Optimizing IoT-based Human Activity Recognition on Extreme Edge Devices

Optimizing IoT-based Human Activity Recognition on Extreme Edge Devices

Angelo Trotta*, Federico Montori*†, Giacomo Vallasciani*, Luciano Bononi*, Marco Di Felice*†
* *Department of Computer Science and Engineering, University of Bologna, Italy*
† *Advanced Research Center on Electronic Systems "Ercole De Castro", University of Bologna, Italy*
Emails: {angelo.trotta5, federico.montori, luciano.bononi, marco.difelice3}@unibo.it, giacomo.vallasciani@studio.unibo.it

***Abstract*—Wearable Internet of Things (IoT) devices with inertial sensors can enable personalized and fine-grained Human Activity Recognition (HAR). While activity classification on the Extreme Edge (EE) can reduce latency and maximize user privacy, it must tackle the unique challenges posed by the constrained environment. Indeed, Deep Learning (DL) techniques may not be applicable, and data processing can become burdensome due to the lack of input systems. In this paper, we address those issues by proposing, implementing, and validating an EE-aware HAR system. Our system incorporates a feature selection mechanism to reduce the data dimensionality in input, and an unsupervised feature separation and classification technique based on Self-Organizing Maps (SOMs). We developed the system on an M5Stack IoT prototype board and implemented a new SOM library for the Arduino SDK. Experimental results on two HAR datasets show that our proposed solution is able to overcome other unsupervised approaches and achieve performance close to state-of-art DL techniques while generating a model small enough to fit the limited memory capabilities of EE devices.**

***Index Terms*—Human Activity Recognition, HAR, Unsupervised Learning, Self Organizing Maps, IoT, Edge Computing**

I. Introduction

Human Activity Recognition (HAR) has gained significant attention in recent years due to its potential to improve the quality of life of individuals in many different use cases, including healthcare, fitness, and entertainment [1]. HAR systems can also facilitate advanced, context-aware services in smart home and smart city scenarios [2]. These systems aim to automatically identify and classify human activities using either cameras [3] or inertial measurement units (IMUs). While camera-based solutions offer high accuracy, concerns about privacy and cost limit their applicability on a large scale. Conversely, IMU-based solutions leverage low-power sensors such as accelerometers, gyroscopes, and magnetometers, which are commonly embedded in mobile and wearable devices [4].
Smartphone-based HAR systems have been extensively studied in the literature. At the same time, their application is typically restricted to specific classes of human activities, due to the device's orientation variability and pattern of usage. Recent research has focused on integrating HAR functionalities within the Internet of Things (IoT) wearable devices [2], whose market has grown exponentially in the last few years. Besides well-known devices, such as smart bracelets and smartwatches, the miniaturization of the hardware circuitry is expected to open the way to a new generation of Internet of Small Things (IoST) [5] that supports fine-grained activity monitoring. The integration of HAR functionalities into such devices is of paramount importance to support real-time operations, e.g., healthcare applications. However, the deployment of IoT-based HAR systems poses new and significant research challenges that have not been addressed in other domains. First, the classification of human activities is often performed using data-driven approaches, and Deep Learning (DL) architectures have been widely employed for this task [4]. Nevertheless, these solutions do not fit the characteristics of power- and memory-constrained micro-controllers. Although quantization techniques have been proposed to reduce the size of the trained DL model, such as reducing the precision of the coefficients [6] [7], the research on edge intelligence for micro-controllers can be considered still at a preliminary stage [8]. Second, supervised learning techniques assume labeled data in input. The process of data labeling is time-consuming and costly for human operators and it can become even more burdensome in the case of IoT wearable devices due to the lack of input systems. For this reason, recent works have investigated the usage of semi-supervised or unsupervised techniques, enabling the automatic detection of patterns associated with different activities [9]–[12].
In this paper, we investigate the full design and implementation of a HAR system for resource-constrained IoT micro-controllers, which we refer to as Extreme Edge (EE) devices. Our solution involves three stages. First, feature selection is applied to IMU data, in order to reduce its dimensionality since only a few of them can be correlated with the activities to detect. Second, the unlabeled features are first separated and then classified through Self-Organizing Maps (SOM), an unsupervised machine learning (ML) technique based on lightweight artificial neural networks. During the training phase, the user does not need to label each sample, and the set of activities to detect can be easily expanded and personalized. Finally, the trained SOM model is deployed on an IoT computing board (in our case, a M5Stack embedding an Espressif ESP32 micro-controller), to run inference on the EE. Three contributions are proposed in this paper:

- We present a novel architecture for EE-HAR, which involves offline training and onboard inference. Our architecture uses ANOVA-F-based feature reduction and SOM algorithm for human activities separation and classification.

- We describe the implementation of the proposed system on the EE device. To achieve this, we developed a novel SOM library for the Arduino SDK in C++ language.
- We extensively validate the proposed architecture on two HAR datasets, i.e. the UCI one and a new dataset collected through the target EE device. We compared our solution against state-of-art supervised, DL-based solutions and another unsupervised technique (K-Means).

The results show that the SOM-based approach greatly overcomes the K-Means in terms of average accuracy, for both datasets. Moreover, the performance loss compared to complex DL approaches is less than 8%, which is a satisfactory result considering the difference in complexity between the two approaches. Furthermore, our SOM implementation in C++ language can achieve up to five-time model size reduction compared to state-of-art approaches [13].

The rest of this paper is organized as follows: Section II provides an overview of related work in the field of HAR systems focusing on unsupervised and edge AI solutions. Section III shows the methodology used for data collection, processing, and classification. Section IV describes the learning model. Section V presents the results of our experiments and compares our proposed approach with existing methods. Section VI concludes the paper and discusses future work.

II. Related Works

The research on HAR systems spans more than 15 years and includes a considerable variety of solutions and applications. Generally speaking, most approaches fall into two main categories, i.e. knowledge-based or data-driven ones. In the first case, semantic correlations among event types and activity classes are detected [14]. In the second case, datasets of sensory data are first collected and then mined. Activity recognition can be considered an instance of pattern-matching problems, and several DL techniques have been proposed to tackle it [4]. At the same time, supervised solutions require labeled data to enable the training phase. Although several datasets are available for basic activities, novel use cases of HAR systems continuously demand fine-grained and personalized data collection, making the labeling process costly and unfeasible on the large scale. Indeed, recent studies investigate the issue of user generalization caused by the unique activity patterns of each individual user and, consequently, propose unsupervised or semi-supervised techniques [11]. The authors of [10] describe a DL-based clustering architecture performing unsupervised learning and label annotation of multi-dimensional inertial signals. Their architecture follows a two-step approach: first, a recurrent auto-encoder extracts the spatiotemporal features of the human activities, then a clustering technique is applied to predict unlabeled signals. A survey was conducted in [9] related to clustering techniques for the recognition of activities of daily living. Regarding semi-supervised approaches, Unsupervised Domain Adaption (UDA) techniques allow new users to align their unlabeled data to the features of labeled datasets. The SALIENCE architecture described in [11] uses both per-feature and per-sample discriminators to predict whether an input sample is from training users or a new user; then, an attention-based neural network allows to differentiate the importance of different sensors based on the outputs of the previous step, so the sensors with strong feature discrimination are prioritized. In [15], the authors investigate UDA techniques in heterogeneous feature spaces; the proposed architecture combines Bi-directional Generative Adversarial Networks (Bi-GAN) and Kernel Mean Matching (KMM) to enable activity transfer across heterogeneous HAR datasets, including accelerometer and binary sensors. A key contribution of our paper is constituted by the implementation of HAR systems on wearable IoT devices that are resource- and power-constrained and thus unable to host complex DL models. To this aim, the emerging edge intelligence paradigm [8] promises to bring the data analytics tasks as close as possible to the data sources, e.g., in our case, integrated within wearable IoT devices. The preliminary study in [6] applies quantization techniques to reduce the size of a Convolutional Neural Network (CNN) devoted to human activity classification in order to fit the memory constraints of a micro-controller unit. Similarly, in [7] different architectures of binarized neural networks (i.e, neural networks with weights and functions restricted to binary values) are compared on HAR datasets. The most similar works to ours are [12] and [13]. In the first paper, the authors investigate daily gesture detection from accelerometer data produced by an IoT bracelet: unsupervised approaches like K-Means and Gaussian Mixture Model are shown to produce comparable results to supervised techniques. In [13], the authors address a similar problem, but employ a SOM and a CNN for feature extraction; also, they deploy the HAR system on a micro-controller unit. Our results confirm the main findings in [13], however, introducing the following contributions: (*i*) we present the implementation of a novel SOM library for the Arduino SDK through which we achieve a considerable reduction of the model size compared to [13]; (*ii*) we introduce a pre-processing mechanism, constituted by the ANOVA-F feature selection, which allows to further reduce the amount of data processed on the EE; (*iii*) we evaluate the trade-off between the SOM accuracy and the model size for different configurations of both the techniques.

III. System Architecture

In this Section, we introduce our HAR system for wearable IoT devices. The proposed architecture tackles the challenges of EE computing related to limited computational power, storage, and data processing capabilities. In addition, it has been designed in order to meet the following requirements:

- *flexibility*: users can define and add new activities at runtime;
- *customizability*: the system is adaptable, allowing users to personalize the training phase;
- *ease of use*: the system minimizes the need for manual data labeling, thereby simplifying the overall user experience.

To meet these goals, our architecture includes two classes of computational nodes, the IoT wearable device, and an external server. Furthermore, we have split the HAR process into three distinct phases: data gathering, model training, and HAR inference. The data gathering and HAR inference phases are executed on the wearable IoT device. Vice versa, the model training phase is performed on an external server due to its need for computational and storage resources that the microcontroller cannot provide.

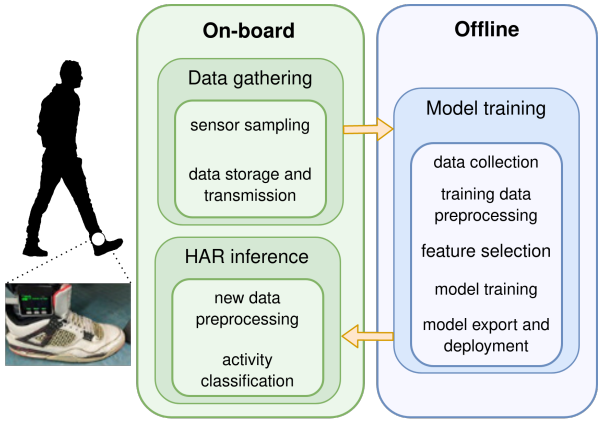


Fig. 1: The proposed HAR architecture

The proposed HAR architecture is described in Figure 1. In the following, we provide additional details for each phase.

- *Data gathering*: during this phase, raw sensor data are sampled on the IoT device while the user is doing actions to be detected. The proposed HAR system utilizes IMU sensors, such as accelerometers, gyroscopes, and magnetometers, which are sampled at a fixed frequency. No user feedback is required at this stage to label the data. Raw IMU data is wirelessly transferred to the external server where unlabeled HAR dataset(s) are built.
- *Model training*: this phase involves training an ML model from the HAR dataset collected from the IoT device(s). First, the IMU data is pre-processed to remove outliers and missing values. Also, techniques based on the Butterworth low-pass filter are employed to filter out the noise. Next, features are extracted from each sensor in the frequency and temporal domain, as further detailed in Section V-B. ML models are then trained from the pre-processed datasets. In this article, we rely on the SOM technique described in Section IV-A. The SOM algorithm separates the data into K classes, where K is either provided as inputs or dynamically estimated from data using the elbow method [16]. The association between the class and its semantics (i.e., the name of the human activity) can be provided at this stage. We remark that here we are labeling each cluster after the unsupervised method has been applied, not each training example as in supervised learning. We have further optimized the data pipeline by implementing a feature selection stage aimed at reducing the number of features in input to the SOM, and hence the processing load on the EE. The feature selection is implemented through the ANOVA-F technique further detailed in Section IV-B. Finally, the trained model, with the list of selected features, is transferred back to the IoT device. Multiple SOM models can be generated for different network sizes, and the selected model should maximize the accuracy while fitting the storage size of the IoT device.
- *HAR Inference*: in operation mode, the trained SOM model is automatically offloaded to the IoT device. The features selected in the previous stage are extracted from the IMU samples and provided as input to the SOM for the classification of the human activity. The HAR output can be notified to the user through output peripherals such as a screen, or to other devices. We do not elaborate on context dissemination as it is outside the scope of this paper.

IV. Learning Models for the EE

In this Section, we present the ML and pre-processing techniques used in the model training block of Figure 1. We rely on Self Organizing Maps (SOMs) for human activity classification and the ANOVA-F technique for feature reduction.

A. Self-Organizing Map

The self-organizing map (SOM) [17] is a type of unsupervised learning algorithm that can be used for dimensionality reduction and clustering. SOMs are a form of artificial neural network that can map high-dimensional input data onto a lower-dimensional output space while preserving the topological properties of the input data. SOMs are lightweight and computationally efficient, making them well-suited for deployment on constrained devices [13]. In our HAR system, we used SOMs as ML techniques to address the HAR task for multiple reasons. Firstly, they are lightweight and computationally efficient, making them well-suited for deployment on constrained devices with limited computational power and storage capacity. This is in contrast to other models such as KNN, FNN, and SVM, which can have high storage requirements and long computational times. KNN, for example, requires storing all training data, making it unsuitable for large datasets or constrained devices with limited storage capacity. FNN, on the other hand, has a complex implementation of the transfer function, which can result in high computational times and energy consumption. Similarly, SVM has a complex implementation of the kernel function, which can make it unsuitable for deployment on constrained devices [18]. Secondly, SOMs are well-suited for unsupervised learning, which is particularly useful for HAR systems, where labeled training data may be scarce or expensive to collect, or simply not tailored to the device owner. Finally, SOMs can be easily customized for specific sensor data formats or feature reduction methods, which is essential for deployment on constrained wearable devices. By optimizing the SOM model for the specific constraints of the device, we can achieve

high performance while minimizing energy consumption and storage usage.

More in detail, a SOM is a distinct type of artificial neural network that employs competitive learning for training. In this approach, nodes compete for the opportunity to respond to specific subsets of input data, as opposed to error-correction learning methods (e.g., backpropagation with gradient descent) utilized by other artificial neural networks. Like most artificial neural networks, SOMs function in two primary modes: training and mapping. The training phase uses an input data set, or "input space", to create a lower-dimensional representation known as the "map space". The mapping phase then employs this generated map to classify additional input data. Typically, the training process aims to represent an input space with *p* dimensions as a two-dimensional map space. An input space consists of *p* features, each representing a dimension. The map space is composed of *nodes* or *neurons*, which are organized in a two-dimensional rectangular grid of size $N \times N$. Each node n_i in the map space in the grid can be hence identified via its coordinates $\langle x_i, y_i \rangle$, with $0 < i < N$. The node n_i is linked to a weight vector w_i of *p* dimensions, that signifies the node's position in the input space. Although nodes in the map space remain stationary, the training process involves adjusting weight vectors towards the input data (by minimizing a distance metric such as *Euclidean* distance) without disrupting the topology derived from the map space. Once trained, the map can classify additional observations in the input space by identifying the node with the nearest weight vector (i.e., the smallest distance metric) to the input space. During the initialization of the SOM algorithm, the neurons' weights are assigned random values. Then, iterating over the input data, for each training example *a*, the Best Matching Unit (BMU) is calculated:

$$\text{bmu} = \arg\min_i \{||a - w_i||\}$$

The weight of the BMU node, and its neighborhood, is then updated as follows:

$$w_i = w_i + \eta(t) \cdot h_{i,\text{bmu}}(t) \cdot (a - w_i)$$

Here, $\eta(t)$ is the learning rate at learning iteration t defined with a learning rate decay rule $\eta(t) = \eta_0 \cdot e^{(-t/\hat{\eta})}$, with η_0 and $\hat{\eta}$ as custom parameters. The neighborhood kernel function $h_{i,\text{bmu}}(t)$ defines the neighborhood size and is defined by:

$$h_{i,\text{bmu}}(t) = e^{-\frac{d^2_{i,\text{bmu}}}{2 \cdot \theta^2(t)}}$$

where $\theta(t) = \theta_0 \cdot e^{-t/\hat{\theta}}$ defines the neighborhood size decay rule, with θ_0 and $\hat{\theta}$ as custom parameters. The distance $d_{i,j}$ between two nodes is calculated as a Manhattan distance, i.e., $d_{i,j} = |x_i - x_j| + |y_i - y_j|$.

The critical factors for deployment on a constrained EE device are determined by the number of weights w_i. The size of the SOM map is thus defined by the number of neurons, i.e., the grid $N \times N$, and the number of features p. Consequently, the total learning model deployment size is $\mathcal{O}(N^2 \cdot p)$. In the following section, we will discuss a feature reduction mechanism to decrease the value of p. Additionally, in Section V-E, we will assess various SOM sizes to attain satisfactory classification outcomes while enabling model deployment on constrained EE devices.

B. ANOVA-F Feature Reduction

Feature reduction is an essential step in building efficient ML models for HAR. Indeed, complex ML models tend to use all possible features as inputs because they are designed to run on computers or other devices that are not constrained to low resources. In fact, most of the ML algorithms are designed to internally disregard features that have little to no influence in discriminating the predicted class. However, since wearable devices have limited computational power and storage capacity, it is crucial to reduce the number of features *before* building the final model without compromising its accuracy. Numerous methods can be employed for feature reduction, such as Principal Component Analysis (PCA), which transforms the feature space by combining the original features into distinct aggregations. However, this approach demands additional computation and storage during the final inference stage. In the proposed HAR system illustrated in Figure 1, this phase would have to take place on the wearable device, rendering it unsuitable for extreme edge deployment.

In the EE scenario, we chose a static method that is able to reduce offline the number of features: the ANOVA-F technique, which can be conducted a priori and not during the inference phase, making it suitable to run outside the device. This technique is a statistical method used for feature selection that can identify the most significant features for the model [19]. As a matter of fact, in HAR scenarios, we could achieve the same accuracy of a complete model by only selecting a subset of highly significant features. In order to calculate the ANOVA-F value of a feature f we first calculate, for each of the K classes, the variance of that feature within such class, then we divide it by the variance of such feature over the whole dataset. More formally, The ANOVA-F value of feature f for class $c \in K$ is given by:

$$\text{An}_c(f) = \frac{\sigma^2(f)_c}{\sigma^2(f)}$$

Ideally, a low value of $\text{An}_c(f)$ means that f is highly discriminant for class c, because the values of f do not change significantly among the members of class c compared to how much they change normally in the dataset. Conversely, values of $\text{An}_c(f)$ that approach 1.0 or above signify that feature f is likely to just add noise for class c. Next, we would need to come up with a single ANOVA-F value for each feature f, for this reason, we run an aggregation function over all instances of ANOVA-F values of f for all K classes. We conducted our experiments by selecting the *average* and the *minimum* as aggregation functions, which are defined as follows:

$$\text{An}_{avg}(f) = \frac{\sum_{c \in C} \text{An}_c(f)}{K},$$
$$\text{An}_{min}(f) = min\{\text{An}_c(f) | c \in K\}$$

The first associates each feature with an ANOVA-F value by taking into account its overall influence over all classes, while the second only considers the class for which the feature is most influential. Once calculated, we would ideally rank all the p features in ascending order by their ANOVA-F value and keep only the ones ranked best. Therefore, we introduce the ANOVA-F Threshold (An_{thr}), which is the ANOVA-F value below which we keep all the features. This variable becomes a parameter of our system, which depends on the available space on the device and has to be balanced with other parameters in order to achieve the best possible accuracy. The ANOVA-F technique can then be executed offline, so that the considered features are known a priori and, in the inference phase, only such features are computed by the device.

V. Performance Evaluation

In this Section, we evaluate the performance of the proposed HAR system for resource-constrained IoT devices. The evaluation includes five stages: (*i*) implementing the system on the microcontroller, (*ii*) defining the HAR datasets, (*iii*) analyzing the supervised learning techniques on the HAR datasets to establish a reference baseline, (*iv*) assessing the effectiveness of the ANOVA-F technique for feature reduction, and (*v*) analyzing the SOM on the two datasets, to evaluate the accuracy and memory size occupation for different map sizes.

A. Implementation on the IoT device

The deployment of the HAR system depicted in Figure 1 is based on the *M5Stack*[1], which is a modular, stackable, and programmable device designed to support pervasive IoT applications. It is equipped with multiple IMU sensors, including an accelerometer, gyroscope, and magnetometer, making it ideal for HAR data collection tasks. The computational unit is based on the popular ESP32 microcontroller with 240MHz dual-core processors and 320KB of RAM memory. We choose this device to validate our HAR architecture due to its compact and energy-efficient profile; however, we emphasize that our firmware can be easily adapted for other IoT boards that support the Arduino SDK.

During the experiments, we positioned the device on users' ankles as this location provides optimal gait analysis, as reported by [20]. Generally speaking, the development of ML models, including the SOM, on micro-controllers is highly challenging and requires optimizing the code in order to fit the memory constraints, which, for the case of the M5Stack, is less than 400 KB. Traditional approaches involve quantization techniques [8], which aim to reduce the size of an ML trained out of the microcontroller by reducing the precision of floating point coefficients [6]. To this aim, frameworks like TensorFlow Lite [21] support the quantization process and the exportation of the model in a binary format; the latter is loaded by an interpreter running on the microcontroller. However, we were unable to follow such an approach as the generated code exceeded the memory capacity of the M5Stack device.

To overcome these challenges, we opted to implement a SOM library from scratch in *C++* for micro-controller, by taking into account the hardware characteristics of the latter. The SOM library for the Arduino SDK is available online[2] and can be easily customized to support different micro-processors than the ESP-32 one.

Through our implementation, the size of the final deployed SOM model is restricted to several tens of kilobytes. The M5Stack has 320 KB of RAM memory, and after an initial code optimization phase that encompasses sensor reading and wireless communication, the maximum memory available for the SOM model is 140 KB only. As described in Section IV-A, the SOM size depends on two variables: the number of features and the map size. In Figure 2a we depict the relationship between these two variables and the final size of the SOM model to be installed in the wearable device. In the Figure, only feasible configurations are shown. In Section V-E we investigate the trade-off between number of input features, map size and overall accuracy of the HAR task.

B. Description of datasets

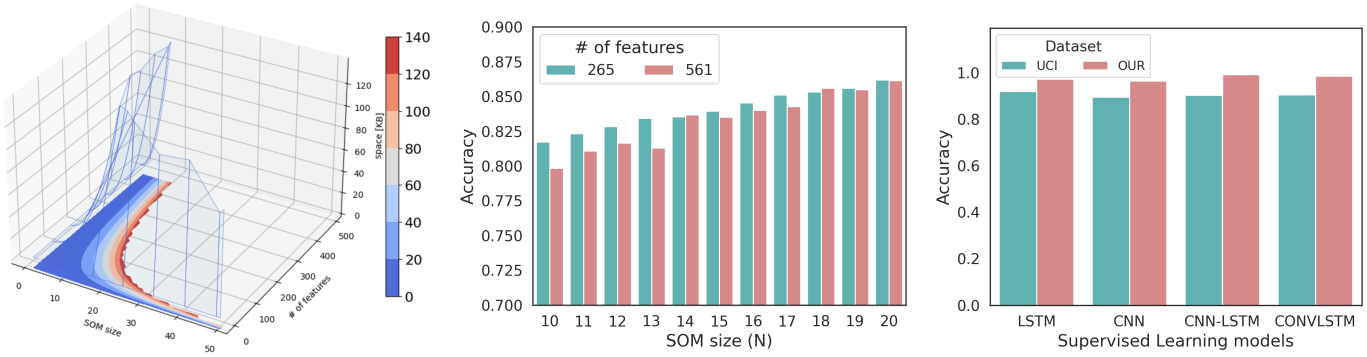
We evaluated our proposed HAR system on two different datasets, one from the literature and a custom one gathered by using the *M5Stack* device.

Regarding the first, we used the well-known *University of California Irvine (UCI) HAR* dataset [22]. It was collected from the accelerometer and gyroscope sensors of a smartphone carried by volunteers performing six different activities: walking, walking upstairs, walking downstairs, sitting, standing, and laying down. The dataset includes time-series data for the three axes of each sensor, as well as the corresponding activity label. The dataset has been widely used in research for developing and evaluating HAR models based on ML techniques. More in detail, each record in the dataset is provided with a 561-feature vector with time and frequency domain variables. The database's features were sourced from the accelerometer and gyroscope 3-axial raw signals and were captured at a constant rate of 50Hz. To eliminate noise, a median filter and a 3rd-order low pass *Butterworth filter* with a corner frequency of 20Hz were employed. Another low-pass Butterworth filter with a corner frequency of 0.3Hz was used to split the acceleration signal into body and gravity acceleration signals. After that, the body's linear acceleration and angular velocity were used to derive *Jerk signals* in time. The magnitude of these three-dimensional signals was determined using the Euclidean norm. Finally, a Fast Fourier Transform (FFT) was applied to some of these signals.

However, not all of these features are equally essential for accurate HAR systems. Some may be redundant or noisy, while others may be more informative. Our study evaluated the UCI dataset's performance using different feature subsets and found that the frequency-based features did not significantly improve our HAR system's performance. Figure 2b demonstrates that

[1]https://docs.m5stack.com/en/core/gray

[2]https://github.com/UniBO-PRISMLab/extreme-edge-som



(a) (b) (c)

Fig. 2: 2a shows the impact of the SOM size and the number of features on the deployed firmware. 2b compares the accuracy for the UCI dataset, when using all 561 features or a subset of 265 features only. 2c shows a comparison between different supervised techniques over the two datasets.

the prediction accuracy using the entire set of 561 features and a subset of 265 features, which excludes frequency-domain inputs, behaved quite similarly for classification accuracy. As a result, we decided to remove the frequency-based features and focus solely on time-domain and statistical features in our deployment.

We used the same selected features and built our custom dataset by collecting data from our wearable device installed on multiple users. The M5Stack device was installed on the user's ankle with a sampling rate of 40Hz to collect sensor data. We applied the same procedures used on the UCI dataset, such as Butterworth filters and Jerk calculation. For the custom dataset, in order to perform a meaningful comparison, we used the exact same classes as the UCI dataset.

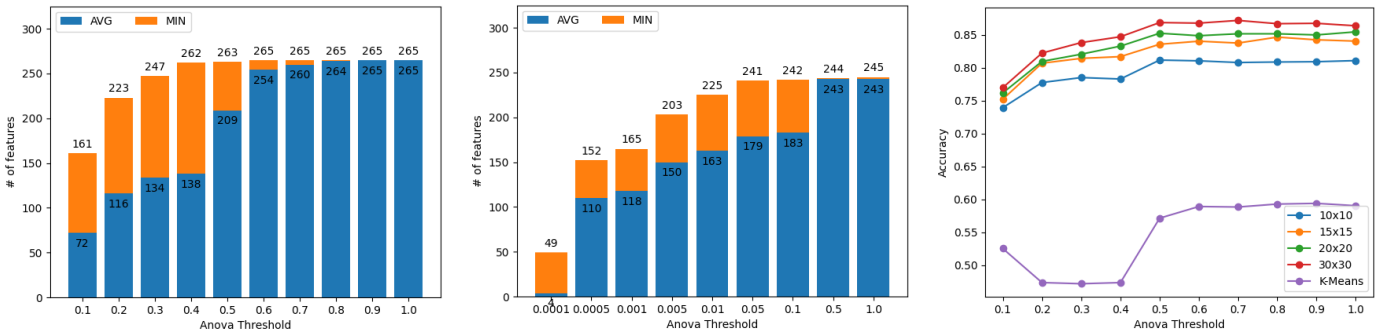
C. Supervised learning for HAR

In this Section, we show the performance of supervised learning methods for HAR applications. These techniques rely on labeled training data to build a model that can predict the activity performed by the user, while unsupervised learning techniques aim to discover patterns or structures in the data without any labeled information. Due to the constraints of the EE device used in our HAR system, we will focus primarily on evaluating the performance of unsupervised learning techniques in the following Sections. However, to have a meaningful comparison perspective, we now analyze different supervised learning techniques for HAR applications as a baseline. Specifically, we evaluated: long short-term memory (LSTM), convolutional neural network (CNN), CNN-LSTM, and convolutional LSTM (CONVLSTM).

Figure 2c shows that supervised learning techniques are very effective in classifying human activities, with all four models achieving high accuracy. In the Figure are depicted the results for both datasets. Here, the CNN-LSTM model achieved the highest overall performance, with an accuracy of 99% with our custom dataset. This evaluation shows also the effectiveness of placing the sensors in the right position, i.e., the ankle, instead of using the sensors inside the smartphone. The results of the supervised learning techniques over the UCI dataset are around 8% less accurate than the one executed on our dataset.
However, while supervised learning methods are highly effective in the classification of HAR systems, they are not suitable for deployment in EE devices. Our M5Stack device, in fact, has 140KB available memory only. On the contrary, the supervised learning models require significant computational resources and storage capacity.

D. Features reduction with ANOVA-F

In this Section, we present the results of the ANOVA-F feature reduction method on both the UCI HAR dataset and our custom dataset. We analyzed the two different variations of the ANOVA-F method described in Section IV-B: one using the average as the aggregation function and another using the minimum as the aggregation function, in the following called *AVG* and *MIN*, respectively. The aim of this analysis was to identify the most relevant features for our ML model and reduce their number to minimize storage usage, as requested in previous Section V-A. Our results show that both variations of the ANOVA-F method were effective in reducing the number of features used in our classification model. Figures 3a and 3b display the outcomes of features reduction obtained by applying distinct An_{thr} on both the UCI and our custom datasets. It should be noted that the scales on the x-axes are different in the Figures (UCI dataset in Figure 3a and our dataset in Figure 3b). This disparity in scaling is due to the fact that in our custom dataset, setting the An_{thr} to 0.5 resulted in almost all of the input features being included without any feature reduction. Therefore, in order to evaluate our dataset, we used exponential steps for the An_{thr}.
The results show that the variation using *AVG* as the aggregation function resulted in a greater reduction in the number of features with respect to the *MIN* version, especially for low levels of the An_{thr}. Based on these results, we selected the *AVG* variation of the ANOVA-F method for the next evaluations. This is because by using this method we can achieve a significant reduction in the number of features used in our model.



(a) (b) (c)

Fig. 3: Feature reduction using *AVG* and *MIN* for different value of An_{thr} applied to the UCI dataset (3a) and for our dataset (3b). The accuracy evaluated with *K-Means*, different SOM sizes by varying An_{thr} for the Uci dataset is shown in Figure 3c.

E. SOM for Extreme Edge Devices

In this Section, we present the evaluation process undertaken to determine the optimal configuration for the SOM and the ANOVA-F threshold (An_{thr}) in our HAR model. Our primary goal was to achieve a satisfactory classification performance while maintaining a compact model size that is suitable for deployment on the M5Stack device. To ensure the robustness and generalizability of our model, we used the two datasets for the evaluation process. Additionally, we analyze the performance of SOMs with the legacy *K-Means* algorithm for comparison purposes. We conducted a series of experiments to assess the performance of our HAR model by varying the size of the SOM and the An_{thr}. The SOM sizes tested ranged from small (e.g., 10x10) to larger maps (e.g., 30x30), while the An_{thr} values were varied between 0.0001 and 1 for our dataset and between 0.1 and 1 for the UCI dataset. This range of values was chosen to cover a wide spectrum of model complexities, thereby providing a comprehensive understanding of the trade-off between model size and performance.

Upon analyzing the results in Figures 3c and 4a from both the UCI HAR dataset and the custom dataset, we observed that the HAR model's performance improved with increasing SOM size. However, for sizes greater than 15×15, the increase in accuracy is not so evident. Regarding An_{thr}, we notice a decrease in performance for low values in both datasets and for high values only in our custom dataset. Here, we have that the optimal performance is different for the two datasets. The UCI dataset performs well starting from An_{thr} of 0.5, i.e., from a number of features of 209 (see Figure 3a). Our custom dataset performs well with An_{thr} values between 0.0005 and 0.1, i.e., with a number of features between 110 and 183. It is clear that with larger SOMs and higher An_{thr}, and hence more features, the model would also demand more computational resources and storage, which may not be available on the M5Stack device.

Taking these considerations into account, we determined that the optimal configuration for our HAR model is a SOM size of 15×15 and An_{thr} of 0.005. For the deployment, we considered the results from our custom dataset. This configuration demonstrated a satisfactory balance between model accuracy and computational complexity, making it well-suited for deployment on our wearable device which has a limit of 140KB. Table I presents the accuracy values and deployment sizes for distinctive configuration values for SOM map size and the number of features, using only our custom dataset.

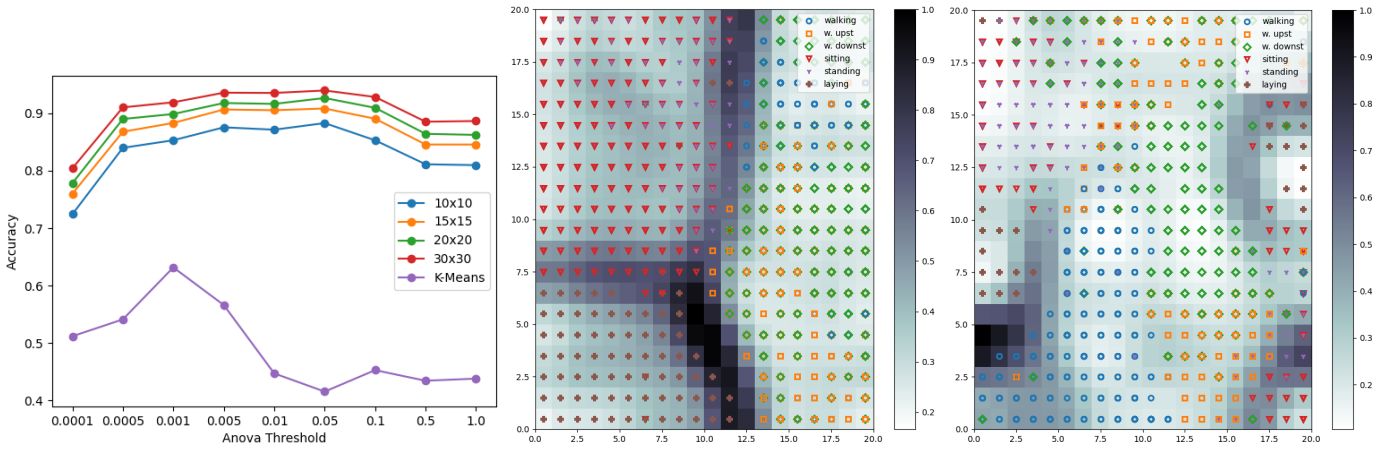
SOM size	An_{thr}	# of features	Accuracy	Size in KB
10×10	1	243	0.81	95.4
15×15	0.005	150	0.905	132.1
15×15	0.01	163	0.905	143.6
15×15	0.05	179	0.91	157.7
20×20	0.0001	4	0.784	6.2
20×20	0.0005	110	0.893	172.1

TABLE I: Detailed results for accuracy and model deployment size with different configurations. Green and red colors in the last column indicate if the model can or cannot be deployed in our device, respectively.

Lastly, comparing the performance of SOMs to that of the K-Means algorithm, we found that SOMs outperformed K-Means for both datasets in terms of accuracy. This is likely due to the ability of SOMs to preserve the topological structure of the data, allowing for more effective clustering and recognition of human activities. In Figures 4b-4c are shown a graphical view of the resulting SOM after the training phase for both datasets, using a SOM of size 20×20. It is clear how this method is able to topologically divide the different classes into two dimensions. These results emphasize the potential of SOMs as a suitable ML model for HAR applications on constrained wearable devices.

VI. Conclusions

In this paper, we have presented the design, implementation and validation of an Extreme Edge (EE)-aware Human Activity Recognition (HAR) system. The system aims at detecting human activity on micro-controlled based, wearable IoT devices equipped with inertial sensor, by taking into account the unique challenges posed by the computational environment, such as the constrained resources and the impracticability of a data labeling phase. Our HAR system incorporates a feature selection mechanism, based on the ANOVA-F technique, to reduce the dimensionality of HAR features at the input stage. Then, it utilizes an unsupervised classification technique based



(a) (b) (c)

Fig. 4: The accuracy evaluated with *K-Means*, different SOM sizes by varying An_{thr} for our custom dataset is shown in Figure 4a. Figures 4b and 4c show a visual 2D representation of the SOM neurons, for the UCI dataset and our custom dataset, respectively. Here, darker areas define the separation between different classes.

on Self-Organizing Maps (SOMs) to enable data separation and effective activity classification. We developed a C++ SOM library for the Arduino SDK, and validated our system through an M5Stack IoT wearable board. The experimental results on two HAR datasets, the UCI HAR dataset and a custom dataset built through our IoT prototype, demonstrated that the proposed SOM solution is capable of outperforming other unsupervised approaches. Furthermore, our system achieved performance close to state-of-the-art DL techniques while generating a model small enough to fit the limited memory capabilities of EE devices. Future work may involve further optimization of the system, exploration of alternative feature selection techniques, and the adaptation of our approach to other wearable devices and application domains.

References

[1] Y. Chen, L. Yu, K. Ota, and M. Dong, "Robust activity recognition for aging society," *IEEE Journal of Biomedical and Health Informatics*, vol. 22, no. 6, pp. 1754–1764, 2018.

[2] D. Bouchabou, S. M. Nguyen, C. Lohr, B. LeDuc, and I. Kanellos, "A survey of human activity recognition in smart homes based on iot sensors algorithms: Taxonomies, challenges, and opportunities with deep learning," *Sensors*, vol. 21, no. 18, 2021. [Online]. Available: https://www.mdpi.com/1424-8220/21/18/6037

[3] D. R. Beddiar, B. Nini, M. Sabokrou, and A. Hadid, "Vision-based human activity recognition: A survey," *Multimedia Tools Appl.*, vol. 79, no. 41–42, p. 30509–30555, nov 2020. [Online]. Available: https://doi.org/10.1007/s11042-020-09004-3

[4] F. Demrozi, G. Pravadelli, A. Bihorac, and P. Rashidi, "Human activity recognition using inertial, physiological and environmental sensors: A comprehensive survey," *IEEE Access*, vol. 8, pp. 210 816–210 836, 2020.

[5] M. Gohar, S. H. Ahmed, M. Khan, N. Guizani, A. Ahmed, and A. Ur Rahman, "A big data analytics architecture for the internet of small things," *IEEE Communications Magazine*, vol. 56, no. 2, 2018.

[6] A. Ghibellini, L. Bononi, and M. Di Felice, "Intelligence at the iot edge: Activity recognition with low-power microcontrollers and convolutional neural networks," in *2022 IEEE 19th Annual Consumer Communications & Networking Conference (CCNC)*, 2022, pp. 707–710.

[7] F. Luo, S. Khan, Y. Huang, and K. Wu, "Binarized neural network for edge intelligence of sensor-based human activity recognition," *IEEE Transactions on Mobile Computing*, vol. 22, no. 3, pp. 1356–1368, 2023.

[8] D. Xu, T. Li, Y. Li, X. Su, S. Tarkoma, T. Jiang, J. Crowcroft, and P. Hui, "Edge intelligence: Architectures, challenges, and applications," 2020. [Online]. Available: https://arxiv.org/abs/2003.12172

[9] P. Ariza Colpas, E. Vicario, E. De-La-Hoz-Franco, M. Pineres-Melo, A. Oviedo-Carrascal, and F. Patara, "Unsupervised human activity recognition using the clustering approach: A review," *Sensors*, vol. 20, no. 9, 2020.

[10] H. Amrani, D. Micucci, and P. Napoletano, "Unsupervised deep learning-based clustering for human activity recognition," in *2022 IEEE 12th International Conference on Consumer Electronics (ICCE-Berlin)*, 2022, pp. 1–6.

[11] L. Chen, Y. Zhang, S. Miao, S. Zhu, R. Hu, L. Peng, and M. Lv, "Salience: An unsupervised user adaptation model for multiple wearable sensors based human activity recognition," *IEEE Transactions on Mobile Computing*, pp. 1–1, 2022.

[12] A. Moschetti, L. Fiorini, D. Esposito, P. Dario, and F. Cavallo, "Daily activity recognition with inertial ring and bracelet: An unsupervised approach," in *2017 IEEE International Conference on Robotics and Automation (ICRA)*, 2017, pp. 3250–3255.

[13] P.-E. Novac, A. Castagnetti, A. Russo, B. Miramond, A. Pegatoquet, and F. Verdier, "Toward unsupervised human activity recognition on microcontroller units," in *2020 23rd Euromicro Conference on Digital System Design (DSD)*. IEEE, 2020, pp. 542–550.

[14] D. Riboni and F. Murru, "Unsupervised recognition of multi-resident activities in smart-homes," *IEEE Access*, vol. 8, 2020.

[15] A. R. Sanabria, F. Zambonelli, and J. Ye, "Unsupervised domain adaptation in activity recognition: A gan-based approach," *IEEE Access*, vol. 9, pp. 19 421–19 438, 2021.

[16] J. Han, M. Kamber, and J. Pei. (2012) Data mining concepts and techniques, third edition. Waltham, Mass.

[17] T. Kohonen, "The self-organizing map," *Proceedings of the IEEE*, vol. 78, no. 9, pp. 1464–1480, 1990.

[18] G. De Leonardis, S. Rosati, G. Balestra, V. Agostini, E. Panero, L. Gastaldi, and M. Knaflitz, "Human activity recognition by wearable sensors: Comparison of different classifiers for real-time applications," in *2018 IEEE International Symposium on Medical Measurements and Applications (MeMeA)*. IEEE, 2018, pp. 1–6.

[19] X. Li and J. Lin, "Linear time complexity time series classification with bag-of-pattern-features," in *2017 IEEE International Conference on Data Mining (ICDM)*. IEEE, 2017, pp. 277–286.

[20] A. R. Anwary, H. Yu, and M. Vassallo, "Optimal foot location for placing wearable imu sensors and automatic feature extraction for gait analysis," *IEEE Sensors Journal*, vol. 18, no. 6, pp. 2555–2567, 2018.

[21] R. David, J. Duke, A. Jain, V. Janapa Reddi, N. Jeffries, J. Li, N. Kreeger, I. Nappier, M. Natraj, T. Wang *et al.*, "Tensorflow lite micro: Embedded machine learning for tinyml systems," *Proceedings of Machine Learning and Systems*, vol. 3, pp. 800–811, 2021.

[22] D. Anguita, A. Ghio, L. Oneto, X. Parra, and J. L. Reyes-Ortiz, "A public domain dataset for human activity recognition using smartphones," *Computational Intelligence and Machine Learning, ESANN*, 2013.