (section IV-D). For reproducibility purposes, the architectures of DL models and the hyper-parameters of SL classifiers are provided in a pdf file attached to this manuscript.

B. MDA Performance Assessment

In this sub-section, data of FallAllID were used to validate the proposed approach. As indicated before, the achieved accuracy is reported using:

- a set of $F_s \in [20, 40, 60, 100, 238]$ Hz
- a set of $L \in [2, 3, 4, 5, 6]$ s

![Fig. 7. The heat map consisting of the achieved accuracy (%) for each ($F_s$, $L$) pair after applying MDA.](image-url)
This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/TIM.2021.3108235, IEEE Transactions on Instrumentation and Measurement

C. Comparison with State-of-the-Art Methods

1) One stage of 8-class classification: the proposed feature vectors, i.e. local mean and SD values, are concatenated to form one vector of length 5L, which constitutes the input of a NN with one hidden layer of 10 neurons. This is the case of traditional ML methods, where extracted features feed one single classifier. In this case, the accuracy drops from 98.03% to 95%. This proves the added-value of the proposed architecture of MDA. Encoding the temporal characteristics is valuable, since the achieved accuracy of 95% is still relatively high, but the way of interpreting them plays a huge role in improving the performance.

2) Deep learning vs Shallow learning: the performance of DL models, especially recurrent ones, depends on data size. Therefore, the considered DL classifiers were trained and tested on the same 9100 samples, i.e. training and test sets (TrSet and TeSet) were the same. Table III illustrates the accuracy of each model. It is obvious that the accuracy of RNN is poor. The same goes for LSTM, even though its performance is somewhat better. This was expected as explained in section II-A. Simple RNN may suffer from vanishing/exploding gradient problem, while LSTM needs larger datasets to learn properly. However, GRU avoids both problems. CNN and SAE were also able to train, displaying quite high performance. Therefore, in a second step, only CNN, GRU and SAE were kept and a 10-fold cross validation (10-CV) was applied on FallAllD. In this step, the aforementioned nine SL techniques were also tested for comparison, considering the fifty handcrafted features extracted from windowed data (section IV-A). Additionally, the FF technique was also considered by combining GRU with handcrafted features. All these classifiers were trained then tested using all (F_s, L) pairs. Table IV illustrates the highest accuracy achieved by each model (10-CV on FallAllD), where values exceeding 98% are underlined. The proposed MDA is among the best performers, by providing competitive results. SAE is the worst DL model in terms of performance. As expected, FF outperforms GRU since it adds another dimension to the recognition process. As for SL models, KNN and GBM are the best performers with excellent accuracies.

Fig. 9 illustrates the variation in accuracy while fixing F_s and increasing L (left graph), and while fixing L and increasing F_s (right graph) for MDA, DL models and the two best SL performers (GBM and KNN). It is clear that very tight windows are not suitable for HAR. Now, for CNN, GBM, and KNN, larger windows increase the accuracy. However, for recurrent DL networks (GRU) and SAE, the ideal L is somewhere in the middle, since relatively larger windows make the accuracy decrease. Furthermore, in general, the accuracy increases while increasing F_s, except for SAE, since its accuracy is saturated around 89%. This is logical since higher F_s provides more data-points in time-series. Hence, the

![Fig. 8. The performance of MDA in terms of precision and recall for each targeted ADL using 1-vs-all strategy.](image)

![Fig. 9. The achieved accuracies (%) when (i) F_s is fixed and L is variable (left graph) and (ii) L is fixed and F_s is variable (right graph).](image)
pattern of signals delivers better knowledge to the machine. Note that, for SL models, the accuracy is somewhat the same for $F_s \geq 40 \text{ Hz}$. It is worth mentioning that it has been already shown [37] that deeper CNNs are required for higher sampling frequencies, since the features extracted by the first layer is more local and thus a deeper architecture is needed to abstract them. This examination is important to choose the best configuration for a HAR system.

3) **Overfitting:** the overfitting phenomenon occurs frequently in ML. If the learning corresponds too closely to a particular dataset, it may fail to fit additional data or predict future observations accurately. Thus, the choice of extracted features and the classifier is delicate. Since 10-fold cross validation may not reveal overfitting, seeing that training and testing are done on data from the same sensing unit, same subjects, and under the same conditions (same clinical trial), we suggest another experiment. The aforementioned models were trained on FallAllD, then tested on the remaining datasets to assess their performance. Table IV illustrates the accuracy for each model on each dataset, where values exceeding 90% are in bold. In general, SL methods were exposed to overfitting when tested on completely unseen data. This is obvious since the handcrafted features (listed in section IV-A) were extracted globally from windowed data instead of exploiting temporal locality of signals. Hence, they reflect the statistical patterns of a signal without taking into consideration its temporal structure. Moreover, large feature vectors may mislead the machine. When it comes to DL, recurrent networks are more robust. This is also expected since they learn and remember over time sequences, contrarily to CNN for example, which exploits the spatial correlation of data. Finally, MDA avoids overfitting since it encodes local temporal characteristics, and thus provides much better results than most of the other approaches. It is worth mentioning that results on mHealth are relatively higher. The detection of recurrent movements and inactivity periods is easier than that of transitions. Besides, MDA and GRU achieved the lowest results on UMAFall since $F_s$ is somewhat low (20 Hz). In conclusion, high accuracies may not always reflect the behavior of the system in real-world conditions.

4) **Computational Cost:** ultimately, we measured the computational time of the proposed algorithm (MDA), DL models, and the top two SL performers. We ran a Python 3.6 implementation of the algorithms on an Intel(R) Xeon(R) Gold 5118 2.3-GHz processor with 64 GB RAM. Table V illustrates the elapsed time during the training phase and feature extraction + prediction phase on 9100 samples. KNN is the fastest when it comes to training, since it only places the labeled vectors in some metric space. However, classifiers are trained offline. Hence, this phase is not related to the latency of the system. On the other hand, it is clear that MDA provides the fastest response when it comes to predicting the ADL. This response is negligible compared to most of the other techniques. With the efficient online computation technique and the tree-structure classification process, the computational cost decreases drastically compared to the other techniques. Meanwhile, the computational load of the handcrafted feature extraction and that of recurrent network are huge. Even though GRU seems suitable for HAR and may achieve satisfactory results (see previous sections), its complexity and high latency are major drawbacks. It is worth mentioning that the resident set size (RSS), the virtual memory size (VMS), and the number of page faults (pFault) vary between [89.91, 89.96] MB, [99.95, 100.04] MB, and [29206, 29234] respectively, when MDA makes a prediction, depending on the number of accessed stages (S1, S2, and S3). With NN models being loaded, the memory usage before launching MDA is defined by RSS = 85.93 MB, VMS = 91.95 MB, and pFault = 26638.

<table>
<thead>
<tr>
<th>Method</th>
<th>Training</th>
<th>Feature Extraction + Prediction</th>
</tr>
</thead>
<tbody>
<tr>
<td>MDA</td>
<td>15.12 s</td>
<td>0.173 s</td>
</tr>
<tr>
<td>CNN</td>
<td>34.16 s</td>
<td>2.54 s</td>
</tr>
<tr>
<td>SAE</td>
<td>10.46 s</td>
<td>0.55 s</td>
</tr>
<tr>
<td>GBM</td>
<td>107.13 s</td>
<td>23.018 s</td>
</tr>
<tr>
<td>KNN</td>
<td>0.55 s</td>
<td>24.235 s</td>
</tr>
<tr>
<td>GRU</td>
<td>4909.2 s</td>
<td>26.12 s</td>
</tr>
<tr>
<td>LSTM</td>
<td>4991.3 s</td>
<td>29.17 s</td>
</tr>
<tr>
<td>FF</td>
<td>4965.4 s</td>
<td>26.22 s</td>
</tr>
</tbody>
</table>

D. Embeddability

The elapsed time to extract features and make a prediction is useful to evaluate the relative computational cost of the approaches. However, it does not reveal the ability of being embedded in wearable devices, seeing the limited power resources of a micro-controller. Consequently, Appendix C shows in detail the number of operations required at each stage of MDA. Since $\alpha = \beta = \gamma = 1$ flop (micro-controllers featuring FPU single precision), the computational complexity is equal to $\Xi = 12d + 11k + 30$ flops, where $d$ and $k$ are the length of feature vector and the number of output classes respectively. Suppose that $L = 5$, the complexity of each stage (leaf) is:

\begin{align*}
&\{‘S1’: 123, ‘S2’: 183, ‘II-c’: 232, ‘III-c’: 243\} \\
&\rightarrow 123 \text{flops (rest)} \leq \Xi \leq 549 \text{flops (transitions)}
\end{align*}

With only few hundreds of flops, MDA is able to provide a
prediction. Note that these three types of operations are done in one cycle using a hardware implementation (which is almost negligible), without any need for a software library (the case of exponential and logarithm for example).

V. PROTOTYPE

A prototype was developed to illustrate the behavior of the proposed HAR system in real world conditions. The corresponding device was developed by RF-Track society in Cesson-Sévigné, France. LIS3DH from STMicroelectronics is the MEMS 3D accelerometer. It has dynamically user selectable full scales of $\pm 2g/\pm 4g/\pm 8g/\pm 16g$ and it is capable of measuring accelerations with output data rates ranging from 1 Hz to 5.3 kHz. It provides a 16-bit output interface. Additionally, it provides a data rate configuration of $\{1, 10, 25, 50, 100, 200, 400\}$ Hz and $\{1.60, 1.344, 5.376\}$ kHz. Raw data are pre-processed by the ultra-low-power micro-controller STM32L431 from STMicroelectronics (with FPU) and saved on a micro SD card for further analysis. Each component of the 3D accelerometer output is multiplied by $\frac{4}{3g\times 10^3}$ to convert it to g units.

As previously mentioned, a decision is taken every second by the micro-controller. Once connected to an electronic device like mobile phone or tablet, and thanks to a developed mobile application, the user can display the performed ADLs instantaneously. The device sends immediately the label of the predicted activity to the electronic device. Now, if a decision is made each second, a total of 86400 predictions are done under 24 hours. With 98% accuracy (which is quite high), 1728 windows are misclassified if we want to consider the ADL of each sliding window. However, the goal of a HAR system in real world conditions is the accurate localization of ADLs (postural transitions and cyclic movements) for an ideal post-processing tool. These data give another dimension to the HAR system, namely the altitude of the human body, which is an important feature for several applications.

VI. CONCLUSION

This paper addressed the challenges of a HAR system. By encoding local temporal characteristics of acquired acceleration signals, ADLs are accurately localized and recognized. The system presents low computational cost thanks to an efficient online feature extraction technique and a novel classification architecture. Extensive performance evaluations on five public datasets have proved the effectiveness of the proposed solution named MDA, which outperforms start-of-the-art handcrafted features and ML techniques in terms of accuracy, computational load, and capacity of avoiding overfitting. Furthermore, the sampling frequency and the window length were discussed to understand their effect on the accuracy of the system. The feasibility of MDA in real world conditions has been also investigated, by developing a prototype which accurately predicts the performed activity in real-time.

In a future work, barometric signals will be investigated as a post-processing tool. These data give another dimension to the HAR system, namely the altitude of the human body, which is an important feature for several applications.

APPENDIX A

UP-DOWN ACCELERATION PATTERN

Let us assume that MEMS accelerometers are equivalent to an ideal case of single spring and single mass system without any force opposing the motion of the components. When we stand up, the spring is released from the equilibrium point with an upward velocity $-v_0$. Hence, two forces are acting upon the system, namely (a) the restoration force of the spring $F_s = kx$, and (b) the weight due to gravity $W = m_0g$. Note that $X = x + s$ thus $F_s = kx + ks$: $ks$ the restoration force of the spring opposing the pulling of gravity and $kx$ the restoration force of the spring trying to get back to the equilibrium point. By applying Newton’s second law:

$$\sum F = -kx + m_0g - ks = m_0a$$  \hspace{1cm} (6)

Now, by setting the equilibrium as the reference point, i.e. $-ks + m_0g = 0$, Eq. (6) could be rewritten:

$$\ddot{x} + \left(\sqrt{\frac{k}{m_0}}\right)^2 x = 0$$  \hspace{1cm} (7)

Assuming that $\sqrt{\frac{k}{m_0}} = U_0$, the general solution is:

$$x(t) = C_1 \cos (U_0t) + C_2 \sin (U_0t)$$  \hspace{1cm} (8)

The mass was released from the equilibrium point:

$$x(t) = x_0 \sin (U_0t) , \quad x_0 < 0$$  \hspace{1cm} (9)
By replacing \( x(t) \) in (7) by its value in (9), we get:
\[
a(t) \sim a_0 \sin \left( w_0 t \right), \quad a_0 > 0 \tag{10}
\]
Note that for “sitting down”, we get the same acceleration with an opposite sinusoidal function.

**APPENDIX B**

**CUSTOMIZED TRANSFER FUNCTION**

The Taylor series for exponential \( \exp(u) \) (where \( u > 0 \)) is:
\[
\exp(u) = \sum_{i=0}^{\infty} \frac{u^i}{i!} = 1 + u + \ldots \tag{11}
\]
The hyperbolic tangent sigmoid transfer function \( h(x) \) is:
\[
h(x) = \frac{2}{1 + \exp(-2x)} - 1 \tag{12}
\]
By replacing \( \exp(u) \) by \( 1 + u \) (order 1) in \( h(x) \) we obtain the customized transfer function \( \tilde{h}(x) \):
\[
\begin{align*}
x > 0 & : h(x) = \frac{2}{1 + \frac{1}{2x}} - 1 = \frac{2}{1 + 1 + 2x} - 1 = \frac{x}{1 + 1 + 2x} \\
x < 0 & : h(x) = \frac{2}{1 + \exp(-2x)} - 1 = \frac{2}{1 + 1 - 2x} - 1 = \frac{x}{1 - x} \\
\rightarrow h'(x) &= \frac{x}{1 + |x|} \quad \forall x \in \mathbb{R} \tag{13}
\end{align*}
\]

**APPENDIX C**

**MDA COMPUTATIONAL COMPLEXITY**

The addition/subtraction \( \alpha \), multiplication/division \( \beta \), and multiply-and-accumulate \( \gamma \) are considered. The “sign” operation is free at the micro-controller level. The NN begins by scaling the feature vector \( \mathcal{F} \) of length \( d \) using the min-max normalization:
\[
\tilde{\mathcal{F}} = \frac{\mathcal{F} - \min(\mathcal{F})}{\max(\mathcal{F}) - \min(\mathcal{F})} \sim a \times \mathcal{F} + b \tag{14}
\]
This step requires \( \xi_1 = d\alpha + d\beta \) operations.

Afterwards, \( \tilde{\mathcal{F}} \) constitutes the input of the hidden layer (fully-connected) of 10 neurons. Hence, it is the subject to the following transformation:
\[
\hat{\mathcal{F}}^T = \mathbf{M} \times \tilde{\mathcal{F}}^T + \mathbf{B}_h \tag{15}
\]
where \( \mathbf{M} \) and \( \mathbf{B}_h \) are of size \( 10 \times d \) and \( 10 \times 1 \) respectively. This step results in \( \xi_2 = 10d\gamma + 10\alpha \) operations.

Subsequently, the customized transfer function \( h'(x) \) is applied on \( \hat{\mathcal{F}}^T \), resulting in \( \tilde{\mathcal{F}} \) as output while involving \( \xi_2 = 10\alpha + 10\beta \) operations.

Finally, \( \tilde{\mathcal{F}} \) feeds the output layer to finalize the k-class classification process:
\[
\text{Out} = \mathbf{H} \times \tilde{\mathcal{F}} + \mathbf{B}_o \tag{16}
\]
where \( \mathbf{H} \) and \( \mathbf{B}_o \) are of size \( k \times 10 \) and \( k \times 1 \) respectively. Here, \( \xi_k = 10k\gamma + k\alpha \) operations are required. Consequently, each stage of MDA requires \( \Xi = \sum \xi_i \) operations:
\[
\Xi = \alpha(d + 20 + k) + \beta(10 + d) + 10\gamma(d + k)
\]

**REFERENCES**


**Manuel Abbas** received the Engineering degree in electronics and telecommunications from the Faculty of Engineering, Lebanese University, Branch II, Roumieh, Lebanon, in 2018, and the master’s degree in signal and image processing from Ecole Centrale de Nantes, France, in 2018. He is currently pursuing the Ph.D. degree with LTSI (Laboratoire Traitement du Signal et de l’Image), Inserm U1099, Université de Rennes 1, France. His research activities mainly focus on signal processing and machine learning in the field of activity and health monitoring.

**Régine Le Bouquin Jeannès** (Member, IEEE) received her Ph.D. degree in Signal Processing and Telecommunications from Rennes University, France, in 1991. She is currently Professor in LTSI (Laboratoire Traitement du Signal et de l’Image, Inserm U1099) at Rennes University, France, and belongs to CRIBs (Centre de Recherche en Information Biomédicale sino-français), a French-Chinese laboratory associated with Inserm (Institut national de la santé et de la recherche médicale). Her research activities mainly focus on biomedical signals processing and modeling in the field of epilepsy and brain connectivity as well as on health monitoring and telecare.