Computational Balancing between Wearable Sensor and Smartphone towards Energy-Efficient Remote Healthcare Monitoring

Alma SECERBEGOVIC[1], Asmir GOGIC[1], Nermin SULJANOVIC[1], Matej ZAJC[2], Aljo MUJCIC[1]
[1]*Faculty of Electrical Engineering, University of Tuzla, Bosnia and Herzegovina*
[2]*Faculty of Electrical Engineering, University of Ljubljana, Slovenia*
alma.secerbegovic@untz.ba

***Abstract*—Recent advances in the development of wearable sensors and smartphones open up opportunities for executing computing operations on the devices instead of using them for streaming raw data. By minimizing power consumption due to the wireless transmission, limited energy resources of wearable devices can be utilized not only for sensing, but also for processing physiological signals. Computational tasks between a wearable sensor and a smartphone can be distributed efficiently in order to provide balance between power consumption of both processing and transmission of the data. In this paper, we have analyzed the computational balancing between a wearable sensor and a smartphone. Presented models show different trade-offs between classification accuracy, processing time and power consumption due to different number and types of extracted features and classification models. Our results are based on a physiological dataset, where electrocardiogram and electro dermal activity signals were collected from 24 individuals in short-term stress and mental workload detection scenario. Our findings show that placing a feature extraction on a wearable sensor is efficient when processing cost of the extracted features is small. On the other hand, moving classification task to the smartphone can improve accuracy of recognition without compromising the overall power consumption.**

***Index Terms*—wearable sensors, mobile computing, body sensor networks, biomedical signal processing, performance evaluation.**

I. INTRODUCTION

The rapid growth of wearable sensors is driving the shift from the traditional hospital-based healthcare service delivery to home-based patient-centric remote monitoring. Diverse range of physiological signals can be captured, processed and transmitted by using low-cost, low-power and small size wearable sensors (WS). The popularity of these devices is expected to grow in the foreseeable future, allowing them to become an important diagnostic and preventive tool in the healthcare industry [1-2]. Strategic use of the WS can help healthcare professionals to provide evidence-based treatments, deliver faster feedback and increase the quality of patient care. Small, body-worn devices can come shaped as an accessory or placed within clothing, which improves patient's compliance for their everyday use.

Wearable sensors are mainly used as sensing devices and they can be easily connected to a smartphone for display, storage and analysis of the physiological signals. The integration of WS and smartphone offers possibilities to provide patient with continuous, unobtrusive monitoring with real-time feedback on their current health state. Along with the increased need for real-time computing, energy efficiency of WS and smartphones has become an important requisite for user-friendly long-term monitoring [3-4]. Small WS come at a cost of limited computational power and shorter lifetime. Smartphones, on the other hand, have larger batteries, but support other functionalities that can lead to faster battery depletion as well. Limitations in hardware resources are preventing their broader deployment in healthcare.

Potential approach for minimizing power dissipation of battery-operated devices in remote healthcare monitoring is by transferring data from a WS to a smartphone and then shifting the intensive computer calculations to a higher-performance platform - cloud. This concept is also known as computational offloading. While cloud-based computational offloading can save energy by migrating complex processing tasks from a mobile device to the cloud, it heavily depends on the reliability of available Wi-Fi or mobile data connections. Furthermore, migrating processing tasks to the cloud is justified only in case when the complexity of the processing tasks is high and the amount of the acquired data is relatively small [5]. Distribution on where the processing tasks should be implemented is usually debated between the cloud and smartphone. Although the establishment of direct connection between a WS and cloud is feasible, this solution is not practical in real-time remote healthcare monitoring due to the high-power consumption of the wireless transceiver. Whenever cloud-based computational offloading is not achievable, integration of the smartphone and wearable sensors can provide a computing platform for real-time processing of biomedical signals. Main advantage of this approach is that balancing of computing tasks between the smartphone and WS can lead to the minimization of power consumption of battery-operated devices, while at the same time also providing reliable monitoring and feedback. This integration requires only the establishment of short-range wireless connection, such as Bluetooth Low Energy (BLE) technology, which enable for moderate speed low-power data transmissions, compared to resource-hungry mobile or Wi-Fi connections that are required for the transmission to the your cloud.

In this paper, we have conducted a study on the trade-off

This work was supported in part by the Federal Ministry of Education and Science of Bosnia and Herzegovina.

Digital Object Identifier 10.4316/AECE.2018.04001

between the accuracy, processing time and energy consumption of three computational balancing models between a wearable sensor and a smartphone. Since computational process consists of pre-processing, feature extraction and classification operations, we have divided the aforementioned processing tasks between a WS and a smartphone. The trade-off between local processing on the wearable sensor and remote offloading to the smartphone has been recently studied. In [6] authors have reported simulation-only results, where computational offloading models are rated based on the amount of data that needs to be processed. Separate processing tasks are analyzed in [7], where power consumption of the wearable device is estimated based on simulations, without measuring the smartphone's power consumption. Taking into consideration remote healthcare monitoring scenarios, activity recognition applications served as case studies for distribution of processing tasks between a WS and a smartphone. The work in [8] compares sensor-only processing tasks with combined WS and smartphone computing. The authors focused on the importance of accuracy on different platforms and reported estimated values of energy savings. By moving classification tasks to the smartphone, higher accuracy levels are accomplished but with higher power consumption due to the wireless transmission. In [9], authors have proposed an architecture called E-gesture, where processing on the smartphone is initiated when a wearable device detects a gesture with accelerometer and gyroscope sensors. Experimental results show energy savings of a WS and a smartphone compared to raw streaming of the data from a WS to the smartphone. Sensor-based computations were reported as energy-efficient in [10], but authors concluded that these results depend on the user's activity.

In this paper, we have focused on the distribution of execution of feature extraction and classification tasks between a wearable sensor and a smartphone. More specifically, three computational balancing models are introduced and evaluated in terms of accuracy, energy-efficiency and processing time. We have carefully analyzed the process of feature selection in remote monitoring since it has strong relation to classification accuracy and energy efficiency of the wearable sensor and smartphone. Computational complexity of the extracted features and classification models is of great importance, since light-weight processing leads to lower power consumption and longer battery lifetime of wearable and mobile devices. Furthermore, computational balancing models are extended with cost-based feature selection, which minimizes energy-consumption, and without consideration of the processing cost, which yields highest classification accuracy. For classification models we have selected a simple decision tree for implementation on the WS, and Random Forest and k-Nearest Neighbor for implementation on the smartphone device. The exploration of the presented models is performed by analyzing a physiological dataset collected in a mental workload and stress detection scenario with electrocardiogram and electro dermal activity biomedical signals.

The rest of the paper is structured as follows: in Section II we provide a description of computational balancing models between a WS and a smartphone along with their energy and processing time estimations. Section III describes cost-based feature selection, while the experimental study, with detailed description of each processing step and algorithm selection are explained in Section IV. Performance evaluation on the described dataset is presented in Section V. Conclusions are drawn in Section VI.

II. Distribution of Computational Tasks Between Wearable Sensor and Smartphone

A. Signal processing pipeline

In general, remote healthcare monitoring applications follow signal processing pipeline, which consists of sensing, pre-preprocessing, feature extraction and classification stages. Acquisition of biomedical signals is performed during the sensing stage. Afterwards, pre-processing of the obtained signals is executed in order to remove or suppress artefacts that were present during the recording of the data. Real-time monitoring introduces signal corruption due to the mobility of the subject, muscle contractions, faulty electrode placement, respiration, etc. Selection of an adequate noise removal technique depends on the desired outcome and available resources in terms of memory and processing power. After filtering, signals are divided into time blocks called segments, which have a predefined size and overlapped intervals. Feature extraction stage refers to the process of extracting relevant information from pre-processed signals. It minimizes the dimensionality of the data to be analyzed, in order to preserve only informative and non-redundant characteristics of the signals. Upon each segment, feature extraction is applied resulting in a multidimensional feature vector. At the end, obtained features are used as an input to the classification model, which provides the user with feedback appropriate to the monitoring scenario. Above listed operations are all executed in real-time. In this paper, different distributions of pre-processing, feature extraction and classification tasks are presented between a WS and a smartphone. Sensing operations are reserved only for the wearable sensor.

B. Computational balancing model 1

Computational balancing model 1 is defined in a way that WS executes all processing tasks. WS and smartphone have an established wireless connection, which is utilized when on-node classification process detects a specific event that needs to be reported to the patient. Therefore, consumption due to transmission is minimized. Fig. 1 shows a block diagram of signal processing pipeline divided between wearable sensor and smartphone for this model. However, implementation of the signal processing algorithms locally on a wearable sensor can have a significant impact on the recognition accuracy in remote monitoring scenario. Resource-constraint WS disallows complex and computationally intensive signal processing algorithms.

Restrictions in terms of noise removal, feature extraction operations, number of features and type of classifier model require careful selection of signal processing routines that can meet memory and real-time execution requirements. Total processing time is the processing time of the WS

$$T = T_{ACQ} + T_C + T_{TXn} \quad (1)$$

where *ACQ* stands for acquisition, *C* for computing and *TX*

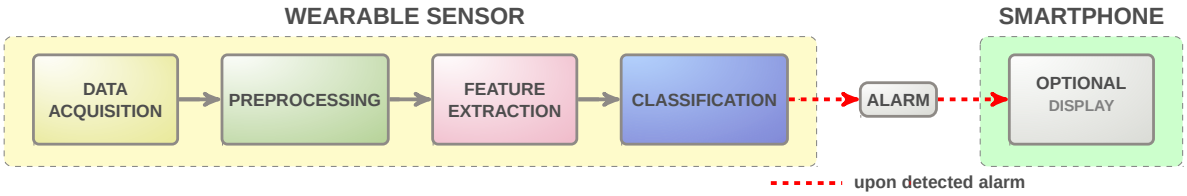

Figure 1. Sensor-based processing with event-driven transmission

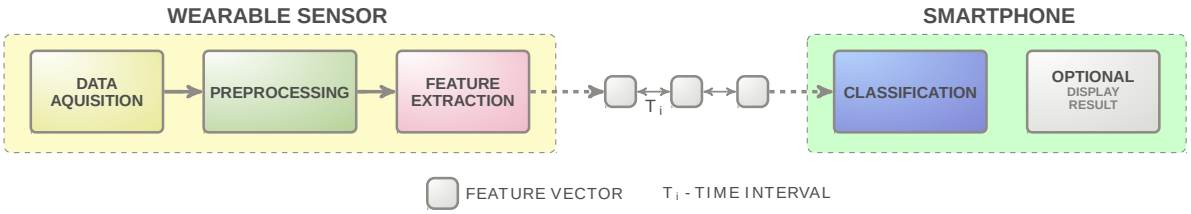

Figure 2. Feature-only transmission to the smartphone

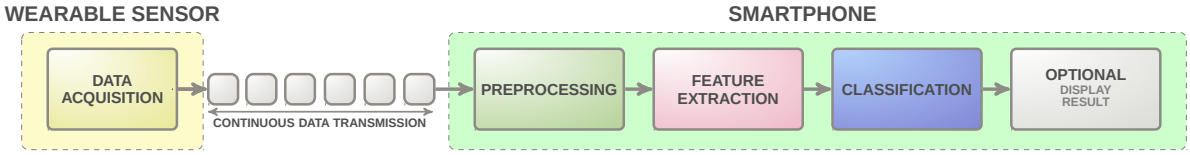

Figure 3. Raw data streaming to the smartphone

for transmission, while subscript n indicates sensor node-based operation. T_{TXn} represents the amount of time needed for the data to be formatted for wireless transmission.

Total energy consumption of the WS includes acquisition, computational and transmission energy,

$$E_{WS} = E_{ACQ} + E_C + E_{TXn} \quad (2)$$

where sensing energy depends on the sampling rate and analog-to-digital resolution of the WS. Computational energy can be defined as

$$E_C = P_{PRn} \cdot T_{PRn} + \sum_{i=1}^{L} P_{FEi} \cdot T_{FEi} + P_{CLn} \cdot T_{CLn} \quad (3)$$

where P_{PR}, P_{FE} and P_{CL} denote power consumption of preprocessing, feature extraction and classification tasks, respectively, while L is the number of features that is extracted on the node.

C. Computational balancing model 2

This computational balancing model requires transmission of the extracted features from the WS to the smartphone (Fig. 2). The advantages of sensor-based processing are exploited, resulting in significant reduction of the energy employed for wireless transmission. The number of features as well as the type of extracted features depends on the processing and memory capacity of a resource-constraint WS. Implementation of the classification model is moved to the smartphone which can lead to employment of complex classification algorithms, resulting in a higher accuracy for the monitoring scenario.

Total processing time represents the duration of local computation on the WS and the smartphone

$$T = T_{ACQ} + T_{PRn} + T_{FEn} + T_{TXn} + T_{CLs} \quad (4)$$

where subscript s defines a smartphone-based task. The computational tasks of the WS include all operations except classification, with total energy consumption of WS and smartphone being derived in (5) and (6), respectively.

$$E_{WS} = E_{ACQ} + P_{PRn} \cdot T_{PRn} + \sum_{i=1}^{R} P_{FEi} \cdot T_{FEi} + E_{TXn} \quad (5)$$

$$E_S = E_{TXs} + P_{CLs} \cdot T_{CLs} \quad (6)$$

with R indicating the number of features extracted on the WS and transmitted to the smartphone. P_{CLs} represents the power consumption of the smartphone due to the classification task of extracted feature vector.

D. Computational balancing model 3

Complete computational offloading from the WS to the smartphone is presented in model 3. Smartphone is the main processing platform for all signal processing tasks, while WS is only provided for sensing. Continuous streaming of raw data is executed by WS, where constant and reliable connection between the two devices is a necessity, as shown on block diagram in Fig. 3.

Processing time for all operations is defined as follows.

$$T = T_{ACQ} + T_{PRs} + T_{FEs} + T_{CLs} + T_{TXn} \quad (7)$$

Energy of the WS is used for sensing and transmission of raw data

$$E_{WS} = E_{ACQ} + E_{TXn} \quad (8)$$

and energy of the smartphone is used for other processing tasks

$$E_S = P_{PRs} \cdot T_{PRs} + \sum_{i=1}^{Q} P_{FEi} \cdot T_{FEi} + P_{CLs} \cdot T_{CLs} + E_{TXs} \quad (9)$$

where Q is the number of extracted features on a smartphone.

Accuracy of the computational model is determined by the selected algorithms for pre-processing, feature extraction and classification. Since pre-processing is moved to the smartphone, improved filtering methods can be executed

which will minimize the influence of artefacts in signals. Feature extraction tasks can include derivation of complex features such as frequency-domain features that can be integrated into the classification model. Compared to the previous computational model, number of extracted features Q can be different from R features extracted on a WS. This might lead to increased classification accuracy of the entire monitoring system. However, smartphone usage is not solely intended for data processing, so it is necessary to take into account other functionalities of the smartphone and their power consumption.

III. Cost-based Feature Selection

Presented computational models are designed in order to evaluate different trade-offs between power consumption, processing time and recognition accuracy. In order to design an energy-efficient and accurate remote healthcare monitoring system, it is important to incorporate offline procedures that include feature selection and training of the classification model. These procedures are computationally intensive operations that are executed off-line before real-time deployment on the WS and the smartphone. Feature selection refers to the process of selecting a subset of features from a total set of features, in order to remove redundant and irrelevant attributes that usually do not contribute to the recognition accuracy. The process of training of the classification model consists of selecting a machine learning algorithm and providing training data for the selected algorithm. Each of these procedures has direct influence on the power consumption of used devices, where feature extraction and classification tasks will be executed. Therefore, it is necessary to carefully consider and analyze these methods.

Feature selection is basically an optimization problem, where a subset of relevant features needs to be selected from a larger set of features, according to the feature selection method. Reduction in computation complexity of the classification model, expedition of the training process and prevention of data overfitting are among positive outcomes in feature selection procedure [11]. While improvement in terms of accuracy is always desirable, another point of view must be considered. Minimizing the number of features reduces the computational cost of the feature extraction and classification stage and therefore can extend battery lifetime of the wearable devices. Therefore, it is advised to do a modification of feature selection process in a way that minimizes the computational cost of feature extraction without compromising the classification error. This variation of feature selection process is defined as cost-based feature selection [12].

Feature selection methods can be divided into three groups: filter, wrapper and embedded methods [11]. Wrapper and embedded methods depend on the selection of a classification model, whereas filter methods provide a list of ranked features according to the specific criterion or merit M, independent of any classifier type. Since the selection of the classification model can also have a significant impact on power consumption of the wearable device, in this paper we focus only on filter-based methods for feature selection. Cost-based feature selection with filter method is defined in a way that features f_i in a feature set F are ranked based on merit M_i and on the user-defined cost C_i. Cost of the individual feature can be referred to different aspects such as acquisition cost, financial cost, energy cost or storage cost. All filter methods use an evaluation function which is calculated for each feature and corresponds to the value of merit Mi. Modification of the merit is provided by adding cost [12]

$$M_{iC} = M_i - \lambda C_i \tag{10}$$

where λ represents the weight factor which stands for the influence of cost in the feature selection process. Feature-specific cost can be defined depending on the goal of the monitoring system. It is of great importance to provide long-term monitoring for the user, which is closely related to the power consumption of the processing devices. In this paper, we have chosen the cost of each feature to be expressed as computational cost. The number and type of arithmetic operations are considered as a representative of the power consumption required for the extraction of each feature. In order to differentiate between different types of arithmetic operations, we have defined the weight factors for six operations, including addition, multiplication, division, comparison, square root and logarithm tasks. For example, a complex operation such as square root is more energy consuming compared to the addition task on all battery-operated devices. In general, the cost C_i for the extraction of feature f_i can be defined as

$$C_i = \sum_{i=1}^{6} \theta_i n_i \tag{11}$$

where n_i is the number of specified operations required for the extraction of the i^{th} feature, and θ_i is the weight of the operation. The value of the processing cost will be higher if the signal processing algorithm for the selected feature requires large number of complex arithmetic operations.

We will present a simple example to motivate our idea of finding feature set. Two biomedical signals were collected during sensing stage. Assume that ten features construct our exhaustive set of features from two signals, represented by $F = \{ f_1, f_2, ..., f_{10} \}$. For each feature we can define merit M_i, which presents the relevance of specific feature in regard to the classes that need to be recognized. In order to select only important but low complexity features, for each feature we have calculated the processing cost C_i by using the number of arithmetic operations required for each feature. E.g. assume that heart rate is f_1 feature, while distance between two R-peaks in electrocardiogram signal, also known as R-R interval, is feature f_2. These two features are commonly used in stress detection scenario [13-14]. Heart rate is calculated by dividing 60 seconds with distance between two R-peaks expressed in seconds. Mentioned features have linear relationship and therefore have the same merit, but when expressing computational cost, f_1 has higher cost C_1 due to the additional operation of division. Therefore, feature f_2 is chosen over f_1. Based on cost-based merit values, features with lower processing cost and higher merits are selected, which directly affects energy-efficiency of feature extraction processing task. Lower number of features implies faster performance and lower power consumption of classification algorithm as well.

IV. Experimental Study

A. Dataset

For the study, 24 healthy subjects (5 female) were recruited. Two biomedical signals were collected during stress initiation protocol, where single-lead electrocardiogram (ECG) signals were recorded from the subject's chest, while electro dermal activity (EDA) data was sampled from the palm of their non-dominant hand. Sampling frequency for both signals was 100 Hz with a 10-bit ADC resolution. The following protocol was applied for all participants. For the first 3 minutes, subjects were asked to comfortably sit and relax in a chair, with their eyes closed while listening to peaceful music. During the *mental workload* condition, the subjects performed simple arithmetic tasks on a stationary computer. They were encouraged to accurately solve as many as possible mathematical expressions during the allotted time slot. This part of the recording is labelled as a mental workload, because it resembles work-related office tasks. As a final step, individuals were assigned to prepare an oral presentation about a miscellaneous topic in front of a fake camera. Subjects had exactly one minute to prepare the presentation, and one minute to continuously speak in front of the camera. This period is labelled as *stress* condition.

B. Device specifications

1) Wearable sensor

For the wearable sensor device, we have selected the sensor board utilizing nRF52840 BLE SoC, with support for a floating point (FP) and digital signal processing (DSP) operations by native hardware FPU and DSP units [13]. This wearable sensor has a low current consumption of 5.5 mA during the BLE communication (Tx/Rx) and 0.5 μA during the sleep mode. In order to perform a real-time current consumption measurement for a WS and capture transients during different computational models, power profiler shield from Nordic Semiconductor is utilized. The power profiler shield allows for a dynamic current consumption measurement with sampling rates up to 50 kHz and currents from 0.2 μA to 70 mA. Current consumption measurement for all computational balancing models is performed when computational tasks were compiled using nRF5x SDK v13 with softdevice S132 v5.0 BLE stack, ARM gcc compiler v4.7, and O3 level of optimization.

2) Smartphone

Smartphone device used for testing of computational balancing models is LG G3, running an Android 5.0 operating system. All signal processing tasks were implemented in C code with Android NDK (Native Development Kit) technology. Power consumption of the Android smartphone is estimated by using Battery historian [14]. It is a tool that inspects battery-related information on an Android smartphone and provides the user with estimated values of battery consumption during the evaluation period.

Connection between Android smartphone and nRF52840 SoC is established with BLE point-to-point communication protocol. It employs a simple master-slave scheme where information is commonly served by a slave/peripheral device and read by a client on a master/central device. Data exchange between the central and peripheral devices is performed during time slots called connection intervals, which can go from 7.5 ms up to 4 seconds. Connection interval is commonly negotiated during the BLE connection setup, but it can be changed at any time during the connection's lifetime. Within each connection interval, the BLE device can transmit maximum of 6 packets of 47 bytes out of which 20 bytes can be used for user data, as stated in the BLE Standard v4.0. On the other hand, Data Length Extension feature in BLE standard v4.2 enables peripheral or central device with larger packet payloads up to 255 bytes per packet. However, maximum achievable data rate is strictly limited by the central device and the BLE stack implementation within the operating system.

3) Implementation

Each computational balancing model was tested on the dataset that was acquired with experimental study. Since ECG and EDA signals were collected during workload and stress conditions, it was necessary to remove the artifacts generated from electronic noise, movement and poor electrode contact. 5-th order low-pass Butterworth filter is used for the removal of high-frequency artifacts in EDA data whereas high-pass IIR filter is applied for baseline wandering removal in ECG. Afterwards, signals were segmented into time blocks of 10 seconds duration with 50% overlap. 10 seconds window was chosen in order to capture the events of interest (stress and mental workload) [17].

From filtered and segmented ECG and EDA signals, we have extracted 28 time-domain features. The features were chosen based on their relevance in the previous research work regarding stress detection [13-14]. In this part, only time-domain features are considered, since their computational complexity is low and calculation time is relatively fast. Certain features required detection of prominent peaks in EDA and ECG signals, such as R-peak in ECG and skin conductivity response (SCR) in EDA. In order to differentiate between smartphone-based and WS-based feature extraction, we have used two different R-peak detection algorithms for implementation; on the WS we have adapted the algorithm proposed by Chen [18], while more efficient and computationally intensive real-time Pan-Tompkins algorithm [19] is implemented on the smartphone. For SCR detection in the EDA signal, simple peak detection algorithm is implemented with 0.02 μS threshold value for detection of SCR. For each feature, a number of arithmetic operations is derived in order to compute the total computational cost with (11). As the feature selection method, we have selected cost-based feature selection based on Minimal-Redundancy-Maximal-Relevance (mRMR) [20]. This method returns a ranked list of features, where merit is calculated based on relevance/redundancy of the feature set and on the computational cost.

Table I contains the list of relevant features obtained after cost-based feature selection. It is evident that features extracted from raw ECG and EDA signals are among cost-based features. Features obtained from R-R intervals extracted from ECG signal or SCR from EDA are more complex in terms of processing, due to the peak detection algorithms executed before the extraction. However, number of these features have appeared as significant when computational cost is not considered. As a final processing

task, different classification models were implemented on WS and smartphone, due to hardware and software restrictions of the devices. For the wearable sensor, simple decision tree model J48 is selected where monitored physiological features are represented by tree nodes and classes are represented by the leaf nodes [21]. Classification model implemented on the smartphone device was k-Nearest Neighbor (k-NN), where used value of k was 3 and Random Forrest (RF) model with 10 combined decision trees.

TABLE I. LIST OF RELEVANT FEATURES FOR IMPLEMENTATION ON WEARABLE SENSOR AND SMARTPHONE

Signal	Feature description	Computation complexity level
ECG	Absolute mean value of successive sample differences	low
ECG	Root mean square value	low
EDA	minimum value	low
EDA	mean value	low
EDA→ SCR	sum of amplitudes	moderate
EDA→ SCR	sum of estimated area under the response	high
ECG→ RR	minimum value	moderate
ECG → RR	root mean square value of successive differences	high

Specifications for each computational model are listed in Table II. Computational balancing model 1 is based on sensor-only local processing, where features are selected based on the cost of processing. This implies that only low-complexity and relevant features will be extracted on the WS in real-time. For our case study, 4 low-cost features are extracted with simple decision tree classification model J48 as the classifier. Second computational balancing model moves the classification task to the smartphone, while transmitting only the selected features after specified time interval.

TABLE II. SETTINGS FOR EACH COMPUTATIONAL BALANCING MODEL IN STRESS DETECTION SCENARIO

Model	Computational balancing model description	Number of features	Classifier
1	*On-node local processing*	4	J4.8
2a	*On-node processing + smartphone classification*	4	kNN
2b		6	kNN
3	*Smartphone local processing*	6	RF

We have defined two modifications of this model; model 2a uses features selected without considering processing cost (parameter λ from (10) equals zero) and model 2b assumes that WS will extract only low-complexity features (λ =10). Classification algorithm for models 2a and 2b implemented on the smartphone device was k-NN, where used value of k was 3. Third model refers to the smartphone-only processing approach, where the WS streams raw data and the smartphone executes all computational tasks. Since smartphones have more resources compared to the wearable sensor, this model demonstrates if it is reliable to place all operations on the smartphone in terms of battery power and accuracy. For this model, we have replaced k-NN classifier with RF classification model due to its simplicity and efficiency in solving classification problems.

V. PERFORMANCE EVALUATION

Processing time and power consumption of WS and smartphone devices are measured for the obtained dataset. The following subsections present results obtained in terms of accuracy, power consumption and processing time.

A. Accuracy

Accuracy is calculated as a percentage of accurately recognized samples divided by a total number of samples. Accuracy, precision and recall were calculated by using Eq. (12)-(14)

$$Accuracy = \frac{TP + TN}{TP + FP + TN + FN} \tag{12}$$

$$Precision = \frac{TP}{TP + FP} \tag{13}$$

$$Recall = \frac{TP}{TP + FN} \tag{14}$$

where TP and FP denote True and False Positives, while TN and FN denote True and False Negatives averaged over all three (relax, mental workload and stress) states. These results were obtained based on an offline analysis of collected data by using Weka software [22].

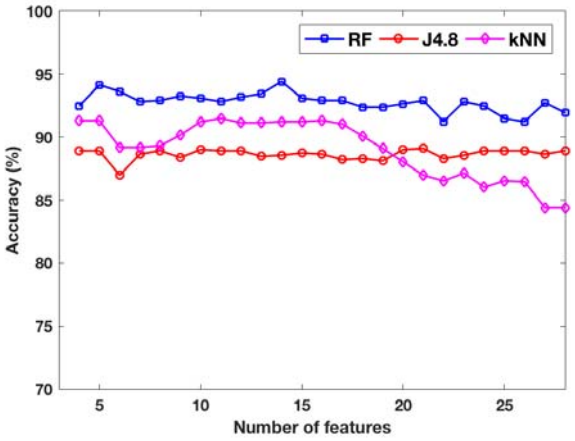

Figure 4. Accuracy for three classifier models in terms of size of feature vector using greedy search method

Best achieved accuracy score for 10-fold cross-validation for J48 classifier was 88.75%, 90.24 % for Random Forrest and 92.42% for k-NN classifier model. These values are referred to as the highest possible accuracy values that can be achieved on the obtained dataset, which are compared to the accuracy values obtained with same classifier for

different computational balancing models.

Fig. 4 demonstrates the overall testing accuracy for three different types of classifier models. Table III lists the results obtained for different computational models. On-node local processing model is tested for J48 decision tree classifier model, whereas remaining models implemented k-NN or RF predictor on the smartphone. Accuracy difference indicates the accuracy error of the classification model implemented on a battery-operated device, compared to the highest achievable accuracy with same classification model. For our case study, results demonstrate that computationally complex features provide lower accuracy levels (Model 2b) compared to low-cost features (Model 2a). However, low complexity classifier J48 implemented on the WS provides lower accuracy with the same low-cost features as in Model 2a. Implementing classification on the WS results in a 5.67% drop in recognition accuracy which is expected due to limited computational capacity of a WS.

TABLE III. AVERAGE CLASSIFICATION MEASURES FOR DIFFERENT COMPUTATIONAL OFFLOADING MODELS

Computational offloading model	Accuracy (%)	Precision (%)	Recall (%)	Accuracy compared to best possible classification (%)
1	86.74	86.3	85.7	-5.67
2a	91.15	91.2	91.15	-1.27
2b	88.02	88.1	87.8	-4.4
3	91.83	91.8	91.8	-0.59

B. *Power consumption*

Current consumption of the WS is shown in Fig. 5, which is captured with power profiling shield for presented computational models.

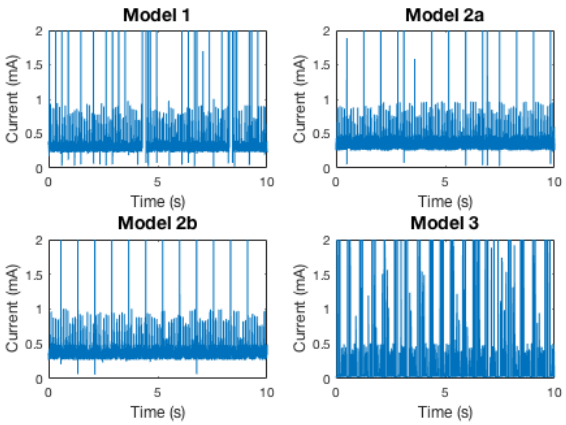

Figure 5. Current consumption of WS for different computational models

Difference between model 1 and model 2a is that the model 1 also implements a simple classifier algorithm, which results in an insignificant increase of current consumption. Model 2b requires a complex feature extraction from the WS, which results in the highest current consumption. Raw streaming in model 3 requires constant wireless connection with the smartphone, which demonstrates constant power consumption due to the wireless transmission.

TABLE IV. AVERAGE CURRENT CONSUMPTION OF SENSOR NODE AND ESTIMATED BATTERY LEVEL CONSUMPTION

Computational offloading model	Average current consumption of WS (μA) for time block	Estimated battery percentage drop during 1 hour of processing
1	363	-
2a	358	0.23
2b	*433*	*0.24*
3	300	3.33

Results of average current consumption of the WS during processing of a 10-seconds time block of data are presented in Table IV. Estimated battery percentage drop of an Android smartphone is also reported. The measured current consumption of the WS show that the processing of all the required tasks in the WS (model 1) is also highly energy efficient. Model 3 places significant burden on the smartphone, which results in the highest power consumption compared to the other presented models.

C. *Processing time*

Table V shows average processing times for a single time-block of 10 seconds window reported for different computational models. Since processing operations in model 1 require no smartphone involvement, the total time depends only on sensor node's execution speed. All other computational models require BLE transmission of the data from the WS to the smartphone. Computation of complex features in model 2b requires the longest computation time for the WS and in model 3 for the smartphone.

TABLE V. PROCESSING TIME FOR EACH SCENARIO FOR 10 SECONDS DURATION OF OBTAINED SIGNALS

Computational offloading model	Processing time (ms)		Total time (ms)
	WS	Smartphone	
1	4.79	0	4.79
2a	4.75	3.28	8.03
2b	24.00	8.41	32.41
3	0.53	355.58	356.11

In Fig. 6 the comparison between different computational models is demonstrated, with relative values of processing time, current consumption of the WS, smartphone battery level and accuracy. When comparing WS-only processing model 1 with a combination of the WS and smartphone processing models (models 2a and 2b), it is evident that the implementation of simple decision tree classification algorithm shows an insignificant increase in the WS's current consumption. This model comes at a cost of lower accuracy but with the fastest processing time.

Placing feature extraction processing on the WS, combined with the classification model implementation on the smartphone (models 2a and 2b), depends only on the complexity of extracted features. While complex features

provide higher accuracy (model 3 vs. model 2a), their implementation is slower and more power-consuming for the WS and the smartphone.

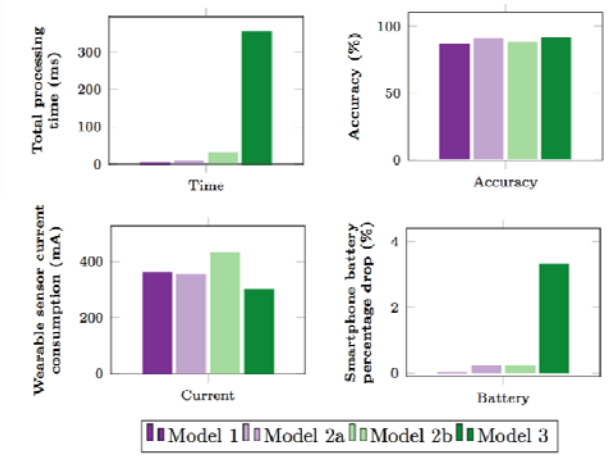

Figure 6. Comparison of computational models in terms of accuracy, WS's current consumption, total processing time and smartphone's battery power

VI. Conclusion

In this work, we have investigated three computational balancing models between a wearable sensor and a smartphone. Distribution of processing tasks between battery-operated devices is important when cloud-based processing is not feasible. Computation operations have been divided into sequential stages, from sensing, pre-processing to feature extraction and classification. Tested healthcare monitoring scenario was mental workload and stress detection, where ECG and EDA biomedical signals are analyzed. By combining different number of features and types of classifiers for real-time implementation on battery-operated devices, our results show different trade-offs between accuracy, processing time and power consumption. In the stress detection case study, experimental results indicate that sensor-based local processing has the lowest power consumption, but it is evident that it is highly dependable on the analyzed dataset and selected algorithms. Moving the classification task to a smartphone and executing feature extraction on a wearable sensor provides possibilities for implementation of computationally complex classification models on a smartphone, which can provide higher accuracy levels. This modification uses wearable sensor and smartphone as processing platforms when no cloud-based platform is available, but without compromising overall power consumption of wearable and mobile devices.

References

[1] M. Patel and J, Wang, "Applications, challenges, and prospective in emerging body area networking technologies", IEEE Wireless communications, vol. 17, no. 1, pp. 80-88, 2010. doi:10.1109/MWC.2010.5416354

[2] U. Varshney, "Pervasive Healthcare and Wireless Health Monitoring", Mobile Network Applications, vol. 12, pp. 113–127, 2007. doi:10.1007/s11036-007-0017-1

[3] T. Rault, A. Bouabdallah, Y. Challal and F. Marin, "A survey of energy-efficient context recognition systems using wearable sensors for healthcare applications", Pervasive and Mobile Computing, vol. 37, pp. 23-44, 2017. doi: 10.1016/j.pmcj.2016.08.003

[4] H. Ghasemzadeh, N. Amini, R. Saeedi and M. Sarrafzadeh, "Power-aware computing in wearable sensor networks: An optimal feature selection", IEEE Transactions on Mobile Computing, vol. 14, no. 4, pp. 800-812, 2015. doi: 10.1109/TMC.2014.2331969

[5] K. Kumar, and Y.H. Lu, "Cloud computing for mobile users: Can offloading computation save energy?", Computer, vol. 43, no. 4, pp. 51-56, 2010. doi: 10.1109/MC.2010.89

[6] C. Ragona, F. Granelli, C. Fiandrino, D. Kliazovich and P. Bouvry. "Energy-efficient computation offloading for wearable devices and smartphones in mobile cloud computing", In Global Communications Conference (GLOBECOM), 2015 IEEE, pp. 1-6. IEEE, 2015. doi: 10.1109/GLOCOM.2015.7417039

[7] H. Kalantarian, C. Sideris, B. Mortazavi, N. Alshurafa and M. Sarrafzadeh, "Dynamic computation offloading for low-power wearable health monitoring systems", IEEE Transactions on Biomedical Engineering, vol. 64, no. 3, pp. 621-628, 2017. doi: 10.1109/TBME.2016.2570210

[8] R. Braojos, I. Beretta, J. Constantin, A. Burg and D. Atienza, "A wireless body sensor network for activity monitoring with low transmission overhead," in 12th IEEE Int Conf EUC, pp. 265–272, 2014. doi: 10.1109/EUC.2014.46

[9] T. Park, J. Lee, I. Hwang, C. Yoo, L. Nachman and J. Song," E-Gesture: a collaborative architecture for energy-efficient gesture recognition with hand-worn sensor and mobile devices", 9th ACM Conference on Embedded Networked Sensor Systems, pp. 260–273, 2011. doi: 10.1145/2070942.2070969

[10] L. Wang, T. Gu, X. Tao and J. Lu, "A hierarchical approach to real-time activity recognition in body sensor networks", Pervasive and Mobile Computing, vol. 8, no. 1, pp. 115-130, 2012. doi: 10.1016/j.pmcj.2010.12.001

[11] I. Guyon, and A. Elisseeff, "An introduction to variable and feature selection", Journal of machine learning research, no. 3, pp. 1157-1182, 2003.

[12] V. Bolon-Canedo, I. Porto-Diaz, N. Sanches-Marono and A. Alonso-Betanzos, "A framework for cost-based feature selection", Pattern recognition, vol. 47, no. 7, pp. 2481-2489, 2014. doi: 10.1016/j.patcog.2014.01.008

[13] nRF52840 Reference manual, Available: http://infocenter.nordicsemi.com/pdf/nRF52840_OPS_v0.5.pdf, accessed on 11.03.2018.

[14] Google. Battery Historian. https://github.com/ google/battery-historian. accessed 01.03.2018

[15] H. Peng, F. Long and C. Ding, "Feature selection based on mutual information criteria of max-dependency, max-relevance and min-redundancy", IEEE Transactions on Pattern Analysis and Machine Intelligence, vol. 27, no. 8, pp.1226-1238, 2005. doi:10.1109/TPAMI.2005.159

[16] J. A. Healey and R. W. Picard., "Detecting stress during real-world driving tasks using physiological sensors", IEEE Transactions on intelligent transportation systems, vol. 6, no. 2, pp. 156-166, 2005. doi: 10.1109/TITS.2005.848368

[17] O. Grigore and L-V. Bornoiu, "Kohonen Neural Network Stress Detection Using Only Electrodermal Activity Features", Advances in Electrical and Computer Engineering, vol. 14, no. 3, pp. 71-78, 2014. doi:10.4316/AECE.2014.03009

[18] I.H. Witten, E. Frank, E., M. A. Hall and C. J. Pal, "Data Mining: Practical machine learning tools and techniques", Morgan Kaufmann, 2016

[19] R. A. Lockhart, "Interrelations between amplitude, latency, rise time, and the Edelberg recovery measure of the galvanic skin response", Psychophysiology, vol. 9, no. 4, pp.437-442, 1972. doi:10.1111/j.1469-8986.1972.tb01791.x

[20] H.C, Chen and S. W. Chen, "A moving average based filtering system with its application to real-time QRS detection", Computers in Cardiology, pp. 585-588, 2003. doi: 10.1109/CIC.2003.1291223

[21] J. Pan and W.J. Tompkins, "A real-time QRS detection algorithm", IEEE Transactions on Biomedical Engineering, vol. 23, pp. 230-236, 1985. doi:10.1109/TBME.1985.325532

[22] M. Hall, E. Frank, G. Holmes, B. Pfahringer, P. Reutemann and I. H. Witten, "The weka data mining software: an update," ACM SIGKDD explorations newsletter, vol. 11, no. 1, pp. 10–18, 2009. doi: 10.1145/1656274.1656278