Robust Real-Time Embedded EMG Recognition Framework Using Temporal Convolutional Networks on a Multicore IoT Processor

Robust Real-Time Embedded EMG Recognition Framework Using Temporal Convolutional Networks on a Multicore IoT Processor

Marcello Zanghieri, Simone Benatti, Alessio Burrello, Victor Kartsch, Francesco Conti, Luca Benini

***Abstract*—Hand movement classification via surface electromyographic (sEMG) signal is a well-established approach for advanced Human-Computer Interaction. However, sEMG movement recognition has to deal with the long-term reliability of sEMG-based control, limited by the variability affecting the sEMG signal. Embedded solutions are affected by a recognition accuracy drop over time that makes them unsuitable for reliable gesture controller design. In this paper, we present a complete wearable-class embedded system for robust sEMG-based gesture recognition, based on Temporal Convolutional Networks (TCNs). Firstly, we developed a novel TCN topology (TEMPONet), and we tested our solution on a benchmark dataset (Ninapro), achieving 49.6% average accuracy, 7.8%, better than current State-Of-the-Art (SoA). Moreover, we designed an energy-efficient embedded platform based on GAP8, a novel 8-core IoT processor. Using our embedded platform, we collected a second 20-sessions dataset to validate the system on a setup which is representative of the final deployment. We obtain 93.7% average accuracy with the TCN, comparable with a SoA SVM approach (91.1%). Finally, we profiled the performance of the network implemented on GAP8 by using an 8-bit quantization strategy to fit the memory constraint of the processor. We reach a $4\times$ lower memory footprint ($460\,\mathrm{kB}$) with a performance degradation of only 3% accuracy. We detailed the execution on the GAP8 platform, showing that the quantized network executes a single classification in $12.84\,\mathrm{ms}$ with a power envelope of $0.9\,\mathrm{mJ}$, making it suitable for a long-lifetime wearable deployment.**

I. Introduction

Decoding hand gestures is an established method for developing advanced Human-Machine Interfaces (HMIs), which leads to a wide range of application scenarios, such as industrial robot control, gaming interfaces, prosthetic control or augmented reality [1], [2]. In the HMI field, gesture recognition relies on the processing of information coming from video cameras [3] or from muscular activity [4]. Camera-based techniques rely on image processing algorithms which recognize users' hand in a scene and recognize different gestures using computer vision. Although this approach can decode a vast number of gestures reliably, it suffers from the line of sight issues and scene illumination variability, and it requires pre-installed environmental cameras. On the other hand, approaches based on muscular signal analysis are inspired by the prosthetics domain, where electromyographic signals are used to control artificial hands [5], [6]. Commercial prosthetic controllers are simple and highly reliable. However, they provide a non-natural interface, unsuitable for intuitive gesture interface design, because of the high level of concentration required by the user and the long learning curve.

New machine learning approaches have been extensively explored to enable the design of natural gesture interfaces. They aim to map muscular contraction patterns onto a set of intended gestures [7], using supervised learning methods such as SVM, LDA or ANN [8], [9], [10]. Such approaches reach accuracy above 80% on classifying several hand gestures (from 4 to 12), making them suitable for the design of human-machine interfaces.

Nevertheless, the EMG signal is affected by high variability caused by subjects' fatigue, perspiration, changes in the skin-to-electrode interface, user adaptation, and mostly by electrode shifts during multi-day usage [8], [11], [10]. These factors severely limit the long-term usage and the reliability of the EMG-based gesture recognition, leading to an inter-sessions accuracy drop of up to 30% [12]. While extending the training dataset and enhancing the algorithms with more features can help to reduce this drop [8], this is still too high ($> 10\%$) to make these approaches suitable for robust and commercially available systems.

A promising possible solution could come from using Deep Learning (DL) techniques. In recent years, they have been successfully proposed for biosignal application scenarios [13], and have achieved state-of-the-art accuracy on a wide range of applications such as activity recognition or neural diseases detection. The major advantage of the DL approaches is the removal of manually-extracted signal features. Indeed, a deep network automatically learns a good representation of the signal during the training step. This automatization is particularly useful for signals that are affected by very high variability like the sEMG; for this class of signals, the algorithm could learn a *signature* of the signal, which is not affected by the variability, but allows to characterize the muscle stimuli and differentiates multiple hand gestures. As a result, such an approach enables a better generalization on more massive datasets.

Most of the DL models are based on complex architectures with a high number of layers and neurons [13], [14], [15] to obtain high recognition accuracy and low performance drop on multi-day sessions. On the other hand, these models require a memory footprint of several MBytes, which is not suitable for a real-time deployment on low-power embedded

M. Zanghieri, S. Benatti, A. Burrello, V. Kartsch, F. Conti and L. Benini are with the Department of Electrical, Electronic and Information Engineering, University of Bologna, 40136 Bologna, Italy (e-mail: marcello.zanghieri2@unibo.it, simone.benatti@unibo.it, alessio.burrello@unibo.it, victorjavier.kartsch@unibo.it, f.conti@unibo.it, luca.benini@unibo.it).

F. Conti and L. Benini are also with the Department of Information Technology and Electrical Engineering at the ETH Zurich, 8092 Zurich, Switzerland.

platforms. Pursuing the creation of a DL reliable framework on an energy-efficient platform requires a multilevel approach with HW/SW codesign. In particular, the designer should *i)* develop a robust and high-accuracy network topology, *ii)* minimize the number of parameters of the algorithm through software DL optimizations and finally *iii)* couple the algorithm with the embedded optimization (e.g. quantization and memory management) of the firmware implementation on the digital platform.

In this work, we tackle the challenge of variability robustness of sEMG-based hand gesture recognition, and we propose a real-time embedded platform for robust sEMG-based gesture recognition. The major contributions we are proposing are:

- TEMPONet, a novel EMG classification algorithm based on a Temporal Convolutional Network (TCN), tested on a benchmark EMG dataset (NinaPro DB6);
- a complete embedded platform for EMG acquisition and processing. The system is based on the combination of a commercial Analog Front End for biopotential acquisition with GAP8, a multicore low-power IoT processor;
- a 20 session dataset, collected with our custom platform, which allows us to validate the algorithm and to profile a quantized version of the TCN, suitable for the deployment on a resource-constrained platform.

We tested the performance of TEMPONet on the NinaPro DB6 dataset [11], achieving 65.2% inter-session accuracy on steady signals and 49.6% inter-session on the full dataset – 7.8% better than the current state-of-the-art [16]. Moreover, after the system design, we tested the same TEMPONet topology on a new dataset we introduce in this work, comprising 20 sessions on three subjects. On this dataset, we achieve 93.7% inter-session accuracy. The 20-session dataset is collected using the same platform on which we deploy our detection algorithm. Therefore, it is representative of the real kind of data that an embedded setup can gather. Our results show that the accuracy drop on entirely unseen sessions can be reduced to 6.6% on NinaPro DB6 and 3.4% on the 20-session dataset, surpassing the current state-of-the-art.

Finally, we performed a full quantization of our TEMPONet, dropping the data representation of the weights and the feature maps from 32-bit floating point to 8-bit integer, thus reducing the network memory footprint by $4\times$. To leverage the 8-core architecture of the GAP8 processor and parallelize the execution of the algorithm, we used highly optimized neural network libraries [17]. The quantized TCN can be executed in real-time on the GAP8 chip (a full inference takes less than $13\,\mathrm{ms}$ and consumes $0.9\,\mathrm{mJ}$), but still achieves 93.3% inter-session accuracy on the 20-session dataset and 61.0% on the NinaPro DB6 while providing up to 54h of battery life, showing a computational efficiency of $10\times$ compared to SoA systems for sEMG processing, suitable for convolutional network deployment, such as [18], [19].

II. Background and Related Work

A. Background

1) sEMG Signal and Datasets

The electromyographic (EMG) signal [20], [21], [22] is the bioelectric potential originating from the current generated by the ionic flow through the membrane of the muscular fibers, and it is, therefore, a major index of the muscular activity. This potential is generated by the electrical stimulus starting from the central nervous system and passing through the motor neurons (motoneurons) that innervate the muscular tissue. Typically, the EMG signal has amplitude ranging from $10\,\mu\mathrm{V}$ to $10\,\mathrm{mV}$, and bandwidth $\sim 2\,\mathrm{kHz}$.

Moreover, it is a very challenging signal as it is affected by several noise sources, such as motion artifacts, floating ground noise, crosstalk, and Power Line Interference [23], [24].

EMG data can be acquired either with invasive or non-invasive methods. In this work, we focus on surface electromyography (sEMG), a non-invasive technique which uses electrodes operating on the surface of the skin. In the sEMG setup, the action potentials (APs) can be detected using an instrumentation amplifier with the positive and negative terminals connected to two metal plates positioned on the skin surface; the sEMG signal results from the superposition of all the detected APs underlying the amplifier [10]. In the Human-Machine Interface (HMI) field, building gesture recognition upon the analysis of sEMG signals is one of the most promising approaches, since non-invasiveness is an essential requirement for many types of interface.

The Non-Invasive Adaptive hand Prosthetics Database 6 (NinaPro DB6) [11] is a public sEMG database realized to investigate the repeatability of sEMG-based hand gesture (grasps) recognition over time. The data were collected from 10 intact (i.e. non-amputee) subjects (3 females, 7 males, average age 27 ± 6 years). The database consists of 10 sessions (5 days, two sessions a day: morning and afternoon), each involving 12 repetitions of 7 grasps, for all the 10 subjects. The grasps were selected from the robotics and rehabilitation literature, covering hand movements typical of Activities of Daily Living (ADL). The sEMG signals were measured with 14 Delsys Trigno sEMG Wireless electrodes, placed on the high half of the forearm, at sampling rate $2\,\mathrm{kHz}$. Each grasp repetition lasts approximately $6\,\mathrm{s}$, followed by $2\,\mathrm{s}$ of rest.

In addition to the analysis performed on NinaPro DB6, we also introduce a study on our dataset, which incorporates the effects of a real-life scenario in our embedded platform. This dataset targets hand gestures that are fully compatible with Human-Machine Interaction. To analyze the sEMG temporal variability and validate our results, we collected a 20-session dataset, using our custom 8-electrodes platform. The dataset is described in detail in Section IV-C.

2) Temporal Convolutional Networks

More recently, Temporal Convolutional Networks (TCNs) have been gaining attention for the analysis of time series [25], [26]. TCNs are a recently proposed class of sequential models able to learn the temporal dependencies of a given input signal. TCNs are the State-of-the-Art in many sequence modeling challenges, outperforming the more expensive and complicated Recurrent Neural Networks (RNNs) [25], [26], [14]. The novel and unique properties granting success to this kind of network are situated in its 1D-convolutional layers operating along the time dimension. These layers present two fundamental novelties:

(1) *causality*, which implies that each output y_{t_N} of the layer

only relies on convolutions of elements x_{t_I} with $t_I \leq t_N$. In this way, only the previous history of the signal is used to predict the label at time t_N.
(2) *dilation*: a fixed step d is introduced between the filter inputs. Using this approach, TCN layers present an increased receptive field with a reduced number of parameters (e.g., with $d = 4$ and filter size = 3, the receptive field is 9). This method allows to take into account a wide signal history without impairing the network with a too high number of parameters (increasing the filter size) or with many stacked layers (reducing the training efficiency). Thus, a convolutional layer of a TCN operates as:

$$\mathbf{y}_n^o = \text{Conv}\,(\mathbf{x}) = \sum_{l=0}^{L-1} \sum_{i=0}^{K-1} \mathbf{x}_{n-d\,i}^{l} \cdot \mathbf{W}_i^{l,m}$$

with $\mathbf{x}$ input feature map and $\mathbf{y}$ output feature map, n the time index, $\mathbf{W}$ the filter weights, L the number of input channels, m the output channel, d dilation, and K the filter size. The lower part of Figure 1 portraits how the presented dilated 1D-convolutional layers work on an input time series.

B. Related Work

In the last few years, several sEMG-based hand recognition approaches have been presented both in academia and in commercial applications. All of them share a typical structure, based on *i)* an analog front end for bio-potential acquisition, *ii)* a data preprocessing and feature extraction/selection step, and *iii)* a final classification back end. Moreover, they usually all rely on Machine Learning (ML) algorithms such as Support Vector Machine (SVM), Random Forest (RF), LDA or artificial neural networks (ANN) [8], [9], [27], [10], [13], [28].

For instance, in [29], [30], the authors presented a 4 hand gesture classification with accuracy above 90%, using ANN with 5 time-domain features (Mean Absolute Value (MAV), Mean Absolute Value Slope (MAVS), number of Slope Sign Changes (SSC), number of zero crossings (ZC), and Waveform Length (WL)). Castellini et al. [31] illustrated a three grasp recognition, achieving 97.1% classification accuracy using the Root Mean Square (RMS) as features extraction for an SVM. On a more general scenario (up to 50 different hand gestures), remarkable results were obtained by Atzori et al. [9] on the Non-Invasive Adaptive hand Prosthetics Databases 1, 2, and 3 (NinaPro DB1, DB2, and DB3), employing a mixture of time- and frequency-domain features. As a downside, all these works are limited to a single-session setup. This setup fails to tackle the issue of the inter-session accuracy drop observed when classifying gesture from a never-seen session after training on just one session.

As a result, the crucial challenge in sEMG-based gesture recognition has shifted from absolute classification accuracy to managing the variability of the signal, which is affected by several factors such as anatomical variability, posture, fatigue, perspiration, changes in the skin-to-electrode interface, user adaptation, and electrode repositioning over multi-day usage [34], [8], [11], [10]. These factors strongly hamper generalization, thus limiting the long-term use and the realization of robust real-time recognition systems.

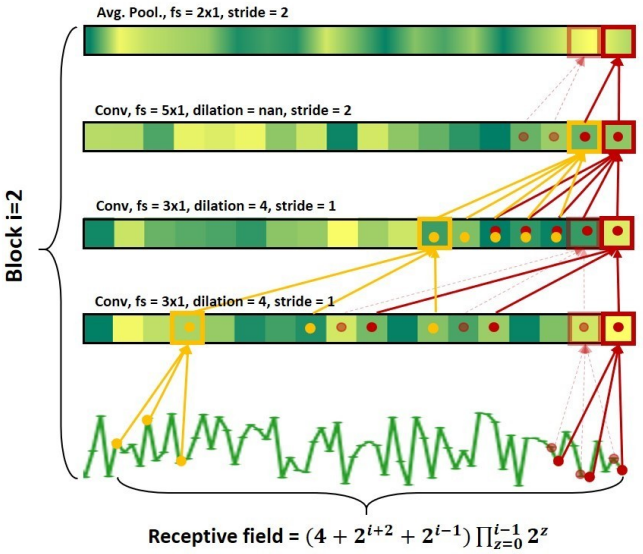

Figure 1: Structure and functioning of Convolutional Block 2: two dilated convolutions ($d = 4$), one strided convolution ($s = 2$) and average pooling. The input of the block is the temporal sequence computed by Convolutional Block 1.

For instance, Benatti et al. [12] and [11] collected sEMG data from several subjects in multi-day sessions to analyze the performance degradation of conventional ML algorithms when donning and doffing the sensory setup. In these experiments, the inter-session accuracy drop after training on a single session was up to 30%. The proposed solutions mostly rely on the extension of the training datasets, the modification of the acquisition setup (e.g. by increasing the electrode count), and the extraction of a broader set of features to improve algorithm convergence. These solutions lower the average accuracy drop, decreasing the average error rate to 12% [8]. However, this performance drop and the lack of generalization are still hampering the deployment of these solutions in reliable, commercially available systems.

A new state-of-the-art strategy to robustify recognition against temporal variability is multi-session training, which is the methodology implemented in this work. This strategy has been made possible by the release of multi-session sEMG datasets such as the Non-Invasive Adaptive hand Prosthetics Database 6 (NinaPro DB6, 10 sessions, 8 classes) [11] and the University of Bologna - INAIL (Unibo-INAIL) database (8 days $\times$ 4 arm postures, 6 gestures) [10].

The NinaPro DB6 is the dataset used as a benchmark in this work. On these data, Palermo et al. [11] reached an inter-session accuracy of 25.4% by feeding Wave Length to a Random Forest. Cene et al. [16] successfully employed Extreme Learning Machines (ELMs) to raise this inter-session accuracy to 41.8%. It is worth to notice that the reason why the accuracy reached on the NinaPro DB6 is much lower than the one reached on other datasets with a similar number of classes and sensors, is that the hand movements of NinaPro DB6 are all grasps, thus much less diverse and discernable than the gestures in ordinary datasets.

On the Unibo-INAIL dataset, Milosevic et al. [10] showed

TABLE I: Comparison between SoA embedded platforms for EMG processing.

Work	Dataset	# subj	# sessions	# classes	# channels	Time window	Features	Algorithm	Accuracy [%] Intra / Inter	Real-time	Embedded
Hudgins [29]	private	18	1	4	1	200 ms	MAV, ZC, SSC, WL	Shallow ANN	88.9 / N.A.	no	no
Park [32]	NinaPro DB1	27	1	6 [1]	8	2000 ms	RMS time×ch.	CNN	N.A./∼94 [2] [3]	no	no
Tsinganos [28]	NinaPro DB1	27	1	53	8	200 ms	RMS time×ch.	CNN	70.5 / N.A.	no	no
Tsinganos [15]	NinaPro DB1	27	1	53	8	300 ms, 1200 ms	RMS	TCN	89.8 / N.A.	no	no
Betthauser [14]	private	9	1	27	8	1675 ms	200 ms-MAV	TCN	69.5 / N.A.	no	no
Hu [13]	NinaPro DB1	27	1	53	8	200 ms	RMS	CNN+LSTM	87.0 / N.A.	yes	no
Kaufmann [8]	private	1	121	10	8	150 ms	MAV, ZC, SSC, WL	SVM	N.A. / 87.7	no	no
Milosevic [10]	Unibo-INAIL	7	8	6	4	1 sample	RMS	SVM	∼90 [4] /∼70 [4]	no	no
Du [33]	CapgMyo	8	2	8	128	1 sample	inst. HD-sEMG images	CNN	98.6 / 63.3 [3]	yes	no
Palermo [11]	NinaPro DB6	10	10	8	14	200 ms	WL	RF	52.4 / 25.4	no	no
Cene [16]	NinaPro DB6	10	10	8	14	200 ms	MAV, VAR, RMS	ELM	69.8 / 41.8	no	no
This work	NinaPro DB6	10	10	8	14	150 ms	raw sEMG	TCN	54.5[5] / 49.6	**yes**	**yes**
	20-session	3	20	9	8	150 ms	raw sEMG	TCN	97.1 / 93.7	**yes**	**yes**

[1] Restricted to 6 functional movements, without rest class. [2] Inter-subject. [3] With domain adaptation. [4] Precise values depending on session and training strategy.
[5] Our lower-than-SoA intra-session accuracy is due to the fact that [16] (1) relies on single-session training, prone to overfit to the sessions; and (2) uses a very aggressive signal filtering over 200 ms time windows, jointly with outlier removal embedded in the algorithm, so as to smooth the transients or even discard them from the accuracy.

that multi-posture and multi-day training improve inter-session generalization. A Radial Basis Function kernel SVM (RBF-SVM) applied on 4-channel single samples of the RMS signal yielded an intra-session recognition accuracy higher than 90%, with an inter-session accuracy drop up to 20% (a value similar to [11], [16]). The aforementioned approaches showed the major limitation of classical ML: it strongly relies on domain-specific knowledge and hand-crafted features, limiting the capability to generalize over time.

To cope with this issue, DL represents a valid approach, since it incorporates feature learning into model training, and can reach a better generalization on the data. DL-based solutions have also been prompted by increased data availability (public sEMG benchmark databases) and great improvements in computing hardware [35]. Table I shows that, DL methods outperform traditional ML approaches when classifying data from different sessions. This conclusion is also reinforced in the revision made by Phinyomark et al. [34]. The first end-to-end DL architecture was proposed by Park and Lee [32], who applied a Convolutional Neural Network (CNN) + RMS on NinaPro DB1, outperforming an SVM in classification accuracy across subjects. From our variability point of view, it is interesting to note that this early work already addresses inter-subject variability, showing that a CNN benefits more than a SVM from an adaptation phase introduced before classifying data from unseen subjects. Atzori et al. [36] also proposed a CNN-based approach to recognize the 52 hand gestures from the NinaPro DB1, DB2 and DB3 (taking 150 ms-windows of RMS, acting on time×channels), reaching classification accuracy comparable to classical methods such as RF.

As to the issue of variability, a strategy typical of DL is Adaptive Batch-Normalization (AdaBN) [33], a domain adaptation consisting in re-training the Batch-Normalization (BN) layers [37] of deep models without fine-tuning the entire network. AdaBN is parameter-free, free of additional components, and computationally simpler than generalized fine-tuning. These qualities make AdaBN interesting for real-time setup, in which it has already shown some success. For instance, Du et al. [38] employed a CNN + instantaneous High-Density (HD) sEMG images, attaining 63.3% accuracy on the 8 classes of their CapgMyo database.

Recently, TCN approaches have started appearing in recent research work, gaining traction for sEMG-based gesture recognition. Tsinganos et al. [15] achieve 89.8% classification accuracy on the 53 classes of NinaPro DB1 with an RMS-fed TCN. This result is 4.8% better than SoA [39] and surpasses by 19.3% the previous results from the same authors obtained with conventional 2d-CNNs [28]. This TCN was evaluated using receptive field (i.e., input sequence lengths) of 300 ms up to 2.5 s. Although the NinaPro DB1 dataset is not multi-session [9] and so does not involve the temporal variability which is the focus of this work, [15] is a valuable demonstration that TCNs can yield good accuracy on this task. Betthauser et al. [14] proved that TCNs outperform Long Short-Term Memory (LSTM) networks in the sEMG-based gesture recognition task, reaching 69.5% accuracy on 27 classes. Also, the TCN used in this work has a very large receptive field: the 1.7 s input windows were generated by

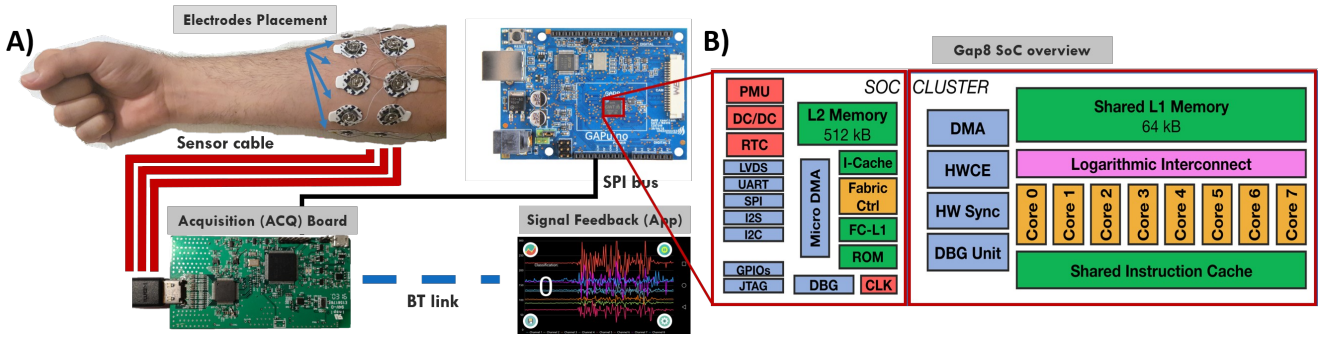

Figure 2: A) Hardware diagram of the proposed system. sEMG sensors on the forearm are connected to the Analog Front End, which sends the data via SPI to the GAP8 processing platform. A Bluetooth link allows streaming the data and the classification to an external gateway. **B)** Detailed block diagram of the GAP8 processor.

computing the MAV from 200 ms-long sequences.

Overall, these works proposing TCNs for sEMG-based gesture recognition share the limitation of using very long (i.e., $\geq$ 300 ms) signal windows. In particular, in [15] the dilated convolution is used to hugely enlarge the receptive field at constant network size, instead of exploiting dilation to work with smaller networks at a constant receptive field. This limitation implies two (related) issues: *i)* the comparison is altered with the other works that comply with the consensus of using time windows $<$ 300 ms [29]; *ii)* the proposed TCNs are evaluated under conditions which are not feasible for a usable real-time implementation. In contrast, in this work, we focus on real-time classification and target full compliance with the upper limit of 300 ms. Our proposed TCN, TEMPONet, uses 150 ms signal windows as input, and needs $<$ 15 ms for inference when deployed on an embedded platform.

III. Materials and Methods

A. Acquisition and Processing Platform

The EMG signal acquisition is based on an 8 channel commercial Analog Front End (AFE) (ADS1298) connected to the GAP8 breakout board [40]. ADS1298 is mostly used in acquisition system design for EMG, EEG and ECG signals, and it is considered the *de facto* standard for such applications. It allows simultaneous sampling of up to 8 bipolar channels with 24-bit resolution, reaching 32k samples per second (ksps). Each channel has a programmable Gain Amplifier with a gain that ranges from 1 to 12. In this application, it drives 8 fully differential channels at sampling rate 4 kHz connected to an array of passive gel-based EMG electrodes, placed in a ring configuration around the forearm. The block diagram of the GAP8 architecture is provided in Figure 2 B. GAP8 has two main functional blocks: a single tiny RISCV core, namely the fabric controller (FC) and an 8-parallel set of RISCV core, i.e. the computational cluster. FC controls SoC and peripherals and can be viewed as a simple microcontroller. The 8 cores cluster is used for vectorized and parallelized computationally-intensive tasks such as embedded artificial intelligence [41].

GAP8 is not equipped with an FPU; hence, algorithms need to be carried out using fixed-point arithmetic. The internal memory of GAP8 is divided into two layers: L1 memory and L2 memory. L2 memory is 512 kB in size and accessible by all cores. L1 memory is split into two parts: a 16 kB memory for the fabric controller and a 64 kB shared memory for the cluster cores. There is also a third level, namely L3, which externally connected via quad-SPI or a HyperBus interface. The GAP8 processor also includes an internal programmable DC/DC converter which provides power supply to the fabric controller and cluster (0.9V to 1.3V, 0.8V for retentive sleep mode). As shown in Figure 2, this setup has been developed for measurement and characterization purpose, using development boards, but, by virtue of the BGA packages of ADC and GAP8, the whole system can be integrated into a single PCB with 30x20mm form factor, suitable for wearable applications.

All firmware has been written in C and runs on the low-power GAP8 processor. For the development, we relied on the GAP8 Software Development Kit (SDK) [42], which embeds all the APIs to access the GAP8 HW features such as DMA engines, hardware timers, and I/O. The GAP8 SDK also includes a customized version of the RISC-V GCC compiler with support for the GAP8 ISA extensions used to accelerate the inference of Deep Neural Networks [17]. As shown in Fig 2 **A)**, GAP8 is connected to the AFE (i.e. ADS1298) via a 20 MHz SPI connection (GAP8 acts as master) in parallel to an interrupt wire (#DRDY) connected to a GAP8 digital pin. Once a new sample is ready, the AFE asserts the #DRDY data ready signal with a pulse. The #DRDY pulse wakes up an interrupt routine on the GAP8 Fabric Controller, which starts acquiring the SPI data using the embedded I/O uDMA. Data loaded via SPI is stored in the GAP8 L2 memory as 24-bit signed fixed-point numbers, with the least significant bit representing a value of $\mathrm{VREF}/(2^{23}-1)$. Acquired sEMG samples are then used as input of the TEMPONet TCN, whose embedded implementation is described in Section III-B2 and profiled in Section IV.

B. TEMPONet TCN

Unlike previous studies in sEMG-based gesture recognition, which are formulated as single-sample recognition [10] or image recognition [13], and mostly rely on extracted features, in this work we address the sEMG signal as a time series, using a small Temporal Convolutional Network (TCN)

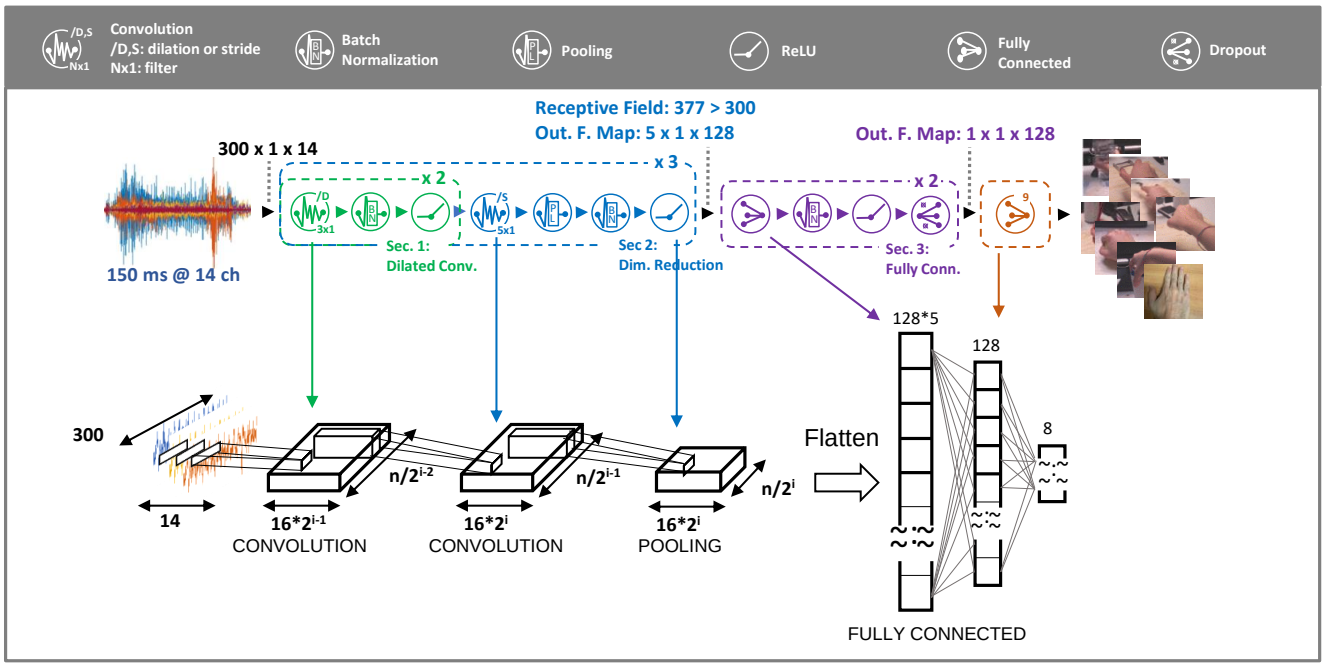

Figure 3: Processing diagram of the proposed algorithm. In the TEMPONet TCN architecture, the three blocks (each one composed of 2 convolutional and one pooling layer) are used to extract temporal features, followed by two fully connected layers that perform the final classification.

based architecture to assign labels to 150 ms raw sEMG time windows.

1) TEMPONet architecture

In this section, we present a novel TCN based topology, TEMPONet (Temporal Embedded Muscular Processing Online Network), depicted in Figure 3.

TEMPONet stacks 3 Convolutional Blocks composed by:
- 2 temporal convolutional layers with filter size 3×1, variable dilation and full padding;
- 1 convolutional layer with filter size 5×1, variable stride and padding, followed by an Average Pooling (AvgPool) with kernel 2×1.

The 3 blocks are characterized by dilation $d = 2, 4, 8$, respectively, and stride $s = 1, 2, 4$, respectively. The strided convolution of the 1st, 2nd and 3rd block raises the number of channels to 32, 64 and 128 respectively, while each AvgPool halves the sequence length immediately after. As an example, Convolutional Block 2 is represented in Figure 1. The Convolutional Blocks are followed by 2 Fully Connected (FC) layers with dropout (to help regularization [43]) and a SoftMax operation. FC layers flatten the input information to compute the final label assigned to the sequence. All layers have ReLU non-linearity as activation function and are equipped with Batch-Normalization (BN) to counter the internal covariate shift [37].

The two main characteristics of this network, block composition, and 1D dilated convolutional layers, are inspired by the novel developments in the deep learning field. Division in *blocks* of several layers where the number of channels and size of the activation tensors is kept constant is typical of many modern networks [44], [45], [46], [47]. It enables to build a network where the temporal dimension is consumed "slowly", therefore enabling a deeper network with more powerful processing of the raw information in the time series. On the other hand, as previously mentioned, dilated layers allow to increase the receptive field of each layer gradually. Dilation factors are chosen so that the receptive field at the end of the network covers an entire time window[1]. The network topology is designed in such a way that the convolutional block increases the receptive field and reduces the width of the signal (i.e., the input time window "visible" from a given neuron in the layer). A larger input window can then be analyzed by stacking more blocks instead of increasing the filter sizes, i.e., by making the network deeper instead of wider and therefore limiting the increase in the number of parameters.

Combining these insights, the TEMPONet 3-blocks configuration presented in this paper can process a 150 ms input window using only 460k parameters, which is well suited for the implementation on our developed processing board as detailed in the following sections.

2) TEMPONet embedded deployment

In this section, we describe the procedure to distill the TEMPONet algorithm for our embedded platform for sEMG acquisition and processing for real-time classification of acquired sEMG signals. As mentioned in Section III-A, the target execution platform (GAP8) has limited memory capacity and no support for floating-point data. To enable deployment of the trained TEMPONet on an embedded platform, therefore, we also targeted its quantization, i.e. reducing it to use only 8-bit integer (INT-8) parameters and feature map tensors. The pre-trained TCNs are fine-

[1]The modular nature of the network structure would allow to stack further blocks and therefore process signals on different timing windows: for the sEMG-based gesture recognition, this is of particular interest since the time-window width could change based on the target application.

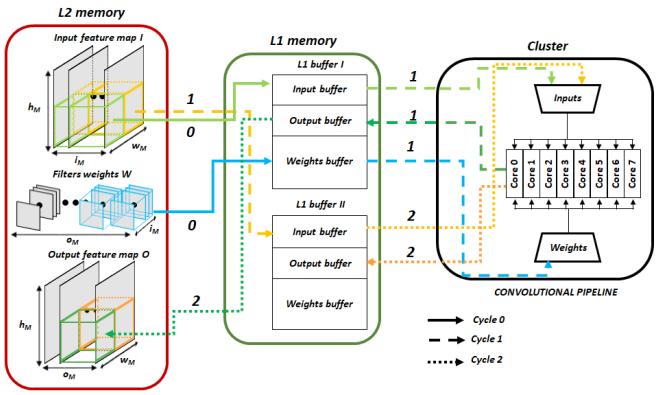

Figure 4: TEMPONet flow. Left: the DMA manages L2-L1 communication using double-buffering. Right: the cluster executes PULP-NN on a tile stored in one of the L1 buffers.

tuned after replacing ReLU activation functions with step functions using the PACT methodology [48]; weights are also quantized using a similar function. Quantization is performed layer-wise. The INT-8 representations of feature maps $\mathbf{y}$ and weights $\mathbf{W}$ are given by (respectively)

$$\widehat{\mathbf{y}} = \left\lfloor \frac{\text{clip}_{[0,\alpha_\mathbf{y})}(\mathbf{y})}{\varepsilon_\mathbf{y}} \right\rfloor; \qquad \varepsilon_\mathbf{y} = \frac{\alpha_\mathbf{y}}{256} \quad (1)$$

$$\widehat{\mathbf{W}} = \left\lfloor \frac{\text{clip}_{[\alpha_\mathbf{W},\beta_\mathbf{W})}(\mathbf{W})}{\varepsilon_\mathbf{W}} \right\rfloor; \qquad \varepsilon_\mathbf{W} = \frac{\beta_\mathbf{W} - \alpha_\mathbf{W}}{256} \quad (2)$$

The quantization procedure operates as follows, starting from a pretrained full-precision network: first, the parameters $\alpha_\mathbf{W}, \beta_\mathbf{W}$ are initialized with the minimum and maximum values of $\mathbf{W}$, respectively, while $\alpha_\mathbf{y}$ parameters are initialized by registering minimum and maximum values of $\mathbf{y}$ over a run on the training set.

All parameters (including $\mathbf{W}$, $\alpha_\mathbf{W}, \beta_\mathbf{W}$, $\alpha_\mathbf{y}$) are then fine-tuned via backpropagation using the Adam optimizer. Learning rate is set to a small value (10^{-6}) for both datasets, and the training is stopped after 30 epochs, or earlier if convergence is achieved (i.e., if the difference in loss between two epochs is bounded to 0.05). The quantized network can be deployed on GAP8 by directly using the INT-8 weights $\widehat{\mathbf{W}}$ and implementing Equation 1 as a set of comparisons against thresholds [49]. Apart from fitting in the GAP8 L2 512 kB memory constraint (INT-8 TEMPONet has a footprint of 460 kB), we also get rid of floating-point multiplications, reducing both the time and energy per classification as GAP8 has no floating-point unit and would emulate these operations in software.

The GAP8 processor receives and accumulates the data to fill an internal 150ms $\times$ 8 channel (i.e. $\sim 10KB$) buffer in L2 and then starts the classification. Meanwhile, a second buffer, also located in L2, receives the data in real-time from the analog front-end in a double-buffering procedure. Data is fed to the network at 2kHz sampling rate and using INT-32 representation only for the input data (as the ADC resolution is 24 bits).

The implementation of TEMPONet on GAP8 is based on the dedicated PULP-NN libraries [17] for optimized ad-hoc convolutional kernels deployment. PULP-NN uses all the cores available in the GAP8 cluster as well as their SIMD extensions and bit-manipulation instructions to achieve the best speed up and energy saving from the chip. As PULP-NN functions work on data in the 64 kB L1 scratchpad, it is necessary to move weights and feature maps between the 512 kB L2 memory and the L1 scratchpad. This process is performed by using an automated tool [50] to *i*) divide the data tensors in each layer in tiles that fit the L1; *ii*) insert appropriate DMA calls to realize a double buffering scheme (separate from the one on ADC data), so that data movement is always overlapped with computation. The DMA calls are asynchronous and non-blocking, allowing to import new activations and weights while the previous calculation is ongoing. Figure 4 describes this flow by highlighting the memory L2-L1 memory traffic, managed by the DMA, and the cluster execution of PULP-NN.

IV. Experimental Results

A. Experimental Setup

We implement our TEMPONet TCN approach using Python 3.5 and the specialized DL development PyTorch [51] 1.1 framework. The TCN is fed with 150 ms time windows (slide 15 ms) of the raw 14-channel and 8-channel sEMG signal, for the NinaPro DB6 and for our 20-session dataset respectively.

To analyze the accuracy drop in multi-session classification, we propose an *incremental training protocol*, where we sweep training data from 1 to a maximum of half dataset sessions (i.e., 5 for NinaPro DB6 and 10 for the 20-session dataset), using the remaining half for testing. The training sessions always precede the testing ones, in a sequential scenario, to maintain temporal coherence among sessions. Regarding the amount of data used for the training, we adopt an internal 2-fold stratified (i.e. equal number of gestures repetition for each fold) cross-validation to evaluate our algorithm also on the same sessions used for training (i.e. without the *temporal variability*). TCN training uses cross-entropy as a loss function and stochastic gradient descent for 20 epochs (batch size = 64) with L_2 regularization (`weight_decay`=10^{-4}). The initial learning rate is set to 0.001 for the NinaPro DB6 dataset and 0.01 for the 20-session dataset; in both cases, it is divided by 10 at epoch 9 and 19.

We introduce two figures of merit to evaluate our approach, based on our protocol: (1) the *intra-session* validation accuracy, computed as the average accuracy on the fold not used for training (alternately), and (2) the *inter-session* validation accuracy, calculated as the average accuracy on sessions 6-to-10 for the NinaPro DB6 and 11-to-20 for the 20-session dataset, which are never used for training. Network training is always performed off-line, hence on *steady* gestures, removing contraction transients. This was obtained by discarding the first and the last 1.5 s of each gesture for NinaPro and 300ms for the dataset we introduce in this work, thus focusing the classification only on steady signal portions.

To evaluate the inference performance, we run TEMPONet on the Ninapro DB6 comparing the results of the multi-session testing described in [16], that represents, at the best

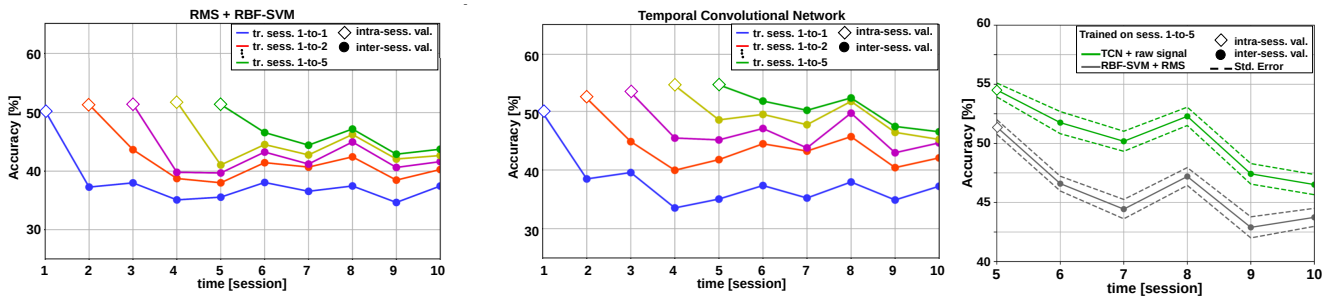

Figure 5: Left and Center: classification accuracy of RMS + RBF-SVM and of TEMPONet, using the incremental training framework on NinaPro DB6. Right: classification accuracy of RMS + RBF-SVM and of TEMPONet, after training on sessions 1-to-5 of NinaPro DB6. All validations are done on steady states + transient states.

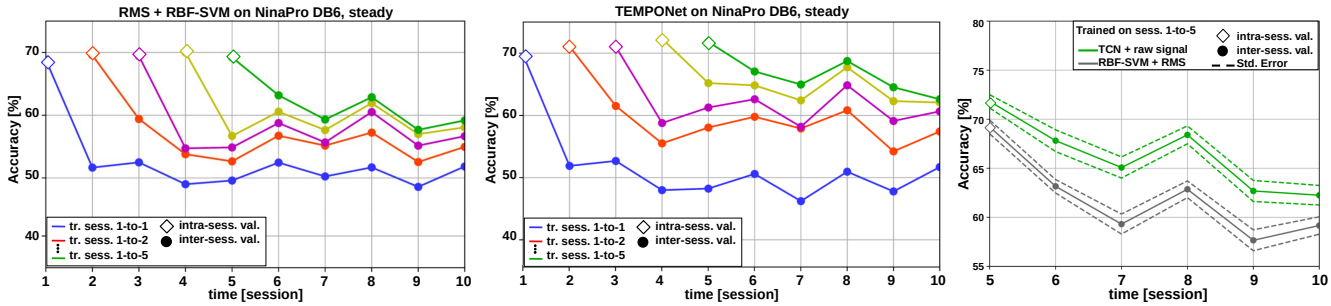

Figure 6: Left and Center: classification accuracy of the baseline RMS + RBF-SVM and our TEMPONet, reached on the different sessions of NinaPro DB6 after transient removal, using the incremental multi-session training framework. Adding training sessions improves the accuracy in the never-seen sessions, with better results for the TEMPONet TCN. Right: classification accuracy of RMS + RBF-SVM and TEMPONet, after training on sessions 1-to-5 of NinaPro DB6. All validations are done on steady states.

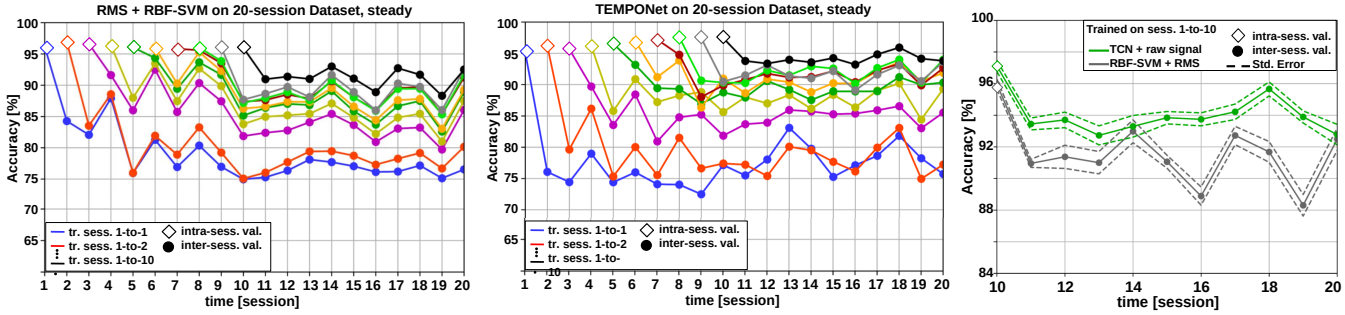

Figure 7: Left and Center: classification accuracy of the baseline RMS + RBF-SVM and our TEMPONet, reached on the different sessions of our 20-session Dataset after transient removal, using the incremental multi-session training framework. Adding training sessions improves the accuracy in the never-seen sessions, with better results for the TEMPONet TCN. Right: classification accuracy of RMS + RBF-SVM and TEMPONet, after training on sessions 1-to-10 of the 20-session Dataset. All validations are done on steady states.

of our knowledge, the previous state-of-the-art inter-session accuracy for the NinaPro DB6. We obtained an average 49.6% accuracy against 41.8% reported in [16]. In this test, the accuracy is evaluated on the entire gestures (i.e. including transients).

Furthermore, we compare our TCN topology against a baseline Radial Basis Function kernel Support Vector Machine (RBF-SVM) applied on the Root Mean Square (RMS) of the sEMG signal, computed on 60 ms time windows. This setup has been chosen as a baseline since it is a widely used approach [52] and it allows to show how the algorithm performs against a well-established classification scheme. We implement the SVM using the Scikit-learn framework (version 0.20.0) [53], and we optimize the SVM parameters, resulting in $C = 1$ and `gamma='scale'`.

Figure 5 depicts the results of RBF-SVM and TCN on the NinaPro DB6 dataset, evaluated on the full validation set. The algorithm comparison shows that TCN performs 4.3% better than the SVM. All accuracy results are reported as mean (i.e. average over subjects, training folds, and validation sessions), or as mean $\pm$ Standard Error.

It is noteworthy that, since most of the classification errors

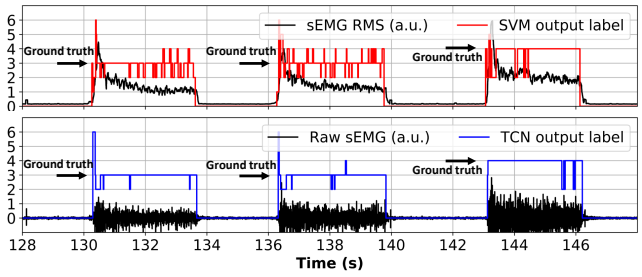

Figure 8: Comparison of the real-time inter-session classification of the RBF-SVM and TEMPONet on our 20-session dataset.

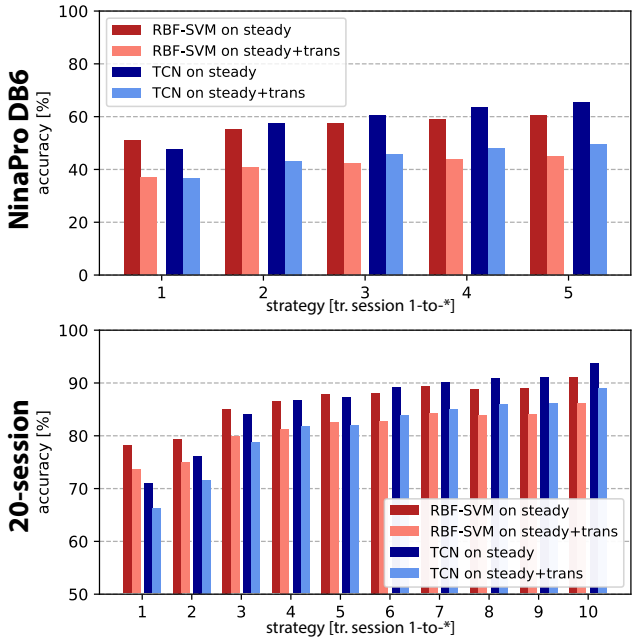

Figure 9: Average inter-session accuracy obtained on steady signals (removing transients, in full color) and on full signals (including transients, in a light color) for both datasets and techniques.

are located in the transients, the design of an end-to-end gesture controller usually requires to remove gesture transients. This is a well-established procedure common to both inter-session [8], [54], [55] and inter-subject [56] studies, and is sufficient for the aim of a steady gesture controller design. It can be done by techniques such as threshold comparison, DTW, or Hidden Markov Models. For this reason, in the following paragraphs, we present the accuracy results also with data purged of transients. To provide better insight in how the accuracy varies when removing transients, Figure 9 compares the average inter-session accuracy achieved testing only on *steady* signals and on *full* ones for both datasets and considering both RBF-SVM and TEMPONet TCN approaches. Similarly to what happens with *steady* signals, the advantage of TEMPONet in terms of accuracy on *full* grows proportionally to the number of sessions involved in the training. The accuracy drop on *full* with respect to *steady* is substantially similar between RBF-SVMs and TCNs for

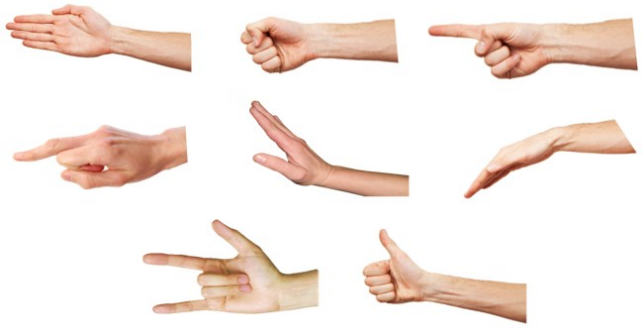

Figure 10: Hand gestures used during experimentation including finger and wrist contractions.

both datasets.

B. NinaPro DB6 (steady)

On NinaPro DB6, both the SVM and the TCN yield the best recognition accuracy when trained with a higher number of sessions, namely 1-to-5 (i.e. the first half of the dataset sessions). Recognition accuracy over time are plotted in Figure 6.

The SVM trained on sessions 1-to-5 reaches an average intra-session validation accuracy of $(69.2 \pm 0.7)\%$, and an average inter-session validation accuracy on sessions 6-to-10 of $(60.4 \pm 0.9)\%$, resulting in a drop of 8.8%. Compared to training on only session 1, the 5-session training maintains the same intra-session accuracy ($+0.4\%$) but increases the inter-session accuracy by 9.5%.

The TEMPONet TCN trained on sessions 1-to-5 reaches a similar intra-session validation accuracy of $(71.8 \pm 0.7)\%$ (2.6% higher than the SVM), while increases the inter-session validation accuracy of 4.8% compared to the SVM ($(65.2 \pm 1.0)\%$), with a resulting drop of 6.6%. Compared to training on only session 1, the 5-session training increases the intra-session accuracy by 5.5%), and strongly increases the inter-session accuracy by 17.5%. Noteworthy, increasing the amount of training data is a key point for our TEMPONet, which strongly increases its performance.

These results confirm the initial assumption that TCNs are more efficient in extracting information directly from raw data, as well as to remove part of the noise due to temporal variability, thus achieving better generalization over time.

C. 20-session Dataset (steady)

Our dataset has been acquired for 10 days and involves 3 subjects (all male, average age of 29 ± 3 years). Each day includes 2 sessions, taking place in the morning and afternoon, for a total of 20 sessions for the complete experimentation.

A single session has an approximate duration of one and a half minutes, including 8 hand gestures and rest, as shown in Figure 10. Each gesture is repeated 6 times, with a contraction time of approximately 3 s. To ease the labeling process, 3 s of rest between contractions of the same gestures and up to 5 s between each new gesture are left.

We tested both the SVM and the TEMPONet TCN again on our 20-session dataset. We maintain the same topology

and the same training parameter, except for the learning rate (Section IV). Recognition accuracy using our incremental training protocol and among time is plotted in Figure 7.

Both the SVM and the TEMPONet TCN reach again the best average intra-/inter-session validation accuracy with the maximum training session (1-to-10). The SVM reaches, $(96.0 \pm 0.3)\%$, $(91.1 \pm 0.6)\%$, and 4.9% of respectively intra-session, inter-session (on sessions 11-to-20), and drop. The TEMPONet increases this performance to $(97.1 \pm 0.3)\%$ intra-session accuracy, $(93.7 \pm 0.5)\%$ inter-session accuracy, and only 3.4% accuracy-drop.

Compared to training on only session 1, the SVM maintains the same intra-session accuracy (-0.2%), while the TEMPONet strongly increases its performance of 9.4%. Noteworthy, both methods enhanced with more training sessions show a sharp performance gain of the inter-session accuracy, namely 12.9% for the SVM and 22.7% for the TCN.

Similarly to NinaPro DB6, the effect of multi-session training on the SVM and the TCN is similar, but with higher gains for the TCN. The fact that our TEMPONet TCN outperforms the SVM also on the new 20-session dataset further validates the initial hypothesis that TCNs can attain better generalization by their higher ability to process raw data and to handle the inter-session sEMG variability noise.

Furthermore, a more explicit comparison between the recognition accuracy of the SVM and our TEMPONet is shown in Figure 8, which displays the output labels of the two classifiers in the real-time inter-session setup on our 20-session dataset (subject 1, training sessions 1-to-10, validation session 20). This visual inspection highlights that the output sequence returned by TEMPONet is more accurate and more stable (i.e., smooth) in inter-session validation than the output of the SVM. The smoothness of the TEMPONet TCN classification is due to the fact that, for each inference, TEMPONet leverages $150\,\mathrm{ms}$ of signal history, rich enough to enhance stability and avoid erratic oscillations as the ones exhibited by the SVM.

We can finally notice that the classification accuracy reached on the 20-session dataset (all $> 90\%$) is consistently much higher than on NinaPro DB6 (all $< 75\%$), even in presence of a similar number of classes (8 for NinaPro DB6 vs. 9 for the 20-session dataset) and sensors (14 for NinaPro DB6 vs. 8 for the 20-session dataset). The cause is that hand movements in NinaPro DB6 are all grasps, thus much less diverse and discernible than the hand gestures of our 20-session dataset.

D. Embedded Deployment Performance

Regarding the embedded implementation, the input sEMG signals are preprocessed digitally before executing the TEMPONet TCN. This process includes a 10-tap notch filter to remove PLI interference, and a 15-tap band-pass filter ($\mathrm{BW} = 2\,\mathrm{Hz} - 1\,\mathrm{kHz}$) to cancel the DC drift and high-frequency components. The signal is then downsampled to $2\,\mathrm{kHz}$ to match the sampling rate of the NinaPro DB6 dataset. The execution time of these steps is negligible ($< 100\,\mu\mathrm{s}$) and does not affect the real-time performance of the classifier. The processing chain later continues with the execution of TEMPONet as described in Figure 4.

TABLE II: Memory footprint and best intra-/inter-session accuracy of the baseline RMS + RBF-SVM, full-precision TEMPONet TCN and 8-bit quantized TCN.

	Memory Footprint	Intra-sess. acc.	Inter-sess. acc.
NinaPro DB6			
RMS + RBF-SVM (float)	1.3 MB	69.2%	60.4%
RMS + RBF-SVM (INT-8)	332 kB	50.7%	44.7%
TEMPONet TCN (float)	1.8 MB	71.8%	65.2%
TEMPONet TCN (INT-8)	460 kB	64.5%	61.0%
20-session dataset			
RMS + RBF-SVM (float)	670 kB	96.0%	91.1%
RMS + RBF-SVM (INT-8)	168 kB	95.8%	78.6%
TEMPONet TCN (float)	1.8 MB	97.1%	93.7%
TEMPONet TCN (INT-8)	460 kB	96.6%	93.3%

TABLE III: Inference time and energy of the TEMPONet deployed on GAP8.

	inf. time	**energy**	**MAC/cycle**
TEMPONet inference on GAP8 @ 1 V, 170 MHz			
Dilated Convolutions	5.40 ms	0.38 mJ	9.54
Non-Dilated Convolutions	5.86 ms	0.41 mJ	6.95
Pooling	0.16 ms	0.01 mJ	n.a.
Fully Connected	1.42 ms	0.10 mJ	4.10
Full net	12.84 ms	0.90 mJ	7.73

To evaluate fairly the accuracy of the quantized version of TEMPONet, we also distilled the RBF-SVM support vectors to INT-8. This was performed offline by evaluating the mean μ and standard deviation σ of the support vectors and applying Eq. 2 by setting $\alpha_{\mathbf{W}} = \mu - 5\sigma$, $\beta_{\mathbf{W}} = \mu + 5\sigma$ [2].

Table II reports the memory occupancy and the accuracy of the full-precision and 8-bit TEMPONet, compared to the SVM baselines. Remarkably, the accuracy drop after quantization decreases given its regularizing effect. On NinaPro DB6, quantization leads to an accuracy loss of 7.3% intra-session and 4.2% inter-session, still above the full-precision RBF-SVM baseline. On our 20-session dataset, quantization causes an accuracy loss of just 0.5% intra-session and only 0.4% inter-session, again above the full-precision SVM, but with a $1.5\times$ lower memory footprint. On the other hand, the quantization of the support vectors, which is still necessary to deploy SVMs on our GAP8 based processing platform ($512\,\mathrm{kB}$ memory constraint), results in a $\sim 15\%$ inter-session accuracy loss for both the datasets.

Table III highlights the performance of our network in terms of inference time, energy, and MAC/cycle, collected with real execution of the net on the GAP8 SoC targeting the most efficient voltage-frequency configuration to save energy ($1\,\mathrm{V}$, $170\,\mathrm{MHz}$). Also, we detail these metrics for the different layer types involved, namely Dilated and Non-Dilated Convolutions, Pooling, and Fully Connected layers.

The last column of Table III, the mean MAC/cycle, is a key indicator of computational efficiency; Dilated Convolutions are not only algorithmically effective but also achieve the highest level of efficiency: 42% of the execution time is spent to run 53% of the overall operations of the network. Overall, exploiting the 8-cores of GAP8, our network reaches a mean MAC/cycle of 7.73. Therefore, TEMPONet can classify a

[2]We tried several other settings for $\alpha_{\mathbf{W}}$, $\beta_{\mathbf{W}}$; we chose the one resulting in the smallest accuracy drop.

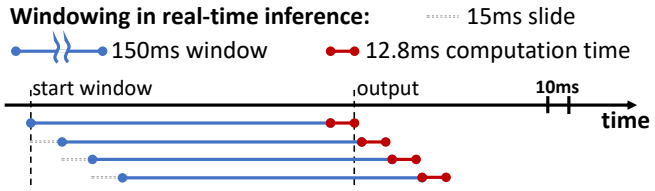

Figure 11: The windowing scheme and inference time. The real-time requirement, i.e. upper limit of 300 ms [29], is fulfilled.

time window in 12.8 ms, consuming 0.90 mJ. Our real-time constraint is given by the 15 ms of the sliding window, and is therefore well-met by the embedded application, as shown in Figure 11. Moreover, Figure 11 highlights that our windowing scheme (same for off-line training and real-time inference) and computation comply with the consensus real-time requirement, i.e. the upper limit of 300 ms [29].

Looking at classification energy, each a time window costs 0.90 mJ per inference. Between two adjacent inferences, the GAP8 SoC is only collecting data (and not processing it) for 2.2 ms . During this phase, we are able to idle the 8-core cluster using its embedded hardware synchronization unit [57], which enables fully state-retentive clock gating and wakeup in a few nanoseconds. The power consumption in this phase is limited to the ~ 10 mW consumed by the SoC to collect data from the sensor. Overall, a 15 ms window costs 0.90 mJ, yielding an average power of 60 mW. Using a small 1000 mAh battery, the sEMG gesture classification system can run continuously for $\sim 13 \cdot 10^6$ classifications, i.e., for a lifetime of ~ 54 h.

To measure the computational impact of Dilated Convolutions as opposed to *conventional* ones, we projected our measured results over a modified version of TEMPONet where dilation factors are removed. Still, the dimension of the receptive field is kept constant, therefore covering the same time window as our TEMPONet. To do so, we increased the filter size of the layers to 5, 9, and 17 in each of the three blocks. The consequence is twofold: first, execution time and energy jump to 28.7 ms and 2.0 mJ, respectively. Second, the dimension of the modified TEMPONet grows to 970 kB, too high to be suitable for embedded deployment in the GAP8 L2 memory (512 kB).

To have fair benchmarking, we need to compare our approach against platforms that are capable of running deep learning algorithms (e.g., Cortex H7 or A8 family) on sEMG signals. There are some embedded systems for sEMG processing and gesture classification, such as [18] and [19] which can execute DL algorithms on a Cortex-A processor, with a power envelope $\geq$ 500 mW, almost one order of magnitude higher than GAP8. Platforms of this class are capable of running inference of Deep Neural Networks [58], but their size and power envelope limit their applicability to real wearable systems. Recently, some attempts have also been made on Deep Neural Network deployment on high-end Cortex M family (ARM Cortex H7), leveraging an energy-efficient software support (i.e., CMSIS-NN, the SoA in SW implementation of Deep Neural Networks). Unfortunately, the deployment of a Neural network on an H7 platform reaches a top performance of 0.69 MAC/cycle @ 346 mW @ 400 MHz measured on an STM32-H7 microcontroller [17], more than 10× slower and 23× less efficient than our proposed TCN implementation, which combines parallel execution of the GAP8 cluster cores and the ISA extensions utilized by the PULP-NN computational backend [17].

V. Conclusion

In this work, we addressed the temporal variability affecting the inter-session generalization of sEMG-based hand gesture recognition. We proposed a new approach based on the novel TEMPONet Temporal Convolutional Network.

Our approach, validated on the NinaPro Database 6 and on our new 20-session sEMG dataset, proves that the best training set compositions are the ones including the highest number of sessions, producing an inter-session classification accuracy of 65.2% on NinaPro DB6, and 93.7% on our 20-session dataset. These accuracies improve by up to 4.8% and 2.6% the results yielded by a reference RBF-SVM, on NinaPro DB6 and on the 20-session dataset respectively. This low inter-session accuracy drop allows the design of a robust long-term sEMG-based controller. Moreover, the TEMPONet does not require any additional hand-crafted feature extraction.

We also distilled the TCN applying deep network quantization to 8-bit, showing that our approach reaches as little as 2.8× and 1.5× lower memory footprint compared to a baseline RBF-SVM, for the NinaPro DB6 and the 20-session dataset respectively, with an inter-session accuracy decrease of only 4.2% and 0.4% respectively, still higher than the reference SVM.

Finally, we showed an implementation of the quantized TEMPONet on the GAP8 microcontroller, achieving 12.8 ms time with 0.90 mJ energy per classification. Considering a 1000 mA h battery, our TEMPONet running on GAP8, reaches up to 13 M gesture classifications, for a total always-on classification time of ~ 54 h.

Acknowledgments

This work has been partially supported by the European H2020 FET project OPRECOMP under Grant 732631, and by project EC Horizon-2020 ALOHA under Grant 780788.

References

[1] R. Meattini *et al.*, "An sEMG-Based Human-Robot Interface for Robotic Hands Using Machine Learning and Synergies," *IEEE Transactions on Components, Packaging and Manufacturing Technology*, vol. 8, no. 7, pp. 1149–1158, 2018.

[2] T. Saponas *et al.*, "Making muscle-computer interfaces more practical," in *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*, ser. CHI '10. New York, NY, USA: ACM, 2010, pp. 851–854. [Online]. Available: http://doi.acm.org/10.1145/1753326.1753451

[3] T. Starner *et al.*, "Real-time american sign language recognition using desk and wearable computer based video," *IEEE Transactions on pattern analysis and machine intelligence*, vol. 20, no. 12, pp. 1371–1375, 1998.

[4] T. Saponas *et al.*, "Enabling always-available input with muscle-computer interfaces," in *Proceedings of the 22nd annual ACM symposium on User interface software and technology*. ACM, 2009, pp. 167–176.

[5] touch bionics, http://www.touchbionics.com/products, 2018.

[6] Ottobock, https://www.ottobockus.com/prosthetics/upper-limb-prosthetics/solution-overview/myoelectric-prosthetics/, 2018.

[7] J. Yousefi and A. Hamilton-Wright, "Characterizing emg data using machine-learning tools," *Computers in biology and medicine*, vol. 51, pp. 1–13, 2014.

[8] P. Kaufmann *et al.*, "Fluctuating emg signals: Investigating long-term effects of pattern matching algorithms," in *2010 Annual International Conference of the IEEE Engineering in Medicine and Biology*. IEEE, 2010, pp. 6357–6360.

[9] M. Atzori *et al.*, "Electromyography data for non-invasive naturally-controlled robotic hand prostheses," *Scientific Data*, vol. 1, p. 140053, dec 2014. [Online]. Available: http://www.nature.com/articles/sdata201453

[10] B. Milosevic *et al.*, "Exploring Arm Posture and Temporal Variability in Myoelectric Hand Gesture Recognition," *Proceedings of the IEEE RAS and EMBS International Conference on Biomedical Robotics and Biomechatronics*, vol. 2018-August, pp. 1032–1037, 2018.

[11] F. Palermo *et al.*, "Repeatability of grasp recognition for robotic hand prosthesis control based on sEMG data," in *2017 International Conference on Rehabilitation Robotics (ICORR)*. IEEE, jul 2017, pp. 1154–1159. [Online]. Available: https://ieeexplore.ieee.org/document/8009405/

[12] S. Benatti *et al.*, "Analysis of robust implementation of an emg pattern recognition based control," in *Proceedings of the International Joint Conference on Biomedical Engineering Systems and Technologies-Volume 4*. SCITEPRESS-Science and Technology Publications, Lda, 2014, pp. 45–54.

[13] Y. Hu *et al.*, "A novel attention-based hybrid cnn-rnn architecture for semg-based gesture recognition," *PloS one*, vol. 13, no. 10, p. e0206049, 2018.

[14] J. Betthauser *et al.*, "Stable Electromyographic Sequence Prediction during Movement Transitions using Temporal Convolutional Networks," *International IEEE/EMBS Conference on Neural Engineering, NER*, vol. 2019-March, no. c, pp. 1046–1049, 2019.

[15] P. Tsinganos *et al.*, "Improved Gesture Recognition Based on sEMG Signals and TCN," *ICASSP 2019 - 2019 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, no. May, pp. 1169–1173, 2019. [Online]. Available: https://ieeexplore.ieee.org/document/8683239/

[16] V. Cene *et al.*, "Open Database for Accurate Upper-Limb Intent Detection Using Electromyography and Reliable Extreme Learning Machines," *Sensors (Basel, Switzerland)*, vol. 19, no. 8, 2019.

[17] A. Garofalo *et al.*, "PULP-NN: Accelerating Quantized Neural Networks on Parallel Ultra-Low-Power RISC-V Processors," *arXiv:1908.11263 [cs]*, Aug. 2019.

[18] J. Liu *et al.*, "An open and configurable embedded system for emg pattern recognition implementation for artificial arms," in *Engineering in Medicine and Biology Society (EMBC), 2014 36th Annual International Conference of the IEEE*. IEEE, 2014, pp. 4095–4098.

[19] X. Zhang *et al.*, "Real-time implementation of a self-recovery emg pattern recognition interface for artificial arms," in *Engineering in Medicine and Biology Society (EMBC), 2013 35th Annual International Conference of the IEEE*. IEEE, 2013, pp. 5926–5929.

[20] L. Tassinary *et al.*, "The Skeletomotor System : Surface," 1985, no. January 1990.

[21] C. De Luca, "The Use of Surface Electromyography," *Journal of applied biomechanics*, vol. 13, no. July 1993, pp. 1–38, 1997.

[22] R. Rangayyan, *Biomedical Signal Analysis: A Case-Study Approach*. IEEE/Wiley, New York, NY, 2002.

[23] B. Milosevic *et al.*, "Design challenges for wearable emg applications," in *Design, Automation Test in Europe Conference Exhibition (DATE), 2017*, March 2017, pp. 1432–1437.

[24] M. Tomasini *et al.*, "Power Line Interference Removal for High-Quality Continuous Biosignal Monitoring with Low-Power Wearable Devices," *IEEE Sensors Journal*, vol. 16, no. 10, pp. 3887–3895, 2016.

[25] C. Lea *et al.*, "Temporal convolutional networks for action segmentation and detection," *2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 1003–1012, 2016.

[26] S. Bai *et al.*, "An empirical evaluation of generic convolutional and recurrent networks for sequence modeling," *arXiv preprint arXiv:1803.01271*, 2018.

[27] S. Benatti *et al.*, "A versatile embedded platform for EMG acquisition and gesture recognition," *IEEE Transactions on Biomedical Circuits and Systems*, vol. 9, no. 5, pp. 620–630, Oct 2015.

[28] P. Tsinganos *et al.*, "Deep learning in emg-based gesture recognition," in *PhyCS*, 2018.

[29] B. Hudgins *et al.*, "A new strategy for multifunction myoelectric control," *IEEE transactions on bio-medical engineering*, vol. 40, pp. 82–94, 02 1993.

[30] K. Englehart and B. Hudgins, "A robust, real-time control scheme for multifunction myoelectric control," *IEEE Transactions on Biomedical Engineering*, vol. 50, no. 7, pp. 848–854, July 2003.

[31] C. Castellini *et al.*, "Fine detection of grasp force and posture by amputees via surface electromyography," *Journal of Physiology-Paris*, vol. 103, pp. 255–262, 2009.

[32] K.-H. Park and S.-W. Lee, "Movement intention decoding based on deep learning for multiuser myoelectric interfaces," *2016 4th International Winter Conference on Brain-Computer Interface (BCI)*, pp. 1–2, 2016.

[33] Y. Li *et al.*, "Revisiting batch normalization for practical domain adaptation," *ArXiv*, vol. abs/1603.04779, 2016.

[34] A. Phinyomark and E. Scheme, "Emg pattern recognition in the era of big data and deep learning," 2018.

[35] I. Goodfellow *et al.*, *Deep Learning*. Cambridge, MA, USA: MIT Press, 2016, http://www.deeplearningbook.org.

[36] M. Atzori *et al.*, "Deep learning with convolutional neural networks applied to electromyography data: A resource for the classification of movements for prosthetic hands," *Frontiers in Neurorobotics*, vol. 10, 09 2016.

[37] S. Ioffe and C. Szegedy, "Batch normalization: Accelerating deep network training by reducing internal covariate shift," *ArXiv*, vol. abs/1502.03167, 2015.

[38] Y. Du *et al.*, "Surface emg-based inter-session gesture recognition enhanced by deep domain adaptation," in *Sensors*, 2017.

[39] W. Wei *et al.*, "A multi-stream convolutional neural network for semg-based gesture recognition in muscle-computer interface," *Pattern Recognition Letters*, vol. 119, 12 2017.

[40] Texas Instruments, 2015. [Online]. Available: http://www.ti.com/lit/ds/symlink/ads1298.pdf

[41] D. Palossi *et al.*, "A 64mW DNN-based Visual Navigation Engine for Autonomous Nano-Drones," *IEEE Internet of Things Journal*, pp. 1–1, 2019.

[42] "GAP8 SDK," https://greenwaves-technologies.com/setting-up-sdk/.

[43] N. Srivastava *et al.*, "Dropout: a simple way to prevent neural networks from overfitting," *J. Mach. Learn. Res.*, vol. 15, pp. 1929–1958, 2014.

[44] A. G. Howard *et al.*, "MobileNets: Efficient Convolutional Neural Networks for Mobile Vision Applications," 2017. [Online]. Available: http://arxiv.org/abs/1704.04861

[45] M. Sandler *et al.*, "Mobilenetv2: Inverted residuals and linear bottlenecks," in *The IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2018.

[46] X. Zhang *et al.*, "ShuffleNet: An Extremely Efficient Convolutional Neural Network for Mobile Devices," *arXiv:1707.01083 [cs]*, Jul. 2017.

[47] C. Szegedy *et al.*, "Rethinking the inception architecture for computer vision," in *The IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2016.

[48] J. Choi *et al.*, "PACT: Parameterized Clipping Activation for Quantized Neural Networks," *arXiv:1805.06085 [cs]*, May 2018.

[49] M. Rusci *et al.*, "Work-in-Progress: Quantized NNs as the Definitive Solution for Inference on Low-Power ARM MCUs?" in *2018 International Conference on Hardware/Software Codesign and System Synthesis (CODES+ISSS)*, Sep. 2018, pp. 1–2.

[50] A. Burrello *et al.*, "Work-in-Progress: DORY: Lightweight Memory HierarchyManagement for Deep NN Inference on IoT Endnodes," in *International Conference on Hardware/Software Codesign and System Synthesis (CODES+ISSS 2019)*, 3001in press.

[51] A. Paszke *et al.*, "Automatic differentiation in pytorch," in *NIPS-W*, 2017.

[52] C. Castellini and P. van der Smagt, "Surface emg in advanced hand prosthetics," *Biological cybernetics*, vol. 100, no. 1, pp. 35–47, 2009.

[53] F. Pedregosa *et al.*, "Scikit-learn: Machine learning in Python," *Journal of Machine Learning Research*, vol. 12, pp. 2825–2830, 2011.

[54] S. Amsüss *et al.*, "Long term stability of surface EMG pattern classification for prosthetic control," *Proceedings of the Annual International Conference of the IEEE Engineering in Medicine and Biology Society, EMBS*, pp. 3622–3625, 2013.

[55] J. He *et al.*, "User adaptation in long-term, open-loop myoelectric training: Implications for EMG pattern recognition in prosthesis control," *Journal of Neural Engineering*, vol. 12, no. 4, 2015.

[56] M. Atzori *et al.*, "Building the NINAPRO Database: A Resource for the Biorobotics Community - HES SO Valais publications - Aigaion 2.0," *Proceedings of the IEEE International Conference on Biomedical Robotics and Biomechatronics*, p. 51, 2012. [Online]. Available: http://publications.hevs.ch/index.php/publications/show/1172

[57] F. Glaser *et al.*, "Hardware-accelerated energy-efficient synchronization and communication for ultra-low-power tightly coupled clusters," in *2019 Design, Automation Test in Europe Conference Exhibition (DATE)*, March 2019, pp. 552–557.

[58] A. Ignatov *et al.*, "Ai benchmark: Running deep neural networks on android smartphones," in *Proceedings of the European Conference on Computer Vision (ECCV)*, 2018, pp. 0–0.

Marcello Zanghieri received his M.Sc. degree in Physics from the University of Bologna, Italy, in 2019. In 2019, he worked at the Energy-Efficient Embedded Systems Laboratory (EEES Lab), DEI, University of Bologna, with a research studentship. Currently, he is pursuing his Ph.D. in Data Science and Computation at EEES Lab, under the supervision of Luca Benini. His research interests focus on the analysis of biosignals by means of machine learning and deep learning, currently focusing on EMG of and EEG, for the development of low-power wearable devices such as sEMG-controlled prostheses and Brain-Machine Interfaces.

Simone Benatti received the Ph.D. degree in Electrical Engineering and Computer Science from the University of Bologna, Italy, in 2016. He currently holds a postdoctoral position at the University of Bologna. His research interests focus on energy efficient embedded wearable systems, signal processing, sensor fusion and actuation systems. This includes hardware/software co-design to efficiently address performance, as well as advanced algorithms. In this field, he has published more than 30 papers in international peer-reviewed conferences and journals. He has collaborated with several international research institutes and companies. Previously, he worked 8 years as electronic designer and R&D engineer of electromedical devices.

Alessio Burrello received his Master's degree in Electronic for Embedded Systems at the Politecnico of Turin. Currently, he is pursuing his Ph.D. at the EEES lab (DEI), under the supervision of Luca Benini, focusing on the vertical implementation of deep learning solutions, from the high-level code to the embedded implementation.

Victor Kartsch received his Master degree in Electronic and Communications Science and Technology at the University of Bologna. Currently, he is pursuing his Ph.D. at the EEES lab (DEI), under the supervision of Luca Benini, focusing on the design and implementation of Low Power Brain-Machine Interfaces for EMG/EEG signals.

Francesco Conti received the Ph.D. degree from the University of Bologna in 2016 and is currently a post-doctoral researcher at the Integrated Systems Laboratory, ETH Zürich, Switzerland, and the Energy-Efficient Embedded Systems Laboratory, University of Bologna, Italy. His current research focuses on deploying advanced Deep Neural Network capabilities on "extreme edge" ultra-low-power nodes; he explores the fields holistically with novel architectures and hardware design, as well as embedded software and high-level algorithmic innovations to bridge the gap from deep learning frameworks to ultra-low-power hardware. He has co-authored more than 30 papers in international conferences and journals, and he has been the recipient of three best paper awards (ASAP'14, EVW'14, ESWEEK'18) and the 2018 HiPEAC Tech Transfer Award for his work on hardware acceleration of Deep Neural Networks.

Luca Benini holds the chair of digital Circuits and systems at ETHZ and is Full Professor at the University of Bologna. In 2009-2012 he served as chief architect in STMicroelectronics France. Dr. Benini's research interests are in energy-efficient computing systems design, from embedded to high-performance. He is also active in the design ultra-low power VLSI Circuits and smart sensing micro-systems. He has published more than 1000 peer-reviewed papers and five books. He is an ERC-advanced grant winner, a Fellow of the IEEE, of the ACM and a member of the Academia Europaea. He is the recipient of the 2016 IEEE CAS Mac Van Valkenburg award and the 2019 IEEE TCAD Donald O. Pederson Best Paper Award.