IEEE Access
Multidisciplinary | Rapid Review | Open Access Journal

Received 9 September 2025, accepted 24 October 2025, date of publication 4 November 2025,
date of current version 11 November 2025.

*Digital Object Identifier 10.1109/ACCESS.2025.3628721*

RESEARCH ARTICLE

# Inference-Driven Window Sizing for Enhanced Human Activity Recognition in IoT Devices

**NICHOLAS KANIA, CHIARA CONTOLI, VALERIO FRESCHI, AND EMANUELE LATTANZI**
Department of Pure and Applied Sciences, University of Urbino, 61029 Urbino, Italy

Corresponding author: Emanuele Lattanzi (emanuele.lattanzi@uniurb.it)

This work was supported in part by European Union-NextGenerationEU through Italian Ministry of University and Research (MUR) National Innovation Ecosystem under Grant ECS00000041-VITALITY-CUP H33C22000430006.

**ABSTRACT** Human activity recognition in the Internet of Things devices has become a pivotal area of research in health monitoring, fitness tracking, and smart homes, especially with the increasing use of wearable and smart devices equipped with inertial sensors. Sizing the length of the signal window used to segment the sensor data stream is challenging. Conventional fixed-time interval methods may not guarantee an optimal level of generalization needed to handle the heterogeneous nature of human activities and their inter-/intra-personal variability at runtime, which may result in suboptimal recognition accuracy. While the fixed window strategy has been largely investigated, the variable window approach still remains little explored. In this paper, we propose an inference-driven method targeting low-power devices to vary the window size dynamically. Unlike traditional methods, the proposed technique dynamically adjusts the window length based on the classification confidence derived from model inference. We also propose an innovative validation methodology, namely leave-clustered-subjects-out, for generalization assessment. This paper describes how the window size adaptability ensures an increased ability to recognize activities whose characteristics exhibit temporal or subject-specific variations. Our approach is validated on four publicly available datasets, namely MotionSense, MobiAct, UCI-HAR (all three using smartphones), and WISDM using smartphone and smartwatch data. Experiments show an improvement in classification accuracy of up to 9% compared to conventional fixed-window approaches, with an energy consumption overhead remaining within a factor of 2. Results also show how our approach balances the tradeoff between recognition accuracy and computational efficiency on a target low-power device.

**INDEX TERMS** Human activity recognition, dynamic-window sizing, deep-learning, resource-constrained IoT devices.

## I. INTRODUCTION

Recognition of specific human activities is increasingly becoming a key task in many Internet of Things (IoT) application domains, ranging from healthcare (e.g., to support rehabilitation, patient assistance, and early diagnosis) to smart homes, from the tracking of sport/fitness activities to security and surveillance. Current approaches can be coarsely classified into computer-vision-based and sensor-based methods. The former entails automated images and/or video frame analysis; the latter uses sensor data (e.g., inertial signals, light, temperature) sampled by specific low-cost sensors integrated into embedded systems. These two solutions differ in: *i)* the complexity and flexibility (in terms of computation and communication requirements) of the underlying infrastructure; *ii)* the energy consumed by devices and at the whole system level; *iii)* the privacy level that can be guaranteed [1], [2]. In this article, we focus on sensor-based human activity recognition (hereafter denoted as HAR) through processing signals generated by inertial measurement units, i.e., accelerometers and gyroscopes.

One of the critical steps in the HAR process is properly segmenting sensor signals to extract relevant features that maximize accurate activity detection. In the context of

[0]The associate editor coordinating the review of this manuscript and approving it for publication was Jose Saldana.

activity recognition, in 2014, Bulling et al. surveyed five different segmentation approaches [3], namely: i) sliding window, ii) signal energy-based, iii) rest-position, iv) sensor modality, and v) context sources. The sliding window is undoubtedly the most widely used technique because of its simplicity in implementation. It defines a window size moved over the time series to find and extract relevant information. A very narrow window size might not be able to capture the characteristics of the activity, but might help in detecting the transition between activities [4], [5]; on the contrary, a large window size might contain information related to multiple activities, thus leading to potential erroneous classification. The signal energy-based approach leverages the principle that different activities produce different intensity levels, thus impacting the energy level of the sensor signals. The rest-position approach is mainly suited for gesture or whole-body recognition in the context of human-computer interaction, and it is based on the principle of identifying a hand position or a posture. The sensor modality approach proposes to leverage information coming from other modalities (e.g., environmental, heart rate) to segment the signal, whereas the context sources approach tries to leverage information (that is coupled to the activities) coming from external sources to discover, for example, the duration of an activity.

More recently, Najeh et al. identified two other approaches, namely sensor events segmentation and activity (or event) based segmentation techniques [6]. Sensor events windowing proposes to identify potential correlations between two sensor events. In particular, the challenge is to determine if two temporally separated (but one after the other) sensor events belong to the same activity or not. It is worth mentioning that this segmentation strategy is based on a sliding window approach, which leads to the same problem. Indeed, the data stream is split into sliding windows containing the same number of sensor events, and each window is labeled with the last event's label in the window, thus providing an activity context. However, one window might span multiple activities, or multiple events related to one activity might be spanned across more than one window. Moreover, the same activity carried out by multiple subjects (those latter possibly differing for gender, age, weight, etc.) has different dynamics and duration, thus requiring a dynamic window to be recognized. One possible way to overcome this issue is to adopt activity-based segmentation, which proposes to identify the boundaries of the activity [7]. The challenge remains how to determine if two sequential events belong to the same activity or not.

It is, therefore, clear that none of the segmentation approaches comes without challenges. Although the fixed window strategy has been largely investigated, our work is motivated by two main reasons: first, the variable window approach still remains little explored; second, to the best of our knowledge, none of the existing work in the literature on adaptive sliding window segmentation has targeted lower-power devices, nor has it explored how their proposed approach performs in terms of computational efficiency on target devices.

In this work, we contribute by proposing a new sliding window approach based on the dynamic window sizing paradigm. In particular, we introduce a novel algorithm that leverages the confidence levels provided by a deep neural network model to possibly explore larger or shorter window lengths that better fit the current inference need. The selection of the candidate window length is carried out using a lightweight search to avoid excessive burden on latency, memory, and energy. Once the best window size is identified across a predetermined set of choices, the inference is performed, and a new sample is loaded from the ongoing signal stream.

Results from an extensive set of experiments highlight that the proposed procedure is capable of accommodating cases where specific temporal variations occur in the dynamics of the activity and in cases of marked inter-/intra-subject variability. Moreover, we also demonstrate that the implementation of the method nicely fits resource-constrained IoT embedded systems, making it suitable for applications in wearable and mobile low-power devices.

### A. OUR CONTRIBUTION

To summarize, we make the following contributions:

- we propose a new algorithm to configure window size dynamically in HAR; the presented solution adaptively explores a set of alternatives based on the confidence of the current prediction and is capable of dealing with the different dynamics of the recognized activity. Examples are the changes in the execution dynamics of the same gesture by the same person and with the inter-personal variability, i.e., different people may execute the same activity, e.g., walking, with different dynamics.
- we introduce a systematic assessment based on clustering datasets according to homogeneous subject typologies (i.e., clustering according to weight, age, height, or according to a specific metric that we derived from the characteristics of the amplitude of the signal). This methodology enables us to: i) evaluate the capability of the proposed method of dealing with variations in the activity patterns; ii) highlight how the proposed system can generalize to a cohort of unseen subjects with inherent dynamics similar between them and different from the rest of the samples in the dataset.
- We provide the results of an extensive set of experiments to characterize the design of the system, also targeting its implementation on an IoT low-power embedded platform to demonstrate the feasibility of deployment on severely constrained devices

The remainder of the article is organized as follows: in Section II we summarize the state-of-the-art solutions currently available in the context of dynamic window sizing; in Section III we describe the proposed approach and provide algorithmic details; in Section IV we illustrate the

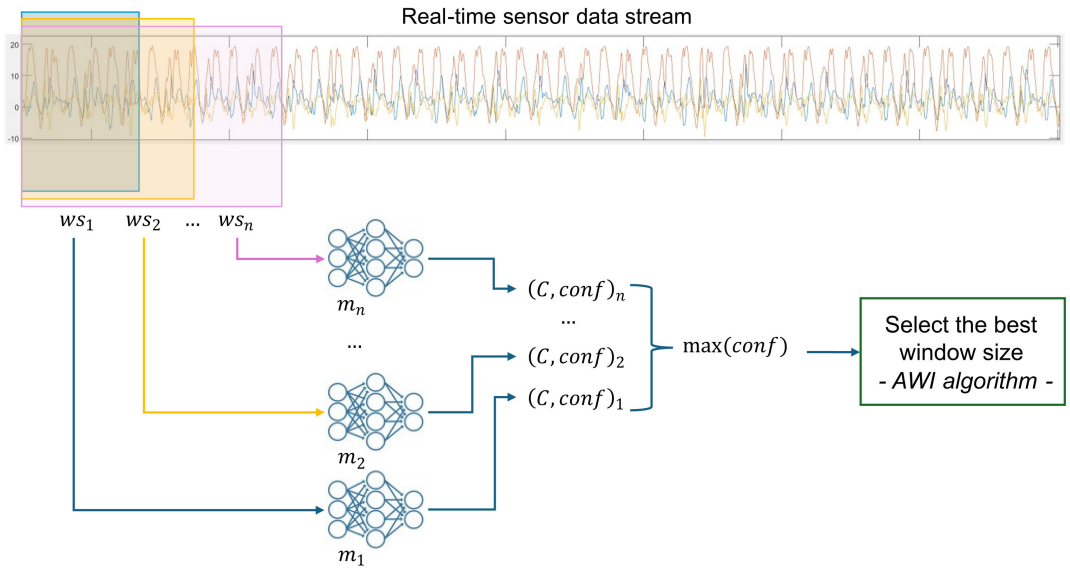



**FIGURE 1.** Main flow of the proposed approach entailing the dynamic sizing of the sliding window.

experimental setup adopted to validate our solution; the results of the experiments are presented in Section V; in Sections VI we discuss the limitations of the presented approach; finally, in Section VII a conclusive summing-up of the main findings and possible directions of future research are given.

## II. RELATED WORKS

Traditional approaches to inertial sensor data segmentation rely on grouping the data stream into sub-groups with the same characteristics. One of the most widely adopted approaches is the sliding window. This approach is also known as time-based window sizing because windows are measured in time intervals, which could be fixed or variable. While a fixed window size approach has always been investigated in the context of activity recognition [7], [8], [9], [10], [11], [12], the variable size approach is a recent attempt to improve the classification of an activity and still deserves investigation. Indeed, the duration of an activity can vary over time, and there is no unique window size that fits all activities.

Many strategies have been proposed to adjust the window length dynamically, and many base the adjustment decision on the Multivariate Gaussian Distribution (MGD). The Gaussian distribution calculates the probability that a signal belongs to a certain activity. Noor et al., for instance, proposed an approach that iteratively expands the window size, starting from an initial one, until the most effective window segmentation, i.e., the one with the highest probability density function value, is found [13]. Similarly, Ma et al. used the MGD to calculate the probability density function to determine the optimal window size for wheelchair user activities and proposed an algorithm that adjusts the window size by increasing or decreasing [14]. The Gaussian approach has been combined with classification entropy to adjust the window length by Santos et al. for body part trajectory recognition in a Cartesian space [15]. The authors increased or decreased the window size based on the entropy and the previous window length.

Other proposed strategies are inspired by speech signal processing, which leverages waveform evaluation, autocorrelation, and transformation methods. Sheng et al. used the autocorrelation method to extract the signal period from motion data of quasi-periodic activity to adjust the window size dynamically [16]. The authors needed to calculate the autocorrelation of at least two periods to discover the period. This approach allows the authors to be more precise in recognizing the activity according to the extracted features and adjust the time window according to the activity period, thus resembling the event-based windowing approach. Another work that leveraged this principle of looking for periodicity characteristics in the signal is the one proposed by Sun et al., where the authors used the fast Fourier transform for signal period extraction and the kernel density analysis method to fit the cycle sampling points of different user activities [17]. The authors then adjust the window size based on the duration of the single occurrence of different activities by selecting the frequency with the highest kernel density. A solution employing peak (and valley as well) detection, selection based on a threshold, and a distance for dynamic size windowing has been proposed in [18] in the context of rehabilitation of individuals with spinal cord injuries; the authors adjust the length by identifying the boundaries of each activity via peaks and valleys.

In line with the boundaries identification approach, Hirawat et al. proposed to identify activity boundaries by looking at the transition in the activity class labels in the

datastream, thus adjusting the window size as the number of activity records in that specific activity class [19]. Najeh et al. proposed to identify an activity's start and end, leveraging instead the event-based window approach [6]. The authors carried out the dynamic segmentation of events by applying the Pearson product-moment correlation coefficient between the sensor events. Therefore, their basic idea is to use a time-correlation-based method between sensors to determine whether they belong to the same activity. Although not using a dynamic approach, other works have been proposed to find the optimal window length for each target activity [9], [20], [21].

Compared to the existing literature, we propose a novel approach that leverages the confidence levels provided by the model, namely a deep neural network, to possibly explore larger or shorter window lengths that better fit the current inference need. Different from other methods that rely on preprocessing techniques to identify transitions in the signal stream, we, therefore, make use of the inference confidence to conveniently drive the selection of a suitable window size.

## III. THE PROPOSED APPROACH

This section outlines the proposed methodology for dynamically sizing the sliding window used to segment the input signals during activity recognition. We also provide a detailed description of the classification models and procedure for evaluating the impact of the proposed approach in terms of performance.

### A. SYSTEM OVERVIEW

Figure 1 shows a system overview of the proposed approach. In particular, starting from a real-time data stream provided by the IMU sensors built into the wearable device, we put forward a methodology based on the classification confidence of a set of different models $\mathcal{M} = [m_1, m_2, \ldots, m_N]$ trained with increasing sliding windows $\mathcal{W} = [ws_1, ws_2, \ldots, ws_N]$.

To identify the actual human activity, the accelerometer and gyroscope data are consumed using windows of different sizes, which are dynamically selected based on the model's classification confidence. In particular, each model corresponds to a given window size (i.e., it is trained with windows of the given length) and it returns the inferred class ($C$) together with its confidence value ($c$). Starting from the set of models $\mathcal{M}$, the Adaptive-Window Inference (AWI) algorithm identifies the best window size leveraging the classification confidence provided by each model. The algorithm may also trigger inference with additional models with still different window lengths.

The AWI algorithm is characterized by a $\alpha$ parameter in the range [0,1], which is used as a confidence threshold to decide when to deeply investigate more window sizes or not. For instance, for a given value of $\alpha$, if the actual model returns a higher confidence value, the algorithm stops testing further window lengths; otherwise, a new classification inference is triggered with a different model. Ultimately, $\alpha$ tunes the tradeoff between adaptivity and computation efforts.

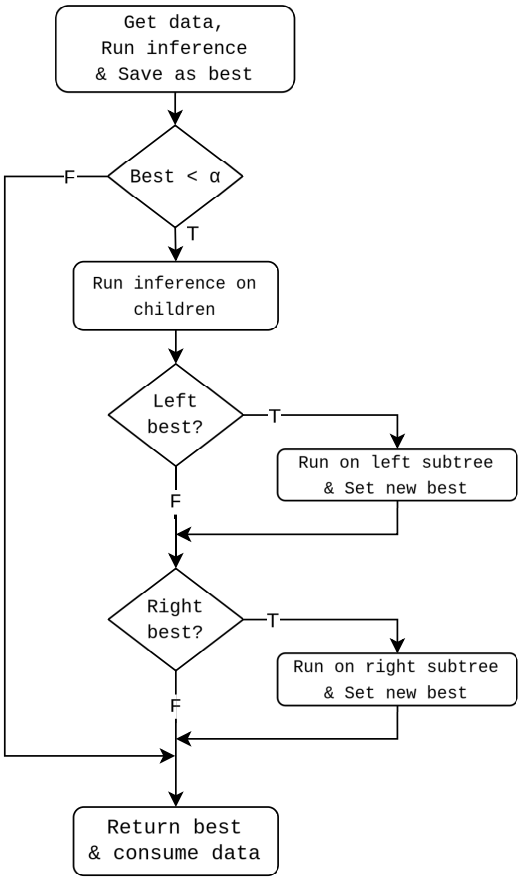



**FIGURE 2.** Adaptive-Window Inference (AWI) algorithm for the best window selection.

In particular, for $\alpha = 0$, no adaptivity is provided and the algorithm always uses a fixed window length; on the contrary, for $\alpha = 1$, all the available models are used and the one that shows the highest confidence is selected.

### B. THE ADAPTIVE-WINDOW INFERENCE ALGORITHM

#### 1) ALGORITHM AT A GLANCE

We propose an adaptive-window inference (AWI) algorithm whose flowchart is shown in Figure 2. A detailed description of the algorithm and involved parameters is provided in the next subsection.

Firstly, a series of classification models must be trained using windowed data acquired from a buffer using a distinct window size for each model. The reason is to encompass a comprehensive range of time intervals. Then, starting from these models, a binary tree is built where each node contains a model and its associated window size. When the algorithm is invoked on a node, the inference result is stored in a tuple containing the predicted class, relative confidence, and associated window size. This output is also saved as the best currently obtained. Upon obtaining the inference result, its confidence is compared to the threshold value $\alpha$. If the confidence of the current prediction is above the threshold, the output is returned as it satisfies the

**Algorithm 1** Adaptive-Window Inference (AWI)

**Input:** *Node* ▷ Node of binary tree. Data: (ws, model)
**Input:** $\alpha$ ▷ Threshold value
**Input:** *InferResult* ▷ Inference result (class, c, ws)
**Output:** *best* ▷ Best result (class, c, ws)

```
1:  function AWI(Node, α, InferResult)
2:      if InferResult = (0, 0, 0) then
3:          curr ← infer(Node)
4:      else
5:          curr ← InferResult
6:      end if
7:      best ← curr
8:      if curr.c < α then
9:          left ← infer(Node.Left)
10:         right ← infer(Node.Right)
11:         if max(best.c, left.c, right.c) = left.c then
12:             best ← AWI(Node.Left, 1.0, left)
13:         end if
14:         if max(best.c, left.c, right.c) = right.c then
15:             best ← AWI(Node.Right, 1.0, right)
16:         end if
17:     end if
18:     return best
19: end function
```

application requirements. Otherwise, a further exploration of the nodes' children is performed. If a child outperforms the current classification by accuracy, the algorithm is executed recursively on the relevant subtree, and the best result is updated. At the end of all the recursions, the best result is found, and the amount of data employed during research process is removed by the buffer.

#### 2) ADAPTIVE-SIZING INFERENCE CRITERIA

The pseudo-code of the proposed AWI method is shown in Algorithm 1. The (*AWI*) takes as input: *i)* the root node of the (sub)tree; *ii)* the confidence threshold $\alpha$; *iii)* a triple (termed InferResult) representing the class predicted by the model (*class*), the confidence of the prediction (*c*), and the corresponding window size (*ws*), respectively. Given its uniqueness, the tree uses the *ws* as a key; therefore, designing a balanced tree enables efficient exploration strategies. It is worth noticing that, in principle, the models associated with each node are not required to have the same architecture by design, allowing the deployment of different, specialized models for the various window sizes.

When the function *AWI* is invoked, it requires the tree's root as a starting node, the $\alpha$ parameter, and the inference result, possibly computed by previous recursions. This latter input is set to be (0, 0, 0) just during the first invocation of the *AWI*. Firstly, the algorithm checks if the inference has already been carried out for the given node using the assigned parameter *InferResult*. If this parameter is (0, 0, 0), a call to an auxiliary function *Infer* (not shown in pseudo-code) is made by passing the root node of the subtree. This function extracts data from a buffer continuously filled with sensor readings (without consuming any value of the stream) according to the nodes' window size; then, *Infer* performs the prediction and stores the result in the *curr* variable, while another variable, called *best*, is also initialized with the same result. It is worth mentioning that *Infer* is also called from *AWI* each time a new recursive exploration of the subtree begins.

Suppose the current confidence level is lower than the threshold $\alpha$. In this case, the inference computation is executed starting from the two child nodes, and the results are assigned to variables *left* and *right*, respectively. Upon completion, the maximum is computed between the best confidence currently achieved and the confidence of the children nodes. Suppose the maximum coincides with the left node result. In that case, the algorithm is recursively invoked on the left child node, with $\alpha$ value set at 1 (since we want the search to continue) and with *left* as *InferResult* variable. The algorithm finally returns the best result (composed by class, confidence, and window size used) in the *best* variable. After the result is computed, the chosen window size is consumed from the head of the buffer, allowing the retrieval of new data upon successive invocations.

#### 3) CONSIDERATIONS

Concerning the computational requirements of the proposed approach, we have to precise that to find the best window size necessitates, in the worst case, using all pre-trained models (i.e., *n*). For simplicity, if we consider a set of models that differ only in their input layer (i.e., the number of input samples in a given window), we can assume that executing each model will take substantially the same amount of time. Consequently, the time requirements in the worst-case scale linearly with the number of models, as does the memory footprint. In practice, it seems feasible to have at most a dozen models for constrained wearable devices. Therefore, more than the asymptotic complexity, we need to pay attention to the models' tuning to ensure they do not exceed memory and time constraints in the worst-case scenario.

The practice of adopting adaptive approaches that simultaneously explore the solution space while balancing the tradeoff between recognition accuracy and computational efficiency proves to be relevant in the context of HAR. Indeed, this practice has also been recently used to discover the optimal model architecture in the context of health monitoring use case [22], highlighting not only the promising results in terms of recognition accuracy but also a more flexible and robust solution.

### C. CLASSIFICATION MODELS

To assess the viability of our proposed methodology, we have developed two distinct classification model families based on deep neural networks. The first model family, derived from the architecture proposed by Bigelli et al. in 2024 [23], allows us to construct models tailored to edge devices.

**TABLE 1.** Models architectures.

| Edge | | Tiny | |
|---|---|---|---|
| Layers | # Elements | Layers | # Elements |
| Input | Input Size | Input | Input Size |
| Conv1D | (32, 2) | Conv1D | (16, 3) |
| Conv1D | (224, 4) | LSTM | 16 |
| Conv1D | (160, 2) | Dense | 256 |
| Conv1D | (128, 3) | Dense | # Classes |
| Dense | 544 | | |
| Dense | 1024 | | |
| Dense | 96 | | |
| Dense | # Classes | | |
| ***# Params*** | ***≈ 960k*** | ***# Params*** | ***≈ 8k*** |

Notice that this model is intended only as a reference for high performance in terms of accuracy, but it is not suitable for deployment on a battery-powered IoT device. Due to its memory and computational requirements, it could be installed on a wire-powered single-board computer such as a Raspberry Pi.

The second model family, built upon the architecture presented by Contoli and Lattanzi in 2023 [24], provides a set of tiny models deployable on MCU-based IoT devices and represents the main foundation on top of which this work is based. In particular, the first model family, called *Edge*, comprises four one-dimensional convolutional layers, followed by an average pooling layer and four dense layers (each alternated with a dropout to prevent overfitting). In comparison, the *Tiny* family is built on a single one-dimensional convolutional layer followed by a Long Short Term Memory (LSTM) recurrent layer and two dense layers.

Table 1 shows the configuration details of the two network families, *Edge* and *Tiny*, respectively. Under column *# Elements*, we specify in brackets the number of filters and kernel size of the convolutional layers, first and second, respectively. Notice that deploying machine learning models on constrained devices implies striking a balance between accuracy and computational requirements. For instance, the proposed *Edge* models contain about 960K parameters and can be executed on top of a Raspberry Pi 4, in less than 500 ms, while *Tiny* models, composed of about 8k parameters, need about 80 ms to be executed on a typical 32-bit MCU-based platform like that used for wearable devices. In HAR, signal processing windows lasting between 2 and 10 seconds are conventionally employed. Considering the MCU platform, we are able to make up to 25 different inferences without compromising real-time requirements, even in the worst-case scenario (i.e., when using the shortest window).

### D. PERFORMANCE ASSESSMENT

Comparison with other approaches is not trivial, in particular with respect to dynamic window methods. In fact, to the best of our knowledge, none of the approaches listed in Section II make the source code available for an accurate comparison. To overcome this reproducibility problem, we therefore opted for a comparison between the proposed system and a static window counterpart. Moreover, to highlight the potential of a runtime, dynamic sizing, especially in the presence of intra-personal variability, we provide two kinds of performance assessment methodology: the leave-subjects-out (LSO) and an ad-hoc designed methodology called leave-clustered-subjects-out (LCSO).

**Leave-subjects-out** is a cross-validation technique used to evaluate the generalization performance of classifiers, especially in the HAR domain where dataset samples belong to several different subjects [25]. The main idea is that, for a given dataset $\mathcal{D}$, the model is trained on data from a subset of participants and then tested on participants not included in the training phase. The subjects contained in training are called *seen subjects*, and the samples belonging to them is used to build the $\mathcal{S}$ set, while those belonging to the subjects that are kept out lead to the set of the *unseen subjects* $\mathcal{U}$. Sets $\mathcal{U}$ and $\mathcal{S}$ are mutually exclusive (i.e., $\mathcal{D} = \mathcal{U} \cup \mathcal{S}$ and $\emptyset = \mathcal{U} \cap \mathcal{S}$).

LSO methodology is a subject-independent evaluation strategy that ensures models are tested on entirely unseen subjects. This makes it possible to assess how well the model performs on unseen individuals, which is critical in real-world applications where the model encounters new users with different movement patterns, behaviors, or physiological characteristics. It should be noted that performance may drop compared to a traditional random split because inter-subject variability is often larger than intra-subject variability.

**Leave-clustered-subjects-out** is an assessment methodology we conceived to more effectively evaluate the generalization capability of a classification model in HAR. In particular, for the datasets containing information regarding participating subjects' attributes, we group samples according to the chosen attribute $x$ (i.e., $x =$ speed, age, weight, height) of the subject to which they belong, defining a *clustering rule* $C_x$. The samples belonging to the subjects characterized by a specific range of values of the given attribute are kept as a hold-out set $\mathcal{U}$ (*unseen subjects*) used only in the training phase. The remaining samples constitute the set $\mathcal{S}$ of *seen subjects*, which is then used for the usual train-test split.

For example, we rank all samples according to the age of the participants and use all records from subjects under a specified age (clustering rule $C_x$, $x =$ age) to obtain the $\mathcal{S}_{C_{age}}$ set used to train the models. Then, we test each model on the remaining samples (i.e., those from all subjects above the chosen value, the set $\mathcal{U}_{C_{age}}$). The rationale behind this procedure is to enable the evaluation of how the learning models under test adapt to subjects with various (and unseen) activity dynamics. In this way, we can verify whether the model can cope with a markedly different typology of inertial signals. In this sense, it represents a stronger benchmark for generalization concerning standard LSO validation procedures to check the robustness of shifts in the statistical distribution of the test set with respect to the training set.

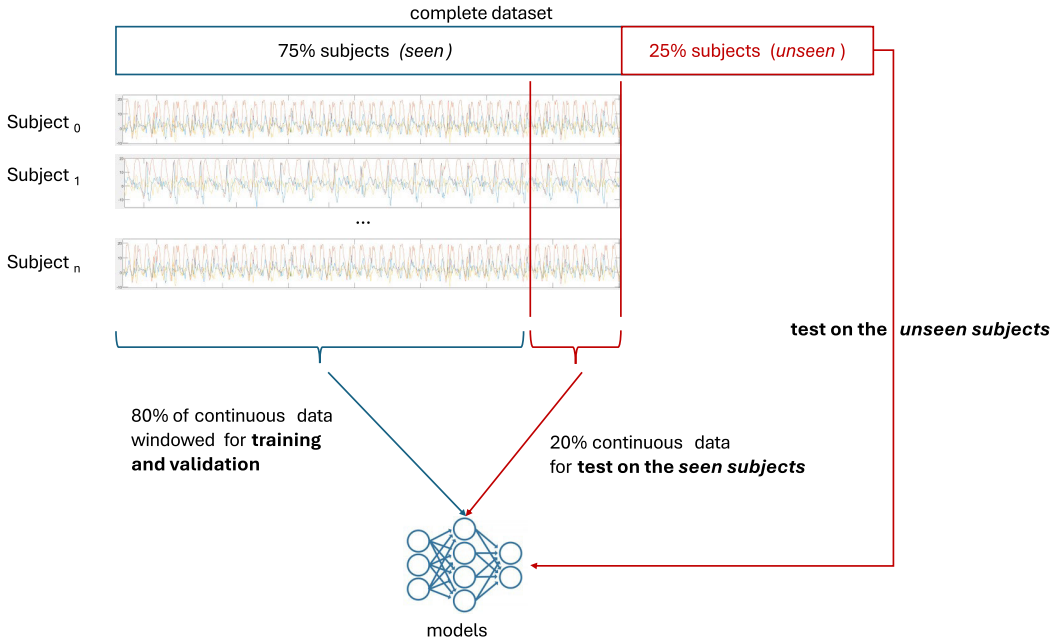



**FIGURE 3.** Dataset splitting procedure for cluster-based assessment.

By assessing how the method generalizes to a cohort of unseen subjects with dynamics potentially different from the rest of the samples in the dataset, we can, therefore, evaluate how the HAR system under study adapts to variability in the execution of activities. Our approach extends LSO methodology by exploring different clustering criteria to build the hold-out split.

We explore groups according to multiple attributes, namely age, height, weight, and the number of peaks per sample of the acceleration signal. This last feature is computed from the vertical axis of series labeled as walking, counting the number of peaks contained in the signal and dividing it by the related number of samples; it represents a reasonable approximation of the number of steps per time unit, thus providing a simple yet effective clue of the dynamics of non-standing activities that can be used to separate subjects that perform these activities with higher frequency content from others with slower dynamics. In other words, it allows, in principle, to separate slow subjects from fast ones. Moreover, it can be used to cluster datasets even in the absence of other types of data associated with the subject, a desirable option in many cases; in fact, while there are several datasets devoted to HAR, only very few report accessory information that can be exploited to perform the above-described experiments. Finally, the nature of the partitioning enforced by our methodology (i.e., clustering and construction of windowed streams of signals), coupled with the significant amount of data needed by deep learning algorithms and models, further reduces the set of possible options.

In Figure 3, we show a graphic summary of the overall train and test flow. The dataset is partitioned according to a given clustering rule into a set of seen and unseen subjects (75% and 25% in the case of the figure and of our experiments as described in Section IV). Then, the set of signals associated with each seen subject is further divided into two parts: the first one (80% in the figure) is segmented into predefined windows and used to train the models; the remaining part (20% in the example) is used as test data to validate the approach on seen subjects. Finally, the trained models are tested on the samples from unseen subjects to assess the generalization capabilities of the models. Notice that, during the testing phase, the size of the window used to consume the stream is selected at runtime. Consequently, with regard to both seen and unseen subjects, it is not possible to segment the testing set in advance, nor to shuffle the samples to maintain their temporal order.

## IV. EXPERIMENTAL SETUP

In this section, we provide a detailed description of the experimental setup used.

### A. DATASETS

Four datasets have been used to evaluate the proposed methodology namely MotionSense, MobiAct, WISDM, and UCI-HAR.

#### 1) MOTIONSENSE

[26] this dataset contains time-series data from accelerometer and gyroscope sensors sampled at 50 Hz, extracted by an iPhone 6s kept in the participant's front pocket using SensingKit from the Core Motion framework on iOS devices. It includes 6 different activities collected from a total of 24 users having different genders, weights, heights, and

ages. Each data extraction occurred in the same environment among 15 different trials [27].

#### 2) MOBIACT

[28] this dataset is widely used for evaluating machine learning and deep learning models in HAR due to its detailed annotations and variety of included activities. It contains data collected from smartphone accelerometers and gyroscopes sampled at 50 Hz from a total of 66 participants, providing a set of motion data for 12 different activities. Each recording is performed by subjects of diverse genders, heights, weights, and ages under controlled conditions.

#### 3) WISDM

[29]: contains data from the accelerometer and gyroscope sensors of both a smartphone and a smartwatch while 51 subjects engaged in 18 different activities of daily living. Each activity was performed for an average duration of 3 minutes so that each subject contributed 54 minutes of data. These activities include general ambulation-related tasks (such as walking, jogging, and climbing stairs), hand-based activities of daily living (like brushing teeth and folding clothes), and a range of eating activities (including eating pasta and consuming chips) [30]. As proposed by the dataset authors, we kept the division in subsets and tested our approach on the *General* and *Non-Hand Oriented (NHO)* activities separately.

#### 4) UCI-HAR

[31] is built from accelerometer and gyroscope samples belonging to seven different activities: walking, walking upstairs, walking downstairs, sitting, standing, and lying down. Data are collected via smartphones at a frequency of 50 Hz. All the activities were gathered from 30 subjects between 18 and 48 years of age and of varying races, heights, and weights [32].

### B. DATASETS PREPARATION

As mentioned in Section III, the proposed methodology uses a series of different models trained using several window sizes. In this work, we chose windows composed of 125, 188, 250, 313, and 375 samples, each of which, according to the sampling rate of the chosen datasets, leads to windows of the duration of approximately 2.5s, 3.8s, 5s, 6.3s and 7.5s respectively. To increase the dimension of the training set, 30% overlapping windows have been used to segment the original signals in the training phase.

Regarding the LSO experiments, the four datasets have been randomly partitioned into seen and unseen subjects ($\mathcal{S}$ and $\mathcal{U}$), with a ratio of approximately 75% and 25%, respectively.

MotionSense and MobiAct, the two datasets for which subject attributes are known and can be used to perform LCSO experiments, have been split into $\mathcal{S}$ and $\mathcal{U}$ by applying each clustering rule $\mathcal{C}_x$ in order to obtain a training set ($\mathcal{S}_{C_x}$) containing approximately 75% of the data and a test set ($\mathcal{U}_{C_x}$) with the remaining 25%

**TABLE 2. Clustering rules.**

| Dataset | Rule | Ranking criteria |
|---|---|---|
| MotionSense | $C_{speed}$ | Slowest subjects |
| | $C_{weight}$ | Heaviest subjects |
| | $C_{height}$ | Shortest subjects |
| | $C_{age}$ | Youngest subjects |
| MobiAct | $C_{speed}$ | Slowest subjects |
| | $C_{weight}$ | Heaviest subjects |
| | $C_{height}$ | Shortest subjects |
| | $C_{age}$ | Oldest subjects |

Table 2 indicates the various clustering rules applied to each dataset together with the corresponding ranking criteria.

It is worth recalling that the rule concerning $C_{speed}$ is computed from the number of peaks detected in a window in the vertical acceleration signal; hence, we can use it as a proxy for discriminating slowest from fastest persons. $C_{weight}$, $C_{height}$, and $C_{age}$, on the other hand, clearly provide a criterion for clustering the heaviest, shortest, and oldest/youngest people. Finally, with regard to the $C_{age}$ rule, the decision was taken to utilize contrasting ranking criteria in the two distinct datasets in order to enhance the inter-cluster variability.

### C. HARDWARE SETUP

Training and testing were performed on a workstation equipped with two Intel ® Xeon® Silver 4314 processors and three NVIDIA® A100 GPUs. The final tiny-models, converted into TensorFlow Lite format, were deployed on a low-resource device from Espressif ®, the ESP32 [33]. This microcontroller features two independently controllable CPU cores with clock frequencies adjustable between 80 MHz and 240 MHz. It also supports a low-power mode for energy-efficient operation during peripheral I/O tasks requiring minimal computational effort. This version of the ESP32 includes 4 MB of flash memory for firmware storage and 400 KB of RAM.

Energy consumption was measured by powering the device with an NGMO2 Rohde & Schwarz dual-channel power supply [34] and connecting an 8.0 Ω shunt resistor in series. The voltage drop across the shunt resistor was sampled using a National Instruments NI-DAQmx PCI-6251 16-channel data acquisition board, interfaced with a BNC-2120 shielded connector block [35], [36].

## V. EXPERIMENTS AND RESULTS

In the following, we present the results obtained during the experiments performed to characterize the effectiveness of the proposed approach. In particular, we report the classification accuracy gain and the energy overhead measured on top of the benchmark datasets.

### A. CLASSIFICATION ACCURACY

To create the reference point, hereafter called baseline (used as comparison terms with the proposed approach) we

**TABLE 3.** Classification accuracy of the fixed-window models on the reference datasets in random LSO experiments.

| Dataset | Edge | | Tiny | |
|---|---|---|---|---|
| | $A_{\mathcal{S}}$ | $A_{\mathcal{U}}$ | $A_{\mathcal{S}}$ | $A_{\mathcal{U}}$ |
| MotionSense | $0.904 \pm 0.011$ | $0.885 \pm 0.036$ | $0.836 \pm 0.039$ | $0.808 \pm 0.069$ |
| MobiAct | $0.934 \pm 0.003$ | $0.852 \pm 0.034$ | $0.904 \pm 0.012$ | $0.801 \pm 0.025$ |
| WISDM [NHO] | $0.870 \pm 0.003$ | $0.855 \pm 0.049$ | $0.813 \pm 0.035$ | $0.810 \pm 0.006$ |
| WISDM [General] | $0.909 \pm 0.013$ | $0.849 \pm 0.029$ | $0.860 \pm 0.014$ | $0.785 \pm 0.030$ |
| UCI-HAR | $0.873 \pm 0.009$ | $0.857 \pm 0.012$ | $0.869 \pm 0.008$ | $0.870 \pm 0.013$ |

**TABLE 4.** Best classification accuracy of the proposed approach on the reference datasets split in random LSO experiments.

| Dataset | Edge ($\Delta$) | | Tiny ($\Delta$) | |
|---|---|---|---|---|
| | $A_{\mathcal{S}}$ | $A_{\mathcal{U}}$ | $A_{\mathcal{S}}$ | $A_{\mathcal{U}}$ |
| MotionSense | 0.931 (**0.029**) | 0.916 (**0.031**) | 0.889 (**0.050**) | 0.873 (**0.065**) |
| MobiAct | 0.941 (**0.007**) | 0.887 (**0.036**) | 0.933 (**0.029**) | 0.843 (**0.041**) |
| WISDM [NHO] | 0.892 (**0.021**) | 0.878 (**0.023**) | 0.873 (**0.058**) | 0.864 (**0.054**) |
| WISDM [General] | 0.929 (**0.022**) | 0.864 (**0.015**) | 0.905 (**0.043**) | 0.820 (**0.035**) |
| UCI-HAR | 0.894 (**0.029**) | 0.896 (**0.039**) | 0.900 (**0.024**) | 0.894 (**0.024**) |

**TABLE 5.** Classification accuracy of the fixed-window models on top of the reference datasets in LCSO experiments following the clustering rules.

| Dataset | Rule | Edge | | Tiny | |
|---|---|---|---|---|---|
| | | $A_{\mathcal{S}}$ | $A_{\mathcal{U}}$ | $A_{\mathcal{S}}$ | $A_{\mathcal{U}}$ |
| MotionSense | $C_{speed}$ | 0.926 | 0.857 | 0.860 | 0.793 |
| | $C_{weight}$ | 0.909 | 0.888 | 0.792 | 0.860 |
| | $C_{height}$ | 0.894 | 0.780 | 0.813 | 0.689 |
| | $C_{age}$ | 0.863 | 0.875 | 0.887 | 0.823 |
| MobiAct | $C_{speed}$ | 0.928 | 0.731 | 0.930 | 0.764 |
| | $C_{weight}$ | 0.925 | 0.802 | 0.923 | 0.813 |
| | $C_{height}$ | 0.922 | 0.789 | 0.915 | 0.806 |
| | $C_{age}$ | 0.896 | 0.815 | 0.907 | 0.818 |

conducted a series of experiments to evaluate the accuracy of the standard fixed-window classifiers on top of the selected datasets using a 5-second window. This size is an intermediate value which arguably should represent the best trade-off between the various experimented lengths. Each of the considered datasets has been divided then into sets of seen and unseen subjects, as described in Section III-D.

In the case of LSO assessment, the classifier has been trained on 80% of samples belonging to the seen subjects ($\mathcal{S}$) and tested on the remaining 20% of samples belonging to the same set, in order to calculate the accuracy over seen subjects ($A_{\mathcal{S}}$). Then, the trained classifier has been tested on top of the unseen subjects ($\mathcal{U}$) to obtain the accuracy over the unseen ($A_{\mathcal{U}}$).

For what concerns LCSO experiments, for each rule $C_x \in \{C_{speed}, C_{weight}, C_{height}, C_{age}\}$, the classifier has been trained on 80% of samples belonging to the seen subjects ($\mathcal{S}_{C_x}$) and tested on the remaining 20% of samples belonging to the same set, in order to calculate the accuracy over seen subjects ($A_{\mathcal{S}}$). Also in this case, predictions on the unseen subjects ($\mathcal{U}_{C_x}$) have been used to obtain the accuracy over the unseen ($A_{\mathcal{U}}$).

Overall, we refer to these results as a baseline as they represent the classification performance achieved by traditional approaches leveraging fixed-size sliding windows.

Table 3 illustrates the results of the LSO experiments for the two models taken into consideration. In particular, the *Edge* model achieves better results with respect to the *Tiny* one (as expected, because of its higher complexity), in terms of $A_{\mathcal{S}}$ and $A_{\mathcal{U}}$. Furthermore, both models clearly reach higher accuracies on the seen subjects (with the exception of *Tiny* on UCI-HAR which remains substantially unchanged).

In Table 4 we reported the outcomes of the LSO experiments when the proposed approach based on inference-driven window selection is used. Results refer to the best classification accuracies obtained across different values of the parameter $\alpha$; the gain with respect to the baseline is reported in brackets. The application of the proposed method is apparently beneficial, with accuracy gains up to 2.9% and 3.9% for the *Edge* model on seen and unseen subjects, respectively and improvements up to 5.8% and 6.5% (in $A_{\mathcal{S}}$ and $A_{\mathcal{U}}$) for the *Tiny* network.

In general, it is worth noticing that: *i)* the performance of both deep learning models decrease when shifting from seen to unseen subjects (which is somehow an expected consistency); *ii)* the proposed approach always improves the prediction accuracies, with a positive gain across all datasets and classifiers.

Table 5 summarizes the results of the LCSO series of experiments using both proposed classifiers. For each clustering rule, the *Edge* model outperforms the *Tiny* model regarding the $A_{\mathcal{S}}$ metric in support of the fact that more complex models perform better on traditional hold-out benchmarks. On the other hand, concerning the classification accuracy measured on top of the unseen subjects ($A_{\mathcal{U}}$), both models see their performances drop significantly (even up to 19.7 percentage points), and the supremacy of the *Edge* model tends to disappear. In several cases, the smaller model surpasses the bigger one.

These findings help us to emphasize: i) the issue of the lack of widely representative datasets; ii) how both complex and simpler models are unable to account for interpersonal variability when employed to predict the activities of

**TABLE 6.** Best classification accuracy of the proposed approach on reference datasets in LCSO experiments following the clustering rules.

| Dataset | Rule | Edge (Δ) | | Tiny (Δ) | |
|---|---|---|---|---|---|
| | | $A_{\mathcal{S}}$ | $A_{\mathcal{U}}$ | $A_{\mathcal{S}}$ | $A_{\mathcal{U}}$ |
| MotionSense | $C_{speed}$ | 0.955 (**0.029**) | 0.898 (**0.042**) | 0.897 (**0.037**) | 0.862 (**0.069**) |
| | $C_{weight}$ | 0.934 (**0.024**) | 0.904 (**0.015**) | 0.860 (**0.068**) | 0.910 (**0.049**) |
| | $C_{height}$ | 0.907 (**0.013**) | 0.846 (**0.067**) | 0.845 (**0.032**) | 0.737 (**0.048**) |
| | $C_{age}$ | 0.905 (**0.041**) | 0.904 (**0.029**) | 0.910 (**0.023**) | 0.913 (**0.090**) |
| | **average** | **(0.027)** | **(0.038)** | **(0.040)** | **(0.064)** |
| MobiAct | $C_{speed}$ | 0.932 (**0.004**) | 0.813 (**0.082**) | 0.946 (**0.016**) | 0.855 (**0.091**) |
| | $C_{weight}$ | 0.940 (**0.015**) | 0.849 (**0.047**) | 0.943 (**0.019**) | 0.849 (**0.035**) |
| | $C_{height}$ | 0.943 (**0.020**) | 0.841 (**0.052**) | 0.942 (**0.027**) | 0.847 (**0.042**) |
| | $C_{age}$ | 0.922 (**0.026**) | 0.843 (**0.027**) | 0.940 (**0.033**) | 0.831 (**0.013**) |
| | **average** | **(0.016)** | **(0.052)** | **(0.024)** | **(0.045)** |

individuals whose characteristics differ from those included in the training set. Moreover, these findings also suggest that the proposed cluster-based performance assessment is a viable methodology for evaluating model generalization in the HAR domain.

The same experiments have been then repeated using the proposed inference-driven window-sizing methodology for several values of $\alpha$. Table 6 reports the best classification accuracies.

In particular, for each clustering rule, the $A_{\mathcal{S}}$ and $A_{\mathcal{U}}$ accuracies are reported together with the achieved gain (values enclosed in brackets), and the average gains among each dataset (bold values) calculated with respect to the baseline.

Regarding the performance measured on top of the data belonging to the seen subjects ($A_{\mathcal{S}}$), the proposed approach is always advantageous concerning the baseline reported in Table 5 (the gain is always positive). For instance, concerning the *Edge* model, the average gain is 2.7% and 1.6% in MotionSense and MobiAct datasets, respectively. In the case of *Tiny* model, these values grow up to 4.0% and 2.4%. Going into detail concerning the different clustering rules, the gain sometimes reaches values close to 7%. For example, for the *Tiny* model, a gain of about 6.8% is obtained by ranking and splitting the MotionSense subjects by weight. In this case, the proposed methodology can effectively capture the considerable inter-subject variability in motor gesture performance among individuals of varying weight, thereby restoring the classification accuracy of the *Edge* model to a more than acceptable level.

Focusing on the performance obtained on top of the unseen subjects $A_{\mathcal{U}}$, higher gain values are measured, which bring the classification performance close to the levels achieved on the seen subjects. In particular, the average gains for the *Edge* model are 3.8% and 5.2% depending on the dataset, with a max value of about 8.2% measured while MobiAct was split on the subject speed ($C_{speed}$). Regarding the *Tiny* model, the accuracy gain grows further, peaking at over 9% in MotionSense $C_{age}$ and MobiAct $C_{speed}$.

To summarize, the proposed methodology effectively compensates for the poor generalization ability of classification models when the training dataset is not fully representative of real-world conditions. Moreover, the gain is even more significant when the methodology is applied to small models, such as those used in the wearable and IoT contexts in general.

### B. COMPARISON WITH THE STATE-OF-THE-ART

One of the primary challenges in HAR research lies in the absence of both benchmark datasets and standardized evaluation protocols for assessing model performance. This limitation and the lack of available source code of state-of-the-art approaches hinder fair comparisons. In support of such claims, Table 7 shows a comparison between our approach and some of the state-of-the-art dynamic window sizing solutions. The comparison is analyzed from different perspectives, not only to support our claims but also to highlight the distinct contributions that each makes. The second column of Table 7 compares our approach with the performance of some state-of-the-art models in terms of the best accuracy gain achieved with respect to the static window reference system provided by the authors in their works. Such a result shows that our approach is effective in achieving a higher accuracy gain with respect to static ones. However, Table 7 underlines the difficulty and some limitations in comparing our approach with the other dynamic approaches.

First, some works use well known datasets [13], [17], whereas others use ad-hoc data [14], [16], [18], which are not publicly available. Moreover, some given datasets are not suitable for testing all approaches. Second, many studies focus on specific human activities, for example: [13] concentrates on transitional activities, [14] on wheelchair users activities, [16] on not specified quasi-periodic activities, [17] on sport activities and vital signal, and [18] on upper-limb rehabilitation activities of wheelchair users. Our approach is activity agnostic instead. Last, but not least, compared to our study, none of the works listed in Table 7 propose a solution suitable for MCU, low-power devices, nor carry out an in-depth performance analysis of the solution deployed on a physical device.

### C. IN DEPTH CLASSIFICATION PERFORMANCE ANALYSIS

To better understand the impact of the proposed methodology on the classification process, we deeply investigated the

**TABLE 7. Comparison between the proposed approach and some of the state-of-the-art dynamic window sizing solutions.**

| Authors | Accuracy gain [%] | Dataset | Methodology | Comparison limits |
|---|---|---|---|---|
| Noor et al., [13] | 6.9 | SBHAR [37] | Multivariate Gaussian distribution; probability density function | Public dataset; transitional activities; no MCU-based solution; no performance analysis |
| Ma et al., [14] | 4.8 | Ad-hoc data | Multivariate Gaussian distribution; probability density function | Private dataset; wheelchair activities; no MCU-based solution; no performance analysis |
| Sheng et al., [16] | 2.3 | Ad-hoc data | Two steps classification; autocorrelation | Private dataset; quasi-periodic activities; no MCU-based solution; no performance analysis |
| Sun et al., [17] | 6.1 | MHEALTH [38] | Fast Fourier transform; kernel density analysis | Public dataset; sport activities, vital sign; no MCU-based solution; no performance analysis |
| Alhammad et al., [18] | 6.5 | Ad-hoc data | Boundaries identification; threshold,distance | Private dataset; rehabilitation activities; no MCU-based solution; no performance analysis |
| **Proposed approach** | **9.1** | **MobiAct [28]** | **Inference-driven window sizing** | **Public dataset; agnostic about activities; MCU-based solution; performance analysis** |

per-class accuracy and the distribution of the applied corrections among the different classes. As the two proposed models exhibit a similar trend, and because our focus is on models targeting IoT devices, we report in the following sections the results from the *Tiny* model.

Figure 4 reports the confusion matrices related to the unseen subjects in the MotionSense dataset with ($\alpha = 1$) and without the proposed methodology ($\alpha = 0$). Note that $\alpha = 1$ means that our methodology is applied to every sample, which usually corresponds to the best-achieved performance. Conversely, a value of $\alpha = 0$ implies that the condition on the confidence returned by executing the model on the first window (i.e. the one corresponding to the root of the binary tree built with nodes corresponding to different models) is always satisfied, thus no further exploration of models with different windows is done (i.e. the inference is carried on with a fixed window).

By comparing the two matrices, it is evident that the gain in accuracy is not uniformly distributed. For example, the classes that have benefited the most are ''Standing'' and ''Sitting,'' which were previously confused with each other, and ''Stairs down,'' which was confused with ''Stairs up.'' However, the latter lost approximately 6 percentage points due to the misclassification with ''Walking'' and ''Jogging''. Finally, a significant enhancement is observed in the accuracy of the ''Walking'' class, exhibiting an increase of more than 15 percentage points.

Figure 5.(a) illustrates how the corrections made to the classification by our methodology have been classified as 'Right', 'Wrong' or 'Useless' corrections. In this case, most of the corrections made resulted in an enhancement of the classification, while the proportion of wrong or useless corrections is relatively insignificant in most cases. This result also shows that the number of useless corrections, which do not affect the gain in accuracy but can waste the device's energy, remains very low despite the high value of $\alpha$ used in this experiment.

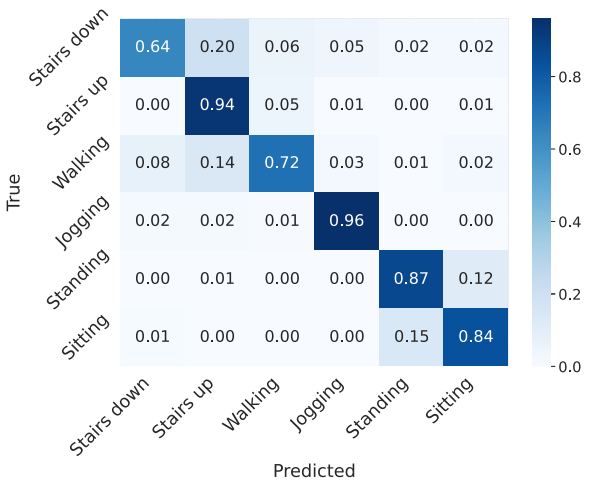
Figure 5.(b) shows which window size resulted in the corrections applied for each class in the dataset. Notably, a single window size does not make the most corrections, as classes vary. This evidence supports the hypothesis that each type of activity may have a different optimal classification window size. Moreover, the necessity to utilize different window sizes within the same class to enhance performance illustrates the importance of employing a dynamic methodology in real operational contexts. Different window sizes are, therefore, crucial to capturing inter-subject variability and intra-variability inside one subject performing the same activity at various times and under different boundary conditions.

### D. ENERGY OVERHEAD

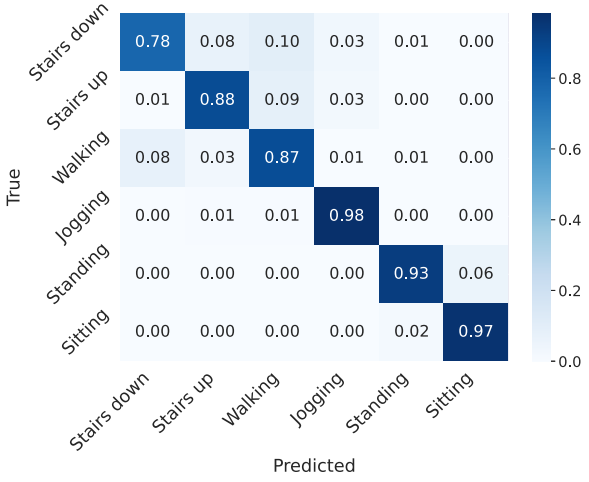
In the context of IoT devices, energy expenditure is a relevant factor in ensuring the validity of the proposed methodology. Given the increased energy expenditure associated with repeated execution of the model under varying window sizes, there is a clear requirement for accurate and precise quantification.

This section presents the results of the investigation into the energy overhead as the parameter $\alpha$ varies to identify the optimal compromise between energy and classification capability.

Figure 6 shows the average energy expenditure required to classify a single sample by the ESP32 MCU for each clustering rule in MotionSense while increasing $\alpha$. For each clustering rule, as $\alpha$ increases, the energy consumption rises gradually up to an $\alpha$ of approximately 0.7, after which it continues to grow exponentially, reaching a maximum value for $\alpha = 1$. It should be noted that, for a given value of $\alpha$, the actual number of model executions required to classify a single sample is not constant. Rather, it depends on the accuracy of the model itself, which determines whether a descent is triggered within the tree data structure used by the proposed algorithm. Remarkably, Figure 6 substantiates




(a) $\alpha = 0$




(b) $\alpha = 1$

**FIGURE 4. Confusion matrices for the $C_{age}$ clustering rule in MotionSense dataset before (a) and after (b) the integration of our proposed methodology.**

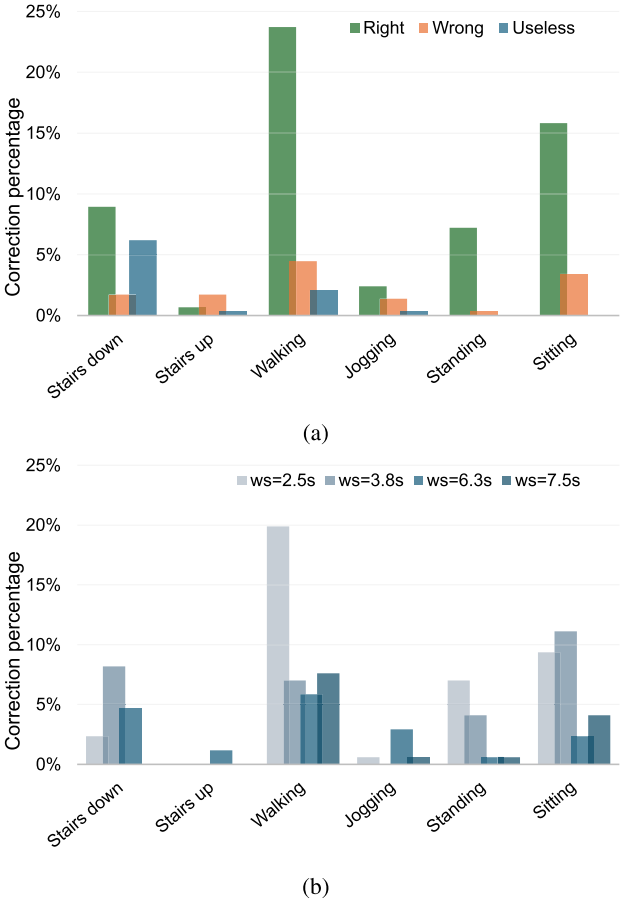



**FIGURE 5. Distribution of the percentage of corrections belonging to "Right", "Wrong", or "Useless" (a), and to the window size (b) for $C_{age}$ in MotionSense with $\alpha = 1$.**

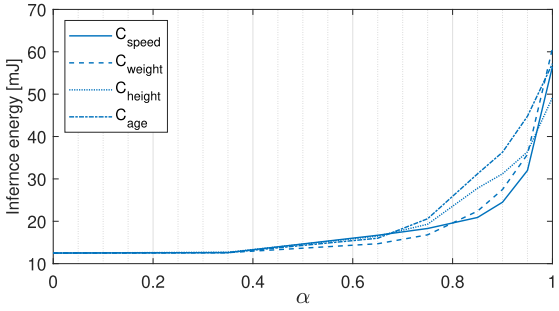



**FIGURE 6. Average energy expenditure required to classify a single sample by the ESP32 MCU for each clustering rule in MotionSense while increasing $\alpha$.**

the role of $\alpha$ as a knob to be dynamically adjusted at runtime to span the accuracy-energy tradeoff; for instance, by controlling the state of charge of the battery of a wearable device, the value of $\alpha$ can be chosen to save energy or to increase accuracy according to the value of energy storage.

To investigate the best value of the $\alpha$ parameter, Figure 7 shows the Pareto chart related to the MotionSense dataset. A Pareto chart is valuable for visualizing the relationship between different cost metrics and identifying the optimal trade-off points. In Figure 7, the energy overhead, expressed as an energy multiplier concerning the baseline, is plotted versus the variation in the misclassification rate ($\Delta_{MCR}$) measured between the baseline and the proposed approach across the range of investigated $\alpha$ values. Notice that a negative value of $\Delta_{MCR}$ represents a gain in classification accuracy, so that the optimal points in the plot are those that simultaneously minimize both the energy overhead and the $\Delta_{MCR}$. For enhanced readability, these points are indicated with red circles, and the corresponding values of $\alpha$ are provided in the plot for each clustering rule. Overall, the proposed approach demonstrates a notable trade-off across all clustering rules, achieving an optimal accuracy improvement ranging from approximately 3.5% to 8%, while maintaining an energy impact that remains within a factor of 2 compared to the baseline. Finally, it is noteworthy that the optimal

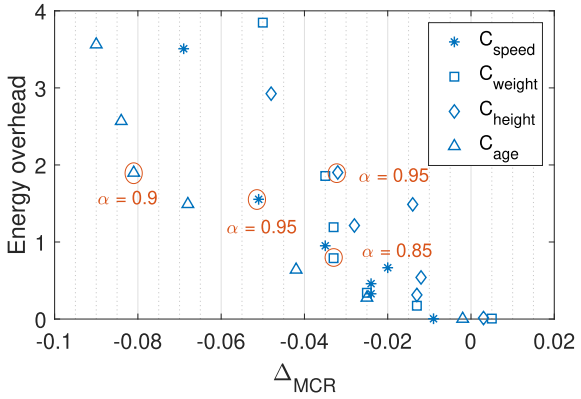



**FIGURE 7.** MotionSense clusters Pareto graphs.

compromise is achieved with high values of $\alpha$, particularly within the range of 0.85 to 0.95.

## VI. LIMITATIONS

There are some limitations that are worth underlining in the presented study. The experimental results refer to a setting with five windows whose length is decided a priori. The spanned range and the level of granularity are reasonable for main HAR applications, but they could clearly change for differing conditions (e.g., different types of activities to be classified), possibly yielding different results. The empirical findings suggest that the proposed approach presents a clear trade-off between energy expenditure and accuracy gain for the embedded platform chosen as reference. While this trade-off is expected to hold also with different hardware platforms, it is not immediate to establish to what extent the energy consumption could be balanced with accuracy in other settings that could therefore be subject to investigation. Finally, we experimented, for the sake of simplicity, only one model that is used for different window sizes, but since different models (e.g., different network architectures) could be used for as many windows, this could in principle lead to improvements w.r.t to a one-size-fits-all solution.

## VII. CONCLUSION AND FUTURE RESEARCH

This study addresses the critical challenge of optimizing the sliding window size used to segment signals in human activity recognition in IoT devices. Recognizing the limitations of conventional fixed-window approaches, we propose a dynamic, inference-driven method tailored for low-power devices. By leveraging classification confidence to adjust the window size adaptively, the proposed method demonstrates improved recognition accuracy and robustness concerning the inter-/intra-subject variability of human activities.

Experimental evaluations on a range of four publicly available datasets validate the effectiveness of this approach, yielding up to a 9% improvement in classification accuracy compared to static window methods while maintaining energy consumption overhead within acceptable bounds. These results highlight the potential of the proposed technique to enhance human activity recognition systems' adaptability and performance in real-world applications, such as health monitoring, fitness tracking, and smart homes.

Several directions could be worth exploring for future work. First, investigate the deployment (and the related performance) of our approach on hardware platforms different from the embedded system that we used for this study; second, tightly interrelated with the previous point, many pruning and optimization techniques can be applied to the neural network architectures adopted, in order to lighten the memory and computation burden.

## REFERENCES

[1] D. R. Beddiar, B. Nini, M. Sabokrou, and A. Hadid, "Vision-based human activity recognition: A survey," *Multimedia Tools Appl.*, vol. 79, nos. 41–42, pp. 30509–30555, Nov. 2020.
[2] D.-A. Nguyen, C. Pham, and N.-A. Le-Khac, "Virtual fusion with contrastive learning for single-sensor-based activity recognition," *IEEE Sensors J.*, vol. 24, no. 15, pp. 25041–25048, Aug. 2024.
[3] A. Bulling, U. Blanke, and B. Schiele, "A tutorial on human activity recognition using body-worn inertial sensors," *ACM Comput. Surveys*, vol. 46, no. 3, pp. 1–33, Jan. 2014.
[4] J.-H. Li, L. Tian, H. Wang, Y. An, K. Wang, and L. Yu, "Segmentation and recognition of basic and transitional activities for continuous physical human activity," *IEEE Access*, vol. 7, pp. 42565–42576, 2019.
[5] S. Irfan, N. Anjum, N. Masood, A. S. Khattak, and N. Ramzan, "A novel hybrid deep learning model for human activity recognition based on transitional activities," *Sensors*, vol. 21, no. 24, p. 8227, Dec. 2021.
[6] H. Najeh, C. Lohr, and B. Leduc, "Dynamic segmentation of sensor events for real-time human activity recognition in a smart home context," *Sensors*, vol. 22, no. 14, p. 5458, Jul. 2022.
[7] A. M. A. Baraka and M. H. Mohd Noor, "Similarity segmentation approach for sensor-based activity recognition," *IEEE Sensors J.*, vol. 23, no. 17, pp. 19704–19716, Sep. 2023.
[8] O. Banos, J.-M. Galvez, M. Damas, H. Pomares, and I. Rojas, "Window size impact in human activity recognition," *Sensors*, vol. 14, no. 4, pp. 6474–6499, Apr. 2014.
[9] J. Liono, A. K. Qin, and F. D. Salim, "Optimal time window for temporal segmentation of sensor streams in multi-activity recognition," in *Proc. 13th Int. Conf. Mobile Ubiquitous Syst., Comput., Netw. Services*, Nov. 2016, pp. 10–19.
[10] A. H. Niazi, D. Yazdansepas, J. L. Gay, F. W. Maier, L. Ramaswamy, K. Rasheed, and M. Buman, "Statistical analysis of window sizes and sampling rates in human activity recognition," in *Proc. 10th Int. Joint Conf. Biomed. Eng. Syst. Technol.*, 2017, pp. 319–325.
[11] G. Wang, Q. Li, L. Wang, W. Wang, M. Wu, and T. Liu, "Impact of sliding window length in indoor human motion modes and pose pattern recognition based on smartphone sensors," *Sensors*, vol. 18, no. 6, p. 1965, Jun. 2018.
[12] F. Duan, T. Zhu, J. Wang, L. Chen, H. Ning, and Y. Wan, "A multitask deep learning approach for sensor-based human activity recognition and segmentation," *IEEE Trans. Instrum. Meas.*, vol. 72, pp. 1–12, 2023.
[13] M. H. M. Noor, Z. Salcic, and K. I.-K. Wang, "Adaptive sliding window segmentation for physical activity recognition using a single tri-axial accelerometer," *Pervas. Mobile Comput.*, vol. 38, pp. 41–59, Jul. 2017.
[14] C. Ma, W. Li, J. Cao, J. Du, Q. Li, and R. Gravina, "Adaptive sliding window based activity recognition for assisted livings," *Inf. Fusion*, vol. 53, pp. 55–65, Jan. 2020.
[15] L. Santos, K. Khoshhal, and J. Dias, "Trajectory-based human action segmentation," *Pattern Recognit.*, vol. 48, no. 2, pp. 568–579, Feb. 2015.
[16] Z. Sheng, C. Hailong, J. Chuan, and Z. Shaojun, "An adaptive time window method for human activity recognition," in *Proc. IEEE 28th Can. Conf. Electr. Comput. Eng. (CCECE)*, May 2015, pp. 1188–1192.
[17] L. Sun, X. Yang, and C. Hu, "DSWHAR: A dynamic sliding window based human activity recognition method," in *Proc. IEEE Smartworld, Ubiquitous Intell. Comput., Scalable Comput. Commun., Digit. Twin, Privacy Comput., Metaverse, Auto. Trusted Vehicles (SmartWorld/UIC/ScalCom/DigitalTwin/PriComp/Meta)*, Dec. 2022, pp. 1421–1426.

[18] N. Alhammad and H. Al-Dossari, "Dynamic segmentation for physical activity recognition using a single wearable sensor," *Appl. Sci.*, vol. 11, no. 6, p. 2633, Mar. 2021.
[19] A. Hirawat, S. Taterh, and T. K. Sharma, "A dynamic window-size based segmentation technique to detect driver entry and exit from a car," *J. King Saud Univ.-Comput. Inf. Sci.*, vol. 34, no. 10, pp. 8514–8522, Nov. 2022.
[20] H. Li, G. D. Abowd, and T. Plötz, "On specialized window lengths and detector based human activity recognition," in *Proc. ACM Int. Symp. Wearable Comput.*, Oct. 2018, pp. 68–71.
[21] K.-S. Lee, S. Chae, and H.-S. Park, "Optimal time-window derivation for human-activity recognition based on convolutional neural networks of repeated rehabilitation motions," in *Proc. IEEE 16th Int. Conf. Rehabil. Robot. (ICORR)*, Jun. 2019, pp. 583–586.
[22] F. Alam, P. Pławiak, A. Almaghthawi, M. R. C. Qazani, S. Mohanty, and A. R. Alizadehsani, "NeuroHAR: A neuroevolutionary method for human activity recognition (HAR) for health monitoring," *IEEE Access*, vol. 12, pp. 112232–112248, 2024.
[23] L. Bigelli, C. Contoli, V. Freschi, and E. Lattanzi, "Privacy preservation in sensor-based human activity recognition through autoencoders for low-power IoT devices," *Internet Things*, vol. 26, Jul. 2024, Art. no. 101189.
[24] C. Contoli and E. Lattanzi, "A study on the application of TensorFlow compression techniques to human activity recognition," *IEEE Access*, vol. 11, pp. 48046–48058, 2023.
[25] D. Gholamiangonabadi, N. Kiselov, and K. Grolinger, "Deep neural networks for human activity recognition with wearable sensors: Leave-one-subject-out cross-validation for model selection," *IEEE Access*, vol. 8, pp. 133982–133994, 2020.
[26] M. Malekzadeh. (2019). *MotionSense Dataset*. [Online]. Available: https://github.com/mmalekzadeh/motion-sense
[27] M. Malekzadeh, R. G. Clegg, A. Cavallaro, and H. Haddadi, "Mobile sensor data anonymization," in *Proc. Int. Conf. Internet Things Design Implement.*, Apr. 2019, pp. 49–58.
[28] MBILab. (2017). *The MobiFall and MobiAct Datasets*. [Online]. Available: https://bmi.hmu.gr/the-mobifall-and-mobiact-datasets-2/
[29] U. M. L. Repository. (2019). *WISDM Smartphone and Smartwatch Activity and Biometrics Dataset*. [Online]. Available: https://archive.ics.uci.edu/dataset/507/wisdm
[30] G. Weiss, "Wisdm smartphone and smartwatch activity and biometrics dataset," UCI Machine Learning Repository, Univ. California, Irvine, CA, USA, Tech. Rep. 507, 2019.
[31] U. Irvine. (2025). *Human Activity Recognition Using Smartphones*. [Online]. Available: https://archive.ics.uci.edu/dataset/240/human
[32] J.-L. Reyes-Ortiz, L. Oneto, A. Samà, X. Parra, and D. Anguita, "Transition-aware human activity recognition using smartphones," *Neurocomputing*, vol. 171, pp. 754–767, Apr. 2015.
[33] *Esp32-c3-wroom-02 Datasheet*, Espressif, Shanghai, China, 2022.
[34] *Ngmo2 Datasheet*, Rohde-&-Schwarz, Munich, Germany, 2024.
[35] *Pc-6251 Datasheet*, National-Instruments, Austin, TX, USA, 2024.
[36] *Installation Guide BNC-2120*, National-Instruments, Austin, TX, USA, 2024.
[37] J. Reyes-Ortiz, D. Anguita, L. Oneto, and X. Parra, "Smartphone-based recognition of human activities and postural transitions," Univ. California, Irvine, CA, USA, Tech. Rep. 341, 2015, doi: 10.24432/C54G7M.
[38] O. Baños, R. B. García, J. A. Holgado-Terriza, M. Damas, H. Pomares, I. Rojas, A. Sáez, and C. Villalonga, "MHealthDroid: A novel framework for agile development of mobile health applications," in *Proc. Int. workshop ambient Assist. living*, 2014, pp. 91–98.

**NICHOLAS KANIA** received the bachelor's degree in applied computer science from the University of Urbino, Italy, in early 2023, where he is currently pursuing the master's degree. Since June 2023, he has been doing research on machine learning technologies and low-power embedded devices with the University of Urbino. His research interests include machine learning, the Internet of Things, embedded systems, and vision-based systems.

**CHIARA CONTOLI** received the degree in computer engineering and the Ph.D. degree in electronics, telecommunications, and information technologies engineering from the University of Bologna, Italy, in 2013 and 2017, respectively. For two years, she was a Software Developer in the industry. She is currently a tenure-track Assistant Professor of computer engineering with the Department of Pure and Applied Sciences (DiSPeA), University of Urbino, Italy. Her research interests include network management and network softwarization, network architectures and protocols, the Internet of Things, machine learning, and power management. She is also interested in programming languages for networks and enjoys working on interdisciplinary problems.

**VALERIO FRESCHI** received the Laurea degree in electronic engineering from the University of Ancona, Italy, in 1999, and the Ph.D. degree in computer science engineering from the University of Ferrara, Italy, in 2006. He is currently an Associate Professor of computer engineering with the Department of Pure and Applied Sciences (DiSPeA), University of Urbino, Italy. His research interests include energy-efficient algorithms, optimization, and machine learning.

**EMANUELE LATTANZI** received the Laurea degree, in 2001, and the Ph.D. degree from the University of Urbino, Italy, in 2005. In 2003, he was with the Department of Computer Science and Engineering, The Pennsylvania State University, as a Visiting Scholar with Prof. V. Narayanan. Since 2020, he has been an Associate Professor of computer engineering with the Department of Pure and Applied Sciences (DiSPeA), University of Urbino. His research interests include the Internet of Things, energy-aware embedded programming, machine learning, and human activity recognition.