Neural Architecture Search for Tiny Detectors of Inter-beat Intervals

Rafael G. de Lima[*], Pedro G. Freitas[§], Giovani D. Lucafo[*],
Vanessa Fioravanti[†], Ismael Seidel[‡], and Otávio A. B. Penatti[*]
[*]Samsung R&D Institute Brazil (SRBR), Campinas, São Paulo, Brazil
[§]Department of Computer Science, University of Brasília, Brasília, Brazil
[†]McKinsey & Company, São Paulo, Brazil
[‡]Department of Informatics and Statistics, University of Santa Catarina, Florianópolis, Brazil

***Abstract*—Inter-Beat Interval (IBI) is a clinically established proxy indicator of Heart Rate Variability (HRV), as well as a number of cardiac rhythm abnormalities. Its relevance for healthcare continuous monitoring solutions has called for precise, yet lightweight IBI estimators, compatible with real-time applications on resource-constrained devices, such as wearables. While the increasing relevance of Deep Learning-based models, with their advantageous automatic feature extraction properties, has also made its way onto the healthcare technology domain, the design of IBI estimators using such an approach would typically involve extensive handcrafted architectures. To mitigate the manual effort of finding satisfactory models for many tasks, automatic search methods for these neural Machine Learning models have been proposed. In the present work, we propose a Neural Architecture Search (NAS) approach to discover compact neural models for IBI estimation. In this work, we report the best-found models that outperform the current state-of-the-art (SOTA). The reported models were obtained under such strict memory-performance trade-off considerations in a standardized medical database of photoplethysmographic sensor data.**

***Index Terms*—Inter-beat Interval, Neural Architecture Search, Peak Detection, Signal Quality**

I. Introduction

Wearable healthcare devices have received increasing public interest from the consumer electronics industry these days and play a crucial part in human life, notably, in Health Monitoring Applications (HMA) [1]. Among the diverse wearable-based HMA domains, cardiovascular monitoring applications are one of the most important ones, mostly motivated by various reports indicating that Cardiovascular Diseases (CVDs) have become the leading cause of death worldwide [2]. For instance, in 2012, the World Health Organization (WHO) reported that 17.6 million deaths occurred worldwide due to CVD causes, a growth since 2000 [3]. Moreover, in addition to mortality rates, CVDs also leads to infirmity and decreased quality-of-life [3]. Thus, to provide a better life for people, continuous cardiac monitoring is essential in preventing and managing CVDs.

The conventional cardiac monitoring methods commonly employ Electrocardiogram (ECG) to monitor various physiological parameters, especially Heart Rate (HR). HR is a key indicator of cardiac health and it is computed as the average value of heart contractions during a determined time frame. Its variation, i.e., the HRV, is one of the most important factors in identifying cardiac status, and heart conditions, and it serves as a guide to the assessment of cardiac wellness. HRV is the physiological measure of variations between consecutive heartbeats that can be measured via IBI [2].

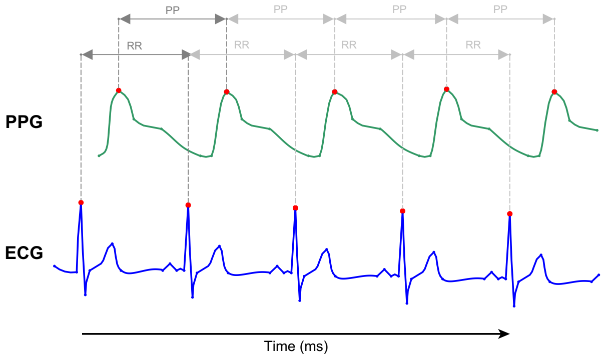

Fig. 1: Inter-Beat Interval (IBI) estimation using RR Interval (RRI) from ECG signals and the corresponding Peak-to-Peak Interval (PPI) from Photoplethysmography (PPG) sinals.

IBI is traditionally computed using ECG signals, which is still the gold standard for HR and heart rhythm analysis [4]. Because of its unique signature, ECG provides detailed information about the heart's function due to its characteristic waveform. Considering the ECG waveform, IBI can be estimated by computing the difference between R peaks. The R peak is typically the largest spike in a period of the ECG signal and it corresponds to the depolarization and contraction of the left ventricle. Usually, ECG signals are measured considering the electrical activity of the heart, using multiple electrodes, which can be a big hindrance for some applications, especially on wrist-worn wearable devices.

Photoplethysmography (PPG) is another common physiological signal used in medical applications that can be used to measure IBI. It is an optical measurement of an organ's blood flow volume by illuminating the skin and subcutaneous tissue with light of a specific wavelength. This light is either absorbed, passed through, or reflected back. A photodiode then converts the measured lights (before and after reflection) into

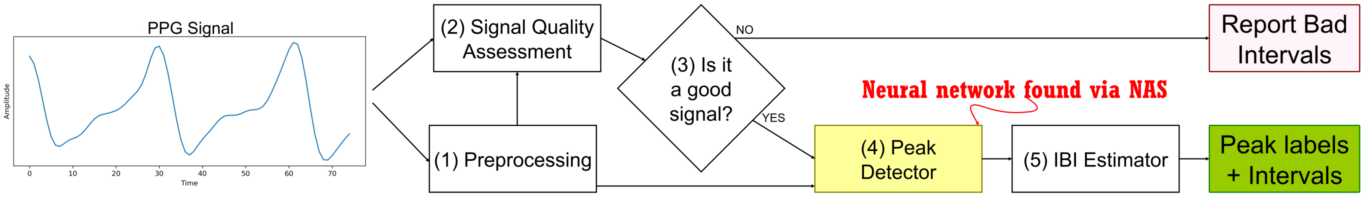

Fig. 2: The two-stage framework for IBI peak classification.

an electrical signal, which produces a characteristic waveform. Considering the PPG waveform, IBI can be estimated by computing the difference between the largest peaks of consecutive periods, the Peak-to-Peak Interval (PPI). PPG compares to ECG as depicted in Figure 1. Notice how both signals have the same period, meaning that either one can be studied to determine IBI.

PPG usage presents some advantages for wearable applications, especially on wrist-worn or forefinger devices. It is easy to set up, convenient, simple, and economically efficient. Modern PPG devices use a single optical sensor, with a near-infrared emitter and detector integrated into a mechanically robust, reusable, and comfortable to wear, which can be readily integrated with computational resources. This allows continuous measurements, which is advantageous to enable continuous monitoring solutions for wearable electronics, like fitness trackers and smartwatches. Because of these aforementioned advantages, this paper is focused on IBI estimation using PPG signals.

In the context of using PPG for measuring IBI, some researchers propose techniques for enabling the production of feasible wearable health applications. Most of these studies rely on rule-based methods for guiding the decision on whether to retain or remove candidate systolic peaks from signal [5] Regularly, these rule-based method employs threshold functions based on the amplitude value and the time gap to the latest peak position. Since these methods employ only a set of thresholds functions (i.e., a series of `if` and `elses`), they have the remarkable advantage of being computationally simple and suitable for being employed in embedded devices with severe memory restriction and computational constraints. Nonetheless, because rule-based methods usually define their threshold functions and parameters experimentally or throughout brute-force searches, the performance of such methods does not perform well when the amplitude of the PPG signal is subject to wide variations.

Recently, solutions based on Machine Learning (ML) and Deep Learning (DL) approaches have enjoyed increasing visibility due to their enhanced robustness to more challenging scenarios [6] .The learning-based methods, despite having the advantage of enhanced robustness in challenging scenarios, still carry the need for extensive handcrafted parameters and manual decisions over architectures. Furthermore, these methods are customarily computationally costly and demand considerable hardware resources.

In this paper, we propose an approach that combines the advantages of learning-based with the advantages of rule-based methods. In other words, we employ ML to generate robust and well-performing models to classify PPG peaks for IBIs estimation purposes. We make use of NAS [7] techniques for finding lightweight neural solutions of competitive performance under strict memory and computational constraints.

II. Proposed Method

Figure 2 depicts the overall steps employed during this study. It includes five main stages, namely (1) Preprocessing, (2) Signal Quality Assessment, (3) Signal Quality Classifier, (4) Peak detector, and (5) IBI estimator. The Preprocessing step is responsible for preprocessing the raw sensed data to reduce noise from the raw PPG sensor. We implemented an algorithm inspired by Khalak & Wiggins [8], where we used a Butterworth filter of second-order whose pass-band frequencies are 0.8Hz and 4.5Hz to filter the green channel of the PPG signal and the three-axis (i.e., x, y, and z) of inertial sensors. Then, a cascaded Least-Mean-Square (LMS) filter is employed for reducing motion artifacts using the inertial sensors data.

The Signal Quality Assessment and Signal Quality Classifier steps are responsible for analyzing the quality of the PPG signal and classifying it as ‘reliable’ or ‘unreliable’, respectively. These steps can be implemented separately (as illustrated in Figure 2) or jointly (e.g., as in a neural network). Since there is a vast literature proposing signal quality evaluation methods [9], and since all these methods could be virtually used for signal quality classification tasks in our study, we will not detail the algorithms used for signal quality in the next sections. Details about the algorithms are described in [10]. The ‘IBI Estimator’ step just gets the peak positions detected by the ‘Peak Detector’ step and their corresponding instants to compute the difference between consecutive peak instants. This difference corresponds to the IBI. Therefore, the main scope of this paper comprises the Peak Detector step, which is detailed in the next section.

A. *Peak Detector*

Let a PPG sensor data reading discretely captured over a session of time range $[0, T]$, with sampling frequency f_s^{ppg}, defined as a monotonically increasing vector

$$\boldsymbol{x}_{[0,T]} = \{x_0, x_1, ..., x_N\} \quad x_i \in [0, T], \tag{1}$$

and an equally monotonically increasing reference ECG vector, simultaneously captured with sampling frequency f_s^{ecg}, defined as

$$\boldsymbol{e}_{[0,T]} = \{e_0, e_1, ..., e_{N_{ecg}}\} \quad e_i \in [0, T]. \tag{2}$$

The goal of IBI estimation is to extract a set of tuples

$$\boldsymbol{p}_{[0,T]} = \{(p_i, \mathbb{I}_i)\} \quad i \in \{0, 1, ..., N_p\}, \tag{3}$$

where $p_i \in \boldsymbol{x}_{[0,T]}$, and $\mathbb{I}_i$ is an indicator function, with

$$\mathbb{I}_i = \begin{cases} 1, & \text{if peak } p_i \text{ is classified as a true peak;} \\ 0, & \text{if peak } p_i \text{ is classified as a false peak.} \end{cases} \tag{4}$$

The correctness of the estimation is then computed peak-by-peak, by checking if, for each peak p_i, there is a single corresponding peak $p^e_j \in \boldsymbol{e}_{[0,T]}$ subject to $||p_i - p^e_j|| < \delta$, where p^e_j is the jth peaks previously labeled as either 'true' or 'false' by human experts.

The peak detection of all the candidate peaks uses the derivative of the PPG signal computed by taking the difference between consecutive preprocessed signal samples. If its derivative sign switches from positive to negative, it is detected as a 'candidate peak'. A given candidate peak, p_i, is identified using the three-point sliding window method, i.e., $x[t_i - 1] < x[t_i] > x[t_i + 1]$. On the other hand, finding zero-crossing points in which the derivative sign switches from negative to positive indicates a signal valley point. Valley points are identified whenever the zero crossing points v_i lie on $x[v_i - 1] > x[v_i] < x[v_i + 1]$.

After detecting the peaks and their corresponding instants, we use this information to extract features to classify which peaks are systolic or not. The features are extracted directly from the signal using the relative signal amplitudes and the relative time gaps between the peak of interest t, a and the previous and the following fiducial points (i.e., peaks and valleys). These features and fiducial points are illustrated in Figure 3, where the peak indicated as "?" denotes a peak of interest whose extracted features are associated with it. The features are calculated from the predecessor and the following coordinates, which are associated with fiducial points in the neighborhood of the candidate peak. The scalar a refers to the amplitude of the peak of interest. The relative amplitude features of neighbor peaks a_{pp} and a_{np} refer to the amplitude difference between a and the previously detected peak, a_{pp}, or the following peak, a_{np}. We follow the same reasoning for the relative valley amplitude features, i.e., a_{pv} and a_{nv}. The remaining features fetch the time gaps between the peak under assessment and its neighbor peaks and valleys, i.e., d_{pp}, d_{pv}, d_{nv} and d_{np}. Therefore, in summary, the feature vector $\alpha \in \mathcal{A} \subseteq \mathbb{R}^9$ is given as $\alpha = [a_{pp}, a_{pv}, a_{nv}, a_{np}, a, d_{pp}, d_{pv}, d_{nv}, d_{np}]^\intercal$.

Detecting whether a peak is systolic or not is essential because the IBI is actually the time difference between two consecutive systolic peaks, as illustrated in Figure 1. Any classifier model receiving the feature vector α as input and mapping it into binary label can be used for this task (e.g., Support Vector Classifier, Random Forests, etc). However, since we are interested in develop techniques for wearable devices with severe hardware restrictions, considering the constraints of low hardware consumption and high predictive efficiency, we adopted a NAS-based strategy to produce a set of neural networks that met them.

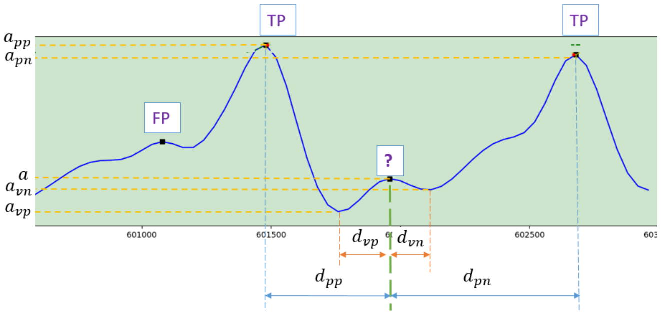

Fig. 3: Coordinates of fiducial points used as features.

Formally, our goal is to design a NAS method for finding a binary peak classifier model $PK \in \mathcal{MPK}(\cdot) : \mathcal{A} \rightarrow \{0, 1\}$ aiming to label a peak as a truly systolic peak (1) or a false one (0). In this case, the model space $\mathcal{M}$ is composed as the set of all Multilayer Perceptrons (MLPs) which can be defined by the following three parameters:

- $h \in \{0, 1, ..., 3\}$, number of hidden layers
- $l \in \{1, ..., 10\}^h$, number of neurons in each hidden layer,
- and $\sigma \in \Sigma^h$, where Σ is the activation functions set, which contains the Sigmoid, Tanh, ReLU, Mish, LeakyRely, and Softplus functions.

The last layer of the sampled model is always fixed to one hidden neuron and Sigmoid activation function, for generating the one-dimensional binary label as output. The NAS then works through the following steps:

1) Initially, all possible combinations of hidden layers are initialized with equal probability;
2) Next, for each NAS epoch, a batch of models is composed by generating architectures following the constraints of h, l, and σ;
3) Each model of the batch is trained on the training samples and its validation accuracy is obtained after a predefined number of training epochs;
4) After obtaining the performance of each model in the batch, a Long Short-Term Memory (LSTM) network [12], denominated *controller*, is trained with a Policy Gradient method with the reward of each model sampled set as its corresponding validation accuracy. That way, layer configurations with better performance tend to be sampled more often in the next NAS epochs;
5) Steps 2, 3, and 4 are repeated until the predefined maximum number of NAS epochs is reached.

Besides these steps, a weight-transferring strategy is employed to speed up the training of architectures with layers that overlap with those of previously sampled models. In this case, if a given hidden layer of a model can be found in an already trained model, its final weights are re-used in the corresponding layer of the new model.

III. Results

We used a dataset collected for this study that includes 46 volunteer subjects. These subjects include 9 volunteers

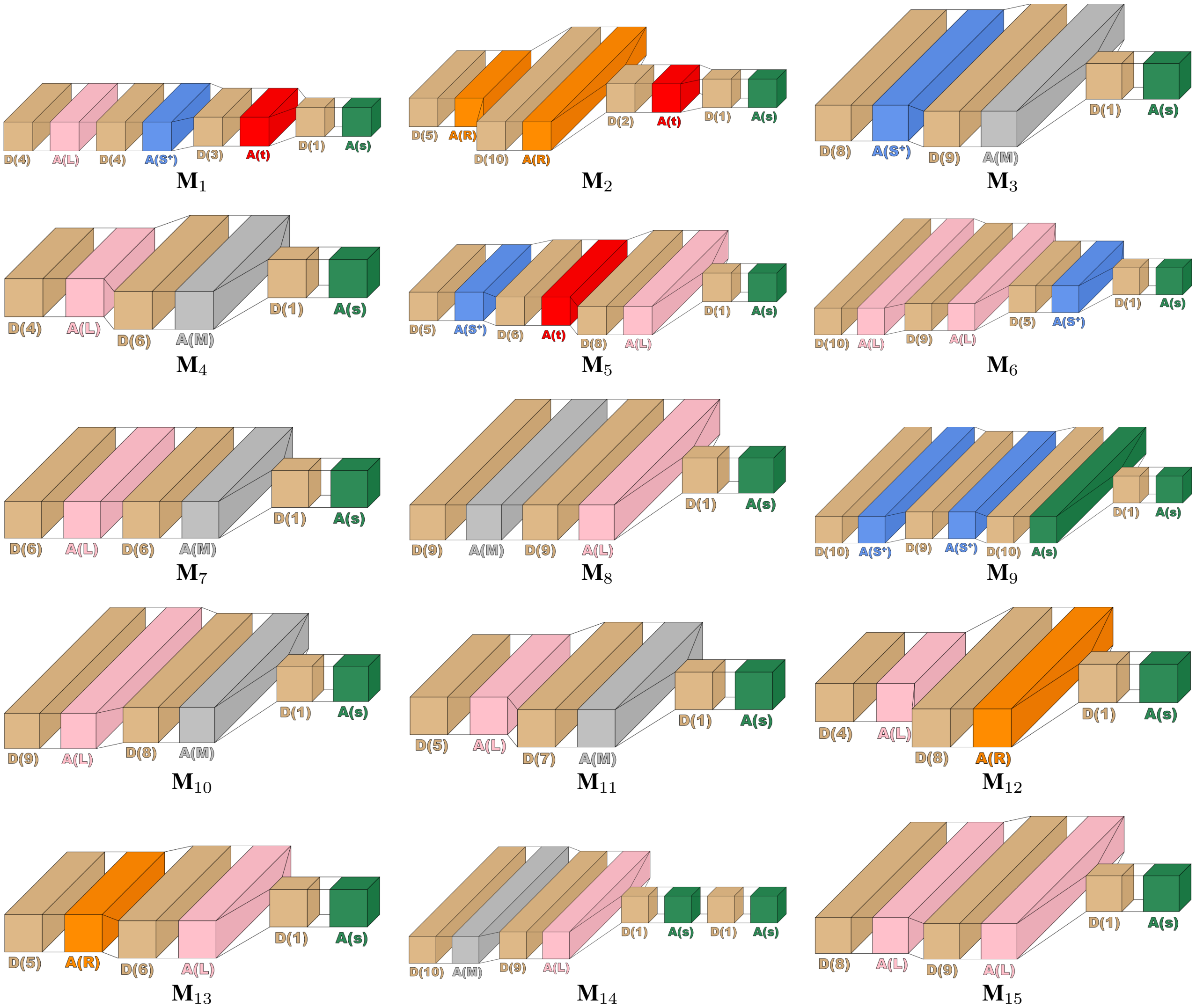
D(4) A(L) D(4) A(S⁺) D(3) A(t) D(1) A(s)
$\mathbf{M}_1$
D(5) A(R) D(10) A(R) D(2) A(t) D(1) A(s)
$\mathbf{M}_2$
D(8) A(S⁺) D(9) A(M) D(1) A(s)
$\mathbf{M}_3$
D(4) A(L) D(6) A(M) D(1) A(s)
$\mathbf{M}_4$
D(5) A(S⁺) D(6) A(t) D(8) A(L) D(1) A(s)
$\mathbf{M}_5$
D(10) A(L) D(9) A(L) D(5) A(S⁺) D(1) A(s)
$\mathbf{M}_6$
D(6) A(L) D(6) A(M) D(1) A(s)
$\mathbf{M}_7$
D(9) A(M) D(9) A(L) D(1) A(s)
$\mathbf{M}_8$
D(10) A(S⁺) D(9) A(S⁺) D(10) A(s) D(1) A(s)
$\mathbf{M}_9$
D(9) A(L) D(8) A(M) D(1) A(s)
$\mathbf{M}_{10}$
D(5) A(L) D(7) A(M) D(1) A(s)
$\mathbf{M}_{11}$
D(4) A(L) D(8) A(R) D(1) A(s)
$\mathbf{M}_{12}$
D(5) A(R) D(6) A(L) D(1) A(s)
$\mathbf{M}_{13}$
D(10) A(M) D(9) A(L) D(1) A(s) D(1) A(s)
$\mathbf{M}_{14}$
D(8) A(L) D(9) A(L) D(1) A(s)
$\mathbf{M}_{15}$

Fig. 4: Neural network architectures discovered via Neural Architecture Search (NAS). In the above diagrams, $D(x)$ denotes a dense/linear layer with x units/neurons. The blocks $A(t)$, $A(L)$, $A(S^+)$, $A(R)$, $A(M)$, and $A(s)$ represent the Hyperbolic Tangent (Tanh), Leaky version of a Rectified Linear Unit (LeakyReLU), Softplus, Rectified Linear Unit (ReLU), Mish [11], and Sigmoid activation functions, respectively.

with permanent Atrial Fibrillation (AF), 16 volunteers with Normal Sinus Rhythm (NSR), and 21 volunteers with other non-specified arrhythmias. The majority of these volunteers are older than 60 years, with a mean age of 66 years old and a median age of 70 years. The dataset used throughout this study was collected using a Samsung Galaxy Watch Active 2 at 25Hz.

We assessed the performance of the proposed architectures depicted in Figure 4 by comparing the found neural classifier models with SOTA ML models using the same proposed features. The performance was evaluated in terms of the Accuracy, Precision, Recall, and F-measure classification scores. Additionally, we defined

$$\Psi(z) = \frac{z}{\text{\# of parameters}} \times 100\%, \tag{5}$$

where $\{\Psi(z)\}$ is the "Metric-Parameters Efficiency Ratio (MPER)", a proposed way of evaluating the memory-performance trade-off of all models, and z is a classification metric, such as Accuracy, Precision, Recall, etc.

The performance of models illustrated in Figure 4 is shown in Table I. From this table, we can verify that these models surpass or achieve competitive performance in comparison with several SOTA algorithms which require significantly more parameters. Moreover, from Table I, it is possible to see that, for the Accuracy metric, the highest performing NAS model (M2) achieves 0.9724, very close to the top-performing baseline BaggingModel, while using about 17000x fewer parameters. For Precision and Recall, the top-performing methods are NAS Models 15 and 8, respectively. For the f1-Score, the highest performing NAS model (M2) achieves a performance of 0.9811, comparable to the top baseline Ex-

TABLE I: Performance metrics for NAS top-performing candidates and baseline comparison methods.

	Model	Accuracy	Precision	Recall	F1-Score	Size (kB)	Parameters	Ψ(Accuracy)
Proposed neural models	**M1**	0.9704	0.9779	0.9816	0.9798	18	79	12.284
	M2	0.9724	0.9776	0.9846	0.9811	18	135	7.203
	M3	0.9703	0.9780	0.9813	0.9796	16	171	5.674
	M4	0.9695	0.9764	0.9819	0.9791	16	77	12.591
	M5	0.9696	0.9739	0.9848	0.9793	18	151	6.421
	M6	0.9704	0.9765	0.9831	0.9798	18	255	3.805
	M7	0.9701	0.9730	**0.9863**	0.9796	16	109	8.900
	M8	0.9699	0.9779	0.9809	0.9794	16	190	5.105
	M9	0.9714	0.9790	0.9818	0.9804	18	310	3.134
	M10	0.9702	0.9784	0.9807	0.9796	15	179	5.420
	M11	0.9708	0.9746	0.9856	0.9801	16	100	9.708
	M12	0.9701	0.9771	0.9820	0.9795	16	89	10.900
	M13	0.9696	0.9775	0.9809	0.9792	16	93	10.426
	M14	0.9707	0.9759	0.9840	0.9800	18	211	4.600
	M15	0.9698	**0.9800**	0.9785	0.9794	16	171	5.671
SOTA ML models	AdaBoost	0.9695	0.9756	0.9827	0.9791	29	29376	0.033
	Bagging	**0.9727**	0.9781	0.9846	0.9813	2317	2371708	0.000410
	Extra Trees	0.9732	0.9772	**0.9863**	**0.9817**	107763	110349210	0.00000882
	GBM	0.9731	0.9769	**0.9863**	0.9816	172	175763	0.00554
	KNN	0.9580	0.9629	0.9802	0.9715	9201	9420890	0.000102
	LDA	0.9606	0.9786	0.9671	0.9728	2	1662	0.578
	QDA	0.9569	0.9799	0.9605	0.9701	3	2732	0.350
	RFC	0.9724	0.9769	0.9854	0.9811	2833	2900430	0.000335
	SVC	0.9469	0.9640	0.9632	0.9636	1206	1234473	0.000767

TABLE II: Performance comparison of the top-performing found model (M2) and state-of-the-art using ICON dataset.

Method	Accuracy	Precision	Recall	F1-Score
Heartpy [13]	0.6611	0.7479	0.8507	0.7960
Elgendi [14]	0.5403	0.9407	0.5593	0.7015
Li et. al. [5]	0.6157	0.7663	0.7581	0.7622
Heo et. al. [15]	0.4388	0.9559	0.4479	0.6100
M2 (Ours)	**0.9724**	**0.9776**	**0.9846**	**0.9811**

traTreesModel, while using about 800000x fewer parameters. From the MPER column, it is possible to verify that the proposed neural models perform much better in many orders of magnitude higher than all the SOTA models. As shown in Table II, M2 also outperforms several state-of-the-art peak detection methods, as well as traditional ones.

IV. Conclusions

This paper describes an approach for generating efficient neural network architectures to classify whether a given peak of PPG signals is systolic or not. The difference between two consecutive truly systolic peaks corresponds to a IBI measurement. Using NAS-based techniques, we found a set of architectures that are tiny enough to be deployed in embedded devices, enabling real-time wearable applications for cardiac monitoring. Therefore, the main goal of this paper is to report the discovered neural models to the biomedical engineering and signal processing communities to help the advancement of continuous cardiac monitoring solutions running on embedded devices such as smartphones and smartwatches.

References

[1] T. Page, “A forecast of the adoption of wearable technology,” *International Journal of Technology Diffusion*, vol. 6, no. 2, pp. 12–29, 2015.

[2] A. Aygun, H. Ghasemzadeh, and R. Jafari, “Robust interbeat interval and heart rate variability estimation method from various morphological features using wearable sensors,” *IEEE Journal of Biomedical and Health Informatics*, vol. 24, no. 8, pp. 2238–2250, 2019.

[3] C. J. McAloon, L. M. Boylan, T. Hamborg, N. Stallard, F. Osman, P. B. Lim, and S. A. Hayat, “The changing face of cardiovascular disease 2000–2012: An analysis of the world health organisation global health estimates data,” *International journal of cardiology*, vol. 224, pp. 256–264, 2016.

[4] T. Vandenberk, J. Stans, C. Mortelmans, R. Van Haelst, G. Van Schelvergem, C. Pelckmans, C. J. Smeets, D. Lanssens, H. De Cannière, V. Storms *et al.*, “Clinical validation of heart rate apps: mixed-methods evaluation study,” *JMIR mHealth and uHealth*, vol. 5, no. 8, p. e7254, 2017.

[5] B. N. Li, M. C. Dong, and M. I. Vai, “On an automatic delineator for arterial blood pressure waveforms,” *Biomedical Signal Processing and Control*, vol. 5, no. 1, pp. 76–81, 2010.

[6] V. S. Tallapragada, D. V. Reddy, K. S. Varma, and G. Sarma, “Improved atrial fibrillation detection using cnn-lstm,” in *ICOEI 2022*. IEEE, 2022, pp. 1050–1055.

[7] T. Elsken, J. H. Metzen, and F. Hutter, “Neural architecture search: A survey,” *J. Mach. Learn. Res.*, vol. 20, pp. 55:1–55:21, 2019.

[8] A. Khalak and M. C. Wiggins, “Method for denoising and data fusion of biophysiological rate features into a single rate estimate,” Aug. 4 2016, uS Patent App. 14/924,565.

[9] S.-H. Liu, R.-X. Li, J.-J. Wang, W. Chen, and C.-H. Su, “Classification of photoplethysmographic signal quality with deep convolution neural networks for accurate measurement of cardiac stroke volume,” *Applied Sciences*, vol. 10, no. 13, p. 4612, 2020.

[10] G. Lucafo, P. G. Freitas, R. Lima, G. Luz, R. Bispo, P. Rodrigues, F. Cabello, and O. Penatti, “Signal quality assessment of photoplethysmogram signals using hybrid rule- and learning-based models,” in *CBIS 2022*.

[11] D. Misra, “Mish: A self regularized non-monotonic neural activation function,” *arXiv preprint arXiv:1908.08681*, 2019.

[12] S. Hochreiter and J. Schmidhuber, “Long short-term memory,” *Neural computation*, vol. 9, no. 8, pp. 1735–1780, 1997.

[13] P. van Gent, H. Farah, N. van Nes, and B. van Arem, “Heartpy: A novel heart rate algorithm for the analysis of noisy signals,” *Transportation research part F: traffic psychology and behaviour*, vol. 66, pp. 368–378, 2019.

[14] M. Elgendi, I. Norton, M. Brearley, D. Abbott, and D. Schuurmans, “Systolic peak detection in acceleration photoplethysmograms measured from emergency responders in tropical conditions,” *PLoS One*, vol. 8, no. 10, p. e76585, 2013.

[15] S. Heo, S. Kwon, and J. Lee, “Stress detection with single ppg sensor by orchestrating multiple denoising and peak-detecting methods,” *IEEE Access*, vol. 9, pp. 47 777–47 785, 2021.