ECG Signal Analysis and Arrhythmia Detection on IoT wearable medical devices

Dimitra Azariadi, Vasileios Tsoutsouras, Sotirios Xydis, Dimitrios Soudris
Institute of Communication and Computer Systems, Greece
Contact info: {dimitra, billtsou, sxydis, dsoudris}@microlab.ntua.gr

***Abstract*—Healthcare is one of the most rapidly expanding application areas of the Internet of Things (IoT) technology. IoT devices can be used to enable remote health monitoring of patients with chronic diseases such as cardiovascular diseases (CVD). In this paper we develop an algorithm for ECG analysis and classification for heartbeat diagnosis, and implement it on an IoT-based embedded platform. This algorithm is our proposal for a wearable ECG diagnosis device, suitable for 24-hour continuous monitoring of the patient. We use Discrete Wavelet Transform (DWT) for the ECG analysis, and a Support Vector Machine (SVM) classifier. The best classification accuracy achieved is 98.9%, for a feature vector of size 18, and 2493 support vectors. Different implementations of the algorithm on the Galileo board, help demonstrate that the computational cost is such, that the ECG analysis and classification can be performed in real-time.**

ECG analysis, ECG heartbeat classification, Support-Vector-Machine (SVM), Discrete-Wavelet-Transform (DWT), Internet-of-Things (IoT), Embedded Systems

I. Introduction

Electrocardiogram (ECG) signal processing has been extensively used for the diagnosis of many cardiac diseases, the leading cause of premature death globally. In recent years, numerous algorithms have been developed for the automatic, computer-based and accurate recognition of arrhythmias in an ECG record. The methods used for the analysis and the classification of the ECG signal, as well as the the number of arrhythmia types examined, show a great deal of variance. Feature extraction techniques used, include morphology and the waveform geometry [1], Wavelet transform [2], Hilbert transform, Fourier transform, Hermite function, power spectral features, higher order spectral methods and nonlinear transformations such as Lyapunov exponents. The classifying methods proposed, include Fuzzy Logic methods, Artificial Neural Network, Hidden Markov Model, Genetic Algorithm, Support Vector Machines [2], Self-Organizing Map and Cluster analysis.

With the development of technology, wearable devices are playing an important role in ECG monitoring. Such devices can be autonomous and provide constant monitoring of patients without them being hospitalized, helping improve their quality of life and minimizing healthcare cost. Thus, the research conducted in the field of automatic ECG analysis has been, in a big part, focused in developing efficient algorithms that overcome the constraints of embedded portable and wearable devices. Furthermore, these devices are usually intended to be part of an IoT-based ECG monitoring design. In such a design, the ECG signal is detected by wearable sensors, sent through a short-rage communication (e.g. Bluetooth) to the wearable device where it is processed and analyzed, and finally sent via wide area connection (e.g. Wi-Fi) to a remote device for further analysis and storage.

In this work, we develop the software infrastructure that supports ECG signal analysis for feature extraction and the corresponding classification technique for diagnosis of the heart condition. The implementation is done on Intel's Galileo embedded platform, which is one of Intel's proposed IoT devices. We choose the Discrete Wavelet Transform (DWT) as feature extraction method and Support Vector Machine (SVM) as classification method. In order to derive an optimized and cost effective solution, we perform exploration over the ECG extracted features space, and achieve 98.8% accuracy. The execution times we obtained from implementing the application on the Galileo board show that the ECG analysis and classification can be performed in real-time.

II. Design and Exploration of ECG Analysis Flow

In this study, data from the MIT-BIH Arrhythmia database [3] were used. This database, provided online by PhysioNet, is a result of the collaboration between Beth Israel Deaconess Medical Center and MIT, and it is one of the most utilized databases for research purposes. The database is composed of 48 half-hour excerpts of two-channel (two leads) ambulatory ECG recordings, obtained from 47 subjects. The data are band-pass filtered at 0.1-100Hz and digitized at 360 samples per second, per channel. The first-channel lead of 45 records are used, which is a modified lead II (MLII), leaving out records 102, 104 and 114, whose first-channel lead is not a MLII. The MIT-BIH database also provides annotations for each record, where cardiologists have placed a diagnosis label for every heartbeat included in the record. The American Heart Association (AHA) heartbeat classes (N,V,F,E,P,Q,O) were used as reference for the two arrhythmia groups examined, 'Normal' (N) and 'Abnormal' (V,F,E,P,Q,O).

The initial stage of the analysis was implemented in Matlab environment. The WFDB Toolbox [4] for Matlab was used for the reading and processing of the ECG records of the database. Figure 1 illustrates the proposed structure of a typical ECG analysis and heartbeat classification.

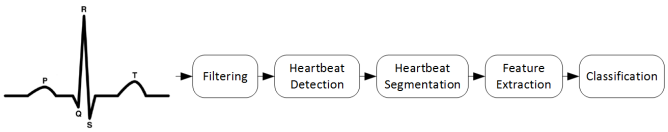

Fig. 1: Proposed ECG analysis flow

1. Filtering

The power line interference and the baseline wandering are

978-1-4673-9680-6/16/$31.00 ©2016 IEEE

significant noise sources that can strongly affect the ECG signal analysis. The signals were band-pass filtered at 1-50Hz [5].

2. Heartbeat detection

The WFDB function $wqrs()$ is applied to the signal, which gives us the locations of all QRS complexes found in the signal. This information along with the ECG signal, are the inputs to WFDB function $ecgpuwave()$, which gives us the exact position of all the R peaks found in the signal. QRS detection, especially detection of R wave, is easier than other parts of the ECG signal due to its structural form and high amplitude. Each R peak detection corresponds to the detection of a single heartbeat.

3. Heartbeat segmentation

Having located the R peaks, we proceed to segment the ECG signal into single heartbeats. We choose a window width of 257 samples, as suggested in [2], which having as center the detected position of the R peak, will cover the whole heartbeat waveform.

4. Feature extraction

A stage for feature extracton follows, in order to extract a feature vector for each produced heartbeat, containing a smaller number of elements than the ECG samples forming the heartbeat. We use Discrete Wavelet Transform (DWT) as a feature extraction mechanism, since it has been proven to produce very accurate results. The wavelet base for the DWT is Daubechies of order 2 (db2) and we perform 4 levels of decomposition as proposed in [2]. The 4 levels of decomposition produce 8 sets of coefficients each one for 4 levels of detailed and 4 levels of approximate coefficients. Since the heartbeat on which the DWT is applied consists of 257 samples, the number of wavelet coefficients for the first, second, third and fourth level, are respectively 130, 66, 34 and 18. Thus, 494 wavelet coefficients are obtained for each heartbeat. The final feature vector that serves as input to the classification stage, resulted from a design space exploration performed on all combinations of these 8 sets of coefficients.

5.Classification

The last stage consists of a binary classifier, which labels each heartbeat as either 'Normal' or 'Abnormal'. We focus on using a Support Vector Machine (SVM) classifier, with RBF as kernel function, mainly due to its ability to support non-linear classification with efficient accuracy and computation cost. The Matlab interface of LIBSVM [6]was used for the implementation of the SVM classifier. We use the 104581 heartbeats detected in the 45 selected ECG records to form the training and testing data sets. As feature vector we use the DWT coefficients extracted in the previous stage, and as target value we use the diagnosis label given for each heartbeat in the annotation files.

The training process of the SVM model can be done offline. Having a training data set and a testing data set for evaluation, we perform the design space exploration that is described below to decide upon the feature vector which gives the model with the best performance results. Then, with a fixed feature vector, the SVM model is produced offline. This model is used for classification of heartbeats inputted in the algorithmic flow, in real time. The complete algorithmic analysis including the offline training and the online heartbeat classification is shown in Fig. 2.

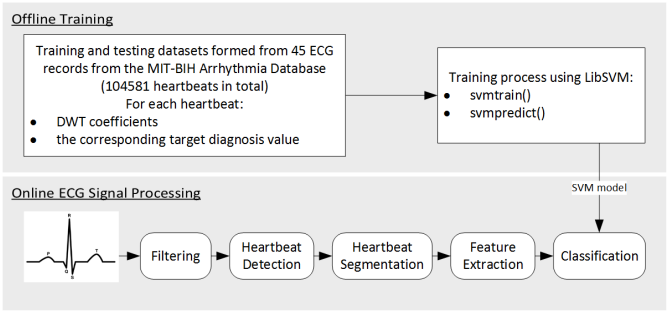

Fig. 2: Offline training and online classification

III. Design space exploration

A design space exploration is performed over all combinations of the 8 sets of DWT coefficients produced in the feature extraction stage. To reduce exploration time, DWT is calculated only once, producing a vector containing all sets of coefficients for each heartbeat detected in the database. An iteration takes place, choosing a different combination of these sets to serve as feature vector, and producing a different classifier. The goal is to produce a classifier with

- maximized accuracy ($\frac{Number\ of\ correctly\ classified\ points}{Total\ number\ of\ points}$)
- minimized computational cost

In this initial stage, the computational cost was not measured as the absolute execution time of classification on the platform used, but it was rather computed theoretically as the product of the number of support vectors and the size of feature vector, since this is the number of multiplications required for a heartbeat to be classified [7].

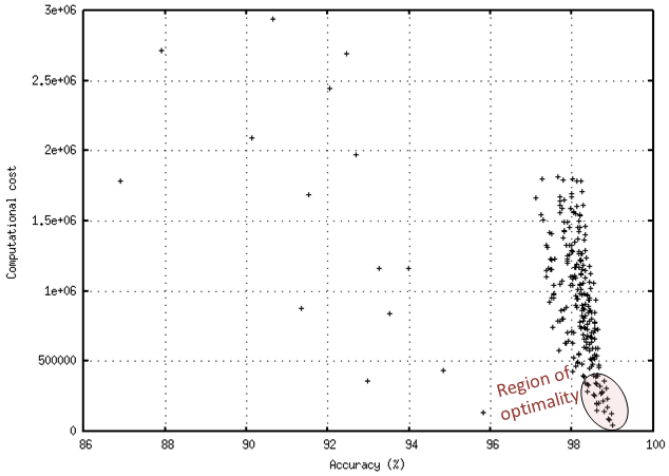

Fig. 3: Design space exploration of SVM classifier

Figure 3 presents the results of the design space exploration, regarding accuracy and computational cost. As we can see, in almost all cases the accuracy is above 97%, with only a few exceptions. The best result comes of the feature vector which only contains the approximate coefficients of the 4th level of decomposition. In this case, we have an accuracy of 98.9%, a feature vector of size 18, and 2493 support vectors.

IV. Implementation on embedded

IoT PLATFORM

In the second part of the study, the code is converted in C and implemented on Intel's Galileo board. The Galileo is the first product to feature the Intel Quark SoC X1000, a chip designed for small-core products and low power consumption, and targeted at markets including the Internet of Things and wearable computing. The Quark SoC X1000 is a 32-bit, single core, single-thread, Pentium (P54C/i586) instruction set architecture (ISA)-compatible CPU, operating at speeds up to 400 MHz. The use of the Pentium architecture gives the Galileo the ability to run a fully-fledged Linux kernel. What's more, an on-board Ethernet port provides network connectivity, while also the underside provides a mini-PCI Express slot, designed for use with Intel's wireless network cards to add Wi-Fi connectivity to designs.

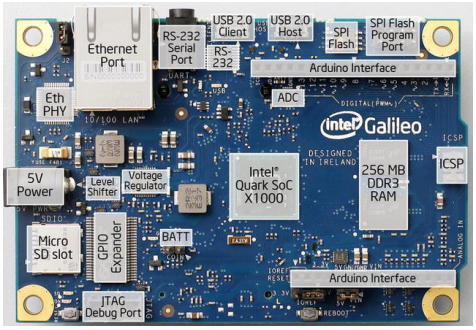

Fig. 4: Intel Galileo board [8]

The Galileo board's technical specifications:

- **Operating system:** Yocto Project-based Linux
- **Processor:** Single-Core 400MHz Intel Quark X1000
- **Memory:** 256MB RAM
- **Dimensions:** 107mm x 74mm x 23mm
- **Weight:** 50g (excluding PSU)
- **Networking:** 1x Wired 10/100 Ethernet, Optional PCIe Wireless

The Quark X1000 features:

- Up to 400MHz clock speed
- 16KB L1 Cache
- 512KB SRAM
- Single core, single thread
- Integrated SDIO, UART, SPI, USB, I2C, Ethernet, RTC

The application implemented on the board, reads sample by sample a digitized (at 360 samples per second) ECG signal and the analysis flow is executed for every set of 3000 samples that is read.

1. Filtering: A band-pass filter at 1-50Hz is implemented in C.

2. Heartbeat detection: since the source code of just $wqrs()$ is available in C, we alter the algorithm to only apply wqrs to the signal, with no substantial divergence in the output of this stage. Wqrs gives back the onsets of the QRS complexes. Instead of passing this information to $ecgpuwave()$, in order to get the exact position of the R peak, we keep the QRS onset information as the heartbeat reference point, and adapt the window applied to cover the heartbeat waveform.

3. Heartbeat segmentation: we decide on a window of 86 samples before the QRS onset, and 170 samples after the QRS onset (window width of 257 samples). Comparing the output of the two implementations up to this stage, we see that there is no substantial difference between the Matlab and the C implementation.

4. Feature extraction: The convolution of the signal with the wavelet decomposition low-pass and high-pass filters associated with wavelet Daubechies of order 2, is implemented to produce the approximation and detail coefficients. This stage is implemented parametrically in terms of the coefficients that compose the feature vector of the heartbeat used in the classification stage. For each decomposition level, the coefficients produced are stored in a matrix if they are part of the final feature vector, or else they are only used for the computation of the next decomposition level. The process stops whenever all required coefficients have been produced.

5. Classification: we use the LIBSVM library which includes the source code of $svmpredict()$ in C. We convert the SVM model under examination, which has already been produced by the same function in the Matlab environment, into the format that is used in the C implementation. The SVM model is only created once, in an initialization section of the program, and used in the classification stage for each heartbeat.

In fig. 5 we illustrate the overall framework. This includes the exploration phase conducted in the Matlab environment, meaning the DSE which resulted in a set of optimal solutions. In the customization phase, having selected a desirable solution, the feature vector is set accordingly and the corresponding SVM model is produced by a generator which converts the Matlab SVM model to the appropriate format for the C implementation. Finally, the run-time phase is the implementation of the algorithm in C, for the configuration selected, conducted on the Galileo board.

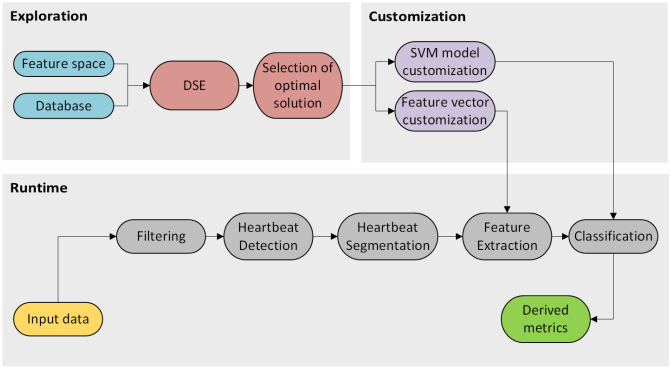

Fig. 5: The overall ECG analysis framework

We selected the 10 optimal, the 10 most computationally demanding, and 11 configurations from inbetween, as resulted from the DSE, to implement on the platform. Table II shows the results of the time averages and the percentage of total execution time, for each stage of the analysis flow, for the 5 optimal configurations (Table I). The time averages for the filtering and the heartbeat detection stage, correspond to the 3000 sample input, while the time averages of the feature extraction and the classification stage are per heartbeat detected. For the estimation of the percentage of the total execution time of each stage, it is assumed that for every 3000 sample input, 10 heartbeats are detected. In the best cases examined, the total execution time is much less than the approximate 9 seconds required for the 3000 samples of

TABLE I: 5 optimal configurations

DWT coefficients	number of support vectors	feature vector size	Accuracy (%)	Computational cost
cA4	2493	18	98.99	44874
cA3	2490	34	98.93	84660
cA4, cD4	2513	36	98.90	90468
cA3, cA4	2408	52	98.99	125216
cA3, cD4	2696	52	98.80	140192

TABLE II: Execution times for 5 optimal configurations

filtering		beat detection		feature extraction		classification	
average (in sec)	% of total ex. time	average (in sec)	% of total ex. time	average (in sec)	% of total ex. time	average (in sec)	% of total ex. time
0.77127	70.01	0.01691	1.53	0.00210	1.91	0.02925	26.55
0.77807	61.88	0.01707	1.36	0.00192	1.53	0.04430	35.23
0.62330	60.49	0.01373	1.33	0.00171	1.66	0.03762	36.52
0.62322	54.80	0.01381	1.21	0.00166	1.46	0.04836	42.53
0.62386	52.48	0.01379	1.16	0.00171	1.44	0.05341	44.92

the signal to be read. This means that the ECG signal analysis and classification proposed, can be performed in real-time.

In fig. 6, we demonstrate that a correlation indeed exists between the execution time of the classification stage and the theoretical model used to examine the computational cost, which was the product of the number of support vectors and the size of feature vector. In all cases examined, the accuracy was above 90%, for the optimal cases being above 98.6%.

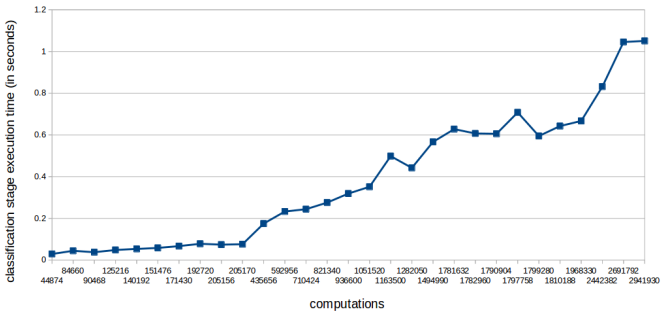

Fig. 6: Computational cost model: theoretical vs measured

Setting as time constraint the heart rate, which is approximately 0.7-1 sec (60-90 bpm) for the normal resting case, and assuming a worst resting case at 0.3 sec (200 bpm), we see in fig. 7 that all optimal design solutions satisfy this constraint.

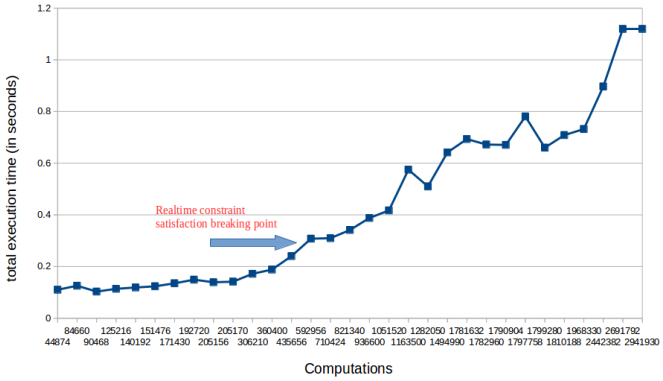

Fig. 7: Total Execution Time of Analysis Flow Implementation on the Target Platform

In fig. 8, we present the average % of total execution time for each stage of the algorithm, of the 10 optimal configurations examined, the 11 configurations from the inbetween and the 10 most computationally demanding configurations. In the last case, the classification stage takes up more than 90% of the total execution time.

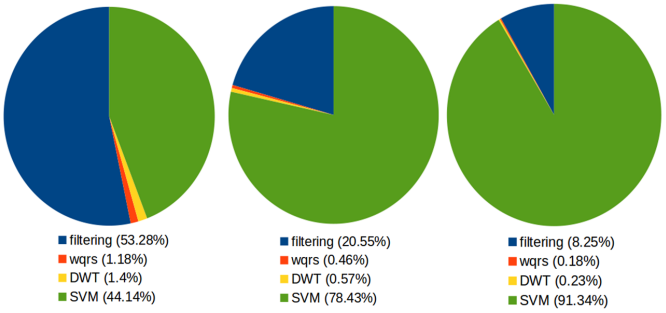

Fig. 8: Avarege % of total execution time for each stage

V. Conclusion

In this paper we develop an algorithm for ECG analysis and classification, and implement it on an IoT-based embedded platform. This algorithm is our proposal for a wearable ECG diagnosis device, suitable for 24-hour continuous monitoring of the patient. In almost all cases, the classification accuracy achieved is above 97%. The best result comes of the feature vector which contains the approximate coefficients of the 4th level of decomposition, with 98.9% accuracy, a feature vector of size 18, and 2493 support vectors. Implementing the best configurations of the design space exploration, on the Galileo board, we demonstrate that the computational cost is such, that the ECG analysis and classification can be performed in real-time.

Acknowledgments

This work was partially supported by the DAAD-funded project "Teacher".

References

[1] Philip De Chazal, Maria O Dwyer, and Richard B Reilly. Automatic classification of heartbeats using ecg morphology and heartbeat interval features. *Biomedical Engineering, IEEE Transactions on*, 51(7):1196–1206, 2004.

[2] Elif Derya Übeyli. Ecg beats classification using multiclass support vector machines with error correcting output codes. *Digital Signal Processing*, 17(3):675–684, 2007.

[3] George B Moody and Roger G Mark. The impact of the mit-bih arrhythmia database. *Engineering in Medicine and Biology Magazine, IEEE*, 20(3):45–50, 2001.

[4] Ary L Goldberger, Luis AN Amaral, Leon Glass, Jeffrey M Hausdorff, Plamen Ch Ivanov, Roger G Mark, Joseph E Mietus, George B Moody, Chung-Kang Peng, and H Eugene Stanley. Physiobank, physiotoolkit, and physionet components of a new research resource for complex physiologic signals. *Circulation*, 101(23):e215–e220, 2000.

[5] Durgesh Kumar Ojha and Monica Subashini. Analysis of electrocardiograph (ecg) signal for the detection of abnormalities using matlab. *World Academy of Science, Engineering and Technology, International Journal of Medical, Health, Pharmaceutical and Biomedical Engineering*, 8(2), 2014.

[6] Chih-Chung Chang and Chih-Jen Lin. Libsvm: a library for support vector machines. *ACM Transactions on Intelligent Systems and Technology (TIST)*, 2(3):27, 2011.

[7] Corinna Cortes and Vladimir Vapnik. Support-vector networks. *Machine learning*, 20(3):273–297, 1995.

[8] Manoel Carlos Ramon. *Intel galileo and intel galileo gen 2*. Springer, 2014.