Optimization and Deployment of Deep Neural Networks for PPG-based Blood Pressure Estimation Targeting Low-power Wearables

Optimization and Deployment of Deep Neural Networks for PPG-based Blood Pressure Estimation Targeting Low-power Wearables

Alessio Burrello*‡, Francesco Carlucci*, Giovanni Pollo*, Xiaying Wang†, Massimo Poncino*, Enrico Macii*, Luca Benini†‡, Daniele Jahier Pagliari*

*Dept. DAUIN, Politecnico of Turin, Italy †Dept. ITET, ETH Zurich, Switzerland ‡DEI, University of Bologna, Italy

***Abstract*—PPG-based Blood Pressure (BP) estimation is a challenging biosignal processing task for low-power devices such as wearables. State-of-the-art Deep Neural Networks (DNNs) trained for this task implement either a PPG-to-BP signal-to-signal reconstruction or a scalar BP value regression and have been shown to outperform classic methods on the largest and most complex public datasets. However, these models often require excessive parameter storage or computational effort for wearable deployment, exceeding the available memory or incurring too high latency and energy consumption. In this work, we describe a fully-automated DNN design pipeline, encompassing HW-aware Neural Architecture Search (NAS) and Quantization, thanks to which we derive accurate yet lightweight models, that can be deployed on an ultra-low-power multicore System-on-Chip (SoC), GAP8. Starting from both regression and signal-to-signal state-of-the-art models on four public datasets, we obtain optimized versions that achieve up to 4.99% lower error or 73.36% lower size at iso-error. Noteworthy, while the most accurate SoA network on the largest dataset can not fit the GAP8 memory, all our optimized models can; our most accurate DNN consumes as little as 0.37 mJ while reaching the lowest MAE of 8.08 on Diastolic BP estimation.**

***Index Terms*—PPG, Neural Architecture Search, Blood Pressure, DNN**

I. Introduction and Related Works

Blood pressure (BP) is a crucial health parameter that necessitates continuous monitoring for a large population of vulnerable individuals, being linked to various heart-related diseases, such as hypertension, cardiomyopathy, and heart failure [1]. Various monitoring solutions exist, from cuffless to invasive procedures, but wearable technologies such as smartwatches would enable non-invasive monitoring of larger coorts of individuals at an affordable cost and without affecting their normal lives, thereby contributing to saving many lives.. In this domain, one of the most common monitoring techniques relies on Photoplethysmography (PPG).

PPG uses a light-emitting diode (LED) to illuminate the skin. A photodiode then collects the reflected light, whose intensity depends on the blood volume variation due to heart activity [2]. While various medically relevant parameters can be derived from PPG, including heart rate (HR) and respiratory rate, this paper focuses on its usage for the estimation of Systolic Blood Pressure (SBP) and Diastolic Blood Pressure (DBP), reflecting the blood pressure during and in between heart muscle contractions, respectively.

Corresponding email: alessio.burrello@polito.it

A broad set of machine learning techniques have been employed in the literature for this task, ranging from classical methods like Random Forest (RF) [3], and Support Vector Regression (SVR) [4] to Deep Neural Networks (DNNs) [5], [6]. Further, ensemble learning frameworks have been proposed to reduce the risk of overfitting [4], [7]. In comparison to classic methods, DNNs offer the advantage of not needing an often expensive feature extraction process and have been shown to generalize better on unseen data in several biosignal processing tasks [8]–[10]. Several DNN architectures have been considered for PPG-based BP estimation [5], [6], [11], [12], with most recent works focusing on 1D Convolutional Neural Networks (CNNs) [10]. Some works train these networks as regressors to directly predict a scalar DBP or SBP value based on a time window of PPG readings. ResNet-like networks achieve state-of-the-art performance in this category [13]. Others adopt a signal-to-signal (sig2sig) approach, where the CNN is tasked to reconstruct the entire DPB/SBP time series starting from the PPG one. In this group, architectures based on UNet [14] are the best-performing ones.

However, existing deep learning models for BP estimation have a large number of parameters and high computational complexity. When pursuing continuous monitoring on resource-constrained, low-power devices such as wearables, those models either exceed the available memory or incur excessive latency and energy consumption.

This paper attempts to mitigate this issue through the use of a fully automated DNN design pipeline, which encompasses two main optimization steps, i.e., Neural Architecture Search (NAS) [15] and Quantization [16]. Starting from state-of-the-art regression and sig2sig CNNs, we first apply a gradient-based NAS to automatically select each layer's operation from a pool, and tune the network depth, discovering architectures that balance BP prediction error and model size. We then select a subset of the Pareto-optimal DNNs identified by the NAS and quantize them to `int8` precision to further reduce their size, latency, and energy consumption. Lastly, we employ a DNN compiler [17] to automatically convert the quantized models to optimized C code, targeting GreenWaves' GAP8 [18], an ultra-low-power System-on-Chip (SoC) suitable to be embedded in a wearable device for practical, efficient and continuous BP monitoring. To the best of our knowledge, ours is the first work

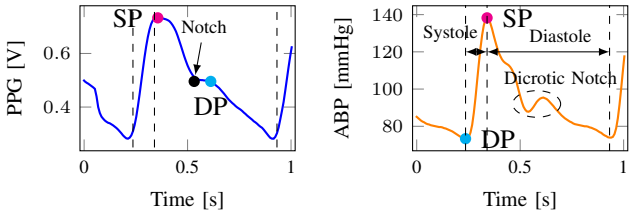

Fig. 1. SBP and DBP estimation from PPG (left) and ABP (right) signals.

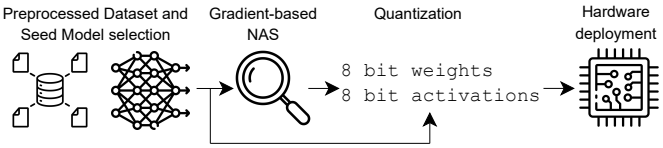

Fig. 2. Overview of the proposed automated DNN optimization flow.

to apply a cost-aware NAS for BP estimation DNNs and to consider their deployment on wearable-class devices.

With experiments on four publicly available datasets [19]–[22], we obtain models that reduce the BP estimation error by up to 4.99% with respect to the best state-of-the-art DNN, or maintain the same error with up to 73.36% fewer parameters. Furthermore, on the most complex dataset, UCI [22], we also outperform classic ML methods, obtaining a Mean Absolute Error (MAE) of 7.86 (versus 8.07), while also using fewer parameters. When deployed on GAP8, our models require 7.12-8.91ms per inference while consuming 0.36-0.45 mJ.

II. Materials & Methods

A. Blood Pressure Estimation using PPG Signal

Blood pressure monitoring techniques can be continuous or intermittent and invasive or non-invasive. The invasive monitoring, usually performed through an intra-arterial catheter [23], directly measures the atrial blood pressure signal (ABP). Common cuff-based methods like sphygmomanometer, although being gold standard and minimally invasive, are cumbersome and don't allow continuous monitoring.

On the other hand, PPG optical signal is strongly related to changes in blood volumes, and its effectiveness is already proven in various clinical applications. Although PPG is morphologically similar to ABP, and various studies have shown how the two signals share most of the informative features [24], extracting a blood pressure estimation from it is not a trivial task, given that the signal is subject to artifacts related to movements or air between the sensor and the skin. Fig. 1 shows examples of clean PPG and ABP signals with the points corresponding to systolic and diastolic blood pressure marked on the two plots.

B. Datasets

In our study, we adopt the same four datasets, data preprocessing, and training protocols as the extensive survey in [25], which is state-of-the-art for this task. All datasets are resampled to 125 Hz. PPGBP [20] is the smallest dataset, with 619 total PPG segments, each lasting 2.1s, but it involves a large number of patients (218) with different cardiovascular diseases. BCG [19] is a bed-based ballistocardiography dataset comprising around 4 hours of cumulative measurements on 40 individuals, split into 5s windows. Sensors [21] is a subset of the MIMIC III dataset, comprising 11102 non-overlapping 5s data segments from 1195 patients. Lastly, UCI [22] is a subset of the MIMIC II waveform. It's considerably bigger than all the others, with ≈411k segments from an unknown number of subjects. All datasets include only measurements on resting patients in a clinical setting. Therefore, motion artifact removal using acceleration data [8], [26] can be neglected. Along with PPG signals, BCG, UCI, and Sensors provide the complete blood pressure time series as ground truth for prediction. PPGBP, instead, only includes two scalar values (SBP and DBP) per sample. Thus, sig2sig models cannot be trained on this dataset. All model performances are evaluated using the test set MAE on SBP and DBP separately. The training protocol uses a 5-fold per-subject Cross-Validation for all datasets except UCI. Given its size, single-held-out validation and test sets are adopted for UCI.

Notably, cross-patient inference following these protocols yields significantly higher estimation errors than medical-grade device requirements, which can only be reached through personalized fine-tuning [27]. However, this additional training is orthogonal to our work, which aims to demonstrate the feasibility of deploying efficient DNNs for BP estimation onboard wearable hardware.

C. Network Optimization

To optimize our DNNs, we leverage the open-source library PLiNIO [28], which provides an easy-to-use interface to implement various NAS and Quantization algorithms. A scheme of the optimization steps is shown in Fig. 2. Together with the training dataset, preprocessed and windowed as discussed above, the other key input of the pipeline is a *seed* network, i.e., an initial DNN, which serves as a blueprint to generate optimized models. We use the two best-performing DNNs from [25] as seeds. Both are 1D CNNs, but while the first belongs to the scalar regressors category and is derived from a ResNet [13], the second is a UNet-like [14] sig2sig model. We refer the reader to [25] for further details on the seeds.

Notably, these two architectures have already been optimized in [25] for each dataset, albeit only for maximizing accuracy. In contrast, in our work, we perform *cost-aware* optimizations, showing that this permits us to find similarly or better performing models, which are additionally smaller and more efficient.

1) NAS: For each seed, the first optimization step consists of the application of a gradient-based NAS called *SuperNet*, inspired from [29], whose working principle is depicted in Fig. 3. This method replaces each convolutional layer in the seeds with a pool of alternatives, all receiving the same input. The output of each layer is obtained as a linear combination of the various alternatives' outputs, weighted by softmax-ed trainable parameters θ_i (Fig. 3b). Intuitively, finding a good architecture corresponds to setting, for each layer, one of the $\theta_i = 1$ (and the others $= 0$). The NAS solves a continuous relaxation of this problem by inserting the multi-path DNN

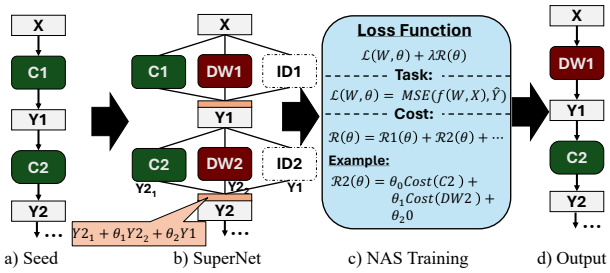

Fig. 3. SuperNet-based NAS.

in a standard training loop, where both the normal network weights W and the newly added θ are optimized jointly by gradient descent. This training uses the modified loss function shown in Fig. 3c, where $\mathcal{L}$ is the standard task loss, i.e., in our case, the Mean Squared Error (MSE) between the network's output $f(W,X)$ and the ground truth $\hat{Y}$. The newly added term $\mathcal{R}$, instead, is the *expected cost* of the network as a function of the layer selection parameters. An example of its calculation is shown in Fig. 3c. In this work, we use *model size* as a cost metric. At the end of the training, the output architecture is generated by selecting, for each layer, the alternative associated with the largest θ_i. Varying the scalar *regularization strength* λ, which controls the balance between the two loss terms, allows the generation of multiple output DNNs with different error vs cost trade-offs.

In our work, we use the SuperNet to select, for each layer, between a standard 1D convolution (C), a Depthwise-separable block (DW), and an identity operation (ID). The original models only include standard convolutions. The DW block, made of a sequence of a depthwise convolution and a pointwise layer, was first made popular by [30], and has been shown to provide a lower-size yet similar-accuracy approximation of standard convolutions, leading to tiny yet capable networks. The ID, instead, is added only when input and output tensors have the same shape, and lets the NAS modulate the network depth by skipping some layers.

2) Quantization: In a second optimization step, we select some of the Pareto-optimal DNNs generated by the NAS and quantize them to `int8` format. For this, we use PLiNIO's Quantization-Aware Training (QAT) capabilities [16]. We use a standard min-max affine quantization format for weights and the Parametrized Clipping Activation (PaCT) method for layer's inputs and outputs [31]. Accumulation and biases are on 32 bits, as supported by our target inference library [32].

Note that the adopted NAS and quantization methods are not new per se. However, to our knowledge, we are the first to apply HW-aware optimizations for BP estimation.

D. Network Deployment

We deploy our networks on the GreenWaves GAP8 [18], a low-power, RISC-V-based multi-core IoT processor designed specifically for signal processing tasks on edge devices. GAP8 features a cluster of eight general-purpose cores used to accelerate compute-intense workloads. It also includes a 2-level scratchpad memory, with 512 kB of main memory, used to store the application code and DNN weights, and a 64 kB last-level cache with single-clock access latency for the cluster. A DMA engine moves the data between memory levels.

To convert our optimized DNNs into inference code for GAP8, we employ the DORY compiler [17]. DORY automatically generates C code that handles the entire inference process, including memory management, DMA transfers scheduling, and optimized AI primitives invocation. It can directly take as input quantized DNNs generated by PLiNIO. As backend library for implementing each layer, we use [32]. We profile our deployed models on the GAP8 evaluation board, utilizing the internal performance counters for measuring latency, and the Nordic Power Profiler Kit II for power [33].

III. Results

We performed all trainings using the Adam optimizer with a learning rate of 0.001 for the network weights and a separate Adam optimizer with a learning rate of 0.01 for the NAS parameters (θ). In each epoch, the network weights were optimized on the training set, while the NAS parameters were optimized on the validation set, as in [29]. For all datasets and both ResNet and UNet seeds, we tested 18 different values of λ, evenly spaced on a log scale between 10^{-11} and 10^{-7}. We compared our optimized models with the original ResNet and UNet from [25], as well as with two classic model families also considered in that paper, namely Random Forests (RFs) and Support Vector Regressors (SVR).

A. Pareto Analysis

Fig. 4 depicts the results of NAS on all four datasets and for DBP and SBP prediction. All results are reported in a MAE vs model size (n. of parameters) plane. Red and green diamonds correspond to the optimization seeds (ResNet and UNet from [25] respectively). Correspondingly colored circles are the Pareto-optimal architectures found with NAS. All results refer to floating point DNNs, before quantization.

On all datasets, we obtain models that either dominate the seeds or are on the memory vs error Pareto front. On the smallest one, PPGBP, we obtain a rich curve of Pareto architectures starting from the ResNet. We are able to reduce the seed size by 16%, with a small increase in the MAE of only 3.9% and 3.5% on DBP and SBP prediction, respectively. As mentioned, we do not have results with the UNet seed for PPGBP, given that this dataset does not include the full BP signal ground truth. On BCG, we Pareto-dominate both seed networks, improving both MAE and size. Our best UNet-derived model obtains 11.139 mmHg MAE on SBP prediction and 7.52 mmHg MAE on DBP, being 6.7%/4.7% better than the best seed (ResNet). Simultaneously, this network reduces the total number of parameters by 3.8×. However, it shall be noted that for these two datasets, classical ML methods still outperform our optimized DNNs in both performance and size, as reported in [25]. SVR achieves the lowest MAE in DBP estimation for both PPGBP and BCG datasets (8.04 and 7.34 mmHg, respectively) and a MAE of 13.15 mmHg and 11.45 mmHg on SBP estimation, being outperformed by the UNet, solely on BCG.

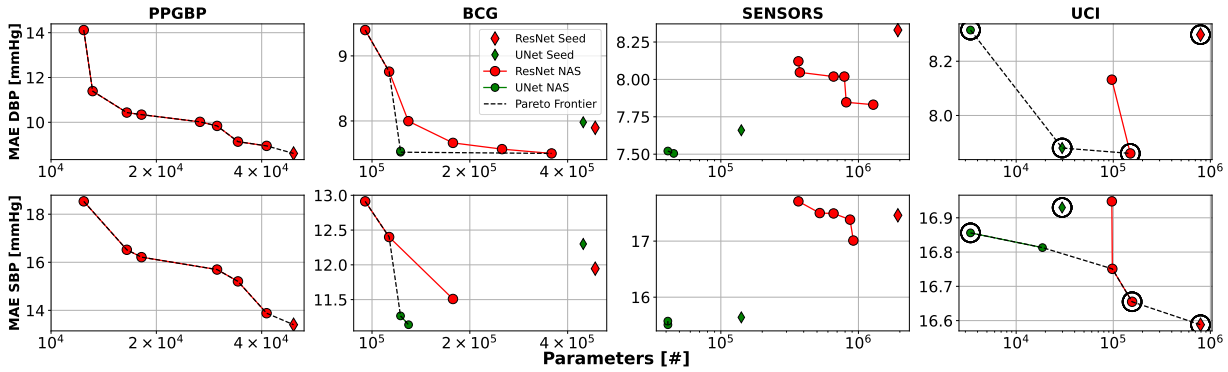

Fig. 4. NAS results on all datasets and on DBP and SBP prediction.

While being interesting for small datasets, classic ML models fail to benefit from the availability of larger amounts of data. On the second largest dataset, Sensors, classical ML methods have slightly better performance than the seeds, but inferior to our NAS-optimized DNNs. Namely, SVR, which achieved the best results on both metrics (15.60 mmHg for SBP and 7.50 mmHg for DBP), is now outranked by our UNet NAS models (15.51 mmHg for SBP and the same DBP) with a parameters reduction up to 40×. On UCI, the dataset with the most samples, classic methods are outperformed even by the seeds, as shown in [25]; the best one (a RF) achieves a SBP MAE of 16.85 mmHg (versus 16.59 mmHg of the ResNet), while SVR is outperformed by UNet, with a DBP MAE of 8.07 vs 7.88 mmHg respectively. Moreover, the higher complexity of these datasets causes the number of parameters of both the SVR (with RBF kernel) and of the RF to increase exponentially. For instance, on UCI, the SVR becomes 998× larger than our best NAS output.

Conversely, on these two larger datasets, thanks to our NAS, we are again able to obtain Pareto-dominant solutions. On Sensors, our UNet-derived architectures reduce the size of the most accurate seed (UNet) by 3.4×, while achieving a similar or lower MAE of 7.51 mmHg / 15.51 mmHg on DBP/SBP, respectively. Interestingly, on BCG and Sensors, Unet-based architectures outperform ResNets. We attribute this behavior to the ability of this network topology to learn faster from a lower amount of data, thanks to the richer training signal provided by the full time series reconstruction task. The situation reverses in UCI, where ResNet-derived DNNs achieve the best performance. The most accurate networks found with our NAS on UCI require only 149.8k/156.3k parameters to achieve a close-to-optimal MAE of 16.655 mmHg on SBP estimation, and the lowest overall (7.86 mmHg) on DBP estimation. While the seed ResNet is able to achieve an even lower MAE on SBP, with its 792k parameters, it would be impossible to deploy on GAP8's internal memory of 512KB, even when quantized.

B. Quantization & Deployment

For the sake of space, we report our deployment results only on the largest and most challenging dataset, UCI. We quantize and deploy the DNNs marked with black circles in Fig. 4. Namely, the two seeds, and the NAS outputs at the extremes of the Pareto front. All the results are reported in Table I, where *ResNet-B* and *ResNet-S* are the biggest NAS models on the Pareto front of the SBP and DBP graphs, respectively; *UNet-S* is the smallest Pareto model, which has the same architecture for both SBP and DBP estimation. The table reports the error, size, latency (Lat.), and energy per inference (E.) for each model. For reference, the MAE and size of the seeds in floating point are also reported, although these models are not deployable on the FPU-less GAP8. The size and latency reductions, thanks to quantization, are paid with a slight increase in MAE (up to 9.8%). ResNet models tend to be more susceptible to this degradation. After quantization and deployment, the best results are achieved by the Resnet-S, which achieves an 8.08 mmHg MAE on DBP estimation, and by the UNet-S, achieving 17.2 mmHg of MAE on SBP estimation. Due to the too-high number of parameters, the seed ResNet can not be deployed on GAP8's onboard memory. On the other hand, all NAS produced models fit the platform. Compared to the seed UNet, we achieve a similar latency and energy consumption; the UNet-S model, which achieves a latency of 8.91 ms with an energy consumption of as low as 0.45 mJ, is indeed made mostly of DW layers, which are smaller but also less efficient when deployed, reducing the memory occupation at the cost of a limited increase in latency.

TABLE I
DEPLOYMENT RESULTS ON GAP8.

Model	MAE-SBP	MAE-DBP	Size [B]	Lat. [ms]	E. [mJ]
Floating Point Models (fp32)					
ResNet	16.59	8.3	3.17M	n.a.	n.a.
UNet	16.93	7.88	118.9k	n.a.	n.a.
Quantized Models (int8)					
ResNet	18.23	8.17	791.8k	o.o.m.	o.o.m.
UNet	17.63	8.19	29.8k	7.04	0.36
ResNet-B	17.83	8.44	156.3k	7.12	0.36
Resnet-S	17.48	**8.08**	149.8k	7.27	0.37
UNet-S	**17.2**	8.26	23.4k	8.91	0.45

IV. Conclusion

The efficient execution of PPG-based BP estimation algorithms is critical for the prevention of important diseases associated, for instance, to hypertension. With our experiments, we demonstrated the possibility of embedding accurate DNN models on low-power wearable-class devices, achieving SoA performance. Future work will focus on the fine-tuning of our models on patient-specific data to reach competitive accuracy with the golden standard of non-intrusive PB.

REFERENCES

[1] F. D. Fuchs *et al.*, "High blood pressure and cardiovascular disease," *Hypertension*, vol. 75, pp. 285–292, 2 2020.

[2] G. Joseph *et al.*, "Photoplethysmogram (ppg) signal analysis and wavelet de-noising," in *2014 Annual International Conference on Emerging Research Areas: Magnetics, Machines and Drives (AICERA/iCMMD)*, 2014, pp. 1–5.

[3] R. He *et al.*, "Beat-to-beat ambulatory blood pressure estimation based on random forest," in *2016 IEEE 13th International Conference on Wearable and Implantable Body Sensor Networks (BSN)*, 2016, pp. 194–198.

[4] M. W. K. Fong *et al.*, "Svr ensemble-based continuous blood pressure prediction using multi-channel photoplethysmogram," *Computers in Biology and Medicine*, vol. 113, p. 103392, 10 2019.

[5] C. El Hajj *et al.*, "Cuffless and continuous blood pressure estimation from ppg signals using recurrent neural networks," in *2020 42nd Annual International Conference of the IEEE Engineering in Medicine & Biology Society (EMBC)*, 2020, pp. 4269–4272.

[6] J. Cheng *et al.*, "Prediction of arterial blood pressure waveforms from photoplethysmogram signals via fully convolutional neural networks," *Computers in Biology and Medicine*, vol. 138, p. 104877, 11 2021.

[7] J. Pan *et al.*, "Improved blood pressure estimation using photoplethysmography based on ensemble method," in *2017 14th International Symposium on Pervasive Systems, Algorithms and Networks & 2017 11th International Conference on Frontier of Computer Science and Technology & 2017 Third International Symposium of Creative Computing (ISPAN-FCST-ISCC)*, 2017, pp. 105–111.

[8] A. Burrello *et al.*, "Q-ppg: Energy-efficient ppg-based heart rate monitoring on wearable devices," *IEEE Transactions on Biomedical Circuits and Systems*, vol. 15, no. 6, pp. 1196–1209, 2021.

[9] A. Burrello *et al.*, "Bioformers: Embedding transformers for ultra-low power semg-based gesture recognition," 2022.

[10] K. Zhou *et al.*, "Methods for continuous blood pressure estimation using temporal convolutional neural networks and ensemble empirical mode decomposition," *Electronics*, vol. 11, no. 9, 2022. [Online]. Available: https://www.mdpi.com/2079-9292/11/9/1378

[11] B. Huang *et al.*, "Mlp-bp: A novel framework for cuffless blood pressure measurement with ppg and ecg signals based on mlp-mixer neural networks," *Biomedical Signal Processing and Control*, vol. 73, p. 103404, 3 2022.

[12] N. F. Ali *et al.*, "An efficient hybrid lstm-ann joint classification-regression model for ppg based blood pressure monitoring," *Biomedical Signal Processing and Control*, vol. 84, p. 104782, 7 2023.

[13] C. Qin *et al.*, "Cuff-less blood pressure prediction based on photoplethysmography and modified resnet," *Bioengineering*, vol. 10, no. 4, 2023. [Online]. Available: https://www.mdpi.com/2306-5354/10/4/400

[14] O. Ronneberger *et al.*, "U-net: Convolutional networks for biomedical image segmentation," 2015.

[15] P. Ren *et al.*, "A comprehensive survey of neural architecture search: Challenges and solutions," *ACM Comput. Surv.*, vol. 54, no. 4, may 2021. [Online]. Available: https://doi.org/10.1145/3447582

[16] B. Jacob *et al.*, "Quantization and training of neural networks for efficient integer-arithmetic-only inference," *CoRR*, vol. abs/1712.05877, 2017. [Online]. Available: http://arxiv.org/abs/1712.05877

[17] A. Burrello *et al.*, "DORY: automatic end-to-end deployment of real-world dnns on low-cost iot mcus," *CoRR*, vol. abs/2008.07127, 2020. [Online]. Available: https://arxiv.org/abs/2008.07127

[18] G. Technologies, "Gap8, https://greenwaves-technologies.com/gap8_mcu_ai/," Access 16/05/2024.

[19] C. Carlson *et al.*, "Bed-based ballistocardiography: Dataset and ability to track cardiovascular parameters," *Sensors*, vol. 21, no. 1, 2021. [Online]. Available: https://www.mdpi.com/1424-8220/21/1/156

[20] Y. Liang *et al.*, "A new, short-recorded photoplethysmogram dataset for blood pressure monitoring in china," *Scientific Data*, vol. 5, 2 2018.

[21] N. Aguirre *et al.*, "Blood pressure morphology assessment from photoplethysmogram and demographic information using deep learning with attention mechanism," *Sensors*, vol. 21, no. 6, 2021.

[22] M. Kachuee *et al.*, "Cuff-less high-accuracy calibration-free blood pressure estimation using pulse transit time," in *2015 IEEE International Symposium on Circuits and Systems (ISCAS)*, 2015, pp. 1006–1009.

[23] A. B. Casabianca *et al.*, "Cardiovascular monitoring: Physiological and technical considerations," *Anesthesia Progress*, vol. 56, no. 2, p. 53, 2009.

[24] G. Martínez *et al.*, "Can photoplethysmography replace arterial blood pressure in the assessment of blood pressure?" *Journal of Clinical Medicine*, vol. 7, no. 10, 2018. [Online]. Available: https://www.mdpi.com/2077-0383/7/10/316

[25] S. González *et al.*, "A benchmark for machine-learning based non-invasive blood pressure estimation using photoplethysmogram," *Scientific Data 2023 10:1*, vol. 10, pp. 1–16, 3 2023. [Online]. Available: https://www.nature.com/articles/s41597-023-02020-6

[26] S. B. Song *et al.*, "Nas-ppg: Ppg-based heart rate estimation using neural architecture search," *IEEE Sensors Journal*, vol. 21, no. 13, pp. 14 941–14 949, 2021.

[27] T. P. Almeida *et al.*, "Aktiia cuffless blood pressure monitor yields equivalent daytime blood pressure measurements compared to a 24-h ambulatory blood pressure monitor: Preliminary results from a prospective single-center study," *Hypertension Research*, vol. 46, no. 6, pp. 1456–1461, Jun. 2023.

[28] D. Jahier Pagliari *et al.*, "Plinio: A user-friendly library of gradient-based methods for complexity-aware dnn optimization," 2023.

[29] H. Liu *et al.*, "DARTS: differentiable architecture search," *CoRR*, vol. abs/1806.09055, 2018. [Online]. Available: http://arxiv.org/abs/1806.09055

[30] A. G. Howard *et al.*, "Mobilenets: Efficient convolutional neural networks for mobile vision applications," *CoRR*, vol. abs/1704.04861, 2017. [Online]. Available: http://arxiv.org/abs/1704.04861

[31] J. Choi *et al.*, "PACT: parameterized clipping activation for quantized neural networks," *CoRR*, vol. abs/1805.06085, 2018. [Online]. Available: http://arxiv.org/abs/1805.06085

[32] A. Garofalo *et al.*, "PULP-NN: accelerating quantized neural networks on parallel ultra-low-power RISC-V processors," *CoRR*, vol. abs/1908.11263, 2019. [Online]. Available: http://arxiv.org/abs/1908.11263

[33] N. Semiconductor, "Nordic ii, https://www.nordicsemi.com/," Access 16/05/2024.