Real-Time Event-Driven Classification Technique for Early Detection and Prevention of Myocardial Infarction on Wearable Systems

Dionisije Sopic, *Student Member, IEEE*, Amin Aminifar, Amir Aminifar, *Member, IEEE*, and David Atienza, *Fellow, IEEE*

***Abstract*—A considerable portion of government health-care spending is allocated to the continuous monitoring of patients suffering from cardiovascular diseases, particularly myocardial infarction (MI). Wearable devices present a cost-effective means of monitoring patients' vital signs in ambulatory settings. A major challenge is to design such ultra-low energy devices for long-term patient monitoring. In this paper, we present a real-time event-driven classification technique based on the random forest classification scheme, which uses a confidence-related decision-making process. The main goal of this technique is to maintain a high classification accuracy while reducing the complexity of the classification algorithm. We validate our approach on a well-established and complete MI database (Physiobank, PTB Diagnostic ECG database [1]). Our experimental evaluation demonstrates that our real-time classification scheme outperforms the existing approaches in terms of energy consumption and battery lifetime by a factor of 2.60, with no classification quality loss.**

***Index Terms*—Myocardial infraction, wearables, random forest, event-driven.**

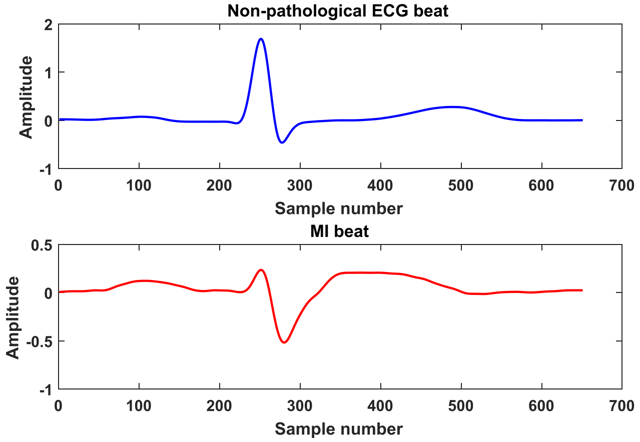

Fig. 1. An ECG beat from a healthy subject versus an ECG beat from a person with MI. MI beats have an ST elevation, i.e., the ST segment is abnormally high above the baseline.

I. Introduction

Myocardial infarction (MI), also known as a heart attack, remains one of the leading life-threatening conditions nowadays. MI is a very serious cardiac disease affecting people all around the world. Based on the latest statistics, in the USA alone, every 40 seconds someone gets a heart attack [2]. The economic burden of US hospitalizations due to MI in 2010 was already estimated at over 45 billion US dollars [3].

MI occurs due to fatty deposits called plaques that gradually form on the inner walls of coronary arteries. Due to smoking, hypertension, diabetes as well as family medical history, these plaques can build up significantly throughout the years. Then, a sudden rupture of these plaques at an unexpected moment triggers a blood clot to form. The blood clot can completely block an artery impeding the normal circulation of the blood. The muscle cells of the part of the heart that was getting supplied by the blocked artery become starved for oxygen and nutrients. Due to this lack of oxygen and nutrients, these cells begin dying.

The histological death of these starved cells begins in as little as 20 minutes after the artery occlusion [4], [5]. As soon as one of the coronary arteries becomes blocked, the person must receive treatment in less than 90 minutes. The earliest possible heart attack detection minimizes the number of dead cells. The mortality rate drastically increases by 41–62% if the delay between the hospital arrival and the performance of the treatment is longer than 2 hours [6]. The prolongation of this detection period results in irreversible consequences of the affected myocardial cells (heart attack). These consequences can be observed based on the altered electrocardiogram (ECG) beat morphology, as shown in Fig. 1. This altered ECG beat morphology is often reflected onto the ST segment abnormality (elevation or depression) and is the same across all ECG beats. Additionally, patients who have suffered a heart attack remain at an increased risk of recurrent heart attacks. The annual death rate of survivors is six times higher than in people who have not had a heart attack [7].

Manuscript received February 14, 2018; revised April 12, 2018; accepted May 16, 2018. This work was supported in part by the Hasler Foundation (Project 15048), and in part by the Office of Naval Research Global Award Grant N62909-17-1-2006. This paper was recommended by Associate Editor S. Carrara. *(Corresponding Author: Dionisije Sopic.)*

D. Sopic, Amir Aminifar, and D. Atienza are with the Embedded Systems Laboratory, Swiss Federal Institute of Technology Lausanne (EPFL), Lausanne 1015, Switzerland (e-mail: dionisije.sopic@epfl.ch; amir.aminifar@epfl.ch; david.atienza@epfl.ch).

Amin Aminifar is with the Department of Artificial Intelligence, Khajeh Nasir Toosi University of Technology, Tehran 470, Iran (e-mail: amin.aminifar@email.kntu.ac.ir).

Color versions of one or more of the figures in this paper are available online at http://ieeexplore.ieee.org.

Digital Object Identifier 10.1109/TBCAS.2018.2848477

In order to prevent a heart attack, adequate care should be provided to these patients. Thus, real-time patient monitoring is performed in hospitals through common tests for heart attack diagnosis. These tests include ECG and echocardiogram. However, all these tests are usually performed by bulky medical equipment. The lack of portability of this equipment, as well as its high energy consumption, make it unsuitable and uncomfortable for ambulatory monitoring.

Wearable devices can overcome the limitation of medical equipment for real-time patient monitoring. These low-cost devices are portable and can be used autonomously by patients [8]. In particular, the portability of these devices allows real-time remote patient monitoring on a daily basis. These devices allow physicians to access patient data remotely. Hence, they prevent further patient state deterioration by detecting early cardiac irregularities. This is done by sending patient data acquired from wearable devices using Bluetooth to a mobile phone, which is later on sent to physicians through the cloud. However, sending data using Bluetooth consumes a lot of energy, which drains the battery of wearable devices [9], [10]. Furthermore, not only is sending data to the cloud power-hungry [11], but its latency and reliable communication are also affected by the connection quality [12]. Nonetheless, many commercial wearable devices have stringent latency requirements. Thus, as constant good-quality network connections are only available at limited locations and at a high cost, and that even with good network connectivity the required latency often cannot be acquired [13], streaming data to the cloud is not suitable for real-time patient monitoring. Therefore, in order to overcome the aforementioned problems, the latest trend is to use smart wearable devices, or so called edge computing techniques [14], in which the entire processing is performed on the on-board microcontroller of the wearable device.

Nowadays, the algorithms running on smart wearables are mostly based on simple machine learning techniques [15], [16]. However, the major challenge of highly-accurate machine learning algorithms is their high computational complexity. Hence, these algorithms cannot be implemented on wearable devices for real-time monitoring. Therefore, these algorithms need to be highly optimized, which promotes the need for a paradigm shift in the classifier design. In this context, the event-driven computing approach offers a promising solution to reduce the computational complexity of embedded machine learning algorithms. Moreover, it can substantially lower the energy consumption of smart wearable devices.

In this paper, we extend our previous work [17] to the general case of nonlinearly-separable classes based on random forest classifications [18]. We refer to classes that cannot be separated by a linear line (hyperplane in N-dimensional space) as nonlinearly-separable classes. Moreover, we also generalize our previous work to several classification levels, which is important for the overall energy efficiency of wearable systems. Finally, we implement our hierarchical classification technique on an existing wearable device, namely SmartCardia INYU device [19], and evaluate its performance in terms of energy consumption and battery lifetime. The main contributions of this paper are:

1) Real-time early detection and prevention of myocardial infarction using an event-driven classification technique that uses a hierarchical classifier with multiple levels.
2) Analysis of computational complexity and energy efficiency of the proposed hierarchical classifier, as well as the design flow that allows users to synthesize high-accuracy event-driven hierarchical classifiers that meet users' battery lifetime requirements.
3) Validation on the MI database (Physiobank - PTB Diagnostic ECG database [1]).
4) Porting of our classification technique on a real-life wearable device, including a detailed evaluation of energy consumption and battery lifetime.

The remainder of this paper is organized as follows. In Section II, we review the previous work on different classification techniques proposed in the literature. We motivate the research in this paper by using an example described in Section III. Then, in Section IV, we propose a real-time event-driven classifier to reduce the energy consumption of wearable devices, while maintaining a high classification accuracy. The analysis of the computational complexity, battery lifetime, and energy efficiency of our approach is presented in Section V. The experimental setup used for validating our approach is presented in Section VI, whereas the experimental results are given in Section VII. Finally, the main conclusions of the paper are summarized in Section VIII.

II. Previous Work on Classification Techniques

With the revolution of wearable technology, many researchers have significantly invested in developing and refining these novel technologies. For instance, in [20] the authors propose a wearable, accurate, and energy-efficient system for obstructive sleep apnea monitoring using a single-channel ECG. A wearable system based on four electroencephalogram (EEG) electrodes for real-time detection of epileptic seizures has been proposed in [21]. Nonetheless, none of these devices target the detection of MI.

Several studies have been conducted concerning the detection of different cardiovascular diseases using various classification techniques. For instance, in [22], the authors use deep neural networks for classification of heart murmurs using a neuromorphic auditory sensor. In [23], it is proposed a patient-specific ECG heartbeat classifier with an adaptive implementation of 1-D CNNs. Then, in [24], the authors report a high accuracy in classifying normal and MI ECG beats from 47 features using the k-nearest neighbors classifier. However, none of these studies are performed on existing wearable devices, taking into consideration the stringent energy and memory constraints of these devices.

The need to reduce the energy consumption of wearable devices has given rise to a plethora of studies. In [25], a real-time classification scheme for automatic detection of abnormal heartbeats targeting embedded and resource-constrained wearables has been proposed. This scheme also incorporates an advanced digital signal processing block that is activated just when abnormal beats are detected, which considerably decreases the computational requirements and the energy consumption. First, from a medical reliability point of view, the authors do not investigate the confidence level of the obtained results, which is a key parameter in medical applications. Secondly, from an energy-efficiency point of view, their work focuses on a context where pathological heartbeats occur less frequently than normal ones. Therefore, in the case of many pathological heartbeats happening one after another, the advanced digital signal processing block will be successively invoked, which will increase the computational complexity and energy consumption.

In [17], we proposed a real-time event-driven classification technique for early detection and prevention of myocardial infarction through means of ultra-low energy wearable systems. This technique uses a hierarchical classifier that incorporates two linear support vector machine (SVM) classifiers [26]. The first-level classifier considers only a few features and is computationally efficient, but it cannot reach a very high classification accuracy. Based on anomaly detection [27], our work first assesses the decision confidence of the first-level classifier. Then, if the first-level classifier cannot make a confident decision, the second-level classifier is invoked. In this case, we keep the features from the first-level classifier and calculate the rest of the features used in the second-level one. The final classification decision is that of the second-level one. This classifier uses much greater number of features and is very accurate, but computationally complex. Therefore, by combining the benefits of the two classifiers, i.e., the computational complexity from the first-level classifier and the high classification accuracy from the second-level one, it is possible to reduce the computational complexity of the classification algorithm, while still maintaining a high classification accuracy. The reduction in computational complexity, in turn, results in a longer battery lifetime, which is an important factor for wearable devices. However, our proposed approach in [17] is restricted to linearly-separable classes. Furthermore, it is also restricted to only two levels, which limits the classification performance and energy efficiency of our system.

In this work, we overcome the limitations of our previous work by extending our event-driven hierarchical technique to the case of nonlinearly-separable classes. We also enable a generalization with multiple levels through a confidence-related decision-making process, and we present a design flow of our event-driven classification technique that satisfies the battery lifetime requirements.

III. Motivational Example

In this section, we illustrate the main idea of our approach using a small example. Without loss of generality, and for the simplicity of the presentation, we herein consider two sets of features for the binary classification problem, which is shown in Fig. 2. We suppose that the computational complexity of the feature set that is along the vertical axis (feature set 2) is higher than the computational complexity of the one along the horizontal axis (feature set 1). For instance, feature set 1 can contain time-domain features of the dataset, whereas feature set 2 can contain frequency-domain features. Time-domain features have a complexity order of $\mathcal{O}(n)$, where n is the signal's length, whereas the frequency-domain features have a complexity order of $\mathcal{O}(n \log_2 n)$, as the calculation of frequency-domain features requires additional signal transformations, such as, the Fourier transform.

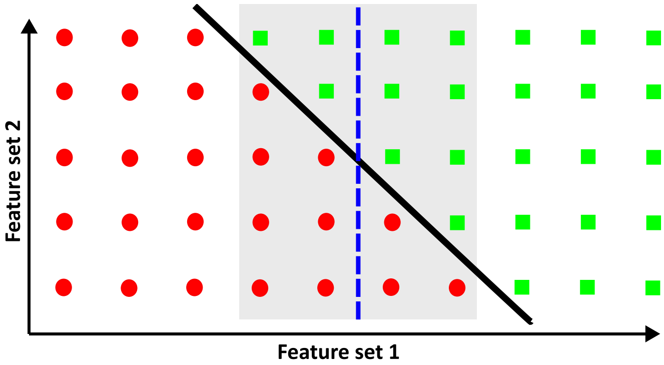

Fig. 2. Motivational example.

In this example, we consider 25 circle-shaped samples of class 1 and another 25 square-shaped samples of class 2. For instance, in the case of MI, circle-shaped samples belong to people suffering from MI, whereas square-shaped ones belong to healthy subjects. Let us suppose that $n = 2^{10}$. Depending on the confidence level, we can build three different linear classifiers.

The first classifier is shown by the dashed line in Fig. 2. This classifier uses only feature set 1 to separate two classes. As it can be observed in Fig. 2, if we use this classifier some samples within the shaded gray area will be misclassified. The accuracy and the expected computational complexity of this classifier are: $Accuracy_{\text{dashed}} = 88\%$, $Complexity_{\text{dashed}} = n = 2^{10}$. Hence, the expected computational complexity of this classifier is low, whereas its classification accuracy is lower than that of the optimal solution.

Another alternative is to use both feature sets. This is done using the second classifier shown by the solid line in Fig. 2. The accuracy and the expected computational complexity of this classifier are $Accuracy_{\text{solid}} = 100\%$, $Complexity_{\text{solid}} = n \log_2 n = 10240$, respectively. This classifier outperforms the first classifier in terms of classification accuracy, but it is ten times more computationally complex.

Finally, the hierarchical classifier combines the benefits of the two previous classifiers. Based on the desired confidence level, we either use the classifier that uses feature set 1 (i.e., the dashed line, the first classifier) or the one that uses both sets of features (i.e., the solid line, the second one). The main goal of this scheme is to reduce the classifier complexity in terms of the number of features that will be used for the final classification, while maintaining a high classification accuracy. As shown in Fig. 2, the first classifier cannot make confident decisions for samples that happen to be in the shaded gray area. For these samples, the second classifier, i.e., the classifier that uses all available features should be used to target medical applications that truly require a high confidence level. Hence, once the region in which the first classifier does not provide high confidence results is identified, the next step is to check for each testing example if it falls into this shaded area and, if so, to use the second classifier. Otherwise, we use the first classifier, i.e., the classifier with the reduced number of features. For this particular example, let us suppose that we have found the region in which the first classifier does not provide high confidence results, shown in Fig. 2. If we use the hierarchical classifier, the classification accuracy is $Accuracy_{\text{hierarchical}} = 100\%$, whereas its expected classification complexity is calculated as:

$$E(C) = \frac{30}{50} \cdot n + \frac{20}{50} \cdot n \cdot \log_2 n.$$

For $n = 2^{10}$, the expected classification complexity is $E_{\text{hierarchical}}(C) = 4710.4$, whereas with the classical approach that uses all available features (shown by the solid line in Fig. 2) we obtain $E_{\text{solid}}(C) = \frac{50}{50} \cdot n \cdot \log_2 n = 10240$. Hence, for this motivational example and $n = 2^{10}$, our approach reduces the classification complexity by a factor of 2.

In summary, for our motivational example shown in Fig. 2, we have presented three different classifiers. The first classifier (the dashed line) has a low computational complexity, but its classification accuracy is much lower than the performance of the second one (the solid line). On the other hand, the second classifier has a high classification accuracy, but it is significantly more complex than the first one. Thus, our hierarchical classifier combines the two previous classifiers, such that we get a classifier that has a high classification accuracy and a low computational complexity.

IV. Real-time Event-driven Classification Technique

In this section, we propose a real-time event-driven classification technique for early detection and prevention of MI based on random forest [18]. Our real-time event-driven technique incorporates $m + 1$ different classification levels. The number of features at level i is equal to $i \cdot K$, where the parameter K is used by the designer to make a trade-off between the performance of our system, in terms of classification accuracy and energy efficiency, and the complexity of design space exploration.

In an event-driven computing paradigm, the execution of a particular action depends on the occurrence of predefined trigger events [28]. Therefore, the event-driven schemes perform the most computationally expensive processing only in case a particular trigger event occurs, which reduces the computational complexity and, therefore, enhances the battery lifetime of wearable devices. In our case, based on the required confidence level, we choose the classification level that is used for classifying a sample data. The overall flow of our approach is shown in Fig. 3. Our proposed classification technique consists of two main phases: the offline phase that is explained in Section IV-A, and the online phase explained in Section IV-B.

A. *Offline Phase of Our Real-time Event-driven Classification Technique*

In an event-driven classification scheme very often we do not need to compute all available features to make confident decisions, as often they can be made based on only a few features.

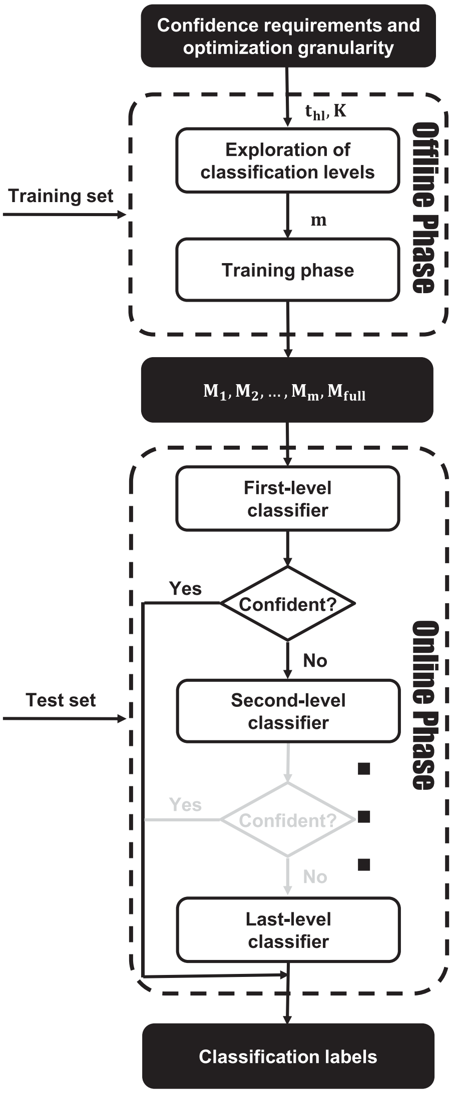

Fig. 3. Diagram of our event-driven classification technique that consists of $m + 1$ classification levels. Each classification level i contains $i \cdot K$ features, where $i = 1, \dots, m$, whereas the last classification level contains all available features. Parameter K is used by the designer to make a trade-off between the classification accuracy performance of our system, its energy efficiency, and the complexity of design space exploration.

Reducing the number of features that we need to compute for making confident decisions, in turn, reduces the energy consumption of our system. Therefore, the basic idea is to use a hierarchical classifier, where all features are computed only when needed.

Consequently, in our technique the classifiers at lower levels consider a limited number of features and, therefore, are computationally efficient, but do not provide as high classification accuracy as those at higher levels. On the other hand, classifiers at higher levels can provide a high classification accuracy, but are computationally complex. In our real-time event-driven classification technique, classifiers at higher levels are invoked only if classifiers at lower levels are unable to classify a sample data with the required level of confidence.

The inputs of our proposed hierarchical classification techniques are:

Algorithm 1: Exploration of classification levels.

```
function Training$t_{hl}, K, T_{min}, \mathbf{X}, \mathbf{y}$
    $m = 1;$
    $Calculate\ T_{lifetime}(m)\ based\ on\ Eq.\ (8)$
    while $(T_{lifetime}(m) < T_{min})$ do
        $m = m + 1$
        $Calculate\ T_{lifetime}(m)\ based\ on\ Eq.\ (8)$
    end while
    $Sort\ features$
    $Shuffle\ the\ data$
    $S_1 \leftarrow 0.8 \cdot length(data)$
    $S_2 \leftarrow 0.2 \cdot length(data)$
    $\mathbf{mean_v}, \mathbf{sig_v}, \overline{S_1} \leftarrow normalization(S_1)$
    for $j = 1$ to $m$ do
        $M_j \leftarrow Train(\overline{S_1}, K, number_{levels} = j)$
    end for
    $M_{full} \leftarrow Train(all\ features)$
    $\overline{S_2} \leftarrow normalization(S_2)\ based\ on\ \mathbf{mean_v}, \mathbf{sig_v}$
    $Test(\overline{S_2}, t_{hl}, K, M_1, \cdots, M_m, M_{full})$
    $Calculate\ Gmean\ based\ on\ Eq.(10)\ on\ S_2$
end function
```

1) input feature matrix, denoted by $\mathbf{X}$, where the overall number of available features is denoted by n,
2) class labels for each training example, denoted by $\mathbf{y}$.

The input feature matrix contains features for both classes. Each column of the input matrix corresponds to a feature, whereas each row corresponds to an observation. The main goal is to design a high-accuracy hierarchical classifier satisfying the lifetime requirements, in case such classifier can be designed.

We first extract the features from the input signals. Then, we sort the features based on their relevance by using the infinite latent feature selection algorithm to find the most informative ones. Here, we consider an algorithm that uses a robust probabilistic latent graph-based feature selection algorithm that performs feature ranking by considering all possible subsets of features as paths on a graph [29]. The relevancy of each feature is modelled using the probabilistic latent semantic analysis (PLSA) technique [30]. However, our approach is not restricted to this feature selection method.

The number of classification levels is found based on Algorithm 1. Note that $T_{lifetime}$ refers to the lifetime of our system, whereas T_{min} refers to the minimum expected lifetime of the wearable device. The condition that uses $T_{lifetime}$ in Algorithm 1 is thoroughly explained in Section V. Each classification level i contains features from the previous classification level, along with additional features that belong to level i. The last classification level contains all available features. In order to make sure that all features are on a similar scale, before training any of the classifiers, we normalize each feature by subtracting its mean value and dividing it by its standard deviation. We store the mean value ($\mathbf{mean_v}$) and the standard deviation of each feature ($\mathbf{sig_v}$).

Next, we apply the random forest algorithm [31] based on the confidence-related decision making process for training each classification level. Random forest is an ensemble of decision trees that are combined to classify a sample data by aggregating decisions of all individual trees. This aggregation of decisions of all trees in the forest reduces the variance of the prediction, resulting in a low-variance model and a robust outcome. Each individual decision tree in the forest is constructed using a different bootstrap sample of data. For each bootstrap sample, we grow a tree based on the randomly selected subsets of features at each node [31]. Finally, the tree is grown until no further splits are possible. The usual random forest classifies a new sample data based on the majority votes among the other trees in the forest.

In this work, we adopt the following approach for aggregating decisions from individual trees. We define the parameter t_h which represents a percentage of mutually agreed trees. This parameter is used for inspecting the confidence level of the obtained results. Comparing the value of this parameter to the value of the decision-making threshold set in the design process t_{hl}, we decide which classifier is invoked. Random forest classifiers at each classification level use 100 weak learners.

The input feature matrix $\mathbf{X}$ and the class labels for every training example $\mathbf{y}$ are used in the offline phase, as shown in Fig. 3. The outputs of this phase are the number of classification levels m, as well as the $m+1$ classifiers, namely $M_1, \ldots, M_m, M_{full}$. The last-level classifier M_{full} uses all of the available features.

B. Online Phase of Our Real-time Event-driven Classification Technique

The overall flow of the online phase of our proposed hierarchical classification technique is shown in Fig. 4. Before applying any of the classifiers, we first normalize a new testing example. When classifying it, we first calculate features from the first-level classifier. We inspect a decision of each tree and calculate the parameter t_h in the following way. Assuming that the number of trees in the forest is N_T, we define functions $F(t_i)$ and $G(t_i)$ for each decision tree t_i within the forest, as follows:

$$F(t_i) = \begin{cases} 1, & \text{if } \mathsf{decision}(\mathsf{t_i}) = +1 \\ 0, & otherwise \end{cases}$$

$$G(t_i) = \begin{cases} 1, & \text{if } \mathsf{decision}(\mathsf{t_i}) = -1 \\ 0, & otherwise. \end{cases}$$

Labels $\mathsf{t}_i = +1$ and $\mathsf{t}_i = -1$ are assigned to different classes. Feature vectors that correspond to non-MI ECG beats are labelled as -1, whereas those extracted from MI ECG beats are labelled as 1. The value of t_h is calculated as follows:

$$t_h = \frac{\mathsf{max}(\sum_{i=1}^{N_T} F(t_i), \sum_{i=1}^{N_T} G(t_i))}{N_T}.$$

If the first-level classifier cannot make a confident decision, i.e., if the value of the parameter t_h is below the decision-making threshold t_{hl}, we keep the first k_1 features in the feature set, and we calculate the rest of $k_2 - k_1$ features of the second-level classifier. This process is repeated until one of the classifiers matches the criteria for making a confident decision. As previously

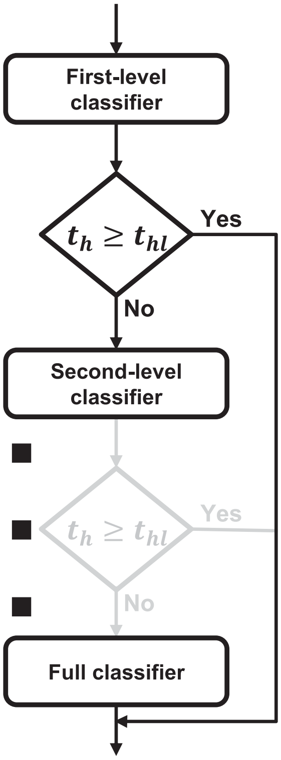

Fig. 4. Flowchart of the online phase of our event-driven classification technique.

mentioned, the last classification level contains all available features. Therefore, in case none of previous m classifiers can make a confident decision, the full classifier is invoked.

V. Analysis of Our Real-time Event-driven Technique

In this section, we analyze the complexity, lifetime and energy efficiency of our proposed approach. In Section V-A, we estimate the expected complexity of our real-time event-driven technique. The energy consumption of our approach is estimated in Section V-B.

A. *Complexity of Proposed Real-time Event-driven Classification Technique*

For the sake of simplicity, in this discussion we focus on a two-level classifier in which the first-level considers $k_1 < n$ features, while the second-level considers all of n available features, including k_1 features from the first-level classifier. We first invoke the first-level classifier to classify a new data sample by calculating its k_1 features. In case the first-level classifier is unable to classify a sample data with the required level of confidence, the second-level classifier is invoked. In this case, we keep k_1 calculated features and we calculate the other $n - k_1$ features. Therefore, the expected computational complexity (the mathematical expectation denoted by $\mathbb{E}(\cdot)$) of this two-level classifier $\mathbb{E}(C_2)$ is computed as follows:

$$\mathbb{E}(C_2) = \sum_{i=1}^{k_1} c_i + (1 - p_1) \cdot \sum_{i=k_1+1}^{n} c_i, \qquad (1)$$

where p_1 is defined as the probability that the first-level classifier is sufficient for making confident decisions, whereas c_i represents the computational complexity of feature i. By reorganizing addends in Eq. (1), we derive the following equation:

$$\mathbb{E}(C_2) = p_1 \cdot \sum_{i=1}^{k_1} c_i + (1 - p_1) \cdot \sum_{i=1}^{n} c_i, \qquad (2)$$

where $\sum_{i=1}^{k_1} c_i$ represents the complexity of the first-level classifier and $\sum_{i=1}^{n} c_i$ that of the second-level (which uses all available features). The first-level classifier is invoked with a probability of p_1. If the first-level classifier fails to classify a new data sample with the required level of confidence, the second-level classifier is invoked with a probability of $1 - p_1$. Therefore, in Eq. (2), the complexity of the first-level classifier is multiplied by p_1, whereas the complexity of the second-level one is multiplied by $1 - p_1$. For $k_1 < n$, the following inequality holds:

$$p_1 \cdot \sum_{i=1}^{k_1} c_i + (1 - p_1) \cdot \sum_{i=1}^{n} c_i < \sum_{i=1}^{n} c_i, \qquad (3)$$

which indicates that the overall computational complexity of the two-level classifier is always smaller than that of the second-level classifier (the classifier that uses all available features).

Let us now consider the case of a three-level classifier in which the first-level is comprised of k_1 features, the second-level includes $k_2 > k_1$ features (including k_1 features of the first-level classifier), and the third-level has all of n available features (including all of the features of the previous classifiers). First, we invoke the first-level classifier for classifying a new data sample. If the first-level classifier cannot classify a new data sample with the required level of confidence, we invoke the second-level classifier by calculating its remaining $k_2 - k_1$ features. In case neither the first-level nor the second-level classifier can classify a new data sample with the required confidence level, the third-level classifier is invoked. In this case, we keep k_2 features of the previous two classification levels and calculate the other $n - k_2$ features of the third-level classifier. Therefore, the expected computational complexity of the three-level classifier is computed as follows:

$$\mathbb{E}(C_3) = p_1 \cdot \sum_{i=1}^{k_1} c_i + (1 - p_1) \cdot \left(p_2 \cdot \sum_{i=1}^{k_2} c_i + (1 - p_2) \cdot \sum_{i=1}^{n} c_i \right).$$

For $k_2 < n$, the following inequality holds:

$$\mathbb{E}(C_3) < \mathbb{E}(C_2), \qquad (4)$$

showing that the computational complexity of the third-level classifier is smaller than that of the second-level one. Taking into consideration both Eq. (3) and (4), for $k_1 < k_2 < n$ we have the following inequality:

$$\mathbb{E}(C_3) < \mathbb{E}(C_2) < \sum_{i=1}^{n} c_i, \qquad (5)$$

which shows that the computational complexity decreases as the number of classification levels increases.

In the general case, considering that m represents the number of classification levels, k_i the number of features at level i, where

$i = 1, \dots, m$, $k_i = i \cdot K$, c_r the computational complexity of feature r, n the number of all available features, and p_i the probability that the classifier at level k_i is sufficient for making confident decisions, the expected computational complexity of our hierarchical classifier can be estimated as follows:

$$\mathbb{E}(C) = \left(\sum_{i=1}^{m} \prod_{j=1}^{i-1} (1 - p_j) \cdot p_i \cdot \sum_{r=1}^{k_i} c_r \right) + \prod_{j=1}^{m} (1 - p_j) \cdot \sum_{r=1}^{n} c_r, \tag{6}$$

where $k_1 < k_2 < \ldots < k_m < n$, and we have:

$$\sum_{i=1}^{m} \prod_{j=1}^{i-1} (1 - p_j) \cdot p_i + \prod_{j=1}^{m} (1 - p_j) = 1,$$

as the sums of computational complexities in Eq. (6) are multiplied by the product of probabilities, which must add up to 1.

B. Energy Consumption of Our Real-time Event-driven Technique

Following the same analogy as in Section V-A, we estimate the energy consumed by our hierarchical classifier as follows:

$$\mathbb{E}(E_C) = \left(\sum_{i=1}^{m} \prod_{j=1}^{i-1} (1 - p_j) \cdot p_i \cdot \sum_{r=1}^{k_i} E_r \right) + \prod_{j=1}^{m} (1 - p_j) \cdot \sum_{r=1}^{n} E_r,$$

where E_r represents the energy spent on the calculation of feature r, and we have:

$$\sum_{i=1}^{m} \prod_{j=1}^{i-1} (1 - p_j) \cdot p_i + \prod_{j=1}^{m} (1 - p_j) = 1.$$

In order to meet the battery lifetime requirements, let us assume that the minimum required lifetime of the wearable device is T_{min} hours. Considering the fact that we want the battery of the wearable device to last for at least T_{min} hours, we have the following set of inequalities:

$$T_{lifetime} \geq T_{min}, \tag{7}$$

$$\Big[E_S \cdot (1 - d_C) + (E_C + E_P) \cdot d_C \Big] \cdot T_{lifetime} = E_B, \tag{8}$$

where $T_{lifetime}$, E_S, d_C, E_C, E_P, E_B represent the lifetime of the wearable system, the energy of the system consumed in the idle state, the CPU duty cycle, the energy spent by our hierarchical classifier (Eq. (6)), the energy spent on the preprocessing, and the battery storage according to its specifications, respectively. The energy spent on the calculation of each feature along with the energy spent in the preprocessing stage can be estimated on the wearable device. By combining inequality (7) and Eq. (8) we obtain the following inequality:

$$E_C \leq \frac{\left(\frac{E_B}{T_{min}} - E_S \cdot (1 - d_C) \right)}{d_C} - E_P. \tag{9}$$

VI. Experimental Setup

In this section, we validate our proposed technique in terms of classification quality on a set of real-life ECG signals from a control group and from patients suffering from MI. In Section VI-A, we introduce the classification metrics that we use to evaluate our approach in terms of classification performance. Section VI-B describes the real-life MI database used for the evaluation of our hierarchical technique along with the main preprocessing steps. The target platform used for deploying our real-time event-driven classification technique is described in Subsection VI-C.

A. Classification Performance Metrics

The simplest diagnostic test used in the medical community combines different metrics to classify patients into two different groups. The majority of medical studies often report two different metrics used to correctly assess the ability of diagnostic tests: sensitivity and specificity [32]. The classification is based on the presence or the absence of a particular symptom of interest [33]. In order to capture the information obtained from both, sensitivity and specificity, we use their geometric mean (gmean, [34]) to inspect the classification performance of our real-time event-driven classifier. These metrics are defined as follows:

$$\mathsf{sensitivity} = \frac{\mathsf{tp}}{\mathsf{tp} + \mathsf{fn}},$$

$$\mathsf{specificity} = \frac{\mathsf{tn}}{\mathsf{tn} + \mathsf{fp}},$$

$$\mathsf{G}_{mean} = \sqrt{\mathsf{sensitivity} \cdot \mathsf{specificity}}, \tag{10}$$

where tp, tn, fp, fn represent the number of true positive, true negative, false positive, and false negative, respectively. The final classification performance is evaluated based on the test set. Each test sample is classified using the first classifier that can make a confident decision.

B. The PTB Diagnostic ECG Database

We evaluate our technique on the ECG signals from the Physionet (PTB Diagnostic ECG database) open access database [1]. Signals from two groups of subjects are used for ECG beat classification: the control group and the group of patients. The control group contains ECG signals from 52 healthy subjects, whereas the group of patients consists of 52 patients who have already had an MI. All ECG signals are sampled at $f_s = 1000$ Hz.

In order to perform ECG beat classification, we need to subtract ECG beats for each person in the database. Therefore, we first filter ECG signals to remove the baseline wander and high frequency noise. The baseline wander is removed through morphological filtering [35]. Furthermore, we apply a zero-phase FIR band-pass filter of order 32 with cut-off frequencies $f_1 = 0.05$ Hz and $f_2 = 40$ Hz for high-frequency noise removal. Pan-Tompkin's algorithm is used for ECG R-peaks detection [36]. Similar to the previous study in [24], the segmentation of individual beats is performed by taking $f_s/4$ samples to the left

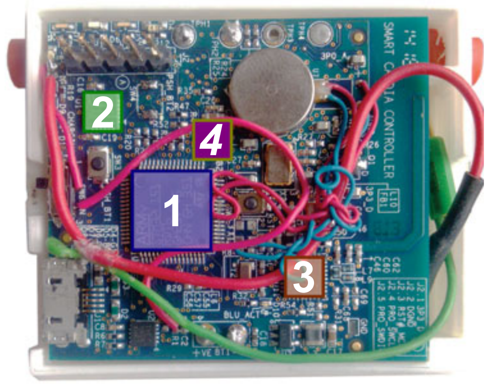

Fig. 5. Front: 1: STM32L151RDT6 (ARM Cortex-M3 MCU, 384 KB Flash, 48 KB RAM), 2: MPU-6000 (6-axis I^2C motion sensor), 3: nRF8001 (Bluetooth low energy v4.0 radio). Back: 4: ADS1191 (Analog front-end for ECG applications).

and $f_s/2.5$ samples to the right of ECG R-peaks. We consider ECG signals coming from Lead 11. Moreover, we decompose each individual ECG beat using a discrete wavelet transform (DWT) down to level four with Daubechies 6 (db6) as a basis function, which results in four detail and four approximation DWT coefficients. The main features used in our classification technique are extracted from these eight coefficients. Namely, the normalized signal energy, the Higuchi's fractal dimension [37], along with the following entropies: approximate, fuzzy, permutation, wavelet, Shannon, Renyi, and Tsallis [38]. These features all together represent the input feature matrix $\mathbf{X}$. The features are sorted based on their relevance using the infinite latent feature selection algorithm [29]. However, our technique is general, thus it is not limited to this particular feature selection algorithm.

C. Target Platform

We consider the SmartCardia INYU wearable sensor [19] as our target device in this work. In this device, a single-lead ECG signal is obtained through an ECG sensor with a 24-bit ADC [39] operating at a frequency that ranges from 125 Hz up to 16 KHz, with up to 16-bit resolution. This ADC is designed specifically for ultra-low power ECG applications. Then, this device features an ultra-low power 32-bit microcontroller STM32L151 [40] with an ARM Cortex–M3 on which the entire processing is performed with the possibility of operation at a maximum frequency of 32 MHz. The SmartCardia device also has a 48 KB RAM, 384 KB Flash, and a standard 710 mAh battery. The prototype of the SmartCardia INYU device is shown in Fig. 5. More detailed information about this device can be found in [41].

VII. Experimental Results

A. Classification Quality Evaluation

We split the entire ECG database into training and test sets. The data of one patient is either put in the training set or in the test set. Therefore, all features extracted from ECG beats of one subject are assigned to one of these sets. The training set contains 80% of randomly selected subjects, whereas the remaining 20% percent of subjects is used in the test set. Considering the fact that we have 104 subjects in our database, we use 83 subjects in the training set, and 21 subjects in the test set. In order to get robust results, this random split is performed ten times, and the geometrical mean of these ten repetitions is reported as the only correct average of normalized measurements [34]. For classification quality evaluation, both the online and offline phases of our classification technique are implemented in MATLAB.

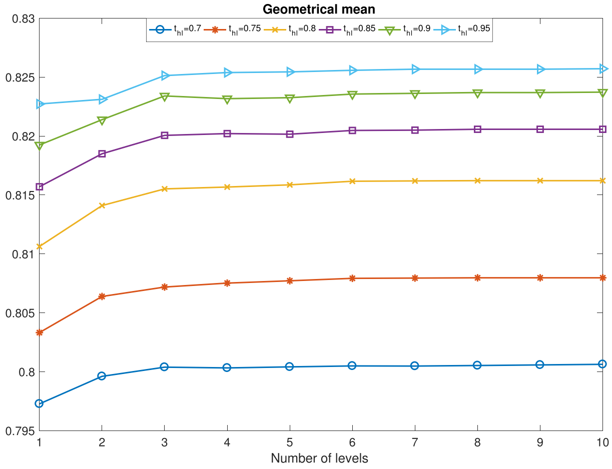

Fig. 6. Geometrical mean of our real-time event-driven classification technique versus the number of classification levels for different values of decision-making thresholds t_{h_l}.

In our implementation, we fix the first level classifier to contain $K = 5$ features. Each succeeding classification level contains features from the previous level and the next five relevant features obtained through feature selection. For instance, for $m = 2$, we have three different classifiers within our classification technique. Namely, the first-level classifier that contains five features, the second-level one that contains ten features including the five features from the first-level classifier, and the full classifier that contains the entire set of available features ($n = 72$). The classifier that uses all available features reaches a geometric mean of 83.26% (Sensitivity = 87.95%, Specificity = 78.82%). The overall geometrical mean of sensitivity and specificity, as well as the expected complexity of our proposed event-driven classification technique, are estimated for different values of decision-making thresholds t_{h_l} and classification levels m. Fig. 6 shows the overall geometrical mean of sensitivity and specificity of our proposed event-driven classification technique (vertical axis) versus the number of classification levels (horizontal axis) for different values of decision-making thresholds. As shown in Fig. 6, an increase in the number of classification levels leads to an increase in the classification performance. This is due to the fact that by increasing the value of decision-making threshold, we start invoking higher level classifiers that are more accurate. However, we see that for a particular value of decision-making threshold, for $m \geq 4$, there is no major improvement in terms of classification quality regarding the number of classification levels.

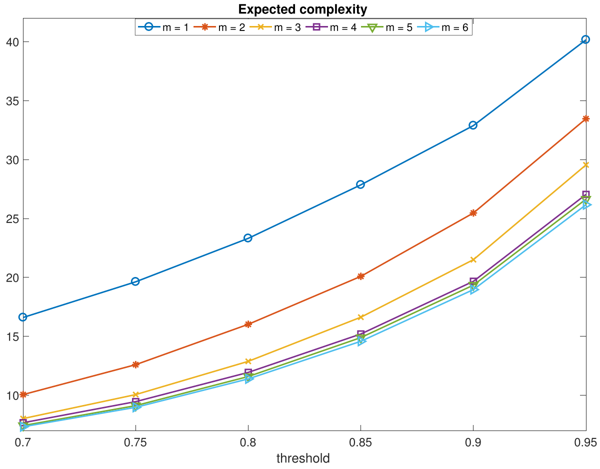

Fig. 7. Expected complexity ($\mathbb{E}(C)$, in Section V) of our real-time event-driven classification technique versus the value of decision-making threshold for different number of classification levels m.

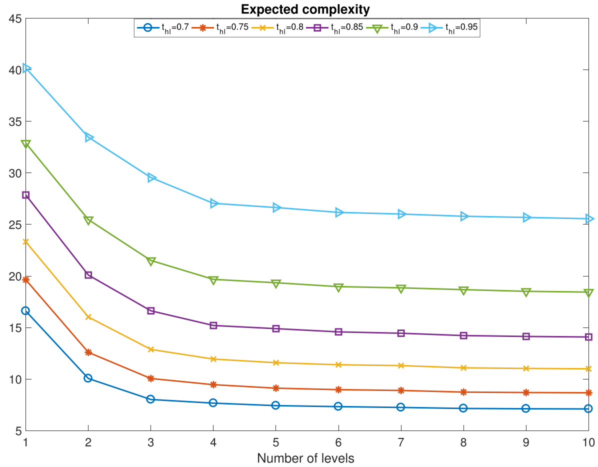

Fig. 8. Expected complexity ($\mathbb{E}(C)$, in Section V) of our real-time event-driven classification technique versus the number of classification levels for different values of decision-making thresholds t_{h_l}.

Fig. 7 shows the expected complexity of our technique versus the value of the decision-making threshold for different number of classification levels m. From this figure, we can see that an increase in the value of decision-making threshold leads to an increase in the expected complexity. This is due to the fact that for higher values of decision-making threshold, we invoke higher level classifiers that use more features, and therefore, are more complex. As seen in Fig. 7, for $m \geq 4$, there is no major improvement in terms of computational complexity regarding the value of decision-making threshold.

Similar to Fig. 6, based on Fig. 8, we see that for a fixed value of decision-making threshold, also after $m = 4$ classification levels, there is no improvement in terms of expected computational complexity either. Therefore, from Fig. 8, we can deduce that for our experiment only four classification levels are needed, they are sufficient to fulfill the required classification performance (with minimum complexity possible). The full classifier that reaches an accuracy of 83.26% (Sensitivity = 87.95%, Specificity = 78.82%) uses $n = 72$ features. As shown in Fig. 8, in the case of having four classification levels and applying $t_{hl} = 0.7$ as a decision-making threshold, we end up with an expected computational complexity of 7.7 and an accuracy of 80.32% (Sensitivity = 81.02%, Specificity = 79.63%). As the expected computational complexity is related to the number of used features, this negligible loss of 3% in terms of classification quality, reduces the computational complexity by a factor of ten.

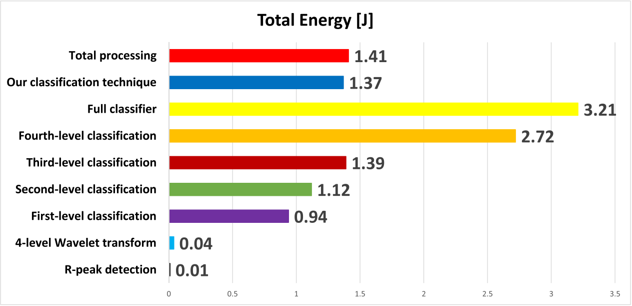

Fig. 9. Overview of the energy consumption of different classification levels.

B. *Energy Consumption and System Lifetime Analysis*

We consider the same code and inputs of the experiments performed in Section VII, and subsequently ported into C code to assess the classification performance and the energy consumption of our event-driven classification technique against the energy consumption of the full classifier on a commercially available Gecko EFM32 development board [42]. This board includes the same ARM Cortex-M3 core as in the SmartCardia INYU device, and provides the Simplicity Studio software in which a full energy profiler is integrated. The detailed energy consumption of our technique along with different classification levels is shown Fig. 9. Herein, we consider $m = 4$ classification levels and we fix the number of features in the first level to $K = 5$. The full classifier uses $n = 72$ features. By running our classification technique on the Gecko EFM32 board, we obtain that the execution time for processing of one heartbeat is equal to $t_1 = 24.27$ s. On the other hand, the time it takes for the full classifier to process one heartbeat is equal to $t_2 = 69.95$ s. Fixing that the processing of one heartbeat is done once every 90 seconds, the CPU duty cycle (d_C, used in Section V) of the full classifier is 77.73%, whereas the CPU duty cycle of our approach is equal to 26.97%. For a standard 710 mAh battery, since the processing is done once every 90 seconds, the full classifier runs for 59.83 hours on a single battery charge. Our proposed classification technique reaches 155.41 hours, thus allowing for more than 6 days of operation. Therefore, our event-driven classification technique extends the battery life by a factor of 2.60. The energy consumption of different components of the SmartCardia INYU device is shown in Table I.

TABLE I
CURRENT USED FOR MI DETECTION ON THE TARGET DEVICE

Operation	Current (mA)	Duty cycle (%)	Avg. current (mA)
ADS1191 [37]	0.427	100	0.427
MPU-6000 [41]	0.005	100	0.005
Signal acquisition subsystem			**0.432**
ECG delineation	14.397	0.25	0.036
MI processing	14.397	33.04 (77.78)	4.757 (11.197)
Idle time	0.018	66.71 (21.97)	0.012 (0.004)
STM32 [38] data processing subsystem			**4.805** (11.2370)
nRF8001 [42]	11	0.0007	0.008
Wireless subsystem			**0.008**
Total			**5.245** (11.677)

Currents drawn by the signal acquisition subsystem are experimentally obtained by running our classification technique on the Gecko EFM32 development board. The currents outside parentheses are currents drawn by our classification technique, while the currents in paretheses are drawn by the full classifier.

VIII. CONCLUSION

In this paper we have addressed the problem of early detection and prediction of MI using smart wearable systems. In order to monitor patients on a long-term basis, we have proposed a real-time event-driven classification technique that reduces the energy consumption while maintaining a high classification accuracy. The experimental evaluation of our proposed classification technique on MI data shows that this scheme reduces the energy consumption by a factor of 2.60, without any real loss in classification performance.

REFERENCES

[1] A. L. Goldberger *et al.*, "PhysioBank, PhysioToolkit, and PhysioNet: Components of a new research resource for complex physiologic signals." *Circulation*, vol. 101, no. 23, pp. E215–E220, 2000.

[2] "Heart Disease Facts & Statistics | cdc.gov." [Online]. Available: https://www.cdc.gov/heartdisease/facts.htm

[3] G. W. Reed, J. E. Rossi, and C. P. Cannon, "Acute myocardial infarction," *Lancet*, vol. 389, no. 10065, pp. 197–210, 2017.

[4] K. Thygesen, J. S. Alpert, A. S. Jaffe, M. L. Simoons, B. R. Chaitman, and H. D. White, "Third universal definition of myocardial infarction," *Circulation*, vol. 126, no. 16, pp. 2020–2035, 2012.

[5] A. J. Camm, T. F. T. F. Luscher, and P. W. Serruys, *The ESC Textbook of Cardiovascular Medicine (European Society of Cardiology)*. London, U.K.: Oxford Univ. Press, 2009.

[6] C. P. Cannon *et al.*, "Relationship of symptom-onset-to-balloon time and door-to-balloon time with mortality in patients undergoing angioplasty for acute myocardial infarction," *JAMA*, vol. 283, no. 22, pp. 2941–2947, Jun. 2000.

[7] "WHO|Prevention of recurrences of myocardial infarction and stroke study," WHO, Geneva, Switzerland, 2013.

[8] S. M. Riazul Islam , Daehan Kwak, M. Humaun Kabir , M. Hossain, and Kyung-Sup Kwak, "The internet of things for health care: A comprehensive survey," *IEEE Access*, vol. 3, pp. 678–708, 2015.

[9] F. Rincon, J. Recas, N. Khaled, and D. Atienza, "Development and evaluation of multilead wavelet-based ECG delineation algorithms for embedded wireless sensor nodes," *IEEE Trans. Inf. Technol. Biomed.*, vol. 15, no. 6, pp. 854–863, Nov. 2011.

[10] H. Mamaghanian, N. Khaled, D. Atienza, and P. Vandergheynst, "Compressed sensing for real-time energy-efficient ECG compression on wireless body sensor nodes," *IEEE Trans. Biomed. Eng.*, vol. 58, no. 9, pp. 2456–2466, Sep. 2011.

[11] Fan Zhang, J. Holleman, and B. P. Otis, "Design of ultra-low power biopotential amplifiers for biosignal acquisition applications," *IEEE Trans. Biomed. Circuits Syst.*, vol. 6, no. 4, pp. 344–355, Aug. 2012.

[12] J. Tang, D. Sun, S. Liu, and J.-L. Gaudiot, "Enabling deep learning on IoT devices," *Computer*, vol. 50, no. 10, pp. 92–96, 2017.

[13] G. Klas, "Edge computing and the role of cellular networks," *Computer*, vol. 50, no. 10, pp. 40–49, 2017.

[14] G. Ananthanarayanan *et al.*, "Real-time video analytics: The killer app for edge computing," *Computer*, vol. 50, no. 10, pp. 58–67, 2017.

[15] S. Patel *et al.*, "Monitoring motor fluctuations in patients with Parkinson's disease using wearable sensors," *IEEE Trans. Inf. Technol. Biomed.*, vol. 13, no. 6, pp. 864–873, Nov. 2009.

[16] D. Apiletti, E. Baralis, G. Bruno, and T. Cerquitelli, "Real-time analysis of physiological data to support medical applications," *IEEE Trans. Inf. Technol. Biomed.*, vol. 13, no. 3, pp. 313–321, May 2009.

[17] D. Sopic, A. Aminifar, A. Aminifar, and D. Atienza Alonso, "Real-time classification technique for early detection and prevention of myocardial infarction on wearable devices," in *Proc. IEEE Biomed. Circuits Syst. Conf.*, 2017, pp. 1–4.

[18] R. Diaz-Uriarte and S. A. de Andres, "Gene selection and classification of microarray data using random forest," *BMC Bioinformatics*, vol. 7, 2006, Art. no. 3.

[19] "INYU - The Inner You - Home." [Online]. Available: http://www.smartcardia.com/inyu/

[20] G. Surrel, A. Aminifar, F. Rincon, S. Murali, and D. A. Atienza, "Online obstructive sleep apnea detection on medical wearable sensors," *IEEE Trans. Biomed. Circuits Syst.*, 2018, available on-line.

[21] D. Sopic, A. Aminifar, and D. Atienza Alonso, "e-Glass: A wearable system for real-time detection of epileptic seizures," in *Proc. IEEE Int. Symp. Circuits Syst.*, 2018, pp. 1–5.

[22] J. P. Dominguez-Morales, A. F. Jimenez-Fernandez , M. J. Dominguez-Morales, and G. Jimenez-Moreno, "Deep neural networks for the recognition and classification of heart murmurs using neuromorphic auditory sensors," *IEEE Trans. Biomed. Circuits Syst.*, vol. 12, no. 1, pp. 24–34, Feb. 2018.

[23] S. Kiranyaz, T. Ince, and M. Gabbouj, "Real-time patient-specific ECG classification by 1-D convolutional neural networks," *IEEE Trans. Biomed. Eng.*, vol. 63, no. 3, pp. 664–675, Mar. 2016.

[24] U. R. Acharya *et al.*, "Automated detection and localization of myocardial infarction using electrocardiogram: A comparative study of different leads," *Knowl.-Based Syst.*, vol. 99, pp. 146–156, 2016.

[25] R. Braojos, I. Beretta, G. Ansaloni, and D. Atienza, "Early classification of pathological heartbeats on wireless body sensor nodes," *Sensors*, vol. 14, no. 12, pp. 22 532–22 551, 2014.

[26] C. Cortes and V. Vapnik, "Support-vector networks," *Mach. Learn.*, vol. 20, no. 3, pp. 273–297, 1995. [Online]. Available: http://link.springer.com/10.1007/BF00994018

[27] M. Ribeiro, "Gaussian probability density functions: Properties and error characterization," , Inst. Syst. Robot., Lisbon, Portugal, Tech. Rep., 2004. [Online]. Available: https://scholar.google.ch/scholar?hl=en&as_sdt=0%2C5&q=M.+Ribeiro%2C+%E2%80%9CGaussian+probability+density+functions%3A+Properties+and+error+characterization%2C%E2%80%9D+2004.&btnG

[28] K.-E. Arzen, "A simple event-based PID controller," 1999. [Online]. Available: https://lup.lub.lu.se/search/publication/20be74b2–9459-40c9-b77a-b8b763d28704

[29] G. Roffo, S. Melzi, U. Castellani, and A. Vinciarelli, "Infinite latent feature selection: A probabilistic latent graph-based ranking approach," in *Proc. Int. Conf. Comput. Vis.*, Jul. 2017, pp. 1407–1415.

[30] T. Hofmann, "Probabilistic latent semantic analysis," in *Proc. 22nd Annu. Int. SIGIR Conf. Res. Develop. Inf. Retrieval*, 1999, pp. 289–296.

[31] A. Liaw and M. Wiener, "Classification and regression by randomforest," *R News*, vol. 2, no. 3, pp. 18–22, 2002.

[32] J. B. Reitsma, A. S. Glas, A. W. S. Rutjes, R. J. P. M. Scholten, P. M. Bossuyt, and A. H. Zwinderman, "Bivariate analysis of sensitivity and specificity produces informative summary measures in diagnostic reviews," *J. Clin. Epidemiol.*, vol. 58, pp. 982–90, 2005. [Online]. Available: http://www.ncbi.nlm.nih.gov/pubmed/16168343

[33] D. G. Altman and J. M. Bland, "Diagnostic tests. 1: Sensitivity and specificity," *BMJ (Clin. Res. ed.)*, vol. 308, no. 6943, 1994, Art. no. 1552.

[34] P. J. Fleming and J. J. Wallace, "How not to lie with statistics: The correct way to summarize benchmark results," *Commun. ACM*, vol. 29, no. 3, pp. 218–221, Mar. 1986.

[35] Y. Sun, K. L. Chan, and S. M. Krishnan, "ECG signal conditioning by morphological filtering," *Comput. Biol. Med.*, vol. 32, no. 6, pp. 465–479, 2002.

[36] J. Pan and W. J. Tompkins, "A real-time QRS detection algorithm," *IEEE Trans. Biomed. Eng.*, vol. 32, no. 3, pp. 230–236, Mar. 1985.

[37] C. Gomez *et al.*, "Use of the Higuchi's fractal dimension for the analysis of MEG recordings from Alzheimer's disease patients," *Med. Eng. Phys.*, vol. 31, no. 3, pp. 306–313, 2009.

[38] U. R. Acharya, H. Fujita, V. K. Sudarshan, S. Bhat, and J. E. Koh, "Application of entropies for automated diagnosis of epilepsy using EEG signals: A review," *Knowl.-Based Syst.*, vol. 88, pp. 85–96, 2015.

[39] "ADS1191 complete low power integrated analog front end for ECG applications|TI.com." [Online]. Available: http://www.ti.com/product/ADS1191

[40] "STM32L1 Series - STMicroelectronics." [Online]. Available: http://www.st.com/web/en/catalog/mmc/FM141/SC1169/SS1295?sc=stm32l1

[41] S. Murali, F. Rincon, and D. Atienza, "A wearable device for physical and emotional health monitoring," in *Proc. IEEE Comput. Cardiol. Conf.*, 2015, pp. 121–124.

[42] "EFM32 Leopard Gecko|Silicon Labs." [Online]. Available: https://www.silabs.com/products/mcu/32-bit/efm32-leopard-gecko

[43] "MPU-6000, Motion Sensor - 6 Axis, TDK InvenSense." [Online]. Available: https://store.invensense.com/ProductDetail/MPU6000-InvenSense-Inc/420595/

[44] "nRF8001 - Nordic Semiconductor." [Online]. Available: https://www.nordicsemi.com/eng/Products/Bluetooth-Smart-Bluetooth-low-energy/nRF8001

Dionisije Sopic (S'17) received the M.Sc. degree in electrical engineering from the Swiss Federal Institute of Technology Lausanne (EPFL), Lausanne, Switzerland, in 2015. He is currently working toward the Ph.D. degree with the Embedded Systems Laboratory, EPFL. His current research includes the acquisition of biomedical signals and optimized processing on low-energy wearable systems for the early detection and prevention of different pathological health conditions, such as myocardial infarction and epilepsy.

Amin Aminifar received the M.Sc. degree in computer engineering from K. N. Toosi University of Technology, Tehran, Iran, in 2017. His current interests include centered around machine learning and data science and their applications in embedded and cyber-physical systems, health-care, and Internet of Things (IoT).

Amir Aminifar (M'16) received the Ph.D. degree from the Swedish National Computer Science Graduate School, Linköping University, Linköping, Sweden, in 2016. During 2014–2015, he visited the Cyber-Physical Systems Laboratory, University of California, Los Angeles (UCLA), Los Angeles, CA, USA, and the Real-Time Systems Laboratory, Scuola Superiore Sant'Anna, Pisa, Italy. He is currently a Research Scientist with the Embedded Systems Laboratory, Swiss Federal Institute of Technology (EPFL), Lausanne, Switzerland. His current research interests include mobile health and smart wearable technologies for early detection/prediction of pathological health conditions.

David Atienza (M'05–SM'13–F'16) received the M.Sc. and Ph.D. degrees in computer science and engineering from Complutense University of Madrid, Madrid, Spain, and Interuniversity Microelectronics Center, Leuven, Belgium, in 2001 and 2005, respectively. He is currently an Associate Professor in electrical and computer engineering, and the Director for the Embedded Systems Laboratory with the Swiss Federal Institute of Technology Lausanne (EPFL), Lausanne, Switzerland. His research interests include system-level design methodologies for high-performance multiprocessor system-on-chip (MPSoC) and low-power Internet-of-Things (IoT) systems, including new 2-D/3-D thermal-aware design for MPSoCs and many-core servers, ultra-low power system architectures for wireless body sensor nodes, HW/SW reconfigurable systems, dynamic memory optimizations, and network-on-chip design. He is a coauthor of more than 250 publications in peer-reviewed international journals and conferences, several book chapters, and eight U.S. patents in these fields. He was the recipient of several best paper awards and is (or has been) an Associate Editor for the IEEE Transactions on Computers, IEEE Design & Test of Computers, IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems, IEEE Transactions on Sustainable Computing, and Elsevier *Integration*. He was the Technical Programme Chair for the IEEE/ACM DATE 2015 and the General Programme Chair for the IEEE/ACM DATE 2017. He was the recipient of the DAC Under-40 Innovators Award in 2018, the IEEE TCCPS Mid-Career Award in 2018, an ERC Consolidator Grant in 2016, the IEEE CEDA Early Career Award in 2013, the ACM SIGDA Outstanding New Faculty Award in 2012, and a Faculty Award from Sun Labs at Oracle in 2011. He is an ACM distinguished member.