Real-Time Probabilistic Heart Beat Classification and Correction for Embedded Systems

Grégoire Surrel[1], Francisco Rincón[2], Srinivasan Murali[2], David Atienza[1]

[1] Embedded Systems Laboratory (ESL), EPFL, Lausanne, Switzerland
[2] SmartCardia Sàrl, Lausanne, Switzerland

Abstract

*With the emergence of wearable and non-intrusive medical devices, one major challenge is the real-time analysis of the acquired signals in real-life and ambulatory conditions. This paper presents a lightweight algorithm for on-line heart beat classification and correction that relies on a probabilistic model to determine whether a heart beat is likely to happen under certain timing conditions or not. It can quickly decide if a beat is occurring at an expected time or if there is a problem in the series (*e.g.*, a skipped, an extra or a misplaced beat). If an error is detected, the series is repaired accordingly. The algorithm has been carefully optimized to minimize the required processing power and memory usage in order to enable its real-time embedded implementation on a wearable sensing device. Our experimental results, based on the PhysioNet Fantasia database, show that the proposed algorithm achieves 99.5% sensitivity in the detection and correction of erroneous beats. In addition, it features a fast response time when the activity level of the user changes, thus enabling its use in situations where the heart rate quickly changes.*

1. Introduction

Our aging society needs autonomous systems to provide quick and continuous health-care of people. Such systems can monitor the vital parameters, such as the electrocardiogram (ECG) of the person at home or outdoor environments, and can alert the doctor or emergency response team of different events. With the spreading of smartphones, researchers have been developing vital parameter wearable sensing nodes that use them as gateways (and display terminals), and directly alert the doctor or hospital in case of a health condition. While continuous mobile health monitoring opens up new care delivery mechanisms, it also creates novel challenges.

According to [1,2], analyzing the Heart-Rate Variability (HRV) is key in cardiovascular analysis because the variation of the heart rate contains indicators of current or impending cardiac diseases. However, the algorithms need to work on long uninterrupted series of beats. This context is not conceivable for non-invasive portable systems where the number of available leads is much smaller and the contact noise is higher. Although the increasing quality of ECG recordings combined with better R-peak detection algorithms improved the situation, having an on-line validation and correction of the RR series is desirable because the HRV indices can be heavily affected by erroneous beats in the series [3]. The correction of ectopic beats is said to be essential regardless of the used spectral estimation technique.

Using a probabilistic model of the heart beats occurrences, it is possible to classify and correct each RR (see Fig. 1) interval in regard to known situations [4]. This solution is better than simply ignoring the problematic parts of the series by removing them because it skews the resulting HRV indices. As recordings include motion artifacts, correcting the series before processing is a necessary step. An automated correction helps to extract the HRV indices live for an immediate feedback for the user.

The rest of the paper is organized as follows. Section 2 presents the reference algorithm investigated in this work, while Section 3 details our workflow. Then, in Section 4, we describe the coefficient-optimization implementation and Section 5 explains the algorithm tuning process. A study of the used dataset is performed in Section 6. Finally, our conclusions are summarized in Section 7.

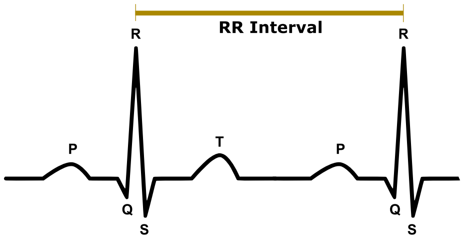

Figure 1. Schematic diagram of normal sinus rhythm for the ECG human heart, including the P-wave, QRS-complex and the T-wave

2. Reference algorithm

The reference algorithm is described in [4] where the RR intervals between heart beat events are modeled as following an Inverse Gaussian (IG) probabilistic distribution where $\mu > 0$ is the mean and $\lambda > 0$ is the shape factor, as follows:

$$ig(x; \mu, \lambda) = \sqrt{\frac{\lambda}{2\pi x^3}} \times \exp\left(\frac{-\lambda(x-\mu)^2}{2\mu^2 x}\right)$$

The μ and λ parameters are estimated by maximizing the log-likelihood applying an exponential decay over the recent history of recorded events. They are computed using the following equations [5]:

$$\hat{\mu} = \frac{\sum_{i=1}^{n} w_i RR_i}{\sum_{i=1}^{n} w_i}, \qquad \frac{1}{\hat{\lambda}} = \frac{1}{n}\sum_{i=1}^{n} w_i \left(\frac{1}{RR_i} - \frac{1}{\hat{\mu}}\right)$$

where $w_i = w(\tau_i) = e^{-\alpha\tau}$ is the weighting function and RR_i is the i^{th} time interval between two beats.

2.1. Beat classification and correction

In our implementation four different situations are identified and corrected. They change how the IG model will be used for beat-classification. For each beat, the four situations are estimated and get a score related to their likelihood. After a relative weighting of individual scores, the highest one determines the final classification.

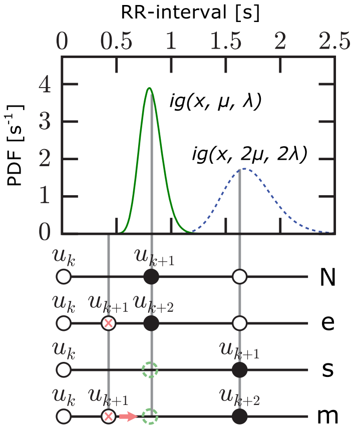

Figure 2. Score for each situation of classification

2.1.1. Normal beat N

For a normal beat, the newest RR interval will match closely the expected timing estimated from the recent history. Thus, the score for beat i being a normal beat is: $p_N = ig(RR_i; \mu_{i-1}, \lambda_{i-1}) - \eta_N$, where η_N is a constant relative weighting factor. In Fig.2 this occurs when the new beat u_{k+1} is near the maximum of the probability density of the inverse Gaussian function.

2.1.2. Extra beat e

If an extra beat u_{k+1} is in the series, the next beat u_{k+2} will be really close to the expected timing (i.e., near the probability density peak). Thus, $p_e = ig(RR_i + RR_{i+1}; \mu_{i-1}, \lambda_{i-1}) - \eta_e$, where η_e is a constant relative weighting factor. This is the second case in Fig. 2, where the new beat u_{k+1} is close to a zero-probability, that is to say very unlikely. An extra beat is corrected by removing it from the series.

2.1.3. Skipped beat s

A skipped beat is detected if the RR interval matches the expected time of two beats. In other words, its score is $p_s = ig(RR_i; 2\mu_{i-1}, 2\lambda_{i-1}) - \eta_s$, where η_s is a constant relative weighting factor. This is the third case in Fig. 2, where the beat u_{k+1} is at the second maximum probability peak.

A skipped beat is corrected by inserting a beat, and the way they are inserted influences the future evaluations of the inverse Gaussian distributions, thus three options exists: (1) splitting the RR interval in two equal parts so $RR_i = RR_{i+1}$; (2) setting the beat to the maximum probability computed from the previous beat as $RR_i = \frac{\sqrt{4\lambda^2\mu^2+9\mu^4}-3\mu^2}{2\lambda}$; (3) use an average between solutions 1 and 2. As the correction influences the classification of the next beats (Fig. 3), each use case should evaluate the best option. In this work, we chose the second solution, using the maximum of the probability density function. Although others methods exist, such as iterating the beat's position using a Newton-Raphson approach to maximize the likelihoods of the missed and next beats together, we did not use any of them due to their heavy computation cost and the unpredictable run-time for embedded systems due to multiple optimization loops.

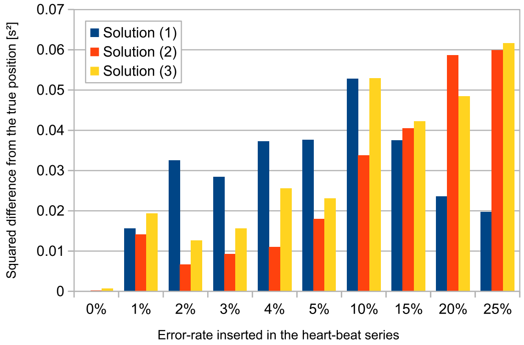

Figure 3. Evaluation of the influence of the correction method on the global performance of the algorithm. As shown, the correction method influences the global algorithm efficiency, as the corrected beat affects the next IG distribution calculation.

3. Embedded Implementation Workflow

We applied different methods to achieve an efficient embedded algorithm with a low processing load. Each method result has been tested using a training phase on one part of the Fantasia Database [6], and then evaluating its performance on the remaining set of files (`f1o01, f1o05, f1o10, f1y01, f1y03, f1y08, f2o04, f2y03 and f2y06`).

The performance is assimilated to the classification error rate on corrupted series where each 4th beat is altered. The generated sequence of beats is `NNNeNNN(s)NNNm` with `N` being a normal beat, `e` an extra beat, `s` indicating a skipped beat (i.e., not in the series) and `m` a misplaced beat. Hence, we know how the classification of each beat and how to reproduce the evaluation.

4. Coefficients optimization

Before evaluating the performance of a given configuration, the relative coefficients η for classification are optimized to get the best results possible. This is a one-time off-line training process. Thus, our run-time implementation uses the best values of coefficients.

As the parameter design space is large, we developed an iterative algorithm starting with a fixed set of values for the parameters η and run a straightforward iterative tuning: as long as the norm of vector of modifications (`detlas`) to apply is greater than a threshold, each parameter η_N, η_e, η_s and η_m to optimize is slightly modified according to the before-mentioned vector. If the final classification rate is improved, the modification is saved and the next modification will be of a higher magnitude. In the other case, the magnitude is lowered and turned in the opposite direction (as the sign is changed). On each iteration, the `deltas` vector is multiplied by 1.1 to have more iterations before reaching the threshold and therefore being closer to the optimal value.

5. Algorithm tuning and simplification

In the following subsections, we detail the three iterative steps to obtain our low-complexity embedded implementation.

5.1. Window-size reduction

Our first step reduces the size of the window containing the beats used for the IG modeling. The longer the window, the more memory is used to store the beats data and more operations are needed to compute the $\hat{\mu}$ and $\hat{\lambda}$ estimates. Fig. 4 shows that the best window size uses the last six RR intervals, having the best trade-off for low resources utilization while yielding good results.

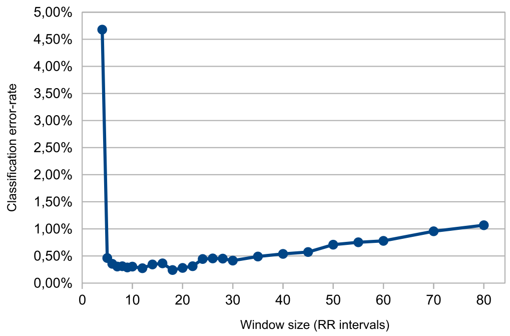

Figure 4. Classification error-rate for different windows lengths of the algorithm. Smaller windows improve results until a lower bound when classification becomes unreliable, as not enough data is used to compute $\hat{\mu}$ and $\hat{\lambda}$ in the IG model.

5.2. Weighting

After the window length optimization is optimized, it is possible to replace the initial exponential decay (i.e., giving more weight on the newest RR intervals) by constant weighting. This replacements gives a classification error-rate of 0.34% instead of 0.36% by the exponential decay function. Thus, the results are improved and uses no inverse exponential computation, which is not adequate for a low-power embedded systems.

5.3. Different estimator for λ

As the standard deviation of the IG distribution is $\sigma = \sqrt{\mu^3 \;/\; \lambda}$, we can use this formula to have another estimate of λ. The complexity is similar as in the original algorithm so all the benefits are due to a custom manual tuning and an efficient implementation of the square-root operation. In fact, this alternative $\hat{\lambda}$ yields a classification error-rate of 0.30% instead of 0.34%. This is a small benefit which is balanced by a higher CPU load using a naïve implementation: the benchmarking of the two estimators gives this alternate estimator a 69% time penalty. As we want the lightest processing possible, we decided to keep the original λ estimate.

6. Dataset analysis

Two results of Table 1 show a much higher classification error-rate: the files `f1y01` and `f1y08` have a 2.22% and 0.56% error rate, whereas the others are between 0% and 0.09%. The reason is that these two files have the highest Heart Rate Variability (HRV) computed both using the SDNN or RMSSD. In fact, when 40 files of the dataset are sorted by increasing RMSSD, seven out of nine recordings used for evaluation of the algorithm performance are in the ten first files, thus having the lowest HRV, which is not representative of the full dataset. Indeed, 2% of classification

Table 1. Classification error rate for the different used files

File	$beats$	Er_{tot}	Er_N	Er_e	Er_s	Er_m
f1o01	7951	0	0	0	0	0
f1o05	6353	2	0	0	1	1
f1o10	9144	2	1	0	1	0
f1y01	9748	216	111	3	47	55
f1y03	8479	8	3	0	1	4
f1y08	8071	45	18	2	5	20
f2o04	7654	1	1	0	0	0
f2y03	7549	1	1	0	0	0
f2y06	7603	6	3	0	2	1

errors exist for the worst-case file `f1y01`.

Given the variability of the results depending of the individual files, some more analysis has been done using both manual selection of files depending on the HRV or random shuffling of the files used in the dataset.

6.1. Manual file selection

Four tests were performed to see the influence of the HRV on the classification. To have a fair comparison with the reference paper, we kept the same size for the training dataset and the performance-evaluation dataset, that is to say respectively 31 files and the remaining nine files. For each test, the files used for the performance evaluation are the files ranked 5 to 13 when sorting the dataset by RMSSD or SDNN, ordering from low to high or the opposite.

After running the algorithm on sets of files sorted by HRV, we notice that the results are significantly changed. Training using the files with a high RMSSD gives the best classification, with a really low error-rate (less than 0.10%). Training on files with low SDNN gave the worst results, with 1.60% classification-error rate. On the other hand, as the performance evaluation is done on the remaining set of files (having a lower RMSSD), the classification is easier for the algorithm. This is why we ran a statistical analysis, shuffling the dataset and the recordings used.

6.2. Statistical analysis

To further check the influence of the recordings used for training, as well as the size of the dataset, an automatic shuffling of the recordings used for training and performance evaluation has been performed. The number of training files range from 10% to 90% of the full dataset, evaluation being always performed on the remaining recordings.

Our results indicate that the training dataset can be reduced to a few recordings, as long as they are representative of the total dataset. In our analysis, only four well-chosen recordings are enough to get good results. In the case we have no information about the individual recordings, using ten out of the 40 files is the best way to lower the variability of the result.

7. Conclusion

This paper has presented a new real-time classification and correction algorithm of heart beats in embedded systems, which provides a reliable continuous monitoring approach for people in a wide range of applications, from wellness to medical care. Starting form a reference algorithm, our approach has optimized the number of loops to computing intermediary results and reduced the amount of data needed to get good classification results. Overall, our obtained results indicate a 10× memory use and computation reduction, which enables a very smooth execution in embedded systems operating at very low frequencies (few Mhz).

8. Acknowledgements

This work has been partially supported by the ObeSense (no. 20NA21_143081) RTD project, evaluated by the Swiss NSF and funded by Nano-Tera.ch with Swiss Confederation financing, as well as the ONR-G grant no. N62909-14-1-N072.

References

[1] A.J. Camm M. Malik JB, et al. Heart rate variability : Standards of measurement, physiological interpretation, and clinical use. ;93(5):1043–1065. ISSN 0009-7322, 1524-4539.

[2] Acharya UR, Joseph KP, Kannathal N, Lim CM, Suri JS. Heart rate variability: a review ;44(12):1031–1051. ISSN 0140-0118, 1741-0444.

[3] Clifford G, Tarassenko L. Quantifying errors in spectral estimates of HRV due to beat replacement and resampling ; 52(4):630–638. ISSN 0018-9294.

[4] Citi L, Brown E, Barbieri R. A real-time automated point-process method for the detection and correction of erroneous and ectopic heartbeats ;59(10):2828–2837. ISSN 0018-9294.

[5] Tweedie MCK. Statistical properties of inverse gaussian distributions. i. ;28(2):367. ISSN 0003-4851.

[6] Goldberger AL, Amaral LAN, Glass L, et al. PhysioBank, PhysioToolkit, and PhysioNet : Components of a new research resource for complex physiologic signals ; 101(23):e215–e220. ISSN 0009-7322, 1524-4539.

Address for correspondence:

Grégoire Surrel
EPFL-STI-IEL-ESL - ELG 130
Station 11, CH-1015 Lausanne, Swtizerland
`gregoire.surrel@epfl.ch`