Probabilistic Cascading Classifier for Energy-Efficient Activity Monitoring in Wearables

Mahdi Pedram, Ramesh Kumar Sah , Seyed Ali Rokni, Marjan Nourollahi, Hassan Ghasemzadeh

***Abstract*— Advances in embedded systems have given rise to integrating several small-size health monitoring devices within daily human life. This trend led to an ongoing extension of wearable sensors in a broad range of applications. Wearable technologies, which are firmly connected with the human body, utilize sensors and machine learning to describe individuals' physical or psychological routines through activity recognition and human movement. Since wearables are used all day long, the power consumption of these systems needs to be reasonably low. Current research considers that such machine learning methods are trained with fixed properties, including sensor sampling rate and statistical features computed from the time series data. However, in reality, wearables require continuous reconfiguration of their computational algorithms due to the personalized nature of human gait and movement. Furthermore, computational algorithms must become energy- and memory-efficient due to these embedded sensors' limited power and memory. In this paper, we propose a resource-efficient framework for real-time, continuous, and on-node human activity recognition. Typically activity recognition problem is a multi-class classification problem. However, we suggest transforming this problem based on MET (Metabolic Equivalent of Task) into a hierarchical classification model, providing personalized structure for each individual. We discuss the design and construction of this new configurable classification paradigm. Our results demonstrate that the proposed probabilistic cascading system accuracy for different personalized scenarios varies between 94.5% and 96.9% in detecting activities using a limited memory, while power usage of the system is reduced by as high as 17.2% compared to the traditional methods.**

***Index Terms*— wearable sensors, machine learning, activity recognition, power optimization**

I. Introduction

Wearable devices have emerged as a promising technology with the potential to play essential roles in the realization of many Internet-of-Things (IoT) applications such as healthcare and medical [38], diet management [22], assistance of industry workers [43], security surveillance [29], home automation [10], and military [39]. The use of these technologies in humans' daily lives is increasing as these devices can be employed almost anywhere. Nowadays, mobile phones consist of embedded sensors such as accelerometers, gyroscopes, magnetometers, GPS, and barometers. In addition, wearable devices with embedded sensors such as activity trackers, smartwatches, smart shoes, smart glasses, and smart clothing are becoming more popular [13]. According to a Pew Research Center survey of June 2019, one-in-five United States adults (21%) regularly wear a smartwatch or wearable fitness tracker. In most applications of these devices, a sensor node is required to obtain physical measurements, and embedded software such as a signal processing algorithm is used for local processing. Furthermore, a sensor node is responsible for storing the results and transmitting them to a gateway [14].

According to the NCHS report, which drew on five years of data from the National Health Interview Survey, about 23% of adults ages 18 to 64 perform at least 150 minutes of moderate or 75 minutes of vigorous-intensity activities. Also, another 32% of the population either perform moderate or vigorous-intensity activities, and 45% do not perform either of these activities [7]. These statistics suggest that activity behaviors vary drastically across individuals. For instance, the report mentioned above indicates that most people are not highly active, concluding that most people do not perform moderate or vigorous daily activities. Therefore, activity monitoring systems measure light intense activities most of the time. On the other hand, light intensity activities such as sitting and sleeping have a lower frequency, and the sensor signal variation is low. As a result, using a lower sampling frequency for activity monitoring systems could be sufficient to recognize most of the activities-of-daily living. In addition, the study shows that sampling rates are up to 57% higher than necessary, which leads to a waste of system resources [24]. The traditional method focuses on extracting potentially complex features from sensor signals and using a sampling frequency according to the activities of interest. However, each individual's daily physical activity is different, and some people may be more active and some less. Therefore, we need a system that can adapt its properties with the user's daily routine to consume energy only when required. For example, if a user is mostly in-active, a lower sampling frequency of motion sensor is sufficient to detect light intense activities. However, the system also needs to detect the higher intense activities that require higher sensor sampling frequency and more complicated features. Computational algorithms prepare the core intelligence of these embedded systems by providing

M. Pedram, R, Sah. A. Rokni, and M. Nourollahi are with the School of Electrical Engineering and Computer Science, Washington State University, Pullman, WA, 99164-2752, USA e-mail: {mahdi.pedram, ramesh.sah, s.roknidezfooli, m.nourollahidarabad} @wsu.edu.
H. Ghasemzadeh is with the College of Health Solutions at Arizona State University, Phoenix, AZ, 85054, USA e-mail: {hassan.ghasemzadeh} @asu.edu.

continuous and real-time processing of clinically important information from sensor readings. Nevertheless, many of these algorithms do not take into account memory and computation constraints of these relatively small embedded systems. As a result, Designing a personalized machine learning algorithm that achieves a reasonable classification performance while satisfying the system's resource constraints is challenging. In fact, these systems need to be low-power and small in size so that individuals can use these devices in their daily life. Furthermore, these embedded devices are often battery-operated, making performing a computationally-intensive process on these devices not feasible.

This paper introduces a probabilistic cascading binary classifier as a resource-efficient machine learning system for human activity classification. Our proposed approach can control the sensor sampling frequency and the number of extracted features in real-time during the system operation. Particularly, we propose to convert the traditional multi-class human activity classification problem into a probabilistic hierarchical binary classification problem that takes into account each individual's activity pattern and achieves a desirable classification accuracy while consuming less energy and memory storage. First, we consider different intensity levels for human activities based on the Metabolic Equivalent of Task (MET) [1]. The intensity levels include light, moderate, and vigorous-intensity groups. We then assign different sampling frequencies and feature sets to each intensity group. As the intensity level increases, the sampling frequency increases, and more sophisticated features are considered for activity recognition. Since lighter intensity activities constitute smaller duty cycles, the changes in their signals are not as frequent as higher intensity activities. Therefore, the system uses lower sampling frequency and fewer complex features to classify lighter intense activities. The construction of our probabilistic cascading classifier reduces the machine learning classification's energy consumption without sacrificing the classification performance. To the best of our knowledge, our work is the first to investigate the problem of energy minimization in wearable sensor systems while considering the sensing, processing, and memory limitations of these systems. In addition to providing a mathematical formulation for the energy minimization problem, we construct a machine learning architecture for the hierarchical classification of the incoming signals. We conduct extensive experiments using sensor data collection with real subjects to demonstrate the effectiveness of our approach compared to conventional multi-class classification. Moreover, we implement our method on a real hardware system to provide practical insight on the extended battery lifetime of the system due to the hierarchical classification architecture.

This article is organized as follows: Section 2 highlights the trend of wearable devices optimization and existing methods and their ability to reduce the required resources for a wearable device. Section 3 identifies the potential problems and challenges of wearable devices. Section 4 presents the solution that we propose for the design and development of a cascading binary classifier. Then, Section 5 illustrates the utility of the proposed method while trying all possible operational scenarios. Section 6 aims to explain the limitations and the future work of the proposed method.

II. Related Work

Processing sensor data and performing continuous human activity recognition quickly drains the battery of embedded sensors and mobile devices [25], [30]. Furthermore, the wearable nature of activity recognition systems puts design constraints on the size and capacity of the battery and demands the design of efficient algorithms to be used in these embedded systems. Therefore, reducing the energy overhead of sensing and computation is essential for continuous human activity recognition and the adoption of these technologies in daily lives [28], [35], [47].

Reducing the energy footprint of an activity recognition system and maximizing the classification accuracy of the machine learning model used for inference are at odds. Sophisticated features and longer computation time are needed for maximum classification performance, whereas shorter computation time with simpler CPU operation results in longer battery life. Prior studies [21], [49] have shown that richer sets of features that result in higher computing costs lead to higher accuracy of activity recognition models. Furthermore, reducing the sampling frequency lowers the energy overhead but increases the classification errors.

[27] uses lower sampling frequency to save battery but suffers from low classification accuracy. In [49], an adaptive model is proposed that is activity-sensitive and adjusts the sampling frequency and classifier features in real-time based on the current activity to lower the power consumption by $20 - 30\%$. In a similar fashion, authors in [45] proposed an adaptive algorithm that selects an appropriate combination of adjustable frequency and classification features for each individual to achieve an energy saving of 28%. The context of a user changes continuously throughout the day, and adapting the algorithm based on the user context reduces the energy requirement of human activity recognition systems. In [17], the user context is used to dynamically adapt the sampling rate of the sensing module to extend the battery life of the activity recognition system by up to 5 times. Furthermore, [26] argues that HAR approaches are energy-inefficient because the sensors are required to run without stopping so that the physical activity of a user can be recognized in real-time. To that end, the author proposed to control the activity recognition duration in conjunction with variable sampling frequency and window size for energy-efficient human activity recognition. The authors were able to reduce the energy consumption by a minimum of about 44.23% without sacrificing on classification accuracy. In machine learning systems, the most power is consumed by the computation needed to process an input. Hence, in [42] a clustering-center based pre-classification strategy is used to reduce the call frequency of deep learning models. The sampling frequency of the inertial sensor is also controlled resulting in energy savings of 49% in Android smartphones.

In [44], a layer-wise convolutional neural networks (CNN) with local loss for human activity recognition was proposed.

This method can be used to alleviate memory requirement of a wearable system. Shallow Convolutional neural networks [18] is also another method that is used in wearable sensors for human activity recognition. This method can be used to develop lightweight deep models for reducing memory requirements. In [37], the authors proposed a technique for memory optimization based on Support Vector Machine (SVM) parameters. This method reduces the number of required support vectors, which results in less memory usage. Moreover, they designed a hierarchical classifier structure for SVM by considering the probability distribution of class labels to reduce the computations. They show that their method can save up to 56% memory for storing the SVM classifier. Moreover, in [46] authors suggested a Convolutional Neural Network which can be used for weakly labeled wearable sensor data. This method can reduce the need for sufficient labeled training data which ultimately results in more efficient data collection process. In [8], [40], [33], [6], [12], [5] authors suggest different effective methods for resource efficient pre-processing and feature extraction for HAR using wearable platforms.

However, these studies only work with a fixed and pre-trained activity recognition model. Designing a personalized activity detection model requires the model to be adjusted upon changes in the system's attributes. Traditionally, changing the model requires classifier retraining by collecting a sufficient amount of labeled training data- a time-consuming, labor-intensive, and expensive process that has been identified as a significant barrier to personalized medicine [50]. Therefore, it is essential to adapt and modify the trained model on the device and in real-time for the maximum benefit of mobile health devices. Also, collecting a large amount of labeled data requires sufficiently large memory storage on the device, which may not be available on tiny wearable sensors. Leveraging the online learning paradigm, we design power and memory-efficient machine learning model training framework that can adjust the trained machine learning model on the device as new labeled observations become available.

III. Problem Statement

An embedded sensing module usually has sensors, such as accelerometer, gyroscope, and magnetometer, for capturing different physical states of an individual. In addition, it has an embedded software module to process the collected signals and a machine learning model used to make inference on the collected data [16]. As shown in Fig 1, these devices are also equipped with a communication module such as Bluetooth for data transmission. The conventional approach for handling computation and memory intensive processes that is not feasible on the sensor node is to transfer these processes to a more powerful unit, referred to as base-station, which gathers the sensed signals for signal processing and machine learning. The base-station is responsible for executing the essential power-hungry computational algorithms. However, there are several disadvantages to this traditional architecture. First, it does not allow embedded sensor nodes from functioning as a stand-alone device and thus presents a single point of failure to the system. Second, this method prevents the system and sensing nodes from providing real-time results. Third, transmitting the collected data to the base-station is a power-hungry procedure [31]. Finally, continuous data transmission from the sensor node to the base-station over a channel raises security and privacy concerns [19]. These problems motivate designing a per-node intelligence resource-efficient module that can independently make decision based on its collected sensor data.

However, performing the required processing locally introduces another challenge. Since the sensor node needs to process the collected data in real-time, we cannot use external memory to store and process the collected data. The reason is that using an external memory increases the response time and power consumption [48]. As a result, we need a fast algorithm and low-power memory to store and process the data. Sensor nodes utilize low-power microcontrollers with limited memory, making it challenging to collect time-series data and process them on-the-fly. We need to use SRAM (static random access memory) to store the collected time series data and extract features from them. However, because SRAM is expensive, the SRAM size on micro-controllers is quite limited [4]. Therefore, while processing the collected data locally, we have to consider this constraint as well.

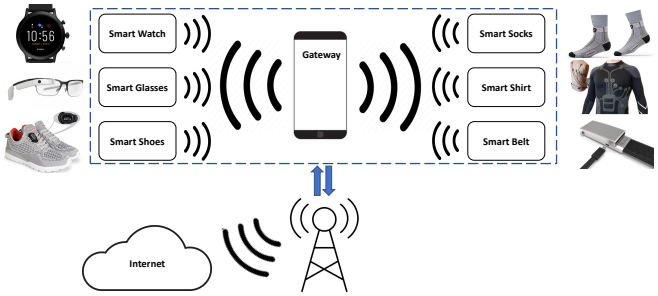

Fig. 1: Wearable devices architecture.

Each sensor node processes sensor readings by using different pre-programmed software for signal processing and machine learning. Physical activity recognition applications start with sensor node Inertial Measurement Unit (IMU). Collected signals from IMU go through signal processing algorithms to remove artifacts and noise. Statistical and spatial features are computed from the processed sensor segments, and the features are used to train machine learning models to recognize human activities such as 'running', and 'sitting'.

Fig 2 shows the human activity recognition machine learning pipeline from sampling a sensor node to the classification. First, a filter is used to diminish high-frequency noise. Second, we need to identify the start and endpoints of each activity classified, which can be done through segmentation. Since 'start' and 'end' of activity could not be easily detected, we use a sliding window with an overlap to segment signals into instances of physical activities. Third, the feature extraction block generates statistical and morphological characteristics of the signal segment. Moreover, in the learning phase, important features are defined, which is based on the application. Features represent different properties of the signal, such as 'amplitude', 'start to end value', and 'mean value'. Finally, a

classification algorithm uses the generated features to recognize the incoming activity of the individual.

Fig. 2: Per-node classification process

An observation x_t, sensor readings over a signal segment, made by a wearable sensor at time 't' can be represented as a K-dimensional feature vector, $x_t = \{d_{t1}, d_{t2}, \ldots, d_{tK}\}$. Each feature is processed from a certain time window. The activity recognition task requires a label space $A = \{a_1, a_2, ..., a_m\}$ including of the set of labels for activities of concern, and a conditional probability distribution $P(A|x_t)$, which is the probability of inferring label $a_j \in A$ given an observed instance x_t. The probability of performing each activity for an individual in a daily basis is given as $P_n = \{p_{a_1}, p_{a_2}, \ldots, p_{a_m}\}$.

A wearable device that works as a stand-alone processing unit needs to store and manipulate sensor readings to extract the required features for later processing. SRAM (Static Random Access Memory) of the micro-controller is where the variables can be manipulated while the system is running. SRAM is an expensive memory compared to DRAM (Dynamic Random Access Memory), resulting in less SRAM space in processing units. Therefore, if we desire to process the data locally, then we need to consider the space limitation of SRAM. Let, for a sensing device, s be the feature complexity, and f be the sampling frequency. For each observation x_t, the computation complexity of feature extraction, $Compute(x_t)$, is defined as the normalized number of required instructions to extract a feature vector from a signal segment. Furthermore, the IMU readings for each observation consist of x_{ta}, and x_{tm}, which represent accelerometer and magnetometer values respectively. In addition, we define the size that is used to store each observation x_t as M, and the available SRAM of a micro-controller is k. Therefore, we have the memory constraint as:

$$M \leq k \tag{1}$$

Furthermore, we define the minimum acceptable accuracy by introducing a bound on classifier errors. In other words, for a set of observations in $\mathcal{D}$, we have:

$$\sum_{x_t \in \mathcal{D}} (y_t - f(x_t)) \leq \epsilon \tag{2}$$

where ϵ denotes the maximum bearable error of the classification. In addition, the power consumption of real-time classification can be divided into the energy consumed by sensing (i.e., sampling) and feature computation as follows.

$$\sum_{x_t \in \mathcal{D}} f \times (Sense(x_t) + Compute(x_t)) \leq \theta \tag{3}$$

where θ represents the maximum acceptable energy consumption (e.g., battery capacity). Assuming a potential solution fulfills these three constraints, we may hold two constraints and try to minimize the third one. For instance, accuracy optimization can be defined as accurately classifying observations made by the embedded sensing device such that the misclassification error is minimized while complying with the energy and memory constraints. Alternatively, we can define energy and memory optimization challenges. However, in this work, we investigate methods of minimizing energy consumption while maintaining an upper bound on the classification error and satisfying the memory requirements. In the following, we formally define this optimization problem.

$$\text{Minimize} \quad \sum_{x_t \in \mathcal{D}} f \times (Sense(x_t) + Compute(x_t)) \tag{4}$$

$$\text{Subject to:} \quad \sum_{x_t \in \mathcal{D}} (y_t - f(x_t)) \leq \epsilon \tag{5}$$

$$M \leq k \tag{6}$$

The optimization problem in (4)–(6) aims to minimize the amount of energy consumption due to sensing and computation subject to an upper bound on the classification error, shown in (5), while satisfying the memory constraint in (6). This problem allows us to set our desired threshold for classification accuracy and memory size and minimize energy consumption when classifying physical activities on a sensor node.

IV. Method

Human activity recognition is a multi-class classification problem that requires comprehensive knowledge of human gait to recognize activities of interest. However, increasing the number of classes and sensor nodes increases the computational complexity of these systems. Designing a single multi-class classifier model to distinguish a class from many other classes is challenging and may not result in an efficient classifier in terms of power and memory requirements. On the other hand, conforming with the Divide-and-Conquer paradigm, by breaking down the problem into multiple hierarchical classifiers, we can save energy and computation without losing the classification accuracy. The properties of our proposed probabilistic cascading classifier include sensing efficiency, personalized structure, and feature computation efficiency. Furthermore, we evaluate the performance of the proposed classifier using two datasets: 1) "Daily and Sports Activities" dataset [2], which consists of 19 activities performed by eight subjects and, 2) "PAMAP2 Physical Activity Monitoring" dataset [36], including 12 activities performed by nine subjects. Both datasets include raw IMU signals, including accelerometer, magnetometer, and gyroscope.

A. Sensing and Computation Efficiency

Human physical activities vary from one another based on their intensity level. For instance, activity such as running is more intense than sitting. If we are to develop a single multi-class classifier to classify all activities, we need to make sure that we use a sampling frequency according to the fastest activity. Only then we can avoid aliasing in the collected

data. Aliasing results in signal distortion because the sampling frequency is too low to regenerate the original analog content accurately. Considering f_s as the sampling frequency of the sensor and f_a as the frequency of the activity, according to Nyquist–Shannon sampling theorem [41] we must have $f_s \geq 2f_a$. Therefore, the required sampling frequency of a single multi-class classifier is at least two times greater than the frequency of the fastest activity of interest. According to [3], human physical activity frequencies are between 0 to 20 Hz. As a result, the required sampling frequency for a multi-class classifier to capture the activity signal needs to be at least 40Hz. However, activity such as normal walking has a frequency between 1.4 to 2.5 Hz [20] and a sampling frequency of 5Hz should be sufficient to avoid aliasing. Furthermore, by setting the sampling frequency according to the most intense activity, we over-sample for the rest of the activities, which do not require such a high sampling frequency. In addition, a multi-class classifier requires a complex set of features to separate all activities of interest, and extracting complex features is both memory and energy inefficient. In this study, we develop a cascading approach to adjust sampling frequency and feature complexity efficiently. We categorize all activities of interest into three categories according to the Metabolic Equivalent of Task (MET) studies, which include light-intensity, moderate-intensity, and vigorous-intensity [1]. For light-intensity activities, a lower sampling frequency is adequate to captures the required information for recognizing the activities. Similarly, for vigorous-intensity activities, we need a higher sampling frequency to detect activities and avoid aliasing. As shown in Fig 3, we consider different sampling frequency values and different feature complexities for each group. The feature complexity and sampling frequency (f_s) increases with the increase in the intensity level of activities.

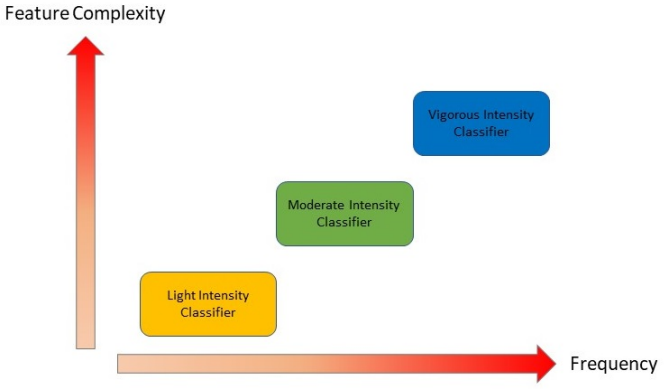

Fig. 3: Cascading classifier resource assignment for each intensity group.

1) Sampling Frequency Determination: As stated earlier, we have different sampling frequencies for each intensity group. We determine the sampling frequencies according to Nyquist–Shannon sampling theorem and define the sampling frequency of each intensity group according to the most intense activity in that group. Because the fastest or most intense activity has the highest frequency, this guarantees the Nyquist-Shannon sampling theorem holds $(f_s \geq 2f_a)$ for all the activities in each intensity group. Equation (7) shows the sampling frequency of each intensity group including light (f_{sl}), moderate (f_{sm}), and vigorous (f_{sv}) groups. f_l, f_m, and f_v are the frequencies of light, moderate, and vigorous activities in each group respectively.

$$f_{sl},\ f_{sm},\ f_{sv} = 2\max(f_l),\ 2\max(f_m),\ 2\max(f_v) \quad (7)$$

If the classifier identifies the current activity as light-intensity, the sampling rate remains low. However, if the classifier detects the current activity belonging to a higher-intensity group, then the system increases the sampling frequency.

2) Feature Selection: Different sets of features are used for each intensity group based on the classification difficulties. We have used two feature selection methods to select the best performing features for each intensity group without sacrificing classification accuracy. We have 3-axial data from accelerometer and magnetometer sensors, and for training human activity recognition models, we compute time-domain statistical features from the sensor window segments. Time-domain features avoid complex pre-processing stages like framing and Fourier transformation required for frequency-domain features. Consequently, computing time-domain features consumes less processing power than frequency-domain features [11]. Since, in this work our aim is to decrease the energy footprint of the HAR system, we have decided to use only time-domain features in our analysis. We extract 9 features for each axis, which includes amplitude, median, mean, maximum, minimum, peak-to-peak amplitude, standard deviation, root mean square power, and start to end value. As a result, we have 81 features from each window-segment, and this requires significant computation resources. Using a large number of features usually results in higher performance of the trained machine learning models [21], [49], however, feature computation requires energy and memory, which are limited in embedded systems. Hence, it becomes crucial that we use a minimal number of features for the maximum attainable accuracy of the activity recognition models. We use Recursive Feature Elimination (RFE) - to select the best features for each level in our cascading model for human activity recognition. To select relevant features, a feature selection criterion is needed to measure the relevance of each feature with the class labels [34]. Recursive feature elimination (RFE) is a wrapper-type feature selection algorithm that uses weights of a machine-learning algorithm to rank features for selection. RFE has two hyperparameters: the number of features to select and a machine learning algorithm used for ranking features. RFE works by searching for an optimal subset of features by starting with all features present in the dataset and successfully removing features until the specified number of features remains. Traditionally, Support Vector Machine (SVM) is used as the ranking machine learning algorithm with recursive feature elimination for binary and multi-class classification problems [9], [15].

B. Cascading Classifier

After categorizing the activities into three groups, we develop the probabilistic cascading classifier to recognize human

activities. The input specifications, including the feature set and sampling frequency of this classifier, are determined by each activity's intensity level. The probabilistic cascading classifier consists of binary classifiers and group classifiers connected to the human activity recognition system. In this subsection, we first explain each component of the classifier, and then we go over the personalized structure and the connections between different components of the system.

1) Binary Classifiers: The first level of the cascading classifier system consists of binary classifiers. Binary classifiers are used to determine the intensity level of an input window-segment. Detecting intensity level for a given input, allows the system to adopt the correct sampling frequency and feature complexity for activity classification. As shown in Fig 3, we need three binary models to separate intensity groups, which include binary classifier for light intensity group BC_l, binary classifier for moderate-intensity group BC_m, and binary classifier for vigorous-intensity group BC_v. Each binary classifier distinguishes that intensity group from the rest. We train three simple binary classifiers based on intensity groups. Each of these binary classifiers works with a unique sampling frequency and feature complexity. For this purpose, we construct a dataset with activities from each intensity group. Then we label our data according to the intensity group of each activity. As a result, we define sub-label according to the intensity group of each classifier. For training BC_l we assign true as the sub-label of all low-intensity activities and false to other intensity groups activities. We follow the same procedure for BC_m and BC_v by assigning the true to moderate-intensity and vigorous-intensity groups. The intensity group activities are similar in some factors, such as changes, amplitude, and max/min of the signal. For instance, for an activity such as standing, we have fewer signal changes, and slower movement of this activity results in low amplitude or low absolute values of min/max. Other light intensity activity works accordingly. However, more intense activity such as walking has higher amplitude or min/max values due to the higher acceleration of the activity. As a result, we expect by just using intensity group labels, it is possible to distinguish intensity groups from one another. We train these classifiers with different models, including Support Vector Machine (SVM), Decision Tree, Random Forest, Logistic Regression, Multilayer Perceptron, and K-Nearest Neighbour. Ultimately, we can choose the model with the best performance for this application.

2) Groups Classifier: After detecting each input's intensity level, we need to classify the input segment as an activity belonging to the detected intensity group. Group classifiers are multi-class classifiers that are responsible for classifying an input segment as an activity class. Group classifiers are trained on a particular feature set to recognize activities for each intensity group. Group classifiers include MC_l for light intensity group, MC_m for moderate-intensity group, and MC_v for vigorous-intensity group. The sampling frequency used in these models is the same as the sampling frequency of the respective intensity group. However, a different set of features is required for each multi-class classifier model than the binary classifier of the same intensity group. We train group classifiers using different popular algorithms, including Support Vector Machine, Decision Tree, Random Forest, Logistic Regression, Multilayer Perceptron, and K-Nearest Neighbour. Therefore, based on the achieved results, we can select the algorithm that has the best classification accuracy as our model.

3) Classifier Flow: After setting variable sampling frequencies and feature sets, we connect different components of the classification system. However, some assumptions need to be discussed before we talk about our experiments and results. First, we assume that we have pre-trained models trained on a limited amount of data and stored on the wearable device. Second, we assume that the initial probability of performing each intensity group for each individual is equally likely. We update the models and probabilities as an ongoing process as the user uses the classification system. This learning on the go provides the personalization for our systems. We consider the personalized variable structure for our probabilistic cascading classifier. We determine the classifier's structure based on the probabilities of intensity groups for each individual. The purpose of this structure is to help save energy by going through less computation for each user. Fig 4 shows the main components of the probabilistic cascading binary classifier. First, the system fetches the initial probability (p_i) of each intensity group for a user. Then it sorts the intensity groups based on their probabilities. Therefore, the sensor node starts sampling and computation according to the frequency and feature level (f_i, s_i) of the most likely group. If the current activity is not a member of the most likely intensity group, then we change the sampling frequency and feature set according to the next likely group. The system keeps executing this process until the current activity is recognized and uses the recognized activity to update the probabilities of intensity groups.

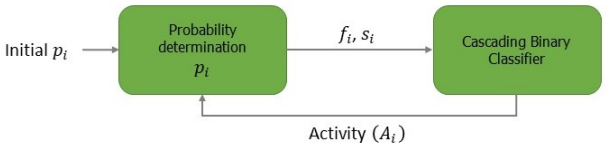

Fig. 4: Probabilistic cascading binary classifier components.

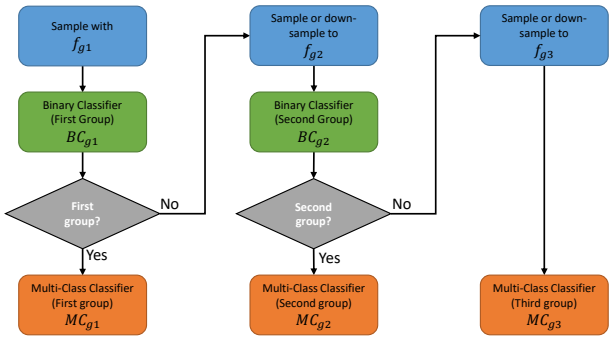

Fig. 5: Flow chart of the cascading binary classifier.

A variable classifier structure aims to minimize the re-sampling and individualize the classifier for each subject. For example, let's consider for a user the activities of daily living has 50% chance of being moderate intensity, 40% chance

Algorithm 1 Probabilistic Cascading Binary Classifier.

$p_1, p_2, p_3 \leftarrow Probabilities(User)$

while $True$ **do**
  $Freq \ = f_{g1}$
  $Features \ = s_{g1}$
  $RawData \ = Read(IMU, Freq)$
  $Features \leftarrow Read(RawData, Features)$
  $ActivityGroup \leftarrow BC_1(ExtractedFeatures)$
  if $ActivityGroup \ = L_1$ **then**
    $ActivityLabel \leftarrow MC_1(ExtractedFeatures)$
  else
    $Features \ = s_{g2}$
    if $f_{g2} \ > f_{g1}$ **then**
      $Freq \ = f_{g2}$
      $RawData \ = SensorReadings(IMU, Freq)$
    else
      $RawData \ = DownSample(RawData, f_{g1}, f_{g2})$
    end if
    $ExtractedFeatures \leftarrow Read(RawData, Features)$
    $ActivityGroup \leftarrow BC_2(ExtractedFeatures)$
    if $ActivityGroup \ = L_2$ **then**
      $ActivityLabel \leftarrow MC_2(ExtractedFeatures)$
    else
      $Features \ = s_{g3}$
      if $f_{g3} \ > f_{g2}$ **then**
        $Freq \ = f_{g3}$
        $RawData \ = Read(IMU, Freq)$
      else
        $RawData \ = DownSample(RawData, f_{g2}, f_{g3})$
      end if
      $ExtractedFeatures \leftarrow Read(IMU, Freq, Features)$
      $ActivityLabel \leftarrow MC_3(ExtractedFeatures)$
    end if
  end if
  $Probabilities \leftarrow UpdateProbability(ActivityLabel)$
  $Report \leftarrow ActivityLabel$
end while

of being light intensity, and 10% chance of being vigorous. Therefore, the system selects the moderate binary classifier when input arrives and checks whether the input belongs to moderate-intensity group or not. Fig 5 and Algorithm 1 shows the flow chart and pseudo code of the cascading binary classifier for the three possible intensity groups. The first group is the intensity group with the greatest probability. The second group is the intensity group, which has the second-largest probability. And the third group is the intensity group with the least probability. We start with the most probable intensity group with sampling frequency f_{g1} and feature set s_{g1} and set the sampling frequency to f_{g1} and we extract feature according to s_{g1}. Then we check if the current activity is in this intensity group (L_1) or not using binary classifiers. If the current activity is not in the most probable intensity group, then we select the next intensity group with the second greatest probability value. We modify the specification of the classifier to f_{g2} and s_{g2}. If the new sampling frequency is greater than the last sampling frequency ($f_{g2} > f_{g1}$), then we need to re-sample sensor data again. However, if the new sampling frequency is less than the previous one, then we down-sample the collected data. This process tries to avoid unnecessary re-samplings that cause energy consumption and add a re-sampling delay to the process. We follow the same procedure for the second and third intensity groups until we identify the activity. We also conclude that if the activity is not in the first two more likely intensity groups, then the activity is in the least likely intensity group without going over the group's binary classifier. Algorithm 1 describes the process of the probabilistic cascading binary classifier.

C. Resource Analysis

We have variable structures for the probabilistic cascading binary classifier, which results in different energy consumption for each scenario. The goal of variable structures is to minimize the energy consumption of each working scenario. The most typical scenario for daily physical activities is the scenario in which the probability of lighter intensity activities is higher than other intensity groups. If we keep this scenario as our fixed approach for all users, then an individual with a higher distribution of activities in moderate or vigorous groups will perform poorly. The system's critical path will be longer, resulting in higher resource usage. This introduces delay and higher power consumption comparing to a normal multi-class classifier method for activity recognition. Therefore, we need to personalize the classifier's structure by considering the probabilities of activities for each user to maximize resource-saving and reduce the response time. The power consumption of the cascading classifier framework is given by:

$$\begin{aligned}&(T_{g1} \ f_{g1}(Sense(x) + Compute_{g1}(x) + Compute_{g11}(x)) \\ &+ (T_{g2} \ f_{g2}(Sense(x) + Compute_{g2}(x) + Compute_{g21}(x)) \\ &+ (T_{g3} \ f_{g3}(Sense(x) + Compute_{g3}(x)) \end{aligned} \quad (8)$$

where T_{g1}, f_{g1} and $Compute_{g1}$ are time, frequency for detecting this level vs. other intensity activities and computation of feature for first intensity level binary classifier. We add $Compute_{g11}$ for the first level activities because after knowing that the activity is in the first intensity group, the classifier uses the second level of this intensity group features to distinguish different activities of this group and infers one single activity as the classification result. The process is similar for the second and thirds intensity groups. The probability consideration helps the classifier minimize the critical path by starting the intensity groups most likely. Therefore, most of the time, the first line of Equation 8 is the total power consumption of the sensor node while executing the probabilistic cascading binary classifier. Since the classifier structure is different for each individual, the power consumption model is different. However, the structure is flexible to reduce the power consumption of the sensor node.

V. Validation

In this section, we provide the results of our experiment using the probabilistic cascading binary classifier. First, we explain the experimental setup of our binary and multi-class classifiers and datasets used in our experiments. Second, we examine each level of our cascading classifier separately. Third, we evaluate the system for activity classification by connecting all the components and discussing the system's performance. Lastly, we report our system's resource analysis

using the actual hardware system and compare our results with the standard way of human activity classification.

A. Experimental Setup

We use two different datasets to evaluate the cascading binary classifier. The first dataset is the "Daily and Sports Activities Dataset" [2] and the second dataset is "PAMAP2 Physical Activity Monitoring" [36]. Our validation focuses on the first dataset, for which we present a comprehensive analysis, and for the second dataset, we offer a summary of our analysis. In the "Daily and Sports Activities dataset", there are apparent inter-subject variations in the speeds and amplitudes of the same activity because the subjects were asked to perform the activities with their desirable style and were not restricted to how the activities should be performed. Subjects performed 19 activities while wearing 5 motion sensor nodes on 5 different body locations, including the torso, right arm, left arm, right leg, and left leg. Each sensor node has a 3-axis accelerometer, 3-axis gyroscope, and 3-axis magnetometer. However, we do not include the gyroscope data in our analysis because the gyroscope is typically considered more power-hungry than an accelerometer and magnetometer [23]. It requires vibration at a certain frequency to measure the angular velocity [32]. Furthermore, we process the data that is collected on the torso. The original sampling frequency of the collected data is 25Hz, and the 5-min signals are divided into 5-sec segments so that 480(=60x8) signal segments are obtained for each activity. Table I shows the activities, their respective intensity groups, and the MET values.

We assigned an intensity group to each activity as their sub-labels, according to MET. As shown in Table I, activities 1 to 5 are light intense activities, activities 6 to 11 are moderate intense activities, and activities 12 to 19 are vigorous intense activities. From each 5 seconds segment of the individual sensor streams, we extract 9 statistical features. Possibly, several different features can be extracted from human physical activities. However, we use the amplitude of the signal (AMP), the median of the signal (MED), mean of the signal (MNVALUE), maximum of the signal (MAX), minimum of the signal (MIN), peak to peak amplitude (P2P), the standard deviation of the signal(STD), root mean square power (RMS), and stand to end value (S2E). As a result, for each IMU with an accelerometer and magnetometer 9 features are extracted for each axis, which gives us $2 \times 3 \times 9 = 54$ total features. These features aim to capture both the shape and amplitude of the signals. For instance, features such as amplitude and mean of the signal can be used to capture the signal's amplitude while the standard deviation of the signal and start-to-end value attributes try to capture the signal's structure. We divide these features into different levels of complexity based on the number of extracted features for each classifier. Each level uses a different set of features which aims to classify the incoming activity efficiently.

The original sampling frequency of the dataset is 25Hz. We downsample the dataset into 1Hz, 2Hz, 5Hz, and 8Hz sampling frequencies for evaluation purposes. We assign 2Hz to light intense activities, 5Hz to moderate-intensity activities, and 8Hz to vigorous intense activities and satisfy the Nyquist theorem to avoid aliasing. We also include 1Hz sampling frequency as the lowest frequency for comparison. The original dataset was constructed by collecting 125 samples for each 5 second time segment. However, we collect 5, 12, 25, and 42 samples during the 5 seconds time window to reach the desired sampling frequencies for each intensity group. Fig 6 and Fig 7 show the raw acceleration data of the X-axis before and after downsampling for a light intensity activity and a vigorous intensity activity respectively. First, we select the desired number of samples from the original 125 samples. Then we regenerate the signal with the selected number of samples. As shown in Fig 7, sampling a vigorous intense activity with low sampling frequency such as 2Hz cannot capture all the detail and shape of the signal. Therefore, we can conclude that low sampling frequency may not be used to train a classifier to achieve a reasonable accuracy for high-intensity activities. On the other hand, in Fig 6, which belongs to a light intensity activity, we can use lower sampling frequency and still capture the signal with a sensible detail. Therefore, the use of high sampling frequency to capture low-intensity activity signal can be avoided, resulting in lower power consumption.

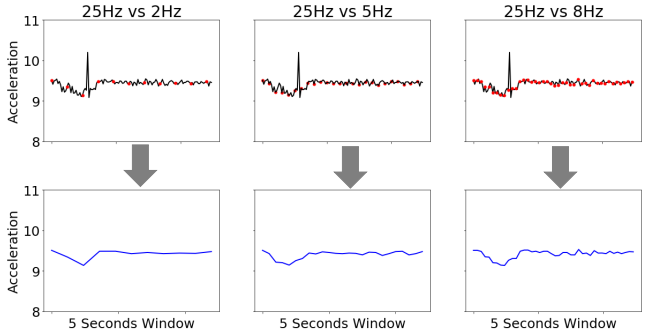

Fig. 6: Down sampling example of a light intense activity.

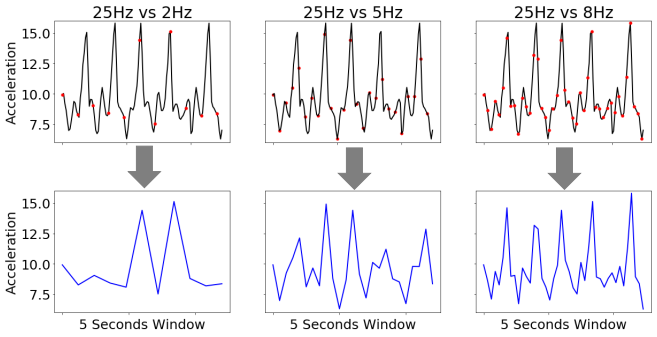

Fig. 7: Down sampling example of a vigorous intense activity.

B. Binary Group Intensity classifiers

As stated earlier, the output of this level of the probabilistic cascading binary classifier determines the intensity level of the activity. We have three binary group intensity classifiers including BC_l, BC_m, and BC_v. BC_l distinguish low-intensity

TABLE I: Activity and group list

Activity ID	Activity Description	Intensity Group	MET
1	Sitting	Light	1.3
2	Lying on back	Light	1.3
3	Lying on right side	Light	1.3
4	Standing	Light	2
5	Standing in elevator	Light	2
6	Walking in a parking lot	Moderate	3.2
7	Walking on a treadmill with a speed of 4 km/h	Moderate	3.5
8	Moving around in elevator	Moderate	3
9	Walking on inclined (15 deg) treadmill at 4km/hr	Moderate	5.3
10	Ascending stairs	Moderate	5
11	Descending stairs	Moderate	3.5
12	Running on a treadmill with a speed of 8 km/h	Vigorous	9.8
13	Exercising on a stepper	Vigorous	9.8
14	Jumping	Vigorous	9
15	Exercising on a cross trainer	Vigorous	9.5
16	Cycling on a horizontal exercise bike	Vigorous	9
17	Cycling on a vertical exercise bike	Vigorous	9
18	Rowing	Vigorous	12
19	Playing basketball	Vigorous	9.3

activities from the rest, BC_m distinguish medium intensity from the rest, and BC_v distinguish high intensity from the rest. We train each model separately using different feature sets computed from data sampled at different sampling frequencies. We use the recursive feature selection method to select the best features for each binary classifier. 50% of the down-sampled data is used as the training dataset for binary classifiers and feature selection. To this end, we choose half of the data for each activity label, at random, to construct the training set. This gives us approximately the same number of instances of each activity in both training and test sets. To achieve the most informative features for each binary classifier, we use the recursive features selection method for which we set the number of desired features, and this gives us the best feature combination based on their performance on the train set. For training the binary group classifiers, we re-label the dataset into binary class considering which binary classifiers we are training. For instance, if we are training BC_l, we assign label 1 to all feature rows for light intensity activities and 0 for feature rows from moderate and high-intensity activities. Ultimately, we have 3 different datasets for each binary classifier. While training each classifier, we assign true labels to the activities of that classifier and false labels to the rest of the activities.

Table II shows an example of the selected features for different feature selection methods for BC_l binary classifier. We analyzed the data using RFE to sort the features according to their importance. This table demonstrate that the feature selection method adds one feature with the most contribution to the class label each time we increase the number of selected features. As it can been from Table II, from the first 10 important features for light intensity group, 9 features are similar between the two tested (RFE and AdaBoost) feature selection methods. Furthermore, we include the selected features of RFE method in later analysis.

TABLE II: The first 10 feature combinations selected by the recursive method.

Feature Order	RFE	AdaBoost
1	Mag RMF(z)	ACC MIN(y)
2	ACC MIN(y)	Mag RMF(z)
3	ACC MIN(x)	ACC MAX(x)
4	ACC MED(x)	ACC MED(x)
5	ACC S2E(z)	ACC MIN(x)
6	Mag STD(z)	ACC MIN(z)
7	ACC MAX(y)	ACC S2E(z)
8	Acc P2P(x)	Mag STD(z)
9	Mag S2E(y)	ACC MAX(y)
10	ACC MIN(z)	Mag S2E(y)

We use the other half of the data as our test set. Therefore, after feature selection and training the model on training data, we use a test set to evaluate the classifier performance. The most challenging situation for each classifier is when each classifier serves as the first component of the cascading classifier structure. Therefore, the first model (the intensity group with the highest probability) has to distinguish the class data from the other two classes. However, the middle model of the classifier structure only needs to distinguish one class from another class because one group is eliminated from the decision process. After the feature extraction phase, we use the constructed train set and test set to evaluate different machine learning algorithms for binary classification. We also start training our models with only one feature and increment the number of features and retrain the model for each feature set obtained from the feature selection algorithm. After training, we test the trained model on the test set.

Fig 8 shows the accuracy of classification on the test set using a variable number of features and different sampling

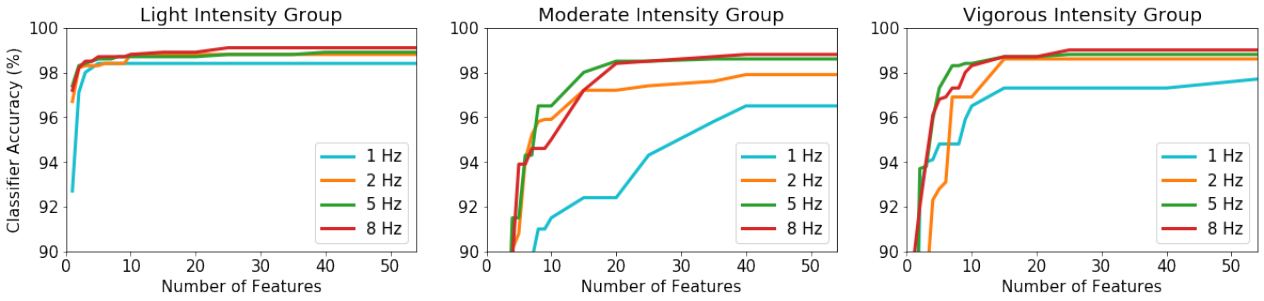

Fig. 8: BC_l, BC_m, and BC_v classifier performance using Random Forest classifier.

frequencies for all the binary classifiers using random forest classifier. For BC_l, we can conclude that increasing the number of features and collecting data with higher frequency results in slightly more accurate predictions. We can also observe that using only a few numbers of features produces a desired accuracy. Moreover, increasing the number of features do not contribute to the classification accuracy after a certain number of features which shows that using more features does not necessarily improve the accuracy on light-intensity vs. rest binary classification task. Therefore, we can extract a few features and classify the current activity with reasonable accuracy. For instance, extracting 2 features while collecting data with a sampling frequency of 1Hz can achieve a classification accuracy of 98%. Therefore, we conclude that we can collect sensor data with lower sampling frequency and extract fewer features while achieving a high classification accuracy for light intensity group vs. rest of activities.

Fig 8 shows the result of classification for BC_m. This classifier distinguishes moderate intense activities from the rest of the activities. From BC_l and BC_m (Fig 8), we realize that the classification accuracy decreased. The reason for this difference is that the moderate-intensity activities are in the middle of MET values. In other words, these activities are not very low intense or very high-intensity activities, which make them more difficult to classify than light intensity activities. Therefore, moderate-intensity activities can be mistaken for either a light intensity group or a vigorous-intensity group. Using different sampling frequency and features, we realize that more features are needed to classify moderate-intensity groups from the rest comparing to BC_l classifier. Sampling frequency seems to be a more critical factor in BC_m comparing to BC_l. This can be explained by the shape and changes of the signals for light intensity activities. Based on Fig 6, it can be seen that changes in the light intensity group's signal are minimal. As a result, it is easier to distinguish these activities from the rest. We conclude that achieving a reasonable accuracy to classify moderate intense activities from other activities needs more resources than light intensity classifiers and requires higher sampling frequency and more features.

Fig 8 also shows the performance of the BC_v binary classifier for variable features and sampling frequencies. BC_v solves the binary classification problem of high-intensity activities vs. the rest. Comparing BC_v and BC_m, we observe that sampling frequency seems to be a more important factor for BC_v. Considering that the frequency of vigorous-intensity activities is higher than moderate intense activities, we can conclude that higher sampling frequency is required to achieve an acceptable classification accuracy. Like the previous binary classifiers, we observe that adding more features to train the classifiers does not necessarily improve the classifier's performance. Therefore, we can recognize activities by extracting a few features while collecting data with lower sampling frequency. In fact, we can reduce the required resources and still achieve reasonable accuracy. For instance, while detecting a vigorous intense group, we can collect data with a sampling frequency of 2Hz and extract 7 features to achieve 97% accuracy.

Comparing the obtained results from BC_l, BC_m, and BC_v, we can define a threshold for the classification accuracy and set the parameters including the number of features and sampling frequency to satisfy the performance threshold. Therefore, we do not over-sample sensor data, which does not contribute to the classifier's performance. At the same time, we also do not extract features when their contributions to the classifiers' performance are negligible. We choose random forest models to use for the binary intensity group classification task. This model gives us two advantages while going forward with constructing the classifier. First, the model is easy to implement. Second, the performance of this model on the test data is more accurate than other tested models. Moreover, the test set that is used for each model includes the data for all 3 groups. However, this scenario is only true if the classifier is the first component of the cascading classifier, which is the most challenging testing scenario of the model. Furthermore, if each binary classifier comes as the second module of the classifier, it needs to distinguish one group intensity from another group intensity, making the prediction less challenging compared to the most difficult scenario, which is having all the intensity groups in the test set. We test the binary classifier in the most challenging scenario for better evaluation of the method. Ultimately, we observe that higher intense activities require higher sampling frequency and more number of features.

C. Multi-Class classification of Activities

We have three multi-class classifiers MC_l, MC_m, and MC_v for each intensity group. Each classifier is used to classify the activity of that intensity group. After group intensity detection, these multi-class classifiers use the original sampling frequency of their intensity groups and extract the required features to classify the activities. The down-sampled data is used for this experiment, and hence we can evaluate

the performance of each classifier using different sampling frequency to collect sensor data. We consider 50% of the data belonging to each intensity group for feature selection algorithm and training the multi-class classifier. The other 50% is considered as a test set of the multi-class classifier. We use the recursive method on the training set to select a set of features from all the possible combinations of the features between having only 1 feature and all features. We train 6 different models, including Support Vector Machine, Decision Tree, Random Forest, Logistic Regression, Multilayer Perceptron, and K-Nearest Neighbour. We assess different types of classifiers to discover the impacts of sampling frequency and features on classification performance. The outcome of the feature selection method is the features with the most contributions to detecting the label. However, each time we increase the number of selected features, we also try all different combinations of the features. Therefore, adding new features does not undermine the ability of the classifier for classification. Adding new features could result in better performance or the same performance if the newly added feature does not contribute to the model. Fig 9 shows the performance of Random Forest classifier to classify all three activity groups using different sampling frequencies and with a different number of features. As can be seen, increasing sampling frequency does not contribute to the classification accuracy for light intensity group. The number of features seems to be more critical for MC_l. Since we expected light intensity activities have fewer changes in their signals, we can conclude that lower sampling frequency can be sufficient to classify these activities. In other words, collecting data with higher sampling frequency does not necessarily improve the classification result for light intense activities. Therefore, by sampling with high frequency, we over-sample, which results in consuming more resources. These resources mainly include power consumption due to data collection and memory usage. This is because of the greater amount of data points collected and could be greatly reduced for light-intensities activities.

Comparing MC_m and MC_l (Fig 9), we can observe that sampling frequency is more critical for MC_m, which classifies moderate intense activities. Since moderate-intensity group activities are faster than light-intensity activities, we need a higher sampling frequency to capture the more intense changes in moderate-intensity activities.

Similar to MC_m we observe that classification accuracy is highly dependent on the sampling frequency for MC_v. Therefore, we justify that to detect vigorous intense activities, the sampling frequency must be high enough to capture changes in the signals in this activity group. In addition, we need to extract more features to reach the saturation point of classifier performance. Comparing MC_m and MC_v classification results, we observe that we need to increase the extracted features for MC_v. In addition, we observe that increasing sampling frequency produces more significant impact on MC_v than MC_m. Therefore, we conclude that for vigorous intense activities, the higher sampling frequency is required comparing to moderate-intensity activities.

D. Cascading Classifier Performance

In this subsection, we examine cascading classifier's overall performance as a human activity recognition system. As mentioned earlier, we use variable sampling frequency and variable feature set for each group of activities. Furthermore, the structure of the classifier depends on the probability of each intensity group. Therefore, we have six different working scenarios for cascading classifier. We evaluate the performance of the cascading classifier for each scenario. Table III shows all possible operational scenarios of the overall system. The first intensity group is the most likely activity group of each scenario, and the third intensity group is the least likely intensity group of each scenario. All the scenarios require at least 1 round of sampling, but some such as scenario 1 may require 3 round of data sampling depending on the activity. Scenarios 5 and 6 do not require re-sampling because we already sampled the sensor with the highest frequency. Therefore, we down-sample the collected data and continue with the rest of the classification process. The rest of the scenarios (1 to 4) may require one or two re-sampling processes, which can be done by increasing the sampling frequency of sensors. Fig 10 shows the process of preparing the test data to evaluate the cascading classifier for the first scenario of Table III. First, we feed the 2Hz test set to BC_l to distinguish light and non-light intensity groups. The light intensity group are then fed to MC_l for light intensity activity classification. We also use the dataset index of non-light intensity groups to select only a non-light group from 5Hz datasets. Therefore, we increase the sampling frequency from 2Hz to 5Hz, which is required for BC_m and MC_l. We perform the same process to exclude the moderate-intensity group from 8Hz test set as well. We feed the 8Hz test set of predicted vigorous intense group to vigorous-intensity classifier (MC_v). A similar process of preparing the test data to evaluate using different sampling frequency is also performed for the rest of the evaluation scenarios, which simulates the real testing situation. To evaluate the performance of the cascading classifier, we use 50% of the data of each activity class for testing purposes, and the rest of the data are used for feature selection. We conduct the experiment using a 5-fold cross validation strategy applied on the test set. We select random forest models for each level in the binary cascading structure which includes these classifiers: BC_l, BC_m, BC_v, MC_l, MC_m, MC_v. We select the Random Forest algorithm because it is simple for implementation, and the performance of these models on the training data is reasonable compared to other machine learning algorithms.

TABLE III: Different operational scenarios of the cascading classifier.

Scenario ID	First Group	Second Group	Third Group
1	Light	Moderate	Vigorous
2	Light	Vigorous	Moderate
3	Moderate	Light	Vigorous
4	Moderate	Vigorous	Light
5	Vigorous	Light	Moderate
6	Vigorous	Moderate	Light

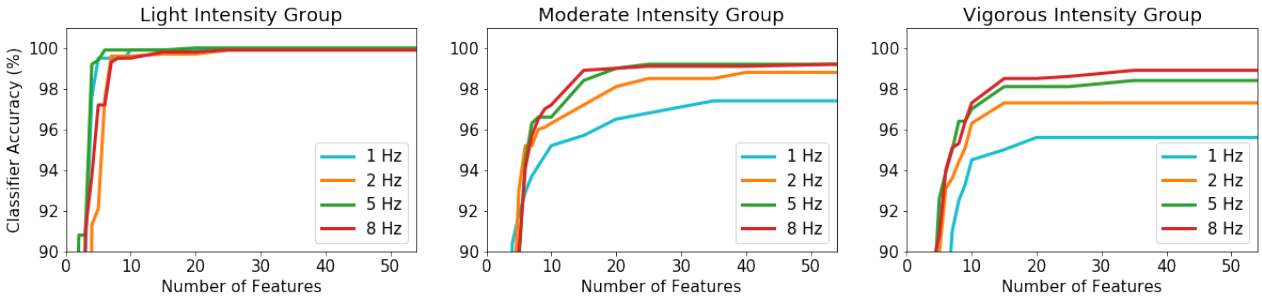

Fig. 9: MC_l, MC_m, and MC_v classifier performance using Random Forest Classifier.

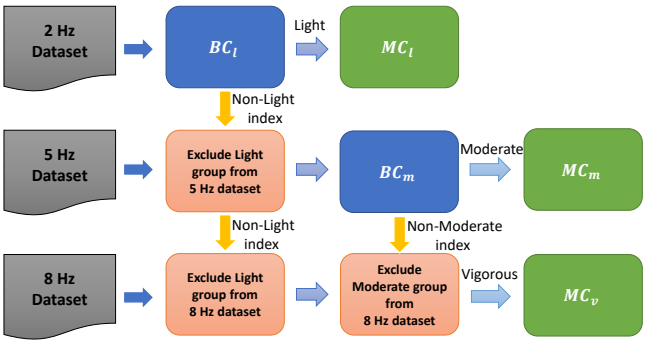

Fig. 10: Test set preparation and classification of the first evaluation scenario.

To test the cascading classifier, we set a threshold for accuracy changes while increasing the number of features for each classifier. The threshold needs to be high enough for a hierarchical classifier to prevent propagation error. We select the minimum number of features, which results in an accuracy of equal or more than 98% for each model. For instance, if the maximum accuracy of a model reported as 100%, then we select the minimum number of features that result in a 98% classification accuracy. As a result, we select the number of features for each model and connect it with the model's input/output to construct the binary classifier. Table IV shows sampling frequency and the number of extracted features for each model. Using these frequency values and feature sets, we achieve at least 98% accuracy for each model. In addition, Table IV shows the performance of each classifier with different metrics obtained through a 5-fold cross validation experiment using a random forest classifier. These results suggest that each component alone is capable of performing the assigned classification task with a relatively high classification performance. Furthermore, Fig 12 shows the performance of cascading classifier for each scenario. Fig 12 shows that each intensity group's overall accuracy is similar across different scenarios. This can be potentially explained by our model's attempt to adjust the sampling frequency and feature set to maintain a minimum accuracy performance in all scenarios. Therefore, we expect to observe similar behavior for all possible operational scenarios of the cascading classifier. However, scenarios that begin with higher intensity activities show higher accuracy since higher sampling frequency is used to classify these activities. Moreover, the higher frequency results in larger data size for the same window interval and better captures the signal changes for higher classification accuracy. Furthermore, Fig 11 shows the confusion matrix of all possible scenarios. It can be seen all scenarios share relatively similar misclassified activities. For instance, "standing in elevator" is repeatedly misclassified as "Moving around in elevator" in all scenarios. Therefore, we conclude that the proposed method is effective in all the scenarios despite variable sampling frequency and different number of features.

E. PAMP2 Dataset Evaluation

We consider using another dataset to evaluate the generalizability of the suggested method. Therefore, we selected "PAMAP2 Physical Activity Monitoring" as the second dataset. We selected 12 activities, which includes four activities from each intensity group. The selected activities include lying, sitting, standing, computer work, walking, ascending stairs, descending stairs, Nordic walking, running, cycling, rope jumping, and playing soccer. This data is collected with a sampling frequency of 100Hz. We down-sampled the sampling frequency to 2Hz, 6Hz, and 10Hz for light intensity, moderate intensity, and vigorous intensity, respectively. We process the data of the IMU that is located on the chest of individuals. Similar to the previous analysis, we only feed accelerometer and magnetometer data to our classifier. Table V shows the specifications of each model of our probabilistic cascading binary classifier, which led to an accuracy of 98%. As expected, we need to sample higher intensity activities with higher frequency and more extracted features. For instance, we require the sampling frequency to be 2Hz for light intensity activities. However, a higher sampling frequency is required for recognizing vigorous intensity activities (10Hz). Fig 13 shows the classification accuracy of the classifier for all possible scenarios according to Table III. For scenarios 1 and 2, we observe better prediction for the light intensity group. This can be justified by mentioning that the classifier first checks the light intensity group in these two scenarios. Therefore, we do not have a propagation error for this group. The same pattern can be observed for the groups that the classifier checks first. Moreover, The average classification performance increases as the intensity level of the first checked group increases. This is because higher intense activities seem to be more challenging to classify. As a result, they are more vulnerable to propagation error. If we check higher intensity groups earlier than other groups, then we decrease the propagation error.

TABLE IV: Models specifications of Daily and Sports Activities dataset.

Model	Frequency (Hz)	Number of Features	Precision	Recall	f-score	MCC
BC_l	2	2	0.984	0.981	0.982	0.965
BC_m	5	8	0.981	0.976	0.978	0.957
BC_v	8	10	0.989	0.986	0.987	0.975
MC_l	2	4	0.987	0.987	0.987	0.984
MC_m	5	10	0.988	0.988	0.988	0.985
MC_v	8	12	0.986	0.986	0.986	0.984

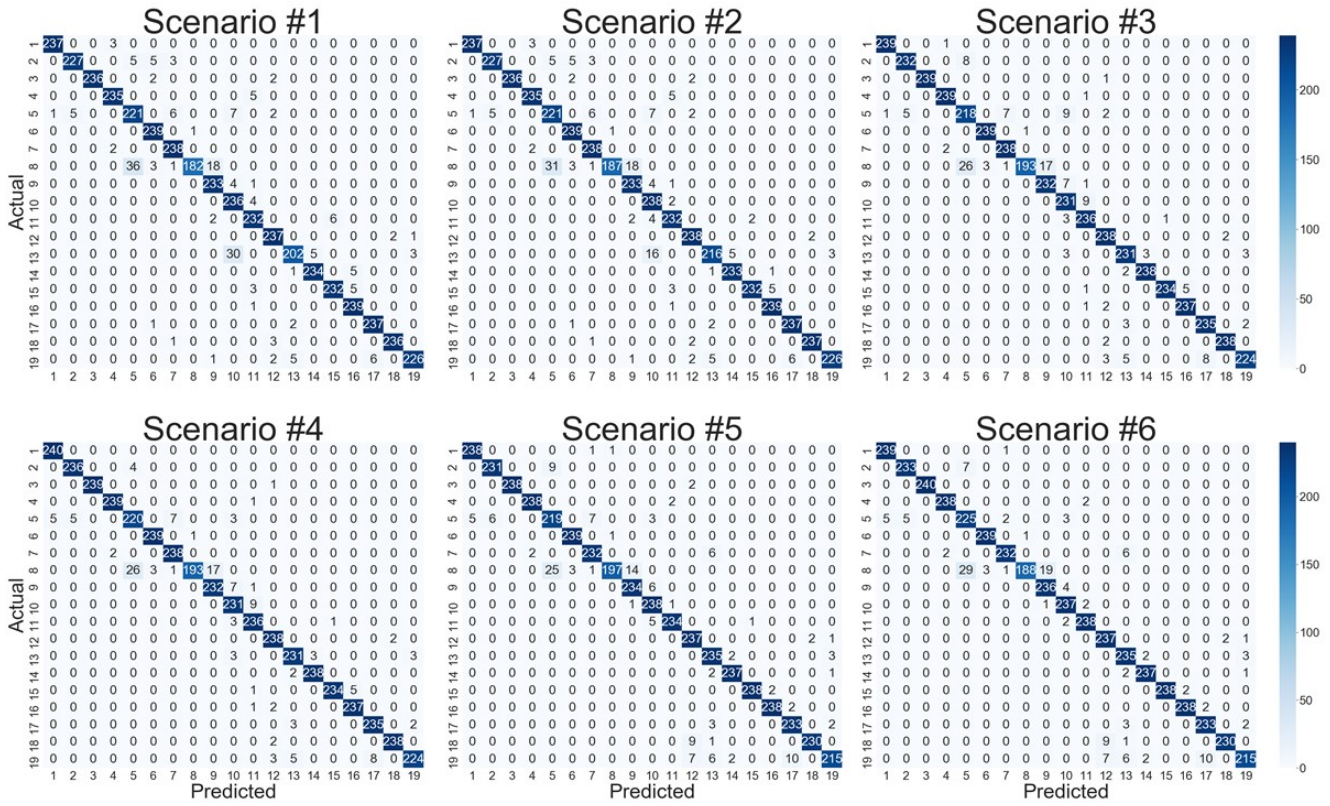

Fig. 11: Confusion matrix of all the scenarios for daily and sports activities dataSet.

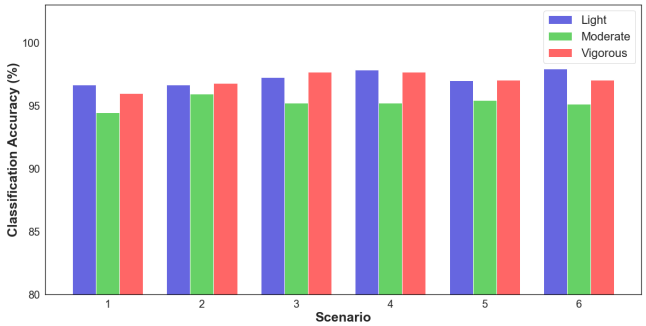

Fig. 12: Cascading classifier performance of Daily and Sports Activities dataset for all possible scenarios.

TABLE V: Models specifications of PAMAP2 dataset for cut-off accuracy of 98% .

Model	Frequency (Hz)	Number of Features
BC_l	2	2
BC_m	6	7
BC_v	10	9
MC_l	2	4
MC_m	6	9
MC_v	10	11

F. Power Consumption and Memory Usage Evaluation

In another experiment, we aim to simulate the probabilistic cascading binary classifier on a real hardware system to experimentally investigate the energy consumption and memory usage of the proposed method. The goal of the proposed method is to save resources in wearable technologies. Therefore, we conduct this experiment to implement the actual method in a real-world setting. Then we compare the energy consumption of the system for different scenarios. For this in-lab experiment, we use Adafruit Feather M0, which is equipped with ARM Cortex M0 processor and the Adafruit LSM303 triple-axis Accelerometer and Magnetometer break-

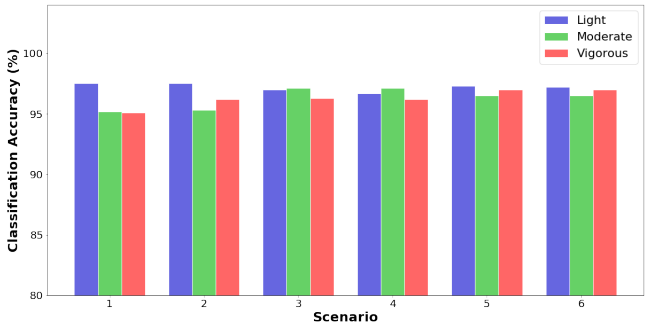

Fig. 13: Cascading classifier performance of PAMAP2 dataset for all possible scenarios.

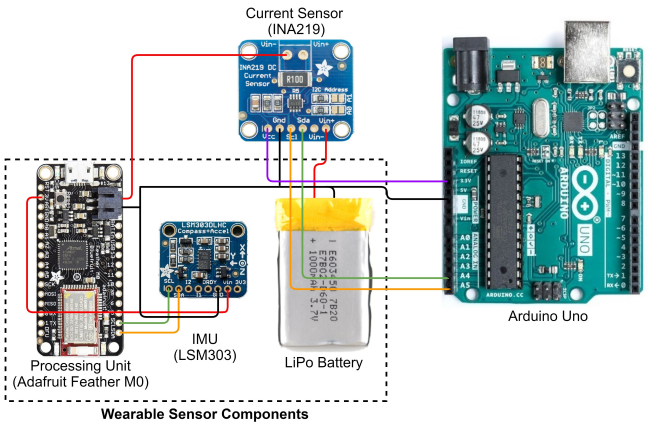

Fig. 14: Current measurement experiment of the cascading binary classifier.

out board. LSM303 is a low power IMU, and the sensor was put in a low-power mode in our experiments. This IMU let us sample data with different sampling frequency at the hardware level, which helps us conduct this experiment. We also use the INA219 current sensor to measure the current consumed by the system. A 1000 mAh LiPo battery powers the system. Fig 14 shows the experiment components and the connections that we use to measure energy consumption. An Arduino Uno is also used to read the current sensor readings. Furthermore, we report the free space of SRAM to compare the required memory for each scenario. We develop each classifier according to Table IV, which comes with a particular sampling frequency and number of features. However, according to LSM303 datasheet, the sampling frequency can be set to $(1, 10, 25, 50, 100, 200, 400)$Hz. Therefore, we consider using $(1, 10, 25)$Hz sampling frequencies as a proof of concept for this experiment. We set the window size to 5 seconds for each sensor observation. The, we start sampling sensor data with the mentioned sampling frequencies, and at the same time, we measure the power consumption by the INA219 current sensor. Table VI shows the results of this experiment, including the average power consumption and memory usage. Battery life is calculated by considering a 1000mAh battery, which we use in this experiment. The required SRAM is determined by the difference between the total size of Cortex M0 SRAM (32KB) and the free space of RAM, which we report via our program. The program reports the space between the heap and stack of SRAM. As shown in Table VI, if we consider equal probability for each intensity level, then the average battery life of the system is 75.2 hours. We can use such a wearable system for 6 hours longer than the traditional multi-class classification method that uses a single classifier using a sampling frequency that is needed to reliably classify the most intense activities in the activity pool of interest.

G. Comparison with State-of-the Art

Table VII shows the specifications of the related studies - energy saving, classification accuracy, number of activities, and tested device - that offers power management in activity recognition systems. These studies are closely related to our work and we compare our results with these studies in Table VII. All these studies intend to reduce energy consumption by reducing sampling frequency of the inertial sensors. However, each study uses a different sensing module or different types of inertial sensors such as accelerometer, gyroscope, and magnetometer with different sampling frequencies. In addition, each study considered a different number of activity classes with varying intensity levels. We note that our proposed method is evaluated on two datasets consisting of 19 and 12 activity classes compared to the competing approaches, which are tested on datasets with 5-11 classes.

VI. Conclusion, Discussion, and Future Work

We designed, developed, and evaluated a probabilistic cascading binary classifier for a recourse-efficient wearable sensor. We aim to reduce the system's power consumption and memory usage by adjusting sampling frequency and feature computation in real-time for activity recognition. The activity intensity level, which is based on MET, is used to determine the number of features and the sampling frequency for recognizing each activity. The system samples sensor data with variable sampling frequencies based on daily activity history. The system also computes a varying number of features depending on the intensity level of the activity. The structure of the classifier changes according to an individual's activity pattern for more efficient resource management. We evaluated our proposed framework on two popular datasets, "PAMAP2 Physical Activity Monitoring Data Set" and " Daily and Sports Activities Data set". The final results show that the classification accuracy for each intensity group of these two datasets varies between 94.2% and 96.9%. Moreover, we measure the power consumption and memory usage of the approach on a micro-controller. The experimental result demonstrates that energy consumption can be reduced by 17.2%, while SRAM usage can be decreased by 58%.

As stated earlier, the probabilistic cascading binary classifier has 6 machine learning models. In this paper, we train each model with an accuracy threshold of 98%. This means that the number of features decided for each intensity level is dictated by the requirement that the classification accuracy of the trained model must be $\geq 98\%$. The accuracy threshold places a hard constraint on the classification accuracy and

TABLE VI: Power consumption and memory usage report for different sampling frequency.

Frequency (Hz)	Required SRAM (KB)	Power Consumption (mW)	Current Draw (mA)	Battery life (h)
1	4.16	48.03	12.10	82.64
10	6.31	52.53	13.29	75.24
25	9.91	58.04	14.51	68.91

TABLE VII: Comparison with state-of-the art methods for power management for human activity recognition systems.

Research	Energy Saving (%)	Classification Accuracy (%)	# Activities	Tested Device
[27]	25	85	11	HTC G11/Samsung i909
[49]	20-25	86.8	10	Samsung Galaxy S2
[45]	28	85.2	5	Samsung Galaxy Note
[26]	44.23-78.85	90	6	LG Optimus Pro
[42]	49	88	8	MI phone
This Study	10-17.2	96.4	19	Adafruit Feather M0

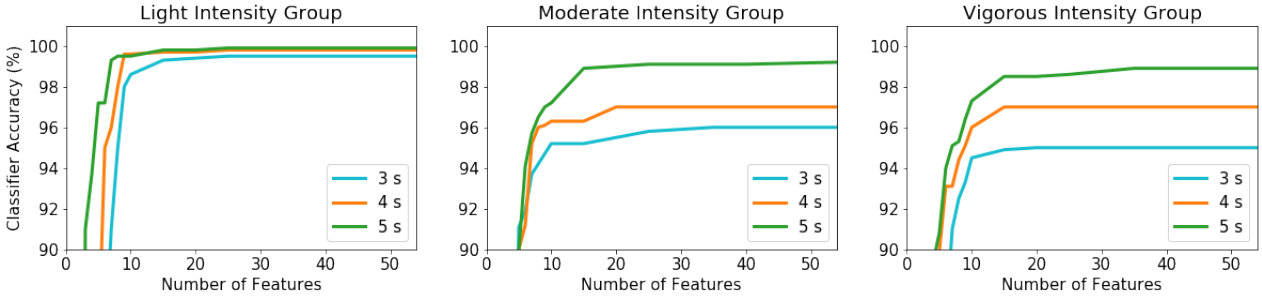

Fig. 15: MC_l, MC_m, and MC_v Random Forest classifier performance using different window sizes.

the resources - power and memory - required to achieve the threshold accuracy. By increasing the accuracy threshold for each component of the binary cascading system, we increase the classification accuracy of the cascading classifier since the cascading classifier is vulnerable to propagation error. Propagation error comes from the misclassification error of each individual component in the hierarchical architecture. However, by increasing the accuracy threshold of each component, we also increase the required resources. Therefore, there is a trade-off between classification accuracy and resources, which needs to be considered. As a result, if we need a lower classification error, we need to increase the accuracy threshold of the classifier's components and allow for flexible resource optimization. One limitation of our framework is that the system cannot fix the propagation error which happens on the go. To address this limitation in our future work, we plan to add an unsupervised (one-class classification method) component to the system that can recognize misclassified instances or instances from unseen activity and update the weight of the system in real-time. Also, we note that if the current activity is in the least likely activity intensity group, the system may need to re-process the data 2 additional times. The worst-case scenario happens when the higher intensity groups are less likely. As a result, the system needs to re-sample sensor data 2 times, which requires another 10 seconds of the activity. Therefore, we conclude that the classifier needs at least 15 seconds of each activity to recognize the activity correctly. If performing activities takes less than 15 seconds, we cannot guarantee the right prediction for such activity. Furthermore, the classifier process may need more than 10 seconds in the worst-case scenario.

Another consideration is the sampling frequency which supported by the sensors. Since sensors support specific sampling frequencies, we need to consider it as another factor to determine the proper sampling frequency of each intensity group. For instance, popular accelerometer sensors such as ADXL345 can be configured to use $(12.5, 25, 50, 100, 200, 400)$Hz as its sampling frequency. Therefore, we also need to consider the supported sampling frequency of the used IMU and other factors that have been mentioned. A comprehensive sampling frequency determination method considers activity frequency, sensor sampling rates, and classification accuracy for different sampling frequencies.

We have three design choices for the probabilistic cascading classifier: 1) sampling frequency, 2) the number of features, 3) window size. We consider a fixed value (5 seconds) for window size while using variable sampling frequency and feature set. Therefore, we only study the effects of variable sampling frequency and features in our analysis. We look deep into the impacts and the benefits of using varying window sizes in our future works. However, we briefly analyze the effects of using variable window size while the sampling frequency is fixed. Fig 15 demonstrates the classification results for each intensity levels (MC_l, MC_m, MC_v) using different window sizes and fixed sampling frequency (8Hz). We use the first dataset to illustrate the impact of window size. It can be seen that using greater window sizes significantly affects the classification accuracy for higher intensity activities.

We do not consider each feature's computation and the difference between processing each feature in this work. As future work, we plan to rank the computed features based on their contributions to energy consumption. This method considers the complexity of processing different features and compares them. It can also consider the sensor that needs to be sampled to extract that feature. For instance, we need to compute the mean of the signal from the accelerometer and gyroscope. As we know, the energy consumption of gyroscope is usually higher than the accelerometer. Therefore, the same feature from the gyroscope turns out to be more expensive, which can be considered for the final model. Also, this ranking method observes the required features for a model. If the majority of features are coming from one sensor of an IMU (e.g., magnetometer), then we are confident that we need to include that specific sensor. As a result, those features are considered less expensive than features from another sensor. Therefore, the ranking method tries to eliminate sampling from other sensors as long as it can replace its features with the features of the sensor that had the majority in our feature set, which can produce a reasonable classification accuracy. Finally, the output of this ranking method is the feature sets of binary and multi-class classifiers.

ACKNOWLEDGMENT

This work was supported in part by the National Science Foundation, under grants CNS-1750679, CNS-1932346, and IIS-1954372, and the Department of Education, under Graduate Assistance in Areas of National Need (GAANN) Grant P200A150115. Any opinions, findings, conclusions, or recommendations expressed in this material are those of the authors and do not necessarily reflect the views of the funding organizations.

REFERENCES

[1] ALINIA, P., SAEEDI, R., FALLAHZADEH, R., ROKNI, A., AND GHASEMZADEH, H. A reliable and reconfigurable signal processing framework for estimation of metabolic equivalent of task in wearable sensors. *IEEE Journal of Selected Topics in Signal Processing 10*, 5 (2016), 842–853.

[2] ALTUN, K., BARSHAN, B., AND TUNÇEL, O. Comparative study on classifying human activities with miniature inertial and magnetic sensors. *Pattern Recognition 43*, 10 (2010), 3605–3620.

[3] ANTONSSON, E. K., AND MANN, R. W. The frequency content of gait. *Journal of biomechanics 18*, 1 (1985), 39–47.

[4] BARR, M. Memory types. *Embedded Systems Programming 14*, 5 (2001), 103–104.

[5] BIJALWAN, V., SEMWAL, V. B., AND GUPTA, V. Wearable sensor-based pattern mining for human activity recognition: deep learning approach. *Industrial Robot: the international journal of robotics research and application* (2021).

[6] BIJALWAN, V., SEMWAL, V. B., AND MANDAL, T. Fusion of multi-sensor-based biomechanical gait analysis using vision and wearable sensor. *IEEE Sensors Journal 21*, 13 (2021), 14213–14220.

[7] BLACKWELL, D. L., CLARKE, T. C., ET AL. State variation in meeting the 2008 federal guidelines for both aerobic and muscle-strengthening activities through leisure-time physical activity among adults aged 18-64: United states, 2010-2015. *National Health Statistics Report*, 112 (2018).

[8] CHALLA, S. K., KUMAR, A., AND SEMWAL, V. B. A multibranch cnn-bilstm model for human activity recognition using wearable sensor data. *The Visual Computer* (2021), 1–15.

[9] CHAPELLE, O., AND KEERTHI, S. S. Multi-class feature selection with support vector machines. In *Proceedings of the American statistical association* (2008), vol. 58.

[10] COOK, D. J., AND DAS, S. K. How smart are our environments? an updated look at the state of the art. *Pervasive and mobile computing 3*, 2 (2007), 53–73.

[11] DARGIE, W. Analysis of time and frequency domain features of accelerometer measurements. In *2009 Proceedings of 18th International Conference on Computer Communications and Networks* (2009), IEEE, pp. 1–6.

[12] DUA, N., SINGH, S. N., AND SEMWAL, V. B. Multi-input cnn-gru based human activity recognition using wearable sensors. *Computing 103*, 7 (2021), 1461–1478.

[13] ERDMIER, C., HATCHER, J., AND LEE, M. Wearable device implications in the healthcare industry. *Journal of medical engineering & technology 40*, 4 (2016), 141–148.

[14] GHASEMZADEH, H., AMINI, N., SAEEDI, R., AND SARRAFZADEH, M. Power-aware computing in wearable sensor networks: An optimal feature selection. *Mobile Computing, IEEE Transactions on 14*, 4 (April 2015), 800–812.

[15] GUYON, I., WESTON, J., BARNHILL, S., AND VAPNIK, V. Gene selection for cancer classification using support vector machines. *Machine learning 46*, 1-3 (2002), 389–422.

[16] HEIDEMANN, J., AND GOVINDAN, R. An overview of embedded sensor networks. *Handbook of Networked and Embedded Control Systems* (2004), 1–20.

[17] HERRMANN, R., ZAPPI, P., AND ROSING, T. S. Context aware power management of mobile systems for sensing applications. In *IPSN* (2012), vol. 12, pp. 16–20.

[18] HUANG, W., ZHANG, L., GAO, W., MIN, F., AND HE, J. Shallow convolutional neural networks for human activity recognition using wearable sensors. *IEEE Transactions on Instrumentation and Measurement 70* (2021), 1–11.

[19] JAN, M. A., KHAN, F., KHAN, R., MASTORAKIS, S., MENON, V. G., ALAZAB, M., AND WATTERS, P. Lightweight mutual authentication and privacy-preservation scheme for intelligent wearable devices in industrial-cps. *IEEE Transactions on Industrial Informatics 17*, 8 (2020), 5829–5839.

[20] JI, T., AND PACHI, A. Frequency and velocity of people walking. *Structural Engineer 84*, 3 (2005), 36–40.

[21] JUNKER, H., LUKOWICZ, P., AND TROSTER, G. Sampling frequency, signal resolution and the accuracy of wearable context recognition systems. In *Wearable Computers, 2004. ISWC 2004. Eighth International Symposium on* (2004), vol. 1, IEEE, pp. 176–177.

[22] KALANTARIAN, H., ALSHURAFA, N., AND SARRAFZADEH, M. A survey of diet monitoring technology. *IEEE Pervasive Computing 16*, 1 (2017), 57–65.

[23] KATEVAS, K., HADDADI, H., AND TOKARCHUK, L. Sensingkit: Evaluating the sensor power consumption in ios devices. In *2016 12th International Conference on Intelligent Environments (IE)* (2016), IEEE, pp. 222–225.

[24] KHAN, A., HAMMERLA, N., MELLOR, S., AND PLÖTZ, T. Optimising sampling rates for accelerometer-based human activity recognition. *Pattern Recognition Letters 73* (2016), 33–40.

[25] KRAUSE, A., IHMIG, M., RANKIN, E., LEONG, D., GUPTA, S., SIEWIOREK, D., SMAILAGIC, A., DEISHER, M., AND SENGUPTA, U. Trading off prediction accuracy and power consumption for context-aware wearable computing. In *Wearable Computers, 2005. Proceedings. Ninth IEEE International Symposium on* (2005), IEEE, pp. 20–26.

[26] LEE, J., AND KIM, J. Energy-efficient real-time human activity recognition on smart mobile devices. *Mobile Information Systems 2016* (2016).

[27] LIANG, Y., ZHOU, X., YU, Z., AND GUO, B. Energy-efficient motion related activity recognition on mobile devices for pervasive healthcare. *Mobile Networks and Applications 19* (06 2013).

[28] LU, H., YANG, J., LIU, Z., LANE, N. D., CHOUDHURY, T., AND CAMPBELL, A. T. The jigsaw continuous sensing engine for mobile phone applications. In *Proceedings of the 8th ACM conference on embedded networked sensor systems* (2010), ACM, pp. 71–84.

[29] MANN, S., NOLAN, J., AND WELLMAN, B. Sousveillance: Inventing and using wearable computing devices for data collection in surveillance environments. *Surveillance & Society 1*, 3 (2002), 331–355.

[30] MILUZZO, E., LANE, N. D., FODOR, K., PETERSON, R., LU, H., MUSOLESI, M., EISENMAN, S. B., ZHENG, X., AND CAMPBELL, A. T. Sensing meets mobile social networks: the design, implementation and evaluation of the cenceme application. In *Proceedings of the 6th ACM conference on Embedded network sensor systems* (2008), ACM, pp. 337–350.

[31] MUKHOPADHYAY, S. C. Wearable sensors for human activity monitoring: A review. *IEEE sensors journal 15*, 3 (2014), 1321–1330.

[32] PASSARO, V., CUCCOVILLO, A., VAIANI, L., DE CARLO, M., AND CAMPANELLA, C. E. Gyroscope technology and applications: A review in the industrial perspective. *Sensors 17*, 10 (2017), 2284.

[33] PATIL, P., KUMAR, K. S., GAUD, N., AND SEMWAL, V. B. Clinical human gait classification: extreme learning machine approach. In *2019 1st international conference on advances in science, engineering and robotics technology (ICASERT)* (2019), IEEE, pp. 1–6.

[34] POLI, A., COSOLI, G., SCALISE, L., AND SPINSANTE, S. Impact of wearable measurement properties and data quality on adls classification accuracy. *IEEE Sensors Journal 21*, 13 (2020), 14221–14231.

[35] RAFFA, G., LEE, J., NACHMAN, L., AND SONG, J. Don't slow me down: Bringing energy efficiency to continuous gesture recognition. In *Wearable Computers (ISWC), 2010 International Symposium on* (2010), IEEE, pp. 1–8.

[36] REISS, A., AND STRICKER, D. Introducing a new benchmarked dataset for activity monitoring. In *2012 16th International Symposium on Wearable Computers* (2012), IEEE, pp. 108–109.

[37] ROFOUEI, M., PEDRAM, M., FRATERNALI, F., ASHARI, Z. E., AND GHASEMZADEH, H. Resource-efficient computing in wearable systems. In *2019 IEEE International Conference on Smart Computing (SMARTCOMP)* (2019), IEEE, pp. 150–155.

[38] ROKNI, S. A., GHASEMZADEH, H., AND HEZARJARIBI, N. Smart medication management, current technologies, and future directions. *Handbook of Research on Healthcare Administration and Management* (2016), 188–204.

[39] SCATAGLINI, S., ANDREONI, G., AND GALLANT, J. A review of smart clothing in military. In *Proceedings of the 2015 workshop on Wearable Systems and Applications* (2015), pp. 53–54.

[40] SEMWAL, V. B., GUPTA, A., AND LALWANI, P. An optimized hybrid deep learning model using ensemble learning approach for human walking activities recognition. *The Journal of Supercomputing 77*, 11 (2021), 12256–12279.

[41] SHANNON, C. E. Communication in the presence of noise. *Proceedings of the IRE 37*, 1 (1949), 10–21.

[42] SHI, J., ZUO, D., AND ZHANG, Z. An energy-efficient human activity recognition system based on smartphones. In *2020 7th International Conference on Soft Computing & Machine Intelligence (ISCMI)* (2020), IEEE, pp. 177–181.

[43] STIEFMEIER, T., ROGGEN, D., OGRIS, G., LUKOWICZ, P., AND TRÖSTER, G. Wearable activity tracking in car manufacturing. *IEEE Pervasive Computing 7*, 2 (2008), 42–50.

[44] TENG, Q., WANG, K., ZHANG, L., AND HE, J. The layer-wise training convolutional neural networks using local loss for sensor-based human activity recognition. *IEEE Sensors Journal 20*, 13 (2020), 7265–7274.

[45] VO QUANG VIET, HOANG MINH THANG, AND DEOKJAI CHOI. Adaptive energy-saving strategy for activity recognition on mobile phone. In *2012 IEEE International Symposium on Signal Processing and Information Technology (ISSPIT)* (2012), pp. 000095–000100.

[46] WANG, K., HE, J., AND ZHANG, L. Attention-based convolutional neural network for weakly labeled human activities' recognition with wearable sensors. *IEEE Sensors Journal 19*, 17 (2019), 7598–7604.

[47] WANG, Y., LIN, J., ANNAVARAM, M., JACOBSON, Q. A., HONG, J., KRISHNAMACHARI, B., AND SADEH, N. A framework of energy efficient mobile sensing for automatic user state recognition. In *Proceedings of the 7th international conference on Mobile systems, applications, and services* (2009), ACM, pp. 179–192.

[48] WU, H., CHEN, C., AND WENG, K. An energy-efficient strategy for microcontrollers. *Applied Sciences 11*, 6 (2021), 2581.

[49] YAN, Z., SUBBARAJU, V., CHAKRABORTY, D., MISRA, A., AND ABERER, K. Energy-efficient continuous activity recognition on mobile phones: An activity-adaptive approach. In *Wearable Computers (ISWC), 2012 16th International Symposium on* (2012), Ieee, pp. 17–24.

[50] ZAKIM, D., AND SCHWAB, M. Data collection as a barrier to personalized medicine. *Trends in pharmacological sciences 36*, 2 (2015), 68–71.