Hardware Implementation of Deep Network Accelerators Towards Healthcare and Biomedical Applications

Mostafa Rahimi Azghadi, *Senior Member, IEEE,* Corey Lammie, *Student Member, IEEE,* Jason K. Eshraghian, *Member, IEEE,* Melika Payvand, *Member, IEEE,* Elisa Donati, *Member, IEEE,* Bernabé Linares-Barranco, *Fellow, IEEE* and Giacomo Indiveri, *Senior Member, IEEE*

***Abstract*—The advent of dedicated Deep Learning (DL) accelerators and neuromorphic processors has brought on new opportunities for applying both Deep and Spiking Neural Network (SNN) algorithms to healthcare and biomedical applications at the edge. This can facilitate the advancement of medical Internet of Things (IoT) systems and Point of Care (PoC) devices. In this paper, we provide a tutorial describing how various technologies including emerging memristive devices, Field Programmable Gate Arrays (FPGAs), and Complementary Metal Oxide Semiconductor (CMOS) can be used to develop efficient DL accelerators to solve a wide variety of diagnostic, pattern recognition, and signal processing problems in healthcare. Furthermore, we explore how spiking neuromorphic processors can complement their DL counterparts for processing biomedical signals. The tutorial is augmented with case studies of the vast literature on neural network and neuromorphic hardware as applied to the healthcare domain. We benchmark various hardware platforms by performing a sensor fusion signal processing task combining electromyography (EMG) signals with computer vision. Comparisons are made between dedicated neuromorphic processors and embedded AI accelerators in terms of inference latency and energy. Finally, we provide our analysis of the field and share a perspective on the advantages, disadvantages, challenges, and opportunities that various accelerators and neuromorphic processors introduce to healthcare and biomedical domains.**

***Index Terms*—Spiking Neural Networks, Deep Neural Networks, Neuromorphic Hardware, CMOS, Memristor, FPGA, RRAM, Healthcare, Medical IoT, Point-of-Care**

I. Introduction

ARTIFICIAL intelligence is uniquely poised to cope with the growing demands of the universal healthcare system [1]. The healthcare industry is projected to reach over 10 trillion dollars by 2022, and the associated workload on medical practitioners is expected to grow concurrently [2]. As the reliability of Deep Learning (DL) improves, it has pervaded various facets of healthcare from monitoring [3], [4], to prediction [5], diagnosis [6], treatment [7], and prognosis [8]. Fig. 1(a) shows how data collected from the patient, which may be a combination of bio-samples, medical images, temperature, movement, etc., can be processed using a smart DL system that monitors the patient for anomalies and/or to predict diseases. DL systems can be used to recommend treatment options and prognosis, which further affect monitoring and prediction in a closed-loop scenario.

The capacity of Artificial Intelligence (AI) to meet or exceed the performance of human experts in medical-data analysis [9], [10], [11] can, in part, be attributed to the continued improvement of high-performance computing platforms such as Graphics Processing Units (GPUs) [12] and customized Machine Learning (ML) hardware [13]. These can now process and learn from a large amount of multi-modal heterogeneous general and medical data [14]. This was not readily achievable a decade ago.

While the field of DL has been growing at an astonishing rate in terms of performance, network size, and training run time, the development of dedicated hardware to process DL algorithms is struggling to keep up. Concretely, the compute loads of DL have doubled every 3.4 months since 2012. Moore's Law targets the doubling of compute power every 18-24 months, and appears to be slowing down [15]. The progress in hardware accelerator development currently relies on advances by a handful of technology companies, most notably Nvidia and its GPUs [16], [17] and Google and its Tensor Processing Units (TPUs) [13], in addition to new startups and research groups developing Application Specific Integrated Circuits (ASICs) for DL training and acceleration. While there are significant advances in tailoring deep network models and algorithms for various healthcare and biomedical applications [18], most computationally expensive deep networks are trained on either GPUs or in data centers [12], [19]. The latter typically requires access to cloud computing services which is not only costly and comes with high power demands, but also compromises data privacy. This is distinct to the effective deployment of DL at the edge on an increasing number of medical IoT devices [20] and PoC systems [21], as illustrated in Fig. 1(b). Edge learning and inference enables the option to move processing away from the cloud. This is critical for highly sensitive medical data and offline operation. Edge-based processing must combine compactness, low-power, and rapid (high throughput) at a low-cost, to make smart health monitoring viable and affordable for integration into human life [22].

M. Rahimi Azghadi and Corey Lammie are with the College of Science and Engineering, James Cook University, QLD 4811, Australia. e-mail: mostafa.rahimiazghadi@jcu.edu.au

J. K. Eshraghian is with the Department of Electrical Engineering and Computer Science, The University of Michigan, Ann Arbor, MI 48109-2122, USA.

M. Payvand, E. Donati, and G. Indiveri are with the Institute of Neuroinformatics, University and ETH Zurich, Switzerland.

B. Linares-Barranco is with the Instituto de Microelectrónica de Sevilla IMSE-CNM (CSIC and Universidad de Sevilla), Sevilla, Spain.

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/TBCAS.2020.3036081, IEEE Transactions on Biomedical Circuits and Systems

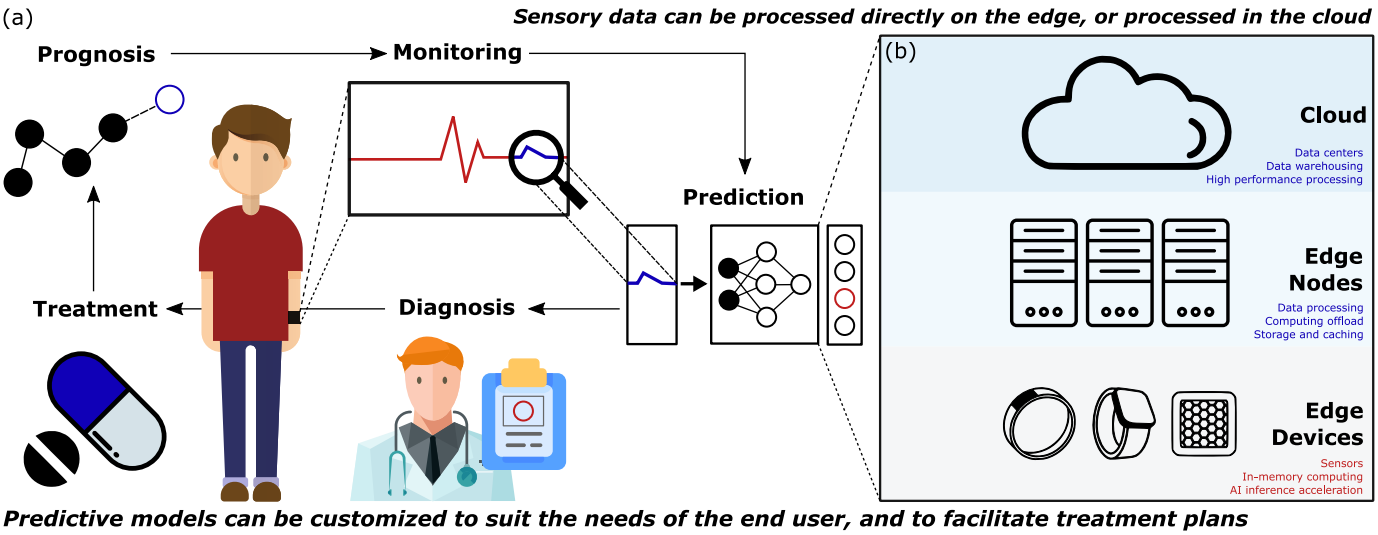

Fig. 1. A depiction of (a) the usage of DL in a smart healthcare setting, which typically involves monitoring, prediction, diagnosis, treatment, and prognosis. The various parts of the DL-based healthcare system can run on (b) the three levels of the IoT, i.e. edge devices, edge nodes, and the cloud. However, for healthcare IoT and PoC processing, edge learning and inference is preferred.

Specialized embedded DL accelerators, such as the Nvidia Jetson and Xavier series [23], and the Movidius Neural Compute Stick [24], [25], have shown the promise of edge computing. More recently, the Nvidia Clara Embedded was released as a healthcare-specific edge accelerator. This is a computing platform for edge-enabled AI on the Internet of Medical Things (IoMT). However, embedded devices remain relatively power hungry and costly, and many state-of-the-art algorithms far exceed the memory bandwidth of resource-constrained devices. They are not yet ideal learning/inference engines for ambient-assisted precision medicine systems. There is a need for innovative systems which can satisfy the stringent requirements of healthcare edge devices to be made affordable to the community at large scales.

To that end, in this paper we focus on the use of three various hardware technologies to develop dedicated deep network accelerators which will be discussed from a biomedical and healthcare application point-of-view. The three technologies that we cover here are CMOS, memristors, and Field Programmable Gate Arrays (FPGAs). It is worth noting that, while our focus targets edge inference engines in the biomedical domain, the techniques and hardware advantages discussed here are likely to be useful for efficient offline deep network learning, or online on-chip learning. Herein, the term DL 'accelerator' is used to refer to a device that is able to perform DL inference and potentially training.

This tutorial on DL accelerators within the biomedical sphere commences with a brief introduction to artificial and spiking neural networks. Next, we introduce the computational demands of DL by shedding light on why they are power- and resource-intensive. This will justify the need for application specific hardware platforms. After that, we discuss recent hardware advances which have led to improvements in training and inference efficiency. These improvements ultimately guide us to viable edge inference engine options.

After reviewing the literature on these DL accelerators, we quantify the performance of various algorithms on different types of DL processors. The results allow us to draw a perspective on the potential future of spike-based neuromorphic processors in the biomedical signal processing domain. Based on our analysis and perspective, we conjecture that, for edge processing, neuromorphic computing and Spiking Neural Networks (SNNs) [26] will likely complement DL inference engines, either through signaling anomalies in the data or acting as 'intelligent always-on watchdogs' which continuously monitor the data being recorded, but only activate further processing stages if and when necessary.

We expect this tutorial, review and perspective to provide guidance on the history and future of DL accelerators, and the potential they hold for advancing healthcare. Our contributions are summarized as follows:

- Our paper is the first to discuss the use of three different emerging and established hardware technologies for facilitating DL acceleration, with a focus on biomedical applications.
- We provide tutorial sections on how one may implement a typical biomedical task on FPGAs or simulate it for deployment on memristive crossbars.
- Our paper is the first to discuss how event-based neuromorphic processors can complement DL accelerators for biomedical signal processing.
- We provide open-source code and data to enable the reproduction of our results.

The remainder of the paper is organized as follows. In Section II, we define the technical terminology that is used throughout this paper and cover the working principles of artificial and spiking neural networks. We also introduce a biomedical signal processing task for hand-gesture classification, which is used for benchmarking the different technologies and algorithms discussed in this paper. In Section III, we step through the design, simulation, and implementation of Deep Neural Networks (DNNs) using different hardware

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/TBCAS.2020.3036081, IEEE Transactions on Biomedical Circuits and Systems

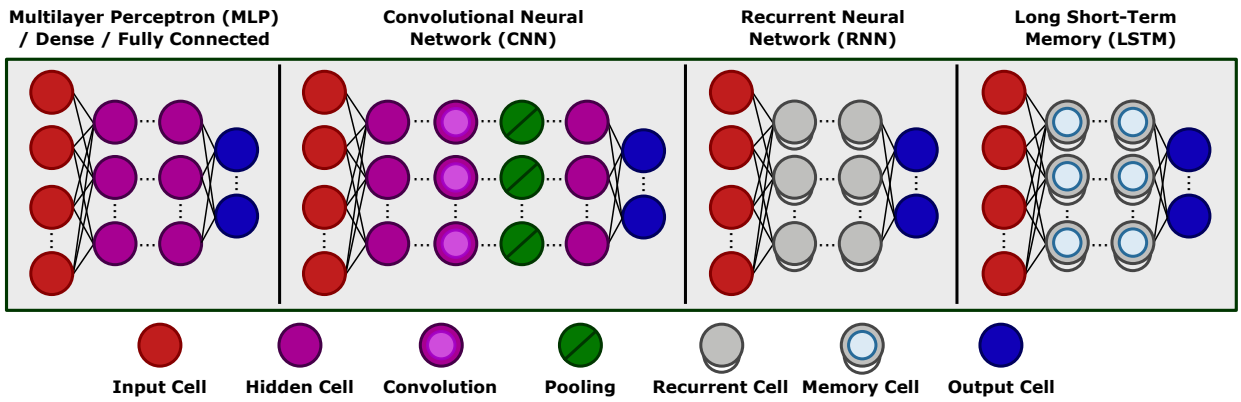

Fig. 2. Popular ANN structures. MLP/Dense/Fully Connected are typically well-suited for cross-sectional quantitative data, whereas RNNs and LSTMs networks are optimized for sequential data. CNNs are equipped for both types.

technologies. We show sample cases of how they have been deployed in healthcare settings. Furthermore, we demonstrate the steps and techniques required to simulate and implement hardware for the benchmark hand-gesture classification task using memristive crossbars and FPGAs.

In Section IV, we provide our perspective on the challenges and opportunities of both DNNs and SNNs for biomedical applications and shed light on the future of spiking neuromorphic hardware technologies in the biomedical domain. Section V concludes the tutorial.

II. Deep Artificial and Spiking Neural Networks

A. Nomenclature of Neural Network Architectures

Although most DNNs reported in literature are ANNs, DNNs refer to more than one hidden layer, independently of whether the architecture is fully connected, convolutional, recurrent, ANN or SNN, or of any other structure. For example, the most widely used DNN type in image processing, i.e. a CNN, can be physically implemented as an ANN or SNN, and in both cases it would be 'deep'. However, in this paper, whenever we use the terms 'deep', DL, or deep network, we refer to Deep Artificial Neural Networks. For Deep Spiking Neural Networks, we simply use the term SNN.

B. Deep Artificial Neural Networks

Traditional ANNs and their learning strategies that were first developed several decades ago [27] have, in the past several years, demonstrated unprecedented performance in a plethora of challenging tasks which are typically associated with human cognition. These have been applied to medical image diagnosis [28] and medical text processing [29], using DNNs.

Fig. 2 illustrates a simplified overview of the structure of some of the most widely-used DNNs. The most conventional form of these architectures is the Multi-Layer Perceptron (MLP). Increasing the number of hidden layers of perceptron cells is widely regarded to improve hierarchical feature extraction which is exploited in various biomedical tasks, such as seizure detection from electroencephalography (EEG) [30], [31]. CNNs introduce convolutional layers, which use spatial filters to encourage spatial invariance. CNNs often include pooling layers to downsample their outputs to reduce the search space for subsequent convolutional layers. CNNs have been widely used in medical and healthcare applications, as they are very well-suited for spatially structured data. Their use in medical image analysis [32] will form a major part of our discussions in subsequent sections.

RNNs are another powerful network architecture recently used both individually [33], and in combination with CNNs [34], in biomedical applications. RNNs introduce recurrent cells with a feedback loop, and are especially useful for processing sequential data such as temporal signals and time-series data, e.g. electrocardiography (ECG) [34], and medical text [35]. The feedback loop in recurrent cells gives them a memory of previous steps and builds a dynamic awareness of changes in the input. The most well-known type of RNNs are LSTMs which are designed to mine patterns in data sequences using their short-term memory of distant events stored in their memory cells. LSTMs have been widely used for processing biomedical signals such as ECGs [33], [36]. Although there are many other variants of DNN architectures, we will focus on these most commonly used types.

1) Automatic hierarchical feature extraction: The above mentioned DNNs learn intricate features in data through multiple computational layers across various levels of abstraction [37]. The fundamental advantage of DNNs is that they mine the input data features automatically, without the need for human knowledge in their supervised learning loop. This allows deep networks to learn complex features by combining a hierarchy of simpler features learned in their hidden layers [37].

2) Learning algorithms: Learning features from data in a DNN, e.g. the networks shown in Fig. 2, is typically achieved by minimizing a loss function. In most cases, this is equivalent to finding the maximum likelihood using the cross-entropy between training data and the learned model distribution. Loss function minimization is achieved by optimizing the network parameters (weights and biases). This optimization process minimizes the loss function from the final network layer backward through all the network layers and is therefore called backpropagation. Widely used optimization algorithms

in DNNs include Stochastic Gradient Descent (SGD) and those that use adaptive learning rates [37].

3) Backpropagation in DNNs is computationally expensive: Despite the continual improvement of hardware platforms for running and training DNNs, reducing their power consumption is a computationally formidable task. One of the dominant reasons is the feed-forward error backpropagation algorithm, which depends on thousands of epochs of computationally intensive Vector Matrix Multiplication (VMM) operations [27], using huge datasets that can exceed millions of data points. These operations, if performed on a conventional von Neumann architecture which has separate memory and processing units, will have a time and power complexity of order $\mathcal{O}(N^2)$ for multiplying a vector of length N in a matrix of dimensions $N \times N$.

In addition, an artificial neuron in DNNs calculates a sum-of-products of its input-weight matrix pairs. For instance, a CNN spatially structures the sum-of-products calculation into a VMM operation. In digital logic, an adder tree can be used to accumulate a large number of values. This, however, becomes problematic in DNNs when one considers the sheer number of elements that must be summed together, as each addition requires one cycle.

4) Transfer learning: A major assumption when training DNNs is that both training and test samples are drawn from the same feature space and distribution. When the feature space and/or distribution changes, DNNs should be retrained. Rather than training a new model from scratch, trained parameters from an existing model can be fixed, tuned, or adapted [38]. This process of transfer learning can be used to greatly reduce the computational expense of training DNNs.

In the medical imaging domain, transfer learning from natural image datasets, particularly ImageNet [39], using standard large models and corresponding pretrained weights has become a de-facto method to speed up training convergence and to improve accuracy [40]. Transfer learning can also be used to leverage personalized anatomical knowledge accumulated over time to improve the accuracy of pre-trained CNNs for specific patients [41], i.e., to perform patient-specific model tuning. This is an important topic in biomedical application domains, which will be further discussed in IV-F.

C. DL Accelerators

In Table I, we depict some popular CNN architectures, accompanied with the total number of weights, and MAC operations that must be computed for a single image (input resolutions of 656×468 for OpenPose, 224×224 for the rest). This table highlights two key facts. Firstly, MACs are the dominant operation of DNNs. Therefore, hardware implementations of DNNs should strive to parallelize a large number of MACs to perform effectively. Secondly, there are many predetermined weights that must be called from memory. Reducing the energy and time consumed by reading weights from memory provides another opportunity to improve efficiency.

TABLE I
NUMBER OF WEIGHTS AND MULTIPLY-AND-ACCUMULATE (MAC) OPERATIONS IN VARIOUS CNN ARCHITECTURES FOR A SINGLE IMAGE AND FOR VIDEO PROCESSING AT 25 FRAMES PER SECOND.

Network architecture	Weights	MACs	@ 25 FPS
AlexNet	61 M	725 M	18 B
ResNet-18	11 M	1.8 B	45 B
ResNet-50	23 M	3.5 B	88 B
VGG-19	144 M	22 B	550 B
OpenPose	46 M	180 B	4500 B
MobileNet	4.2 M	529 M	13 B

Consequently, significant research has been being conducted to achieve massive parallelism and to reduce memory access in DNN accelerators, using different hardware technologies and platforms as depicted in Fig. 3. Although these goals are towards general DL applications, they can significantly facilitate fast and low-power smart PoC devices [21] and healthcare IoT systems.

In addition to conventional DL accelerators, there have been significant research efforts to utilize biologically plausible SNNs for learning and cognition [42]. Spiking neuromorphic processors have also been used for biomedical signal processing [43], [44], [45]. Below, we provide a brief introduction to SNNs, which will be discussed as a method complementary to DL accelerators for efficient biomedical signal processing later in this paper. We will also perform comparisons among SNNs and DNNs in performing an electromyography (EMG) processing task.

D. Spiking Neural Networks

SNNs are neural networks that typically use Integrate-and-Fire neurons to dynamically process temporally varying signals (see Fig. 4(j)). By integrating multiple spikes over time, it is possible to reconstruct an analog value that represents the mean firing rate of the neuron. The mean firing rate is equivalent to the value of the activation function of ANNs. So in the mean firing rate limit, there is an equivalence between ANNs and SNNs. By using spikes as all-or-none digital events (Fig. 4(i)), SNNs enable the reliable transmission of signals across long distances in electronic systems. In addition, by introducing the temporal dimension, these networks can efficiently encode and process sequential data and temporally changing inputs [46]. SNNs can be efficiently interfaced with event-based sensors since they only process events as they are generated. An example of such sensors is the Dynamic Vision Sensor (DVS), which is an event-based camera shown in Fig. 4(h). The DVS consists of a logarithmic photo-detector stage followed by an operational transconductance amplifier with a capacitive-divider gain stage, and two comparators. The ON/OFF spikes are generated every time the difference between the current and previous value of the input exceeds a pre-defined threshold. The sign of the difference corresponds to the ON or OFF channel where the spike is produced. This is different to conventional cameras (Fig. 4(f)), which produce image frames (Fig. 4(g)). Intuitively, it makes sense to use asynchronous event-based sensor data in asynchronous SNNs, and synchronously generated frames (i.e., all pixels are given at a regular clock interval) through synchronous ANNs. But it is worth noting that conventional frames can be encoded as

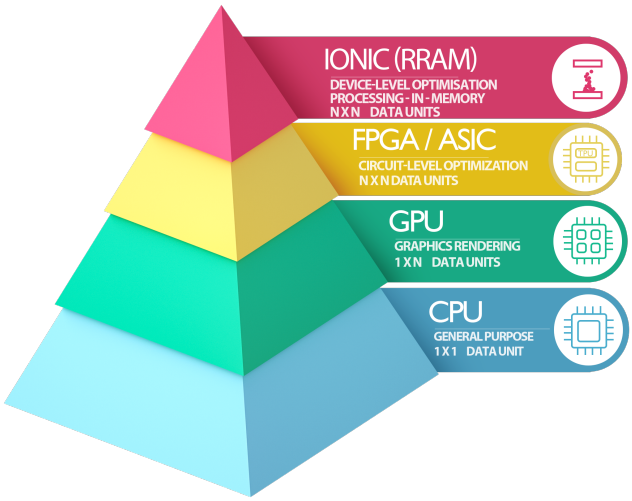

Fig. 3. Typical hardware technologies for DNN acceleration. In this paper we cover the top two layers of the pyramid, which include specialized hardware technologies for high-performance training and inference of DNNs. While the apex is labelled RRAM, this is intended to broadly cover all programmable non-volatile resistive switching memories e.g. CBRAM, MRAM, PCM, etc.

asynchronous spikes with frequencies that vary based on pixel intensity, and event streams can be integrated over time into synchronously generated time-surfaces [47], [48]. Event-based sensors have been used to process biomedical signals [43], [49] (Fig. 4(a)), which can be encoded to spike trains (Fig. 4(b)) to be processed by SNNs or be digitally sampled (Fig. 4(c)) for use in DNNs for learning and inference (Fig. 4(d)).

E. Benchmarking on a Biomedical Signal Processing Task

In Section III we will present a use-case of bio-signal processing where FPGA and memristive DNN accelerators are implemented and simulated. These are later compared to equivalent existing implementations[1] using DNN accelerators and neuromorphic processors from [45]. To perform comparisons, we use the same hand-gesture recognition task as in [45].

Tasks such as prosthesis control can be performed using EMG signals, hand-gesture classification, or a combination of both. Here, the adopted hand-gesture dataset [45] is a collection of 5 hand gestures recorded with two sensor modalities: muscle activity from a Myo armband that senses EMG electrical activity in forearm muscles, and a visual input in the form of DVS events. Moreover, the dataset provides accompanying video captured from a traditional frame-based camera, i.e., images from an Active Pixel Sensor (APS) to feed DNNs. Recordings were collected from 21 subjects including 12 males and 9 females between the ages 25 and 35, and were taken over three separate sessions.

For each implementation, we compare the mean and standard deviation of the accuracy obtained over a 3-fold cross validation, where each fold encapsulates all recordings from a given session. Additionally, for all implementations, we compare the energy and time required to perform inference on a single input, as well as the Energy-Delay Product (EDP), which is the average energy consumption multiplied by the average inference time.

[1]https://github.com/Enny1991/dvs_emg_fusion/blob/master/full_baseline.py

III. DNN Accelerators towards healthcare and biomedical Applications

In this Section, we cover the use of CMOS and memristors in DL acceleration. We discuss how they use different strategies to achieve two of the key DNN acceleration goals, namely MAC parallelism and reduced memory access. We also discuss and review FPGAs as an alternative reconfigurable DNN accelerator platform, which has shown great promise in the healthcare and biomedical domains.

A. CMOS DNN Accelerators

General edge-AI CMOS accelerator chips can be used for DNN-enabled healthcare IoT and PoC systems. Therefore, within this subsection, we first review a number of these chips and provide examples of potential healthcare applications they can accelerate. We will also explore some common approaches to CMOS-driven acceleration of AI algorithms using massive MAC parallelism and reduced memory access, which are useful for both edge-AI devices and offline data center scale acceleration.

1) Edge-AI DNN accelerators suitable for biomedical applications: The research and market for ASICs, which focus on a new generation of microprocessor chips dedicated entirely to machine learning and DNNs, have rapidly expanded in recent years. Table II shows a number of these CMOS-driven chips, which are intended for portable applications. There are many other examples of AI accelerator chips (for a comprehensive survey see [51]), but here we picked several prolific examples, which are designed specifically for DL using DNNs, RNNs, or both. We have also included a few general purpose AI accelerators from Google [52], Intel [53], and Huawei [54].

Although developed for general DNNs, the accelerators shown in Table II can efficiently realize portable smart DL-based healthcare IoT and PoC systems for processing image-based (medical imaging) or dynamic sequential medical data types (such as EEG and ECG). For instance, the table shows a few exemplar healthcare and biomedical applications that are picked based on the demonstrated capacity of these accelerators to run (or train [55]) various well-known CNN architectures such as VGG, ResNet, MobileNet, AlexNet, Inception, or RNNs such as LSTMs, or combined CNN-RNNs. It is worth noting that most of the available accelerators are intended for CNN inference, while only some [56], [57], [58] also include recurrent connections for RNN acceleration.

The Table shows that the total power per chip in most of these devices is typically in the range of hundreds of mW, with a few exceptions consuming excessive power of around 10 Watts [53], [54]. This is required to avoid large heat sinks and to satisfy portable battery constraints. The Table also shows the computing capability per unit time (column 'Computational

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/TBCAS.2020.3036081, IEEE Transactions on Biomedical Circuits and Systems

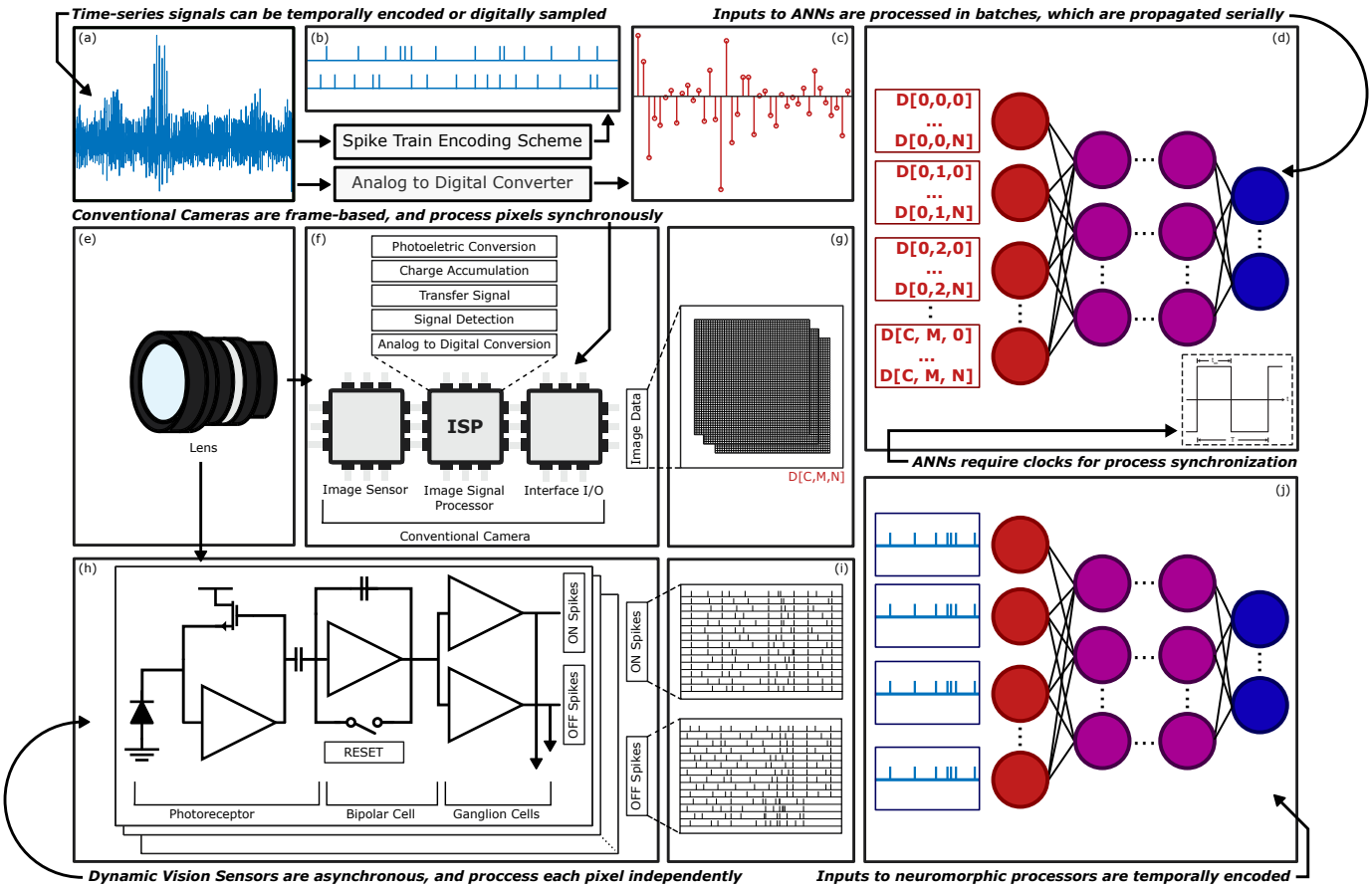

Fig. 4. DNNs and SNN neuromorphic processors adopt different operation models. In DNNs, inputs are processed in batches which propagate serially. Consequently, they require clocks for process synchronization. SNNs are asynchronous and process temporally encoded inputs independently. Time series signals, such as the EMG signal presented in (a) can be either (b) temporally encoded using spike train encoding schemes such as [43], before being fed into (j) neuromorphic processors, or (c) digitally sampled, before being concatenated into batches, to be fed into (d) DNNs. Similarly, photographs captured from (e) lenses can be (i) temporally encoded into spike trains using (h) DVSs [50] or (f) digitally encoded using conventional cameras to build (g) image frames.

Power (GOP/s)'). Regardless of power consumption, this column reveals the computational performance and consequently the size of a network one can compute per unit time. It is demonstrated that several of these chips can run large and deep CNNs such as VGG and ResNet, which enable them to perform complex processing tasks within a constrained edge power budget.

For instance, it has been previously shown in [60] that VGG CNN (shown to be compatible with Cambricon-x [59]), can successfully analyze ECog signals. Therefore, considering the power efficiency of Cambricon-x, it can be used to implement a portable automatic ECog analyzer for PoC diagnosis of various cardiovascular diseases [78]. Similarly, Eyeriss [61] can run VGG-16, which is shown to be effective in diagnosing thyroid cancer [62]. In addition, Eyeriss can run AlexNet for several different medical imaging applications [32]. Therefore, Eyeriss can be used as a mobile diagnostic tool that can be integrated into or complement medical imaging systems at the PoC. Origami [63] is another CNN accelerator chip, which can be used for other healthcare applications based on a CNN. For instance, [64] proposes a CNN-based ECG analysis for heart monitoring, or [65] introduces a two-stage end-to-end CNN for human activity recognition for elderly and rehailitation monitoring, whereas Origami can be used to develop a smart healthcare IoT edge device. Similarly, the CNN processor proposed in [66] is shown to be able to run AlexNet, which can be deployed in a PoC ultrasound image processing system [67]. Envision [68] is another accelerator that has the capability to run large-scale CNNs. It can also be used as an edge inference engine for a multi-layer CNN for EEG/ECog feature extraction for epilepsy diagnosis [69]. Neural processor [70] is another CNN accelerator that is shown to be able to run Inception V3 CNN, which can be used for skin cancer detection [11] at the edge. LNPU [55] is the only CNN accelerator shown in Table II, which unlike the others can perform both learning and inference of a deep network such as AlexNet and VGG-16, for applications including on edge medical imaging [32] and cancer diagnosis [62].

Unlike the above discussed chips that are capable of running only CNNs, DNPU [56], Thinker [57], and UNPU [58] are capable of accelerating both CNNs and RNNs. This feature makes them suitable for a wider variety of edge-based biomedical applications such as ECG analysis for BCI using a cascaded RNN-CNN [34], PoC MRI construction from motion ultrasounds using a long-term recurrent CNN [71], intelligent medical consultation using a CNN-RNN [35], respiratory

TABLE II
A NUMBER OF RECENT EDGE-AI CMOS CHIPS SUITABLE FOR PORTABLE HEALTHCARE AND BIOMEDICAL APPLICATIONS.

CMOS Chip	**Core size (mm^2)**	**Technology (nm)**	**Computational Power (GOP/s)**	**Power (mW)**	**Power Efficiency (TOPS/W)**	**Potential Mobile and Edge-based Healthcare and Medical Applications**
Cambricon-x [59]	6.38	65	544	954	0.5	ECog analysis using a sparse VGG [60] for PoC diagnosis of cardiovascular diseases
Eyeriss [61]	12.25	65	17-42	278	0.06–0.15	- Mobile Image-based cancer diagnosis using VGG-16 [62], - Mobile diagnosis tool based on AlexNet for radiology, cardiology gastroenterology imaging [32]
Origami [63]	3.09	65	196	654	0.8	- Smart healthcare IoT edge device for heart health monitoring using a CNN-based ECG analysis [64] - Two-stage end-to-end CNN for human activity recognition [65]
ConvNet processor [66]	2.4	40	102	25-287	0.3–2.7	PoC Ultrasound processing using AlexNet [67]
Envision [68]	1.87	28	76-408	7.5-300	0.8–10	Multi-layer CNN for EEG/ECog feature extraction for epileptogenicity for epilepsy diagnosis on edge [69]
Neural processor [70]	5.5	8	1900-7000	39–1500	4.5-11.5	On edge classification of skin cancer using Inception V3 CNN [11]
LNPU [55]	16	65	600	43-367	25	- On edge learning/inference using VGG-16 for cancer diagnosis [62], - On edge AlexNet learning/inference for radiology, cardiology, gastroenterology imaging diagnosis [32]
DNPU [56]	16	65	300-1200	35-279	2.1–8.1	Parallel and Cascade RNN and CNN for acECG analysis for BCI [34]
Thinker [57]	14.44	65	371	293	1–5	- PoC conversion of respiratory organ motion ultrasound into MRI using a long-term recurrent CNN [71]
UNPU [58]	16	65	346-7372	3.2-297	3.08–50.6	- Intelligent pre-diagnosis medical support/consultation using a CNN-RNN [35] - A CNN-RNN for respiratory sound classification in wearable devices enabled by patient specific model tuning [72] - A CNN-LSTM for missing Photoplethysmographic data prediction [73]
Google Edge TPU [52]	25	-	4000	2000	2	- Low-cost and easy-to-access skin cancer detection using MobileNet V1 CNN [25] - On edge health monitoring for fall detection using LSTMs [74] - Robust long-term decoding in intracortical BMIs using MLP and ELM networks [75]
Intel Nervana NNP-I 1000 (Spring Hill) [53]	-	10	48000	10000	4.8	- Diagnosis using chest X-ray classification on ResNet CNN family [76] - Long term bowel sound segmentation using a CNN [77]
Huawei Ascend 310 [54]	-	12	16000	8000	2	- Cardiovascular monitoring for arrhythmia diagnosis from ECG using an LSTM [33], - Health monitoring by heart rate variability analysis using ECG analysis by a bidirectional LSTM [36]

sound classification in wearable devices enabled by patient specific model tuning using a CNN-RNN [72], or on-chip online and personalized prediction of missing Photoplethysmographic data [73].

Table II lists three general purpose AI accelerator chips, which have been deployed for low-cost and easy-to-access skin cancer detection using MobileNet V1 CNN [25], on edge health monitoring for fall detection using LSTMs [74], chest X-ray analysis using ResNet CNN [76], long term bowel sound monitoring and segmentation using a CNN [77], cardiovascular arrhythmia detection from ECG using an LSTM [33], or heart rate variability analysis from ECG signals through a bidirectional LSTM [36], just to name a few. These general-purpose chips have the potential to be used for other biomedical edge-based applications such as robust long-term decoding in intracortical BMIs using MLP and ELM networks in a sparse ensemble machine learning platform [75].

In addition to the edge-AI CNN and RNN acceleration chips

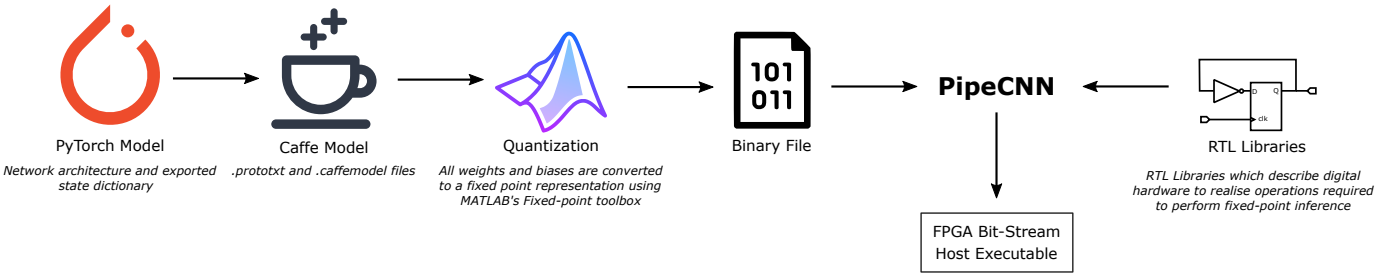

Fig. 5. Compilation flow used to deploy an EMG classification CNN to an OpenVINO FPGA adopting fixed-point number representations using OpenCL.

or general ML chips mentioned in Table II, there have been other works that have developed custom CMOS platforms for biomedical applications. Examples of these CMOS designs include [79] that has developed a 128-Channel ELM-based neural decoder for BMI, and [80] that has implemented an autoencoder neural network as part of a neural interface processor for brain-state classification and programmable-waveform neurostimulation.

2) Common approaches to CMOS-driven DL acceleration: Accelerators will typically target either data center use or embedded 'edge-AI' acceleration. Edge chips, such as those discussed above, must operate under restrictive power budgets (e.g., within thermal limits of 5 W) to cope with portable battery constraints. While the scale of tasks, input dimension capacity, and clock speeds will differ between edge-AI and modular data center racks, both will adopt similar principles in the tasks they seek to optimize.

Most of the accelerator chips, such as those discussed in Table II, use similar optimization strategies involving reduced precision arithmetic [55], [58], [66], [68] to improve computational throughput. This is typically combined with architectural-level enhancements [56], [57], [59], [61], [70] to either reduce data movement (using in- or near-memory computing), heightened parallelism, or both. In addition, there are many other approaches commonly used to make neural network implementations more efficient. Examples of these include tensor decomposition, pruning, and mixed-precision data representation, which are often integrated in hardware with in-memory and near-memory computing. A thorough review of these approaches can be found in [81] and [82].

Sequential and combinational logic research is largely matured, so outside of emerging memory technologies, the dominant hardware benefits are brought on by optimizing data flow and architecture. An early example is the neuFlow system-on-chip (SoC) processor which relies on a grid of processing tiles, each made up of a bank of processing operators and a multiplexer based on-chip router [83]. The processing operator can serially perform primitive computation (MUL, DIV, ADD, SUB, MAX), or a parallelized 1D/2D convolution. The router configures data movement between tiles to support streaming data flow graphs.

Since the development of neuFlow, over 100 startups and companies have developed, or are developing, machine learning accelerators. The Neural Processing Unit (NPU) [84] generalizes the work from neuFlow by employing eight processing engines which each compute a neuron response: multiplication, accumulation, and activation. If a program could be partitioned such that a segment of it can be calculated using MACs, then it would be partially computed on the NPU. This made it possible to go beyond MLP neural networks. The NPU was demonstrated to perform Sobel edge detection and fast Fourier transforms as well.

NVIDIA coupled their expertise in developing GPUs with machine learning dedicated cores, namely, tensor cores, which are aimed at demonstrating superior performance over regular Compute Unified Device Architecture (CUDA) cores [17]. Tensor cores target mixed-precision computing, with their NVIDIA Tesla V100 GPU combining 672 tensor cores on a single unit. By merging the parallelism of GPUs with the application specific nature of tensor cores, their GPUs are capable of energy efficient general compute workloads, as well as 12 trillion floating-point operations per seconds (TFLOPSs) of matrix arithmetic.

Although plenty of other notable architectures exist (see Table II), a pattern begins to emerge, as most specialized processors rely on a series of sub-processing elements which each contribute to increasing throughput of a larger processor [82], [81]. Whilst there are plenty of ways to achieve MAC parallelism, one of the most renowned techniques is the systolic array, and is utilized by Groq [85] and Google, amongst numerous other chip developers. This is not a new concept: systolic architectures were first proposed back in the late 1970s [86], [87], and have become widely popularized since powering the hardware DeepMind used for the AlphaGo system to defeat Lee Sedol, the world champion of the board game Go in October 2015. Google also uses systolic arrays to accelerate MACs in their TPUs, just one of many CMOS ASICs used in DNN processing [13].

B. FPGA DNNs

FPGAs are fairly low-cost reconfigurable hardware that can be used in almost any hardware prototyping and implementation task, significantly shortening the time-to-market of an electronic product. They also provide parallel computation, which is essential when simultaneous data processing is required such as processing multiple ECG channels in parallel. Furthermore, there exists a variety of High Level Synthesis (HLS) tools and techniques [88], [89] that facilitate FPGA prototyping without the need to directly develop time-consuming low-level Hardware Description Language (HDL) codes [90]. These tools allow engineers to describe their

targeted hardware in high-level programming languages such as C to synthesize them to Register Transfer Level (RTL). The tools then offload the computational-critical RTL to run as kernels on parallel processing platforms such as FPGAs [91].

1) Accelerating DNNs on FPGAs: FPGAs have been previously used to realize mostly inference [89], [92], [93], and in some cases training of DNNs with reduced-precision-data [94], or hardware-friendly approaches [95]. For a comprehensive review of previous FPGA-based DNN accelerators, we refer the reader to [89].

Here, we demonstrate an example of accelerating DNNs to benchmark the biomedical signal processing task explained in subsection II-E. For our acceleration, we use fixed-point parameter representations on a Starter Platform for OpenVINO Toolkit FPGA using OpenCL. OpenCL [88] is an HLS framework for writing programs that execute across heterogeneous platforms. OpenCL specifies programming languages (based on C99 and C++11) for programming the compute devices and Application Programming Interfaces (APIs) to control and execute its developed kernels on the devices, where depending on the available computation resources, an accelerator can pipeline and execute all work items in parallel or sequentially.

Fig. 5 depicts the compilation flow we adopted. The trained DNN PyTorch model is first converted to *.prototxt* and *.caffemodel* files using Caffe. All weights and biases are then converted to a fixed point representation using MATLAB's Fixed-point toolbox using word length and fractional bit lengths defined in [96], prior to being exported as a single binary *.dat* file for integration with PipeCNN, which is used to generate the necessary RTL libraries, and to perform compilation of the host executable and the FPGA bit-stream. We used Intel's FPGA SDK for OpenCL 19.1, and provide all files used during the compilation shown in Fig. 5 in a publicly accessible complementary GitHub repository[2].

2) FPGA-based DNNs for biomedical applications: Despite the many FPGA-based DNN accelerators available [89], only a few have been developed specifically for biomedical applications such as ECG anomaly detection [97], or real-time mass-spectrometry data analysis for cancer detection [98], where the authors show that application-specific parameter quantization and customized network design can result in significant inference speed-up compared to both CPU and GPU. In addition, the authors in [99] have developed an FPGA-based BCI, in which a MLP is used for reconstructing ECog signals. In [100], the authors have implemented an EEG processing and neurofeedback prototype on a low-power but low-cost FPGA and then scaled it on a high-end Ultra-scale Virtex-VU9P, which has achieved 215 and 8 times power efficiency compared to CPU and GPU, respectively. For the EEG processing, they developed an LSTM inference engine.

It is projected that, by leveraging specific algorithmic design and hardware-software co-design techniques, FPGAs can provide $>$10 times energy-delay efficiency compared to state-of-the-art GPUs for accelerating DL [89]. This is significant for realizing portable and reliable healthcare applications. However, FPGA design is not as straightforward as high-level designs conducted for DL accelerators and requires skilled engineers and stronger tools, such as those offered by the GPU manufacturers.

[2]https://github.com/coreylammie/TBCAS-Towards-Healthcare-and-Biomedical-Applications/blob/master/FPGA/

C. Memristive DNNs

To achieve the two aforementioned key DNN acceleration goals, i.e. massive MAC parallelism and reduced memory access, many studies have leveraged memristors [101], [102], [103], [104] as weight elements in their DNN and SNN [105], [106] architectures. Memristors are often referred to as the fourth fundamental circuit element, and can adapt their resistance (conductance) to changes in the applied current or voltage. This is similar to the adaptation of neural synapses to their surrounding activity while learning. This adaptation feature is integral to the brain's in-memory processing ability, which is missing in today's general purpose computers. This in-situ processing can be utilized to perform parallel MAC operations inside memory, hence, significantly improving DNN learning and inference. This is achieved by developing memristive crossbar neuromorphic architectures, which are projected to achieve approximately 2500-fold reduction in power and a 25-fold increase in acceleration, compared to state-of-the-art specialized hardware such as GPUs [101].

1) Memristive crossbars for parallel MAC and VMM operations: A memristive crossbar that can be fabricated using a variety of device technologies [106], [107] can perform analog MAC operations in a single time-step (see Fig. 6(a)). This reduces the time complexity to its minimum ($\mathcal{O}(1)$), and is achieved by carrying out multiplication at the place of memory, in a non-von Neumann structure. Using this well-known approach, VMM can be parallelized as demonstrated in Fig. 6(b), where the vector of size M represents input voltage signals ($[V_1..V_M]$). These voltages are applied to the rows of the crossbar, while the matrix (of size $M \times N$), whose elements are represented as conductances (resistances), is stored in the memristive components at each cross point. Taking advantage

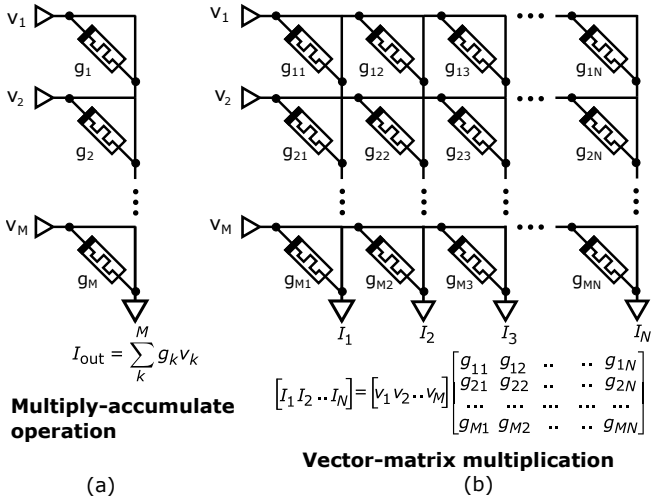

Fig. 6. Memristive crossbars can parallelize (a) analog MAC and (b) VMM operations. Here, V represents the input vector, while conductances in the crossbar represent the matrix.

of the basic Ohm's law ($I = V.G$), the current summed in each crossbar column represents one element of the resulting multiplication vector of size N.

2) Mapping memristive crossbars to DNN layers: Although implementing fully-connected DNN layers is straightforward by mapping the weights to crossbar point memristors and having the inputs represented by input voltages, implementing a complex CNN requires mapping techniques to convert convolution operations to MAC operations. A popular approach to perform this conversion is to use an unrolling (unfolding) operation that transforms the convolution of input feature maps and convolutional filters to MAC operations. We have developed a software platform named *MemTorch* [108], that will be introduced in subsequent sections, to perform this mapping as well as a number of other operations, for converting DNNs to Memristive DNNs (MDNNs). The mapping process implemented in MemTorch is illustrated in the left panel in Fig. 7. The figure shows that the normal input feature maps and convolutional filters (shown in gray shaded area) are unfolded and reshaped (shown in the cyan shaded area) to be compatible with memristive crossbar parallel VMM operations. It is worth noting that the convolutional filters that can be applied to the input feature maps have a direct relationship with the required crossbar sizes. Furthermore, the resulting hardware size depends on the size of the input feature maps [109].

3) Peripheral circuitry for memristive DNNs: In addition to the memristive devices that are used as programmable elements in MDNN architectures, various peripheral circuitry is required to perform feed-forward error-backpropagation learning in MDNNs [103]. This extra circuitry may include: (i) a conversion circuit to translate the input feature maps to input voltages, which for programming memristive devices are usually Pulse Width Modulator (PWM) circuits, (ii) current integrators or sense amplifiers, which pass the current read from every column of the memristive crossbar to (iii) analog to digital converters (ADCs), which pass the converted voltage to (iv) an activation function circuit, for forward propagation, and for backward error propagation (v) the activation function derivative circuit. Other circuits required in the error backpropagation path include (vi) backpropagation values to PWM voltage generators, (vii) backpropagation current integrators, and (viii) backpropagation path ADCs. In addition, an update module that updates network weights based on an algorithm such as SGD is required, which is usually implemented in software. After the update, the new weight values should be written to the memristive crossbar, which itself requires Bit-Line (BL) and Word-line (WL) switch matrices to address the memristors for update, as well as a circuit to update the memristive weights. There are different approaches to implement this circuit such as that proposed in [110], while others may use software ex-situ training where the new weight values are calculated in software and transferred to the physical memristors through peripheral circuitry [103].

4) Memristive device nonidealities: Although ideal memristive crossbars have been projected to remarkably accelerate DNN learning and inference and drastically reduce their power consumption [101], [102], device imperfections observed in experimentally fabricated memristors impose significant performance degradation when the crossbar sizes are scaled up for deployment in real-world DNN architectures, such as those required for healthcare and biomedical applications discussed in subsection III-A. These imperfections include nonlinear asymmetric and stochastic conductance (weight) update, device temporal and spatial variations, device yield, as well as limited on/off ratios [101]. To minimize the impact of these imperfections, specific peripheral circuitry and system-level mitigation techniques have been used [111]. However, these techniques add significant computation time and complexity to the system. It is, therefore, essential to take the effect of these nonidealities into consideration before utilizing memristive DNNs for any healthcare and medical applications, where accuracy is critical. In addition, there is a need for a unified tool that reliably simulates the conversion of a pre-trained DNN to a MDNN, while critically considering experimentally modeled device imperfections [108].

5) Conversion of DNN to MDNN while considering memristor nonidealities: Due to the significant time and energy required to train new large versions of DNNs for challenging cognitive tasks, such as biomedical and healthcare data processing [9], [112], the training of the algorithms is usually undertaken in data centers [9], [13]. The pretrained DNN can then be transferred to be used on memristive crossbars. There are several different frameworks and tools that can be used to simulate and facilitate this transition [113]. In a recent study, we have developed a comprehensive tool named MemTorch, which is an open source, general, high-level simulation platform that can fully integrate any behavioral or experimental memristive device model into crossbar architectures to design MDNNs [108].

Here, we utilize the benchmark biomedical signal processing task explained in subsection II-E to demonstrate how pretrained DNNs can be converted to equivalent MDNNs, and how non-ideal memristive devices can be simulated within MDNNs prior to hardware realization. The conversion process, which can be generalized to other biomedical models using MemTorch, is depicted in Fig. 7.

The targeted MDNNs are constructed by converting linear and convolutional layers from PyTorch pre-trained DNNs to memristive equivalent layers employing 1-Transistor-1-Resistor (1T1R) crossbars. A double-column scheme, in which two crossbars are used to represent positive and negative weight values, is used to represent network weights within memristive crossbars. The converted MDNN models are tuned using linear regression, as described in [108]. The complete and detailed process and the source code of the network conversion for the experiments shown in this subsection are provided in a publicly accessible complementary Jupyter Notebook[3].

During the conversion, any memristor model can be used. For the benchmark task, a reference VTEAM model [114] is instantiated using parameters from Pt/Hf/Ti Resistive Random Access Memory (RRAM) devices [115], to model all memristive devices within converted linear and convolutional

[3]https://github.com/coreylammie/TBCAS-Towards-Healthcare-and-Biomedical-Applications/blob/master/MemTorch.ipynb

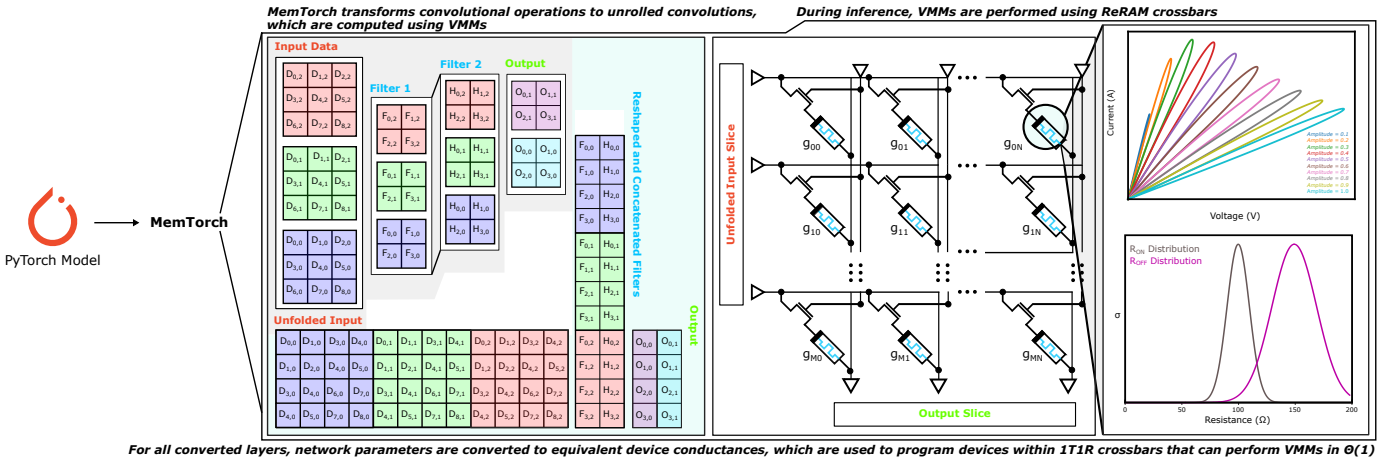

Fig. 7. Conversion process of a DNN trained in PyTorch and mapped to a Memristive DNN using MemTorch [108], to parallelize MVMs using 1T1R memristive crossbars and to take into account memristor variability including finite number of conductance states and non-ideal R_{ON} and R_{OFF} distributions.

layers. As already mentioned, memristive devices have inevitable variability, which should be taken into account when implementing an MDNNs for learning and/or inference. Also, depicted in Fig. 7 are visualizations of two non-ideal device characteristics: the finite number of conductance states and device-to-device variability. Using MemTorch [108], not only can we convert any DNNs to an equivalent MDNNs utilizing any memristive device model, we are also able to comprehensively investigate the effect of various device non-idealities and variation on the performance of a possible MDNN, before it is physically realized in hardware.

In order to demonstrate an example which includes variability in our MDNN simulations, device-device variability is introduced by sampling R_{OFF} for each device from a normal distribution with $\bar{R}_{\text{OFF}}$ = 2,500Ω with standard deviation 2σ, and R_{ON} for each device from a normal distribution with $\bar{R}_{\text{ON}}$ = 100Ω with standard deviation σ.

In Fig. 8, for the converted memristive MLP and CNN that process APS hand-gesture inputs, we gradually increase σ from 0 to 500, and compare the mean test set accuracy across the three folds. As can be observed from Fig. 8, with increasing device-to-device variability, i.e. the variability of R_{ON} and R_{OFF}, the performance degradation increases across all networks. For all simulations, R_{ON} and R_{OFF} are bounded to be positive.

6) Memristive DNNs towards biomedical applications: Although some previous small-scale MDNNs have been simulated for biomedical tasks such as cardiac arrhythmia classification [116], or have been implemented on a physical programmable memristive array for breast cancer diagnosis [117], there is currently no large-scale MDNN, even at the simulation-level, which has realized any practical biomedical processing tasks.

Similar to the recent advances in CMOS-driven DNN accelerator chips discussed in subsection III-A, there have been promises in partial [102] or full [103] realizations of MDNNs in hardware, which are shown to achieve significant energy saving compared to state-of-the-art GPUs. However, unlike their CMOS counterparts, these implementations have been only able to perform simple tasks such as MNIST and CIFAR classification. This is, of course, not suitable for implementing large-scale CNNs and RNNs, which as shown in subsection III-A are required for biomedical and healthcare tasks dealing with image [32] or temporal [33] data types.

In addition, following similar optimization strategies as those used in CMOS accelerators, [118] has simulated the use of quantized and binarized MDNNs and their error tolerance in a biomedical ECG processing task and has shown their potential to achieve significant energy savings compared to full-precision MDNNs. However, due to the many intricacies in the design process and considering the peripheral circuitry that may offset the benefits gained by using MDNNs, full hardware design is required before the actual energy saving of such binarized MDNNs can be verified.

In the next section, we provide our analysis and perspective on the use of the three hardware technologies discussed in this section for DL-based biomedical and healthcare applications. We also discuss how SNN-based neuromorphic processors can

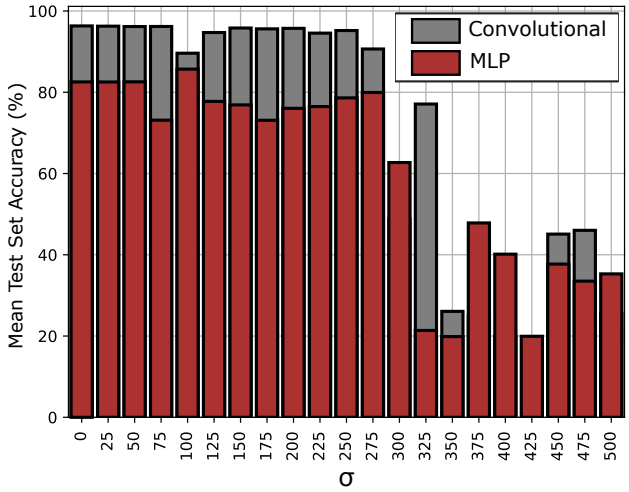

Fig. 8. Simulation results investigating the performance of MDNNs for hand gesture classification adopting non-ideal Pt/Hf/Ti ReRAM devices. Device-device variability is simulated using MemTorch [108].

benefit edge-processing for biomedical applications.

IV. Analysis and Perspective

The use of ANNs trained with the backpropagation learning algorithm in the domain of healthcare and for biomedical applications such as cancer diagnosis [130] or ECG monitoring [131] dates back to the early 90s. These networks, were typically small-scale networks run on normal workstations. As they were not deep and did not have too many parameters, they did not demand high-performance accelerators. However, with the resurgence of CNNs in the early 2010s followed by the rapid spread of DNNs and large data-sets, came the need for high-speed specialized processors. This need resulted in repurposing GPUs and actively researching other hardware and design technologies including ASIC CMOS chips (see Table II) and platforms [13], memristive crossbars and in-memory computing [102], [103], [109], and FPGA-based designs for DNN training [94], [95] and inference [92]. Despite notable progress in deploying non-GPU platforms for DL acceleration, similar to other data processing tasks, biomedical and healthcare tasks have mainly relied on standard technologies and GPUs. Currently, depending on the size of the required DNN, its number of parameters, as well as the available training dataset size, biomedical DL tasks are usually "trained" on high-performance workstations with one or more GPUs [12], [19], on customized proprietary processors such as Google TPU [9], or on various Infrastructure-as-a-Service (IaaS) provider platforms, including Nvidia GPU cloud, Google Cloud, and Amazon Web Services, among others. This is mostly due to (i) the convenience these platforms provide using high-level languages such as Python; (ii) the availability of wide-spread and open-source DL libraries such as TensorFlow and PyTorch; and (iii) strong community and/or provider support in utilizing GPUs and IaaS for training various DNN algorithms and applications.

However, DL inference can benefit from further research and development on emerging and mature hardware and design technologies, such as those discussed in this paper, to open up new opportunities for deploying healthcare devices closer to the edge, paving the way for low-power and low-cost DL accelerators for PoC devices and healthcare IoT. Despite this fact, hardware implementations of biomedical and healthcare inference engines are very sparse. Table III lists a summary of the available hardware implementations and hardware-based simulations of DNNs used for healthcare and biomedical signal processing applications, using the three hardware technologies covered herein. In addition, the table shows existing biomedical signal processing tasks implemented on generic low-power spiking neuromorphic processors.

A. CMOS Technology Has Been the Main Player for DL Inference in the Biomedical Domain

Similarly to general-purpose GPUs, all other non-GPU DL inference engines at present are implemented in CMOS. Therefore, it is obvious that most of the future edge-based biomedical platforms would rely on these inference platforms. In Table II, we listed a number of these accelerators that are mainly developed for low-power mobile applications. However, before the deployment of any edge-based DL accelerators for biomedical and healthcare tasks, some challenges need to be overcome. A non-exhaustive list of these obstacles include: (i) the power and resource constraints of available mobile platforms which, despite significant improvements, are still not suitable for high-risk medical tasks; (ii) the need to verify that a DL system can generalize beyond the distribution they are trained and tested on; (iii) bias that is inherent to datasets which may have adverse impacts on classification across different populations; (iv) confusion surrounding the liability of AI algorithms in high-risk environments [132]; and (v) the lack of a streamlined workflow between medical practitioners and DL. While the latter challenges are matters of legality and policy, the former issues highlight the fundamental need to understand where dataset bias comes from, and how to improve our understanding of why neural networks learn the features they do, such that they may generalize across populations in a manner that is safe for receivers of medical care.

In addition, to make the use of any accelerators possible for general as well as more complex biomedical applications, the field requires strong hardware-software co-design to build hardware that can be readily programmed for biomedical tasks. One successful co-design is the Google TPU [13], which has successfully been used to surpass human experts in medical imaging tasks [9]. Google has used a similar CMOS TPU technology to design inference engines [52], which are very promising as edge hardware to enable mobile healthcare care applications. The main reason for this promise is the availability of the established software platforms (such as TensorFlow Light) and the community support for the Google TPU.

Overall, great advancements have happened for DL accelerators in the past several years and they are currently stemming in various aspects of our life from self-driving cars to smart personal assistants. After overcoming a number of obstacles such as those mentioned above, we may be also able to widely integrate these DL accelerators in healthcare and biomedical applications. However, for some medical applications such as monitoring that requires always-on processing, we still need systems with orders of magnitude better power efficiency, so they can run on a simple button battery for a long time. To achieve such systems, one possible approach is to process data only when available and make our processing asynchronous. A promising method to achieve such goals is the use of brain-inspired SNN-based neuromorphic processors.

B. Towards Edge Processing for Biomedical Applications With Neuromorphic Processors

Although most of the efforts presented in this work focused on DNN accelerators, there are also notable efforts in the domain of SNN processors that offer complementary advantages, such as the potential to reduce the power consumption by multiple orders of magnitude, and to process the data in real time. These so-called neuromorphic processors are ideal for end-to-end processing scenarios, e.g., in wearable devices

TABLE III
EXISTING HARDWARE IMPLEMENTATIONS AND HARDWARE-BASED SIMULATIONS OF DNN ACCELERATORS USED FOR HEALTHCARE AND BIOMEDICAL APPLICATIONS, AND GENERIC SNN NEUROMORPHIC PROCESSORS UTILIZED FOR BIOMEDICAL SIGNAL PROCESSING. †SIMULATION-BASED

Biomedical or Healthcare Task	**DNN/SNN Architecture**	**Hardware**
Image-based breast cancer diagnosis [9]	Ensemble of CNNs	CMOS (Google TPU)
Motor intention decoding [79]	ELM	CMOS
Spatial filtering and dimensionality reduction for brain-state classification [80]	Autoencoder	CMOS
Energy-efficient multi-class ECG classification [44]	Spiking RNN	CMOS
EMG signal processing [45]	Spiking CNN/MLP	CMOS
ECG signal processing [119]	Spiking RNN	CMOS
EMG signal processing [120]	Spiking RNN	CMOS
EMG signal processing [121]	Feed-forward SNN	CMOS
EMG and EEG signal processing [122]	Recurrent 3D SNN	CMOS
EEG and LFP signal processing [123]	TrueNorth-compatible CNN	CMOS
Real-time closed loop neural decoding [124], [125]	Spiking ELM	CMOS
ECG processing for cardiac arrhythmia classification [116]	MLP	Memristors†
Breast cancer diagnosis [117]	MLP	Programmable Memristor-CMOS system
ECG signal processing [118]	Binarized CNN	Memristors†
ECG arrhythmia detection for hearth monitoring [97]	MLP	FPGA
Mass-spectrometry for real-time cancer detection [98]	MLP	FPGA
ECog signal processing for BCI [99]	MLP	FPGA
Signal processing for fall detection [126]	MLP	FPGA
BCI-decoding of large-scale neural sensors [127]	LTSM	FPGA
EEG processing for energy-efficient Neurofeedback devices [100]	LTSM	FPGA and CMOS
PPG signal processing for heart rate estimation [128]	CNN/LTSM	FPGA and CMOS
Multimodal signal classification for physical activity monitoring [129]	CNN	FPGA and CMOS

where the streaming input needs to be monitored in continuous time in an always-on manner.

There are already some works using both mixed analog-digital and digital neuromorphic platforms for biomedical tasks, showing promising results for always-on embedded biomedical systems. Table IV shows a summary of today's large scale neuromorphic processors, used for biomedical signal processing. The first chip presented in this table is DYNAP-SE [133], a multi-core mixed-signal neuromorphic implementation with analog neural dynamics circuits and event-based asynchronous routing and communication. The DYNAP-SE chip has been used to implement four of the seven SNN processing systems listed in Table III. These SNNs are used for EMG [120], [121] and ECG [119], [44] signal processing. The DYNAP-SE was also used to build a spiking perceptron as part of a design to classify and detect High-Frequency Oscillations (HFO) in human intracranial EEG [49].

In [44], [119], [120] a spiking RNN is used to integrate the ECG/EMG patterns temporally and separate them in a linear fashion to be classifiable with a linear read-out. A Support Vector Machine (SVM) and linear least square approximation is used in the read out layer for [119], [44] and overall accuracy of 91% and 95% for anomaly detection were reached respectively. In [120], the timing and dynamic features of the spiking RNN on EMG recordings was investigated for classifying different hand gestures. In [121] the performance of a feedforward SNN and a hardware-friendly spiking learning algorithm for hand gesture recognition using superficial EMG was investigated and compared to traditional machine learning approaches, such as SVM. Results show that applying SVM on the spiking output of the hidden layer achieved a classification rate of 84%, and the spiking learning method achieved 74% with a power consumption of about $0.05\ mW$. This was compared to state-of-the-art embedded system showing that the proposed spiking network is two orders of magnitude more power efficient [134], [135].

The other neuromorphic platforms listed in Table IV include digital architectures such as SpiNNaker [136], TrueNorth [137] and Loihi [138]. SpiNNaker has been used for EMG and EEG processing and the results show improved classification accuracy compared to traditional machine learning methods [122]. In [123], the authors developed a framework for decoding EEG and LFP using CNNs. The network was first developed in Caffe and the result was then used as a basis for building a TrueNorth-compatible neural network. The TrueNorth-compatible network achieved the highest classification, at approximately 76%. In [124], [125], the authors present a low-power neuromorphic platform named Spike-input Extreme Learning Machine (SELMA), which performs continuous state decoding towards fully-implantable wireless intracortical BMI. Recently, the benchmark hand-gesture classification introduced in subsection II-E, was processed and compared on two additional digital neuromorphic platforms, Loihi and ODIN/MorphIC [139], [140]. A spiking CNN was implemented on Loihi and a spiking MLP was implemented on ODIN/MorphIC [45]. The results achieved using these networks are presented in Table V.

On-chip adaptation and learning mechanisms, such as those present in some of the neuromorphic devics listed in Table IV, could be a game changer for personalized medicine, where the system can adapt to each patient's unique bio signature and/or drift over time. However, the challenge of implementing efficient on-chip online learning in these types of neuromorphic architectures has not yet been solved. This challenge lies on

TABLE IV
NEUROMORPHIC PLATFORMS USED FOR BIOMEDICAL SIGNAL PROCESSING

Neuromorphic Chip	**DYNAP-SE**	**SpiNNaker**	**TrueNorth**	**Loihi**	**ODIN**
CMOS Technology	180 nm	ARM968, 130 nm	28 nm	14 nm FinFET	28 nm FDSOI
Implementation	Mixed-signal	Digital	Digital ASIC	Digital ASIC	Digital ASIC
Neurons per core	256	1000 (1M cores)	256	Max 1k	256
Synapses per core	16k	1M	64k	114k-1M	64k
Energy per SOP	17 pJ @ 1.8V	Peak power 1W per chip	26 pJ @ 0.775	23.6 pJ @ 0.75V	12.7 pJ@0.55V
Size	38.5 mm^2	102 mm^2	-	60 mm^2	0.086 mm^2
Biosignal processing application	EMG [121], ECG [119], HFO [49]	EMG and EEG [122]	EEG and LFP [123]	EMG [45]	EMG [45]

two main factors: *locality* of the weight update and *weight storage*.

Locality: There is a hardware constraint that the learning information for updating the weights of any on-chip network should be locally available to the synapse, otherwise most of the silicon area would be consumed by the wires, required to route the update information to it. As Hebbian learning satisfies this requirement, most of the available on-chip learning algorithms focus on its implementation in forms of unsupervised/semi-supervised learning [139], [141]. However, local Hebbian-based algorithms are limited in learning static patterns or using very shallow networks [142]. There are also some efforts in the direction of on-chip gradient-descent based methods which implement on-chip error-based learning algorithms where the least mean square of a neural network cost function is minimized. For example, spike-based delta rule is the most common weight update used for single-layer networks which is the base of the back-propagation algorithm used in the vast majority of current multi-layer neural networks. Single layer mixed-signal neuromorphic circuit implementation of the delta rule have already been designed [143] and employed for EMG classification [121]. Expanding this to multi-layer networks involves non-local weight updates which limits its on-chip implementation. Making the backpropagation algorithm local is a topic of on-going research [144], [145], [146].

Weight storage: The holy grail weight storage for online on-chip learning is a memory with non-volatile properties whose state can change linearly in an analog fashion. Non-volatile memristive devices provide a great potential for this. Therefore, there is a large body of literature in combining the maturity of CMOS technology with the potential of the emerging memories to take the best out of the two worlds.

The integration of CMOS technology with that of the emerging devices has been demonstrated for non-volatile filamentary switches [147] already at a commercial level [148]. There have also been some efforts in combining CMOS and memristor technologies to design supervised local error-based learning circuits using only one network layer by exploiting the properties of memristive devices [143], [149], [150].

Apart from the above-mentioned benefits in utilizing memristive devices for online learning in SNN-based neuromorphic chips, as discussed in subsection III-C, memristive devices have also shown interesting features to improve the power consumption and delay of conventional DNNs. However, as shown in Table III, memristor-based DNNs are very sparse in the biomedical domain, and existing works are largely based only on simulation.

C. Why Is the Use of MDNNs Very Limited in the Biomedical Domain?

Currently there are very few hardware implementations of biomedical MDNNs that make use of general programmable memristive-CMOS, and only one programmed to construct an MLP for cancer diagnosis. We could also find two other memristive designs in literature for biomedical applications (shown in Table III), but they are only simulations considering memristive crossbars. This sparsity is despite the significant advantages that memristors provide in MAC parallelization and in-memory computing paradigm, while being compatible with CMOS technology [151]. These features make memristors ideal candidates for DL accelerators in general, and for portable and edge-based healthcare applications in particular, because they have stringent device size and power consumption requirements. To be able to use memristive devices in biomedical domain, though, several of their shortcomings such as limited endurance, mismatch, and analog noise accumulation must be overcome first. This demands further research in the materials, as well as the circuit and system design side of this emerging technology, while at the same time developing facilitator open-source software [108] to support MDNNs. Furthermore, investigating the same techniques utilized in developing CMOS-based DL accelerators such as limited precision data representation [109], [118] and approximate computing schemes can lead to advances in developing MDNNs and facilitate their use in biomedical domains.

D. Why and When to Use FPGA for Biomedical DNNs?

Table III shows that FPGA is a popular hardware technology for implementing simple DL networks such as MLPs [97], [98], [99], [126] and in a few cases, more complex LSTMs and CNNs [100], [127], [128], [129]. The table also shows that FPGAs are mainly used for signal processing tasks and have not been widely used to run complex DL architectures such as CNNs. This is mainly because they have limited on-chip

memory and low bandwidth compared to GPUs. However, they demonstrate notable benefits in terms of significantly shorter development time compared to ASICs, and much lower power consumption than typical GPUs. Besides, significant power and latency improvement can be gained by customizing the implementation of various components of a DNN on an FPGA, compared to running it on a general-purpose CPU or GPU [98], [100]. For instance, in [100], EEG signals are processed on FPGAs using two customized hardware blocks for (i) parallelizing MAC operations and (ii) efficient recurrent state updates, both of which are key elements of LSTMs. This has resulted in almost an order of magnitude power efficiency compared to GPUs. This efficiency is critical in many edge-computing applications including DNN-based point-of-care biomedical devices [21] and healthcare IoT [20], [64].

Another benefit of FPGAs is that a customized efficient FPGA design can be directly synthesized into an ASIC using a nanometer-node CMOS technology to achieve even more benefits [128], [129]. For instance, [100] has shown near 100 times energy efficiency improvement as an ASIC in a 15-nm CMOS technology, compared to its FPGA counterpart.

Although low-power consumption and affordable cost are two key factors for almost any edge-computing or near-sensor device, these are even more important for biomedical devices such as wearables, health-monitoring systems, and PoC devices. Therefore, FPGAs present an appealing solution, where their limitations can be addressed for a customized DNN using specific design methods such as approximate computing [95] and limited-precision data [92], [94], depending on the cost, required power consumption, and the acceptable accuracy of the biomedical device.

Another programmable low-power device that can be used in biomedical applications are Field Programmable Analog Arrays (FPAAs). These are constructed using programmable Computational Analog Blocks (CBAs) and interconnects. Unlike FPGAs, FPAAs tend to be more application driven than general purpose as they may be current mode or voltage mode devices [152]. FPAAs have been shown to perform computation with 1000 times more power efficiency while reducing the required area by 100 times when compared to FPGAs [153]. Therefore, they are a promising candidate for accelerating biomedical signal processing if machine learning algorithms such as ANNs can be implemented using them.

In 2003, [154] explored ANNs with differential feedback, and in 2006 [155] implemented an ANN using multi-chip FPAAs. More recently, [156] have demonstrated that VMMs can be efficiently computed using FPAAs, which can be used to compute linear and unrolled convolution layers within DNNs. However, while FPAAs have been used in several biomedical applications ranging from knee-joint rehabilitation [153] to the amplification of various bio-electric signals [157], the implementation of a FPAA DNN accelerator, which can be used in biomedical and general applications, is yet to be explored.

E. Benchmarking EMG Processing Across Multiple DNN and SNN Hardware Platforms

In Table V, we compare our FPGA and memristive implementations to other DNN accelerators and neuromorphic processors from [45]. In [45], the authors presented a sensor fusion neuromorphic benchmark for hand-gesture recognition based on EMG and event-based camera. Two neuromorphic platforms, Loihi [138] and ODIN+MorphIC [139], [140], were deployed and the results were compared to traditional machine learning baselines implemented on an embedded GPU, the NVIDIA Jetson Nano. Loihi and ODIN+MorphIC are digital neuromorphic platforms. Loihi is a 128-core neuromorphic chip fabricated on 14 nm FinFET process, designed by Intel Labs. It implements adaptive self-modifying event-driven fine-grained parallel computations used to implement learning and inference with high efficiency. ODIN (Online-learning Digital spiking Neuromorphic) is designed using 28 nm FDSOI CMOS technology and consists of a single neurosynaptic core with 256 neurons and 256^2 synapses that embed a 3-bit weight and a mapping table bit that allows enabling or disabling Spike-timing-dependent plasticity (STDP). MorphIC is a quad-core digital neuromorphic processor with 2k Leaky Integrate and Fire (LIF) neurons and more than 2M synapses in 65 nm CMOS technology [140]. They can be either programmed with offline-trained weights or trained online with a stochastic version of Spike Driven Synaptic Plasticity (SDSP).

For the spiking architectures shown in Table V, the vision input and EMG data were individually processed using spiking CNN and spiking MLP respectively, and fused in the last layer. Loihi was trained using SLAYER [158], a backpropagation framework used for evaluating the gradient of any kind of SNN. It is a dt-based SNN backpropagation algorithm that keeps track of the internal membrane potential of the spiking neuron and uses it during gradient propagation. Both ODIN and Morphic training was carried out in Keras with quantization-aware stochastic gradient descent following a standard ANN-to-SNN mapping approach.

The dataset used is described in Section II-E. It is a collection of 5 hand gestures from sign language (e.g. ILY)[4]. In the comparison proposed in Table V the input and hidden layers are sequenced with the ReLU activation function, and output layers are fed through Softmax activation functions to determine class probabilities. Dropout layers are used in all networks to avoid over-fitting. The DNN architectures are determined in the table caption.

The platforms used for each system in Table V are as follows: ODIN+MorphIC [139], [140] and Loihi [138] neuromorphic platforms were used for spiking implementations; NVIDIA Jetson Nano was used for all embedded GPU implementations; OpenVINO Toolkit FPGA was used for all FPGA implementations, and MemTorch [108] was used for converting the MLP and CNN networks to their corresponding MDNNs to determine the test set accuracies of all memristive implementations.

[4]https://zenodo.org/record/3663616#.X2m5GC2cbx4. Further implementation details can be found in [45].

TABLE V

COMPARISON OF CONVENTIONAL DNNS IMPLEMENTED ON VARIOUS HARDWARE PLATFORMS WITH SPIKING DNN NEUROMORPHIC SYSTEMS ON THE BENCHMARK BIOMEDICAL SIGNAL PROCESSING TASK OF HAND GESTURE RECOGNITION FOR BOTH SINGLE SENSOR AND SENSOR FUSION, AS EXPLAINED IN SUBSECTION II-E. THE RESULTS OF THE ACCURACY ARE REPORTED WITH MEAN AND STANDARD DEVIATION OBTAINED OVER A 3-FOLD CROSS VALIDATION. LOIHI, EMBEDDED GPU, AND ODIN+MORPHIC IMPLEMENTATION RESULTS ARE FROM [45]. THE DNN ARCHITECTURES ADOPTED ARE AS FOLLOWS: $^{\diamond}$8C3-2P-16C3-2P-32C3-512-5 CNN. †16-128-128-5 MLP. ‡16-230-5 MLP. $^{\mp}4 \times 400$-210-5 MLP. $^{\cup}$EMG AND APS/DVS NETWORKS ARE FUSED USING A 5-NEURON DENSE LAYER.

Platform	Modality	Accuracy (%)	Energy (uJ)	Inference time (ms)	EDP (uJ * s)
Loihi (Spiking)	EMG (MLP†)	55.7 ± 2.7	173.2 ± 21.2	5.89 ± 0.18	1.0 ± 0.1
	DVS (CNN$^{\diamond}$)	92.1 ± 1.2	815.3 ± 115.9	6.64 ± 0.14	5.4 ± 0.8
	EMG+DVS (CNN$^{\cup}$)	96.0 ± 0.4	1104.5 ± 58.8	7.75 ± 0.07	8.6 ± 0.5
ODIN+MorphIC (Spiking)	EMG (MLP‡)	53.6 ± 1.4	7.42 ± 0.11	23.5 ± 0.35	0.17 ± 0.01
	DVS (MLP$^{\mp}$)	85.1 ± 4.1	57.2 ± 6.8	17.3 ± 2.0	1.00 ± 0.24
	EMG+DVS (MLP$^{\cup}$)	89.4 ± 3.0	37.4 ± 4.2	19.5 ± 0.3	0.42 ± 0.08
Embedded GPU	EMG (MLP†)	68.1 ± 2.8	(25.5 ± 8.4) $\cdot 10^3$	3.8 ± 0.1	97.3 ± 4.4
	EMG (MLP‡)	67.2 ± 3.6	(23.9 ± 5.6) $\cdot 10^3$	2.8 ± 0.08	67.2 ± 2.9
	APS (CNN$^{\diamond}$)	92.4 ± 1.6	(31.7 ± 7.4) $\cdot 10^3$	5.9 ± 0.1	186.9 ± 3.9
	APS (MLP$^{\mp}$)	84.2 ± 4.3	(30.2 ± 7.5) $\cdot 10^3$	6.9 ± 0.1	211.3 ± 6.1
	EMG+APS (CNN$^{\cup}$)	95.4 ± 1.7	(32.1 ± 7.9) $\cdot 10^3$	6.9 ± 0.05	221.1 ± 4.1
	EMG+APS (MLP$^{\cup}$)	88.1 ± 4.1	(32.0 ± 8.9) $\cdot 10^3$	7.9 ± 0.05	253 ± 3.9
FPGA	EMG (MLP†)	67.2 ± 2.3	(17.6 ± 1.1) 10^3	4.2 ± 0.1	74.1 ± 1.2
	EMG (MLP‡)	63.8 ± 1.4	(13.9 ± 1.8) $\cdot 10^3$	3.5 ± 0.1	48.9 ± 1.9
	APS (CNN$^{\diamond}$)	96.7 ± 3.0	(24.0 ± 1.2) 10^3	5.4 ± 0.2	130.8 ± 1.4
	APS (MLP$^{\mp}$)	82.9 ± 8.4	(23.1 ± 2.6) $\cdot 10^3$	5.7 ± 0.2	131.4 ± 2.8
	EMG+APS (CNN$^{\cup}$)	94.8 ± 2.0	(31.2 ± 3.0) 10^3	6.3 ± 0.1	196.3 ± 3.1
	EMG+APS (MLP$^{\cup}$)	83.4 ± 2.8	(31.1 ± 1.4) $\cdot 10^3$	7.3 ± 0.2	228.2 ± 1.6
Memristive	EMG (MLP†)	64.6 ± 2.2	0.038	6.0 $\cdot 10^{-4}$	2.38 $\cdot 10^{-8}$
	EMG (MLP‡)	63.8 ± 1.4	0.026	4.0 $\cdot 10^{-4}$	1.04 $\cdot 10^{-8}$
	APS (CNN$^{\diamond}$)	96.2 ± 3.3	4.83	1.0 $\cdot 10^{-3}$	4.83 $\cdot 10^{-6}$
	APS (MLP$^{\mp}$)	82.4 ± 8.5	0.18	4.0 $\cdot 10^{-4}$	7.2 $\cdot 10^{-8}$
	EMG+APS (CNN$^{\cup}$)	94.8 ± 2.0	4.90	1.2 $\cdot 10^{-3}$	5.88 $\cdot 10^{-6}$
	EMG+APS (MLP$^{\cup}$)	83.4 ± 2.8	0.33	6.0 $\cdot 10^{-4}$	1.98 $\cdot 10^{-7}$

From Table V, it can be observed that, when transitioning from generalized architectures to application specific processors, more optimized processing of a subset of given tasks can be achieved. Moving up the specificity hierarchy from GPU to FPGA to memristive networks shows orders of magnitude of improvement in both MLP and CNN processing, but naturally at the expense of a generalizable range of tasks. While GPUs are relatively efficient at training networks (compared to CPUs), the impressive metrics presented by memristor (RRAM in this simulations) is coupled with limited endurance. This is not an issue for read-only tasks, as is the case with inference, but training is thwarted by the thousands of epochs of weight updates which limits broad use of RRAMs in training. Rather, more exploration in alternative resistive-based technologies such as Magnetoresistive Random Access Memory (MRAM) could prove beneficial for tasks that demand high endurance.

After determining the test set accuracy of each MDNN using MemTorch [108], we determined the energy required to perform inference on a single input, the inference time, and the Energy-Delay Product (EDP) by adopting the metrics in [159], for a tiled memristor architecture. All assumptions made in our calculations are listed below. Parameters are adopted from those given in a 1T1R 65nm technology, where the maximum current during inference is 3μA per cell with a read voltage of 0.3V. Each cell is capable of storing 8 bits with a resistance ratio of 100, and mapping signed weights is achieved using a dual column representation. All convolutions are performed by unrolling the kernels and performing MVMs, and the fully connected layers have the fan-in weights for a single neuron assigned to one column. Each crossbar has an aspect ratio of 256×64 to enable more analog operations per ADC when compared to a 128×128 array. Where there is insufficient space to map weights to a single array, they are distributed across multiple arrays, with their results to be added digitally. Throughput can be improved at the expense of additional arrays for convolutional layers, by duplicating kernels such that multiple inputs can be processed in parallel. The number of tiles used for each network is assumed to be the exact number required to balance the processing time of each layer. The power consumption of each current-mode 8-bit ADC is estimated to be 2×10^{-4} W with an operating frequency of 40 MHz (5 MHz for bit-serial operation) [159]. The ADC latency is presumed to dominate digital addition of partial products from various tiles. The dynamic range of each ADC has been adapted to the maximum possible range for each column, and each ADC occupies a pair of columns.

The above presumptions lead to pre-silicon results that are extremely promising for memristor arrays, as shown in Table V. But it should be clear that these calculations were performed for *network-specific* architectures, rather than a more general application-specific use-case. That is, we assume the chip has been designed for a given neural network model. The other comparison benchmarks are far more generalizable,

in that they are suited to not only handle most network topologies, but are also well-suited for training. The substantial improvement of inference time over other methods is a result of duplicate weights being mapped to enable higher parallelism, which is tolerable for small architectures, but lends to prohibitively large ADC power consumption for computer vision tasks which rely on deep networks and millions of parameters, such as VGG-16. In addition, the area of each ADC is estimated to be 3×10^{-3}mm^2, which is orders of magnitude larger than the area of each RRAM cell (1.69×10^{-7}mm^2). This disparity implies that pitch-matching is not viable. Instead, to achieve parallelism, weights must be duplicated across tiles which demands redundancy. This improvement in parallelism thus comes at the cost of additional area and power consumption. The use of memristors as synapses in spike-based implementations may be more appropriate, so as to reduce the ADC overhead by replacing multi-bit ADCs with current sense amplifiers instead, and reducing the reliance on analog current summation along resistive and capacitive bitlines.

Spike-based hardware show approximately two orders of magnitude improvement in the EDP from Table V when compared to their GPU and FPGA counterparts, which highlights the prospective use of such architectures in always-on monitoring. This is necessary for enhancing the prospect of ambient-assisted living, which would allow medical resources to be freed up for tasks that are not suited for automation. In general, one would expect that data should be processed in its naturalized form. For example, 2D CNNs do not discard the spatial relations between pixels in an image. Graph networks are optimized for connectionist data, such as the structure of proteins. By extension, the discrete events generated by electrical impulses such as in EMGs, EEGs and ECGs may also be optimized for SNNs. Of course, this discounts any subthreshold firing patterns of measured neuron populations. But one possible explanation for the suitability of spiking hardware for biological processes stems from the natural timing of neuronal action potentials. Individual neurons will typically not fire in excess of 100 Hz, and the average heart rate (and correspondingly, ECG spiking rate) will not exceed 3 Hz. There is a clear mismatch between the clock rate of non-spiking neural network hardware, which tend to at least be in the MHz range, and spike-driven processes. This introduces a significant amount of wastage in processing data when there is no new information to process (e.g., in between heartbeats, action potentials, or neural activity).

Nonetheless, it is clear that accuracy is compromised when relying on EMG signals alone, based on the approximately 10% decrease of classification accuracy on the Loihi chip and ODIN+MorphIC, as against their GPU/FPGA counterparts. This could be a result of spike-based training algorithms lagging behind in maturity compared to conventional neural network methods, or it could be an indication that critical information is being discarded when neglecting the subthreshold signals generated by populations of neurons. But when EMG and DVS data are combined, this multi-sensory data fusion of spiking signals positively reinforce upon each other with an approximately 4% accuracy improvement, whereas combining non-spiking, mismatched data representations leads to marginal improvements, and even a destructive effect (e.g., non-spiking CNN implementation on FPGA and memristive arrays). This may be a result of EMG and APS data taking on completely different structures. This is a possible indication that feature extraction from merging the same structural form of data (i.e., as spikes) proves to be more beneficial than combining a pair of networks with two completely different modes of data (i.e., EMG signals with pixel-driven images). This allows us to draw an important hypothesis: neural networks can benefit from a consistent representation of data generated by various sensory mechanisms. This is supported by biology, where all biological interpretations are typically represented by graded or spiking action potentials.

F. Deep Network Accelerators and Patient-specific Model Tuning

Given the inherent variability between patients, it is difficult to train and deploy a single model to a large group of individuals each with unique signature(s). Consequently, significant efforts are being made to facilitate patient-specific model tuning processes [160], [161], [72]. Patient-specific Modeling (PSM) is the development of computational models of human or animal pathophysiology that are individualized to patient-specific data [160].

In the DL domain, existing ANN and neuromorphic models can be retrofitted to specific patients using transfer learning and tuning algorithms. In this approach, the network is first trained on a large dataset including data from various patients to acquire the domain-specific knowledge of the targeted task. Parts of the large network are then retrained, i.e. tuned, using patient-specific data, to produce better performance for individual patients. This way, the domain-specific features of the large network are transferred to the smaller network that is retrained to learn patient-specific features [72]. Depending on the availability of patient-specific data, PSM can be performed online (on-chip) or offline (off-chip).

1) Online patient-specific model tuning: Considering concerns surrounding the sensitive nature of individual patient data, and the ability of some recent edge-AI CMOS chips such as LNPU [55] to perform online training, patient-specific model tuning can be performed online on the hardware deep learning accelerator. To achieve this, a sufficient amount of patient data that is fed to the accelerator over time can be gathered to individualize the initial generic model. An accelerator that can adapt its working to the specific needs of a patient would be highly beneficial but it may require buffering of data [162], which needs higher on-chip memory and may introduce power overheads.

2) Offline patient-specific model tuning: A convenient approach to tune general models, with domain-specific knowledge, to patient-specific data is offline off-chip transfer learning. However, unlike online tuning, the offline approach requires prior patient data measurements, which may not be readily available. Besides, the offline approach may require undesired remote storage and processing of private patient data to retrain and tune generic models.

V. Conclusion

The use of DL in biomedical signal processing and healthcare promises significant utility for medical practitioners and their patients. DNNs can be used to improve the quality of life for chronically ill patients by enabling ambient monitoring for abnormalities, and correspondingly can reduce the burden on medical resources. Proper use can lead to reduced workloads for medical practitioners who may divert their attention to time-critical tasks that require a standard beyond what neural networks can achieve at this point in time.

We have stepped through the use of various DL accelerators on a disparate range of medical tasks, and shown how SNNs may complement DNNs where hardware efficiency is the primary bottleneck for widespread integration. We have provided a balanced view to how memristors may lead to optimal hardware processing of both DNNs and SNNs, and have highlighted the challenges that must be overcome before they can be adopted at a large-scale. While the focus of this tutorial and review is on hardware implementation of various DL algorithms, the reader should be mindful that progress in hardware is a necessary, but insufficient, condition for successful integration of medical-AI.

Adopting medical-AI tools is clearly a challenge that demands the collaborative attention of healthcare providers, hardware and software engineers, data scientists, policy-makers, cognitive neuroscientists, device engineers and materials scientists, amongst other specializations. A unified approach to developing better hardware can have pervasive impacts upon the healthcare industry, and realize significant payoff by improving the accessibility and outcomes of healthcare.

Acknowledgment

M. Rahimi Azghadi acknowledges a JCU Rising Start ECR Fellowship. C. Lammie acknowledges the JCU DRTPS. This paper is supported in part by the European Union's Horizon 2020 ERC project NeuroAgents (Grant No. 724295). It was also partially funded by EU H2020 grants 824164 "HERMES", 871371 "Memscales", PCI2019-111826-2 "APROVIS3D", and by Spanish grant from the Ministry of Science and Innovation PID2019-105556GB-C31 (NANOMIND) (with support from the European Regional Development Fund).

References

[1] G. Rong, A. Mendez, E. B. Assi, B. Zhao, and M. Sawan, "Artificial Intelligence in Healthcare: Review and Prediction Case Studies," *Engineering*, vol. 6, no. 3, pp. 291–301, 2020.

[2] T. Arevalo, "The State of Health Care Industry- Statistics & Facts," PolicyAdvice, Tech. Rep., Apr. 2020.

[3] V. Jindal, "Integrating Mobile and Cloud for PPG Signal Selection to Monitor Heart Rate during Intensive Physical Exercise," in *Proceedings of the International Conference on Mobile Software Engineering and Systems (MOBILESOFT)*, Austin, TX., May 2016, pp. 36–37.

[4] P. Sundaravadivel, K. Kesavan, L. Kesavan, S. P. Mohanty, and E. Kougianos, "Smart-Log: A Deep-Learning Based Automated Nutrition Monitoring System in the IoT," *IEEE Transactions on Consumer Electronics*, vol. 64, no. 3, pp. 390–398, 2018.

[5] B. Shi, L. J. Grimm, M. A. Mazurowski, J. A. Baker, J. R. Marks, L. M. King, C. C. Maley, E. S. Hwang, and J. Y. Lo, "Prediction of occult invasive disease in ductal carcinoma in situ using deep learning features," *Journal of the American College of Radiology*, vol. 15, no. 3, pp. 527–534, 2018.

[6] X. Liu, L. Faes, A. U. Kale, S. K. Wagner, D. J. Fu, A. Bruynseels, T. Mahendiran, G. Moraes, M. Shamdas, C. Kern *et al.*, "A comparison of deep learning performance against health-care professionals in detecting diseases from medical imaging: a systematic review and meta-analysis," *The Lancet Digital Health*, vol. 1, no. 6, pp. e271–e297, 2019.

[7] F. Liu, P. Yadav, A. M. Baschnagel, and A. B. McMillan, "MR-based Treatment Planning in Radiation Therapy Using a Deep Learning Approach," *Journal of Applied Clinical Medical Physics*, vol. 20, no. 3, pp. 105–114, 2019.

[8] W. Zhu, L. Xie, J. Han, and X. Guo, "The Application of Deep Learning in Cancer Prognosis Prediction," *Cancers*, vol. 12, no. 3, p. 603, 2020.

[9] S. M. McKinney, M. Sieniek, V. Godbole, J. Godwin, N. Antropova, H. Ashrafian, T. Back, M. Chesus, G. C. Corrado, A. Darzi *et al.*, "International evaluation of an AI system for breast cancer screening," *Nature*, vol. 577, no. 7788, pp. 89–94, 2020.

[10] A. Y. Hannun, P. Rajpurkar, M. Haghpanahi, G. H. Tison, C. Bourn, M. P. Turakhia, and A. Y. Ng, "Cardiologist-level arrhythmia detection and classification in ambulatory electrocardiograms using a deep neural network," *Nature medicine*, vol. 25, no. 1, p. 65, 2019.

[11] A. Esteva, B. Kuprel, R. A. Novoa, J. Ko, S. M. Swetter, H. M. Blau, and S. Thrun, "Dermatologist-level classification of skin cancer with deep neural networks," *Nature*, vol. 542, no. 7639, pp. 115–118, 2017.

[12] T. Kalaiselvi, P. Sriramakrishnan, and K. Somasundaram, "Survey of using GPU CUDA programming model in medical image analysis," *Informatics in Medicine Unlocked*, vol. 9, pp. 133–144, 2017.

[13] N. Jouppi, C. Young, N. Patil, D. Patterson, G. Agrawal, R. Bajwa, S. Bates, S. Bhatia, N. Boden, A. Borchers, R. Boyle, P.-l. Cantin, C. Chao, C. Clark, J. Coriell, M. Daley, M. Dau, J. Dean, B. Gelb, T. V. Ghaemmaghami, R. Gottipati, W. Gulland, R. Hagmann, C. R. Ho, D. Hogberg, J. Hu, R. Hundt, D. Hurt, J. Ibarz, A. Jaffey, A. Jaworski, A. Kaplan, H. Khaitan, D. Killebrew, A. Koch, N. Kumar, S. Lacy, J. Laudon, J. Law, D. Le, C. Leary, Z. Liu, K. Lucke, A. Lundin, G. MacKean, A. Maggiore, M. Mahony, K. Miller, R. Nagarajan, R. Narayanaswami, R. Ni, K. Nix, T. Norrie, M. Omernick, N. Penukonda, A. Phelps, J. Ross, M. Ross, A. Salek, E. Samadiani, C. Severn, G. Sizikov, M. Snelham, J. Souter, D. Steinberg, A. Swing, M. Tan, G. Thorson, B. Tian, H. Toma, E. Tuttle, V. Vasudevan, R. Walter, W. Wang, E. Wilcox, and D. H. Yoon, "In-Datacenter Performance Analysis of a Tensor Processing Unit," in *Proceedings of the International Symposium on Computer Architecture (ISCA)*, Toronto, Canada., Jun. 2017.

[14] A. Esteva, A. Robicquet, B. Ramsundar, V. Kuleshov, M. DePristo, K. Chou, C. Cui, G. Corrado, S. Thrun, and J. Dean, "A guide to deep learning in healthcare," *Nature Medicine*, vol. 25, no. 1, pp. 24–29, 2019.

[15] R. Perrault, Y. Shoham, E. Brynjolfsson, J. Clark, J. Etchemendy, B. Grosz, T. Lyons, J. Manyika, S. Mishra, and J. C. Niebles, "The ai index 2019 annual report," *AI Index Steering Committee, Human-Centered AI Institute, Stanford University, Stanford, CA*, 2019.

[16] N. G. Peter *et al.*, "NVIDIA Fermi: The First Complete GPU Computing Architecture," *A White Paper of NVIDIA*, 2009.

[17] Z. Jia, M. Maggioni, B. Staiger, and D. P. Scarpazza, "Dissecting the NVIDIA Volta GPU Architecture via Microbenchmarking," *arXiv preprint arXiv:1804.06826*, 2018.

[18] R. Zemouri, N. Zerhouni, and D. Racoceanu, "Deep Learning in the Biomedical Applications: Recent and Future Status," *Applied Sciences*, vol. 9, no. 8, p. 1526, 2019.

[19] E. Smistad, T. L. Falch, M. Bozorgi, A. C. Elster, and F. Lindseth, "Medical Image Segmentation on GPUs–A Comprehensive Review," *Medical Image Analysis*, vol. 20, no. 1, pp. 1–18, 2015.

[20] B. Farahani, F. Firouzi, and K. Chakrabarty, "Healthcare IoT," in *Intelligent Internet of Things*. Springer, 2020, pp. 515–545.

[21] Q. Xie, K. Faust, R. Van Ommeren, A. Sheikh, U. Djuric, and P. Diamandis, "Deep Learning for Image Analysis: Personalizing Medicine Closer to the Point of Care," *Critical Reviews in Clinical Laboratory Sciences*, vol. 56, no. 1, pp. 61–73, 2019.

[22] M. Hartmann, U. S. Hashmi, and A. Imran, "Edge computing in smart health care systems: Review, challenges, and research directions," *Transactions on Emerging Telecommunications Technologies*, p. e3710, 2019.

[23] I. Azimi, A. Anzanpour, A. M. Rahmani, T. Pahikkala, M. Levorato, P. Liljeberg, and N. Dutt, "Hich: Hierarchical Fog-Assisted Computing Architecture for Healthcare IoT," *ACM Transactions on Embedded Computing Systems (TECS)*, vol. 16, no. 5s, pp. 1–20, 2017.

[24] K. Sethi, V. Parmar, and M. Suri, "Low-Power Hardware-Based Deep-Learning Diagnostics Support Case Study," in *Proceedings of the IEEE Biomedical Circuits and Systems Conference (BioCAS)*, Cleveland, OH., Oct. 2018.

[25] P. Sahu, D. Yu, and H. Qin, "Apply lightweight deep learning on internet of things for low-cost and easy-to-access skin cancer detection," in *Proceedings of Medical Imaging 2018: Imaging Informatics for Healthcare, Research, and Applications*, vol. 10579, Houston, TX, Feb. 2018, p. 1057912.

[26] E. Chicca, F. Stefanini, C. Bartolozzi, and G. Indiveri, "Neuromorphic electronic circuits for building autonomous cognitive systems," *Proceedings of the IEEE*, vol. 102, no. 9, pp. 1367–1388, 2014.

[27] D. E. Rumelhart, G. Hinton, and R. J. Williams, "Learning representations by back-propagating errors," *Nature*, vol. 323, no. 6088, pp. 533–538, 1986.

[28] T. C. Hollon, B. Pandian, A. R. Adapa, E. Urias, A. V. Save, S. S. S. Khalsa, D. G. Eichberg, R. S. DAmico, Z. U. Farooq, S. Lewis *et al.*, "Near real-time intraoperative brain tumor diagnosis using stimulated Raman histology and deep neural networks," *Nature Medicine*, pp. 1–7, 2020.

[29] B. Shickel, P. J. Tighe, A. Bihorac, and P. Rashidi, "Deep EHR: A Survey of Recent Advances in Deep Learning Techniques for Electronic Health Record (EHR) Analysis," *IEEE Journal of Biomedical and Health Informatics*, vol. 22, no. 5, pp. 1589–1604, 2017.

[30] M. A. Sayeed, S. P. Mohanty, E. Kougianos, and H. P. Zaveri, "Neuro-Detect: A Machine Learning-Based Fast and Accurate Seizure Detection System in the IoMT," *IEEE Transactions on Consumer Electronics*, vol. 65, no. 3, pp. 359–368, 2019.

[31] J. Yang and M. Sawan, "From seizure detection to smart and fully embedded seizure prediction engine: A review," *IEEE Transactions on Biomedical Circuits and Systems*, vol. 14, no. 5, pp. 1008–1023, 2020.

[32] N. Tajbakhsh, J. Y. Shin, S. R. Gurudu, R. T. Hurst, C. B. Kendall, M. B. Gotway, and J. Liang, "Convolutional Neural Networks for Medical Image Analysis: Full Training or Fine Tuning?" *IEEE Transactions on Medical Imaging*, vol. 35, no. 5, pp. 1299–1312, 2016.

[33] J. Gao, H. Zhang, P. Lu, and Z. Wang, "An Effective LSTM Recurrent Network to Detect Arrhythmia on Imbalanced ECG Dataset," *Journal of Healthcare Engineering*, vol. 2019, 2019.

[34] D. Zhang, L. Yao, X. Zhang, S. Wang, W. Chen, R. Boots, and B. Benatallah, "Cascade and Parallel Convolutional Recurrent Neural Networks on EEG-based Intention Recognition for Brain Computer Interface," in *Proceedings of the AAAI Conference on Artificial Intelligence*, New Orleans, LA., Feb. 2018.

[35] X. Zhou, Y. Li, and W. Liang, "CNN-RNN Based Intelligent Recommendation for Online Medical Pre-Diagnosis Support," *IEEE/ACM Transactions on Computational Biology and Bioinformatics*, 2020.

[36] J. Laitala, M. Jiang, E. Syrjälä, E. K. Naeini, A. Airola, A. M. Rahmani, N. D. Dutt, and P. Liljeberg, "Robust ECG R-peak detection using LSTM," in *Proceedings of the ACM Symposium on Applied Computing (SAC)*, Brno, Czech Republic., Mar. 2020, pp. 1104–1111.

[37] I. Goodfellow, Y. Bengio, and A. Courville, *Deep learning*. MIT press, 2016.

[38] S. J. Pan and Q. Yang, "A Survey on Transfer Learning," *IEEE Transactions on Knowledge and Data Engineering*, vol. 22, no. 10, pp. 1345–1359, 2010.

[39] J. Deng, W. Dong, R. Socher, L.-J. Li, K. Li, and L. Fei-Fei, "Imagenet: A Large-scale Hierarchical Image Database," in *Proceeding of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, Miami Beach, FL., Jun. 2009.

[40] M. Raghu, C. Zhang, J. Kleinberg, and S. Bengio, "Transfusion: Understanding Transfer Learning for Medical Imaging," in *NeurIPS*, Vancouver, Canada., Dec. 2019.

[41] M. S. Elmahdy, T. Ahuja, U. A. van der Heide, and M. Staring, "Patient-specific finetuning of deep learning models for adaptive radiotherapy in prostate ct," in *2020 IEEE 17th International Symposium on Biomedical Imaging (ISBI)*, 2020, pp. 577–580.

[42] G. Indiveri and S.-C. Liu, "Memory and Information Processing in Neuromorphic Systems," *Proceedings of the IEEE*, vol. 103, no. 8, pp. 1379–1397, 2015.

[43] F. Corradi and G. Indiveri, "A Neuromorphic Event-Based Neural Recording System for Smart Brain-Machine-Interfaces," *IEEE Transactions on Biomedical Circuits and Systems*, vol. 9, no. 5, pp. 699–709, 2015.

[44] F. Corradi, S. Pande, J. Stuijt, N. Qiao, S. Schaafsma, G. Indiveri, and F. Catthoor, "ECG-based Heartbeat Classification in Neuromorphic Hardware," in *Proceedings of the International Joint Conference on Neural Networks (IJCNN)*, Budapest, Hungary., Jul. 2019.

[45] E. Ceolini, C. Frenkel, S. B. Shrestha, G. Taverni, L. Khacef, M. Payvand, and E. Donati, "Hand-gesture recognition based on EMG and event-based camera sensor fusion: a benchmark in neuromorphic computing," *Frontiers in Neuroscience*, no. 520438, p. 36, 2020.

[46] G. Indiveri and Y. Sandamirskaya, "The importance of space and time for signal processing in neuromorphic agents: the challenge of developing low-power, autonomous agents that interact with the environment," *IEEE Signal Processing Magazine*, vol. 36, no. 6, pp. 16–28, 2019.

[47] J. K. Eshraghian, K. Cho, C. Zheng, M. Nam, H. H.-C. Iu, W. Lei, and K. Eshraghian, "Neuromorphic vision hybrid rram-cmos architecture," *IEEE Transactions on Very Large Scale Integration (VLSI) Systems*, vol. 26, no. 12, pp. 2816–2829, 2018.

[48] X. Lagorce, G. Orchard, F. Galluppi, B. E. Shi, and R. B. Benosman, "Hots: a hierarchy of event-based time-surfaces for pattern recognition," *IEEE transactions on pattern analysis and machine intelligence*, vol. 39, no. 7, pp. 1346–1359, 2016.

[49] M. Sharifshazileh, K. Burelo, T. Fedele, J. Sarnthein, and G. Indiveri, "A Neuromorphic Device for Detecting High-Frequency Oscillations in Human iEEG," in *Proceedings of the IEEE International Conference on Electronics, Circuits and Systems (ICECS)*, Genova, Italy., Nov. 2019, pp. 69–72.

[50] P. Lichtsteiner, C. Posch, and T. Delbruck, "A 128X128 120 dB 15 us Latency Asynchronous Temporal Contrast Vision Sensor," *IEEE Journal of Solid-state Circuits*, vol. 43, no. 2, pp. 566–576, 2008.

[51] A. Reuther, P. Michaleas, M. Jones, V. Gadepally, S. Samsi, and J. Kepner, "Survey and Benchmarking of Machine Learning Accelerators," *arXiv preprint arXiv:1908.11348*, 2019.

[52] "Edge TPU," https://coral.ai/docs/edgetpu/faq/.

[53] J. Hruska, "Intel Nervana Inference and Training AI Cards," https://www.extremetech.com/computing/296990-intel-nervana-nnp-i-nnp-t-a-training-inference.

[54] P. Kennedy, "Huawei Ascend 310," https://www.servethehome.com/huawei-ascend-910-provides-a-nvidia-ai-training-alternative/.

[55] J. Lee, J. Lee, D. Han, J. Lee, G. Park, and H.-J. Yoo, "LNPU: A 25.3 TFLOPS/W Sparse Deep-Neural-Network Learning Processor with Fine-Grained Mixed Precision of FP8-FP16," in *Proceedings of the IEEE International Solid-State Circuits Conference (ISSCC)*, San Francisco, CA., Feb. 2019, pp. 142–144.

[56] D. Shin, J. Lee, J. Lee, J. Lee, and H.-J. Yoo, "DNPU: An Energy-Efficient Deep-Learning Processor with Heterogeneous Multi-Core Architecture," *IEEE Micro*, vol. 38, no. 5, pp. 85–93, 2018.

[57] S. Yin, P. Ouyang, S. Tang, F. Tu, X. Li, S. Zheng, T. Lu, J. Gu, L. Liu, and S. Wei, "A high energy efficient reconfigurable hybrid neural network processor for deep learning applications," *IEEE Journal of Solid-State Circuits*, vol. 53, no. 4, pp. 968–982, 2017.

[58] J. Lee, C. Kim, S. Kang, D. Shin, S. Kim, and H.-J. Yoo, "UNPU: An Energy-Efficient Deep Neural Network Accelerator With Fully Variable Weight Bit Precision," *IEEE Journal of Solid-State Circuits*, vol. 54, no. 1, pp. 173–185, 2018.

[59] S. Zhang, Z. Du, L. Zhang, H. Lan, S. Liu, L. Li, Q. Guo, T. Chen, and Y. Chen, "Cambricon-X: An accelerator for sparse neural networks," in *Proceedings of the IEEE/ACM International Symposium on Microarchitecture (MICRO)*, Taipei, Taiwan., Oct. 2016.

[60] J. Zhang, S. Gajjala, P. Agrawal, G. H. Tison, L. A. Hallock, L. Beussink-Nelson, E. Fan, M. A. Aras, C. Jordan, K. E. Fleischmann *et al.*, "A Computer Vision Pipeline for Automated Determination of Cardiac Structure and Function and Detection of Disease by Two-Dimensional Echocardiography," *arXiv preprint arXiv:1706.07342*, 2017.

[61] Y.-H. Chen, T. Krishna, J. S. Emer, and V. Sze, "Eyeriss: An Energy-Efficient Reconfigurable Accelerator for Deep Convolutional Neural Networks," *IEEE Journal of Solid-state Circuits*, vol. 52, no. 1, pp. 127–138, 2016.

[62] Q. Guan, Y. Wang, B. Ping, D. Li, J. Du, Y. Qin, H. Lu, X. Wan, and J. Xiang, "Deep Convolutional Neural Network VGG-16 Model for Differential Diagnosing of Papillary Thyroid Carcinomas in Cytological Images: A Pilot Study," *Journal of Cancer*, vol. 10, no. 20, p. 4876, 2019.

[63] L. Cavigelli and L. Benini, "Origami: A 803-GOp/s/W Convolutional Network Accelerator," *IEEE Transactions on Circuits and Systems for Video Technology*, vol. 27, no. 11, pp. 2461–2475, 2016.

[64] I. Azimi, J. Takalo-Mattila, A. Anzanpour, A. M. Rahmani, J.-P. Soininen, and P. Liljeberg, "Empowering Healthcare IoT Systems with Hierarchical Edge-Based Deep Learning," in *Proceedings of the International Conference on Connected Health: Applications, Systems*

and Engineering Technologies (CHASE), Washington, DC., Sep. 2018, pp. 63–68.

[65] J. Huang, S. Lin, N. Wang, G. Dai, Y. Xie, and J. Zhou, "TSE-CNN: A Two-Stage End-to-End CNN for Human Activity Recognition," *IEEE Journal of Biomedical and Health Informatics*, vol. 24, no. 1, pp. 292–299, 2020.

[66] B. Moons and M. Verhelst, "An Energy-Efficient Precision-Scalable ConvNet Processor in 40-nm CMOS," *IEEE Journal of Solid-state Circuits*, vol. 52, no. 4, pp. 903–914, 2016.

[67] M. Blaivas and L. Blaivas, "Are All Deep Learning Architectures Alike for Point-of-Care Ultrasound?: Evidence From a Cardiac Image Classification Model Suggests Otherwise," *Journal of Ultrasound in Medicine*, 2019.

[68] B. Moons, R. Uytterhoeven, W. Dehaene, and M. Verhelst, "Envision: A 0.26-to-10TOPS/W subword-parallel dynamic-voltage-accuracy-frequency-scalable Convolutional Neural Network processor in 28nm FDSOI," in *Proceedings of the IEEE International Solid-State Circuits Conference (ISSCC)*, San Francisco, CA., Feb. 2017, pp. 246–247.

[69] M.-P. Hosseini, T. X. Tran, D. Pompili, K. Elisevich, and H. Soltanian-Zadeh, "Deep Learning with Edge Computing for Localization of Epileptogenicity Using Multimodal rs-fMRI and EEG Big Data," in *Proceedings of the IEEE International Conference on Autonomic Computing (ICAC)*, Columbus, OH., Jul. 2017, pp. 83–92.

[70] J. Song, Y. Cho, J.-S. Park, J.-W. Jang, S. Lee, J.-H. Song, J.-G. Lee, and I. Kang, "An 11.5TOPS/W 1024-MAC Butterfly Structure Dual-Core Sparsity-Aware Neural Processing Unit in 8nm Flagship Mobile SoC," in *Proceedings of the IEEE International Solid-State Circuits Conference (ISSCC)*, San Francisco, CA., Feb. 2019, pp. 130–132.

[71] F. Preiswerk, C.-C. Cheng, J. Luo, and B. Madore, "Synthesizing Dynamic MRI Using Long-Term Recurrent Convolutional Networks," in *Proceedings of the International Workshop on Machine Learning in Medical Imaging (MLMI)*. Granada, Spain.: Springer, Sep. 2018, pp. 89–97.

[72] J. Acharya and A. Basu, "Deep Neural Network for Respiratory Sound Classification in Wearable Devices Enabled by Patient Specific Model Tuning," *IEEE Transactions on Biomedical Circuits and Systems*, vol. 14, no. 3, pp. 535–544, 2020.

[73] M. S. Roy, B. Roy, R. Gupta, and K. D. Sharma, "On-device reliability assessment and prediction of missing photoplethysmographic data using deep neural networks," *IEEE transactions on biomedical circuits and systems*.

[74] J. P. Queralta, T. N. Gia, H. Tenhunen, and T. Westerlund, "Edge-AI in LoRa-based Health Monitoring: Fall Detection System with Fog Computing and LSTM Recurrent Neural Networks," in *Proceedings of the International Conference on Telecommunications and Signal Processing (TSP)*, 2019, pp. 601–604.

[75] S. Shaikh, R. So, T. Sibindi, C. Libedinsky, and A. Basu, "Sparse Ensemble Machine Learning to Improve Robustness of Long-Term Decoding in iBMIs," *IEEE Transactions on Neural Systems and Rehabilitation Engineering*, vol. 28, no. 2, pp. 380–389, 2020.

[76] I. M. Baltruschat, H. Nickisch, M. Grass, T. Knopp, and A. Saalbach, "Comparison of Deep Learning Approaches for Multi-Label Chest X-Ray Classification," *Scientific Reports*, vol. 9, no. 1, pp. 1–10, 2019.

[77] K. Zhao, H. Jiang, Z. Wang, P. Chen, B. Zhu, and X. Duan, "Long-term bowel sound monitoring and segmentation by wearable devices and convolutional neural networks," *IEEE Transactions on Biomedical Circuits and Systems*, 2020.

[78] G. Zamzmi, L.-Y. Hsu, W. Li, V. Sachdev, and S. Antani, "Harnessing Machine Intelligence in Automatic Echocardiogram Analysis: Current Status, Limitations, and Future Directions," *IEEE Reviews in Biomedical Engineering*, 2020.

[79] Y. Chen, E. Yao, and A. Basu, "A 128-Channel Extreme Learning Machine-Based Neural Decoder for Brain Machine Interfaces," *IEEE Transactions on Biomedical Circuits and Systems*, vol. 10, no. 3, pp. 679–692, 2016.

[80] G. O'Leary, D. M. Groppe, T. A. Valiante, N. Verma, and R. Genov, "NURIP: Neural Interface Processor for Brain-State Classification and Programmable-Waveform Neurostimulation," *IEEE Journal of Solid-State Circuits*, vol. 53, no. 11, pp. 3150–3162, 2018.

[81] L. Deng, G. Li, S. Han, L. Shi, and Y. Xie, "Model compression and hardware acceleration for neural networks: A comprehensive survey," *Proceedings of the IEEE*, vol. 108, no. 4, pp. 485–532, 2020.

[82] V. Sze, Y.-H. Chen, T.-J. Yang, and J. S. Emer, "Efficient processing of deep neural networks: A tutorial and survey," *Proceedings of the IEEE*, vol. 105, no. 12, pp. 2295–2329, 2017.

[83] P.-H. Pham, D. Jelaca, C. Farabet, B. Martini, Y. LeCun, and E. Culurciello, "NeuFlow: Dataflow vision processing system-on-a-chip," in *Proceedings of the IEEE International Midwest Symposium on Circuits and Systems (MWSCAS)*, Fort Collins, CO., Aug. 2012, pp. 1044–1047.

[84] A. Putnam, A. M. Caulfield, E. S. Chung, D. Chiou, K. Constantinides, J. Demme, H. Esmaeilzadeh, J. Fowers, G. P. Gopal, J. Gray *et al.*, "A reconfigurable fabric for accelerating large-scale datacenter services," in *Proceedings of the ACM/IEEE International Symposium on Computer Architecture (ISCA)*, Minneapolis, MN., Jun. 2014, pp. 13–24.

[85] D. Abts, J. Ross, J. Sparling, M. Wong-VanHaren, M. Baker, T. Hawkins, A. Bell, J. Thompson, T. Kahsai, G. Kimmell *et al.*, "Think Fast: A Tensor Streaming Processor (TSP) for Accelerating Deep Learning Workloads," Valencia, Spain., May 2020.

[86] H. Kung and C. E. Leiserson, "Systolic Arrays (for VLSI)," in *Proceedings of Sparse Matrix*, vol. 1. Society for industrial and applied mathematics, 1979, pp. 256–282.

[87] H.-T. Kung, "Why systolic architectures?" *Computer*, no. 1, pp. 37–46, 1982.

[88] J. E. Stone, D. Gohara, and G. Shi, "OpenCL: A parallel programming standard for heterogeneous computing systems," *Computing in Science & Engineering*, vol. 12, no. 3, pp. 66–73, 2010.

[89] K. Guo, S. Zeng, J. Yu, Y. Wang, and H. Yang, "A Survey of FPGA-Based Neural Network Inference Accelerator," *ACM Transactions on Reconfigurable Technology and Systems (TRETS)*, vol. 12, no. 1, pp. 1–26, 2019.

[90] J. Cong, B. Liu, S. Neuendorffer, J. Noguera, K. Vissers, and Z. Zhang, "High-Level Synthesis for FPGAs: From Prototyping to Deployment," *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems*, vol. 30, no. 4, pp. 473–491, 2011.

[91] C. Lammie, W. Xiang, and M. R. Azghadi, "Accelerating Deterministic and Stochastic Binarized Neural Networks on FPGAS using OpenCL," in *Proceedings of the IEEE International Midwest Symposium on Circuits and Systems (MWSCAS)*, Dallas, TX., Aug. 2019, pp. 626–629.

[92] C. Lammie, A. Olsen, T. Carrick, and M. R. Azghadi, "Low-Power and High-Speed Deep FPGA Inference Engines for Weed Classification at the Edge," *IEEE Access*, vol. 7, pp. 51 171–51 184, 2019.

[93] M. Carreras, G. Deriu, L. Raffo, L. Benini, and P. Meloni, "Optimizing Temporal Convolutional Network inference on FPGA-based accelerators," *arXiv preprint arXiv:2005.03775*, 2020.

[94] C. Lammie, W. Xiang, and M. R. Azghadi, "Training progressively binarizing deep networks using fpgas," in *2020 IEEE International Symposium on Circuits and Systems (ISCAS)*, 2020, pp. 1–5.

[95] C. Lammie and M. R. Azghadi, "Stochastic Computing for Low-Power and High-Speed Deep Learning on FPGA," in *Proceedings of the IEEE International Symposium on Circuits and Systems (ISCAS)*, Sapporo, Japan., May 2019.

[96] D. Wang, K. Xu, and D. Jiang, "PipeCNN: An OpenCL-based open-source FPGA accelerator for convolution neural networks," in *Proceedings of the International Conference on Field Programmable Technology (ICFPT)*, Melbourne, Australia., Dec. 2017, pp. 279–282.

[97] M. Wess, P. S. Manoj, and A. Jantsch, "Neural network based ECG anomaly detection on FPGA and trade-off analysis," in *Proceedings of the IEEE International Symposium on Circuits and Systems (ISCAS)*, Baltimore, MD., May 2017.

[98] A. Sanaullah, C. Yang, Y. Alexeev, K. Yoshii, and M. C. Herbordt, "Real-time data analysis for medical diagnosis using FPGA-accelerated neural networks," *BMC Bioinformatics*, vol. 19, no. 18, p. 490, 2018.

[99] R. R. Shrivastwa, V. Pudi, and A. Chattopadhyay, "An FPGA-Based Brain Computer Interfacing Using Compressive Sensing and Machine Learning," in *Proceedings of the IEEE Computer Society Annual Symposium on VLSI (ISVLSI)*, Hong Kong, China., Jul. 2018, pp. 726–731.

[100] Z. Chen, A. Howe, H. T. Blair, and J. Cong, "CLINK: Compact LSTM Inference Kernel for Energy Efficient Neurofeedback Devices," in *Proceedings of the International Symposium on Low Power Electronics and Design (ISLPED)*, Bellevue, WA., Jul. 2018.

[101] G. Burr, P. Narayanan, R. Shelby, S. Sidler, I. Boybat, C. di Nolfo, and Y. Leblebici, "Large-scale neural networks implemented with non-volatile memory as the synaptic weight element: Comparative performance analysis (accuracy, speed, and power)," in *Proceedings of the IEEE International Electron Devices Meeting (IEDM)*, Washington, DC., Dec. 2015.

[102] S. Ambrogio, P. Narayanan, H. Tsai, R. M. Shelby, I. Boybat, C. Nolfo, S. Sidler, M. Giordano, M. Bodini, N. C. Farinha *et al.*, "Equivalent-accuracy accelerated neural-network training using analogue memory," *Nature*, vol. 558, no. 7708, p. 60, 2018.

[103] P. Yao, H. Wu, B. Gao, J. Tang, Q. Zhang, W. Zhang, J. J. Yang, and H. Qian, "Fully hardware-implemented memristor convolutional neural network," *Nature*, vol. 577, no. 7792, pp. 641–646, 2020.

[104] J. K. Eshraghian, S.-M. Kang, S. Baek, G. Orchard, H. H.-C. Iu, and W. Lei, "Analog weights in reram dnn accelerators," in *2019 IEEE International Conference on Artificial Intelligence Circuits and Systems (AICAS)*. IEEE, 2019, pp. 267–271.

[105] M. R. Azghadi, B. Linares-Barranco, D. Abbott, and P. H. Leong, "A Hybrid CMOS-Memristor Neuromorphic Synapse," *IEEE Transactions on Biomedical Circuits and Systems*, vol. 11, no. 2, pp. 434–445, 2017.

[106] M. Rahimi Azghadi, Y.-C. Chen, J. K. Eshraghian, J. Chen, C.-Y. Lin, A. Amirsoleimani, A. Mehonic, A. J. Kenyon, B. Fowler, J. C. Lee *et al.*, "Complementary metal-oxide semiconductor and memristive hardware for neuromorphic computing," *Advanced Intelligent Systems*, vol. 2, no. 5, p. 1900189, 2020.

[107] Q. Xia and J. J. Yang, "Memristive crossbar arrays for brain-inspired computing," *Nature Materials*, vol. 18, no. 4, pp. 309–323, 2019.

[108] C. Lammie, W. Xiang, B. Linares-Barranco, and M. R. Azghadi, "MemTorch: An Open-source Simulation Framework for Memristive Deep Learning Systems," *arXiv preprint arXiv:2004.10971*, 2020.

[109] C. Lammie, O. Krestinskaya, A. James, and M. R. Azghadi, "Variation-aware Binarized Memristive Networks," in *Proceedings of the International Conference on Electronics, Circuits and Systems (ICECS)*, Genova, Italy., Nov. 2019, pp. 490–493.

[110] O. Krestinskaya, K. N. Salama, and A. P. James, "Learning in Memristive Neural Network Architectures Using Analog Backpropagation Circuits," *IEEE Transactions on Circuits and Systems I: Regular Papers*, vol. 66, no. 2, pp. 719–732, 2018.

[111] S. Yu, P.-Y. Chen, Y. Cao, L. Xia, Y. Wang, and H. Wu, "Scaling-up resistive synaptic arrays for neuro-inspired architecture: Challenges and prospect," in *Proceedings of the IEEE International Electron Devices Meeting (IEDM)*, Washington, DC., Dec. 2015.

[112] N. Bien, P. Rajpurkar, R. L. Ball, J. Irvin, A. Park, E. Jones, M. Bereket, B. N. Patel, K. W. Yeom, K. Shpanskaya *et al.*, "Deep-learning-assisted diagnosis for knee magnetic resonance imaging: development and retrospective validation of MRNet," *PLoS Medicine*, vol. 15, no. 11, p. e1002699, 2018.

[113] A. Ankit, I. E. Hajj, S. R. Chalamalasetti, G. Ndu, M. Foltin, R. S. Williams, P. Faraboschi, W. Hwu, J. P. Strachan, K. Roy, and D. S. Milojicic, "PUMA: A Programmable Ultra-efficient Memristor-based Accelerator for Machine Learning Inference," *CoRR*, vol. abs/1901.10351, 2019. [Online]. Available: http://arxiv.org/abs/1901.10351

[114] S. Kvatinsky, M. Ramadan, E. G. Friedman, and A. Kolodny, "VTEAM: A General Model for Voltage-controlled Memristors," *IEEE Transactions on Circuits and Systems II: Express Briefs*, vol. 62, no. 8, pp. 786–790, 2015.

[115] E. Yalon, A. Gavrilov, S. Cohen, D. Mistele, B. Meyler, J. Salzman, and D. Ritter, "Resistive Switching in HfO_2 Probed by a Metal–Insulator–Semiconductor Bipolar Transistor," *IEEE Electron Device Letters*, vol. 33, no. 1, pp. 11–13, 2012.

[116] A. M. Hassan, A. F. Khalaf, K. S. Sayed, H. H. Li, and Y. Chen, "Real-Time Cardiac Arrhythmia Classification Using Memristor Neuromorphic Computing System," in *Proceedings of the International Conference of the IEEE Engineering in Medicine and Biology Society (EMBC)*, Honolulu, HI, Jul. 2018, pp. 2567–2570.

[117] F. Cai, J. M. Correll, S. H. Lee, Y. Lim, V. Bothra, Z. Zhang, M. P. Flynn, and W. D. Lu, "A fully integrated reprogrammable memristor–CMOS system for efficient multiply–accumulate operations," *Nature Electronics*, vol. 2, no. 7, pp. 290–299, 2019.

[118] T. Hirtzlin, M. Bocquet, B. Penkovsky, J.-O. Klein, E. Nowak, E. Vianello, J.-M. Portal, and D. Querlioz, "Digital Biologically Plausible Implementation of Binarized Neural Networks With Differential Hafnium Oxide Resistive Memory Arrays," *Frontiers in Neuroscience*, vol. 13, 2019.

[119] F. C. Bauer, D. R. Muir, and G. Indiveri, "Real-time ultra-low power ECG anomaly detection using an event-driven neuromorphic processor," *IEEE Transactions on Biomedical Circuits and Systems*, 2019.

[120] E. Donati, M. Payvand, N. Risi, R. Krause, K. Burelo, G. Indiveri, T. Dalgaty, and E. Vianello, "Processing EMG signals using reservoir computing on an event-based neuromorphic system," in *Proceedings of the IEEE Biomedical Circuits and Systems Conference (BioCAS)*, Cleveland, Ohio., Oct. 2018.

[121] E. Donati, M. Payvand, N. Risi, R. Krause, and G. Indiveri, "Discrimination of EMG Signals Using a Neuromorphic Implementation of a Spiking Neural Network," *IEEE Transactions on Biomedical Circuits and Systems*, vol. 13, no. 5, pp. 795–803, 2019.

[122] J. Behrenbeck, Z. Tayeb, C. Bhiri, C. Richter, O. Rhodes, N. Kasabov, J. I. Espinosa-Ramos, S. Furber, G. Cheng, and J. Conradt, "Classification and regression of spatio-temporal signals using NeuCube and its realization on SpiNNaker neuromorphic hardware," *Journal of Neural Engineering*, vol. 16, no. 2, p. 026014, 2019.

[123] E. Nurse, B. S. Mashford, A. J. Yepes, I. Kiral-Kornek, S. Harrer, and D. R. Freestone, "Decoding EEG and LFP Signals using Deep Learning: Heading TrueNorth," in *Proceedings of the ACM International Conference on Computing Frontiers (CF)*, Como, Italy., May 2016, pp. 259–266.

[124] S. Shaikh, R. So, T. Sibindi, C. Libedinsky, and A. Basu, "Real-time Closed Loop Neural Decoding on a Neuromorphic Chip," in *Proceedings of the IEEE/EMBS International Conference on Neural Engineering (NER)*, San Francisco, CA., Mar. 2019, pp. 670–673.

[125] ——, "Towards Intelligent Intracortical BMI (i^2BMI): Low-Power Neuromorphic Decoders That Outperform Kalman Filters," *IEEE Transactions on Biomedical Circuits and Systems*, vol. 13, no. 6, pp. 1615–1624, 2019.

[126] P. Škoda, T. Lipić, î Srp, B. M. Rogina, K. Skala, and F. Vajda, "Implementation framework for Artificial Neural Networks on FPGA," in *Proceedings of the International Convention (MIPRO)*, Opatija, Croatia., May 2011, pp. 274–278.

[127] C. Heelan, A. V. Nurmikko, and W. Truccolo, "FPGA implementation of deep-learning recurrent neural networks with sub-millisecond real-time latency for BCI-decoding of large-scale neural sensors (104 nodes)." in *Proceedings of the International Conference of the IEEE Engineering in Medicine and Biology Society (EMBC)*, Honolulu, HI., Jul. 2018.

[128] L. G. Rocha, D. Biswas, B. Verhoef, S. Bampi, C. Van Hoof, M. Konijnenburg, M. Verhelst, and N. Van Helleputte, "Binary CorNET: Accelerator for HR Estimation From Wrist-PPG," *IEEE Transactions on Biomedical Circuits and Systems*, vol. 14, no. 4, pp. 715–726, 2020.

[129] A. Jafari, A. Ganesan, C. S. K. Thalisetty, V. Sivasubramanian, T. Oates, and T. Mohsenin, "SensorNet: A Scalable and Low-Power Deep Convolutional Neural Network for Multimodal Data Classification," *IEEE Transactions on Circuits and Systems I: Regular Papers*, vol. 66, no. 1, pp. 274–287, 2019.

[130] L. Ohno-Machado and D. Bialek, "Diagnosing breast cancer from fnas: variable relevance in neural network and logistic regression models," *Studies in Health Technology and Informatics*, vol. 52, pp. 537–540, 1998.

[131] Y. Ku, W. Tompkins, and Q. Xue, "Artificial neural network for ECG arrhythmia monitoring," in *Proceedings of the International Joint Conference on Neural Networks (IJCNN)*, vol. 2. Baltimore, MD.: IEEE, Jun. 1992, pp. 987–992.

[132] J. K. Eshraghian, "Human ownership of artificial creativity," *Nature Machine Intelligence*, pp. 157–160, 2020.

[133] S. Moradi, N. Qiao, F. Stefanini, and G. Indiveri, "A Scalable Multicore Architecture With Heterogeneous Memory Structures for Dynamic Neuromorphic Asynchronous Processors (DYNAPs)," *IEEE Transactions on Biomedical Circuits and Systems*, vol. 12, no. 1, pp. 106–122, Feb. 2018.

[134] S. Benatti, F. Casamassima, B. Milosevic, E. Farella, P. Schönle, S. Fateh, T. Burger, Q. Huang, and L. Benini, "A Versatile Embedded Platform for EMG Acquisition and Gesture Recognition," *IEEE Transactions on Biomedical Circuits and Systems*, vol. 9, no. 5, pp. 620–630, 2015.

[135] F. Montagna, A. Rahimi, S. Benatti, D. Rossi, and L. Benini, "PULP-HD: Accelerating Brain-inspired High-dimensional Computing on a Parallel Ultra-low Power Platform," in *Proceedings of the ACM/ESDA/IEEE Design Automation Conference (DAC)*, San Francisco, CA., Jun. 2018.

[136] S. B. Furber, D. R. Lester, L. A. Plana, J. D. Garside, E. Painkras, S. Temple, and A. D. Brown, "Overview of the SpiNNaker System Architecture," *IEEE Transactions on Computers*, vol. 62, no. 12, pp. 2454–2467, 2013.

[137] P. Merolla and K. Boahen, "A Recurrent Model of Orientation Maps with Simple and Complex Cells," in *Proceedings of Advances in Neural Information Processing Systems 17 (NIPS)*, Vancouver, Canada., Dec. 2004, pp. 1995–2002.

[138] M. Davies, N. Srinivasa, T.-H. Lin, G. Chinya, Y. Cao, S. H. Choday, G. Dimou, P. Joshi, N. Imam, S. Jain *et al.*, "Loihi: A Neuromorphic Manycore Processor with On-Chip Learning," *IEEE Micro*, vol. 38, no. 1, pp. 82–99, 2018.

[139] C. Frenkel, M. Lefebvre, J.-D. Legat, and D. Bol, "A 0.086-mm^2 12.7-pJ/SOP 64k-Synapse 256-Neuron Online-Learning Digital Spiking

Neuromorphic Processor in 28-nm CMOS," *IEEE Transactions on Biomedical Circuits and Systems*, vol. 13, no. 1, pp. 145–158, 2019.

[140] C. Frenkel, J.-D. Legat, and D. Bol, "MorphIC: A 65-nm 738k-Synapse/mm ^{2} Quad-Core Binary-Weight Digital Neuromorphic Processor With Stochastic Spike-Driven Online Learning," *IEEE Transactions on Biomedical Circuits and Systems*, vol. 13, no. 5, pp. 999–1010, 2019.

[141] N. Qiao, H. Mostafa, F. Corradi, M. Osswald, F. Stefanini, D. Sumislawska, and G. Indiveri, "A reconfigurable on-line learning spiking neuromorphic processor comprising 256 neurons and 128K synapses," *Frontiers in neuroscience*, vol. 9, p. 141, 2015.

[142] M. R. Azghadi, S. Moradi, D. B. Fasnacht, M. S. Ozdas, and G. Indiveri, "Programmable spike-timing-dependent plasticity learning circuits in neuromorphic VLSI architectures," *ACM Journal on Emerging Technologies in Computing Systems (JETC)*, vol. 12, no. 2, p. art. no. 17, 2015.

[143] M. Payvand and G. Indiveri, "Spike-based Plasticity Circuits for Always-on On-line Learning in Neuromorphic Systems," in *Proceedings of the IEEE International Symposium on Circuits and Systems (ISCAS)*, Sapporo, Japan., May 2019.

[144] J. Kaiser, H. Mostafa, and E. Neftci, "Synaptic Plasticity Dynamics for Deep Continuous Local Learning (DECOLLE)," *Frontiers in Neuroscience*, vol. 14, p. 424, 2020.

[145] G. Bellec, F. Scherr, E. Hajek, D. Salaj, A. Subramoney, R. Legenstein, and W. Maass, "Eligibility traces provide a data-inspired alternative to backpropagation through time," Vancouver, Canada., Dec. 2019.

[146] J. Sacramento, R. P. Costa, Y. Bengio, and W. Senn, "Dendritic cortical microcircuits approximate the backpropagation algorithm," in *Proceedings of the Conference on Neural Information Processing Systems (NIPS)*, Montreal, Canada., Dec. 2018, pp. 8721–8732.

[147] A. Valentian, F. Rummens, E. Vianello *et al.*, "Fully Integrated Spiking Neural Network with Analog Neurons and RRAM Synapses," in *Proceedings of the IEEE International Electron Devices Meeting (IEDM)*, San Francisco, CA., Dec. 2019, pp. 14.13.1–14.13.4.

[148] Y. Hayakawa, A. Himeno, R. Yasuhara, W. Boullart *et al.*, "Highly reliable TaOx ReRAM with centralized filament for 28-nm embedded application," in *VLSI Technology*, 2015, pp. T14–T15.

[149] T. Dalgaty, M. Payvand, F. Moro, D. R. Ly, F. Pebay-Peyroula, J. Casas, G. Indiveri, and E. Vianello, "Hybrid neuromorphic circuits exploiting non-conventional properties of RRAM for massively parallel local plasticity mechanisms," *APL Materials*, vol. 7, no. 8, p. 081125, 2019.

[150] M. Payvand, Y. Demirag, T. Dalgaty, E. Vianello, and G. Indiveri, "Analog weight updates with compliance current modulation of binary rerams for on-chip learning," in *To appear in the Proceedings of the IEEE International Symposium on Circuits and Systems (ISCAS)*. IEEE, 2020.

[151] E. Chicca and G. Indiveri, "A recipe for creating ideal hybrid memristive-cmos neuromorphic processing systems," *Applied Physics Letters*, vol. 116, no. 12, p. 120501, 2020.

[152] T. S. Hall, C. M. Twigg, P. Hasler, and D. V. Anderson, "Application Performance of Elements in a Floating-gate FPAA," in *Proceedings of the IEEE International Symposium on Circuits and Systems (ISCAS)*, Vancouver, Canada., May 2004.

[153] S. Shah, H. Toreyin, O. T. Inan, and J. Hasler, "Reconfigurable Analog Classifier for Knee-joint Rehabilitation," in *Proceedings of the IEEE International Conference in Engineering in Medicine and Biology Society (EMBC)*, Orlando, FL., Aug. 2016.

[154] R. Manjunath and K. S. Gurumurthy, "Artificial Neural Networks as Building Blocks of Mixed Signal FPGA," in *Proceedings of the IEEE International Conference on Field-Programmable Technology (FPT)*, Tokyo, Japan., Dec. 2003.

[155] Puxuan Dong, G. L. Bilbro, and Mo-Yuen Chow, "Implementation of Artificial Neural Network for Real Time Applications Using Field Programmable Analog Arrays," in *Proceedings of the IEEE International Joint Conference on Neural Network Proceedings (IJCNN)*, Vancouver, Canada., Jul. 2006.

[156] C. R. Schlottmann and P. E. Hasler, "A Highly Dense, Low Power, Programmable Analog Vector-Matrix Multiplier: The FPAA Implementation," *IEEE Journal on Emerging and Selected Topics in Circuits and Systems*, vol. 1, no. 3, pp. 403–411, 2011.

[157] A. Zbrzeski, P. Hasler, F. Kölbl, E. Syed, N. Lewis, and S. Renaud, "A programmable Bioamplifier on FPAA for In Vivo Neural Recording," in *Proceedings of the IEEE Biomedical Circuits and Systems Conference (BioCAS)*, Paphos, Cyprus., Nov. 2010.

[158] S. B. Shrestha and G. Orchard, "SLAYER: Spike layer error reassignment in time," in *Advances in Neural Information Processing Systems 31*, S. Bengio, H. Wallach, H. Larochelle, K. Grauman, N. Cesa-Bianchi, and R. Garnett, Eds. Curran Associates, Inc., 2018, pp. 1419–1428.

[159] Q. Wang, X. Wang, S. H. Lee, F.-H. Meng, and W. D. Lu, "A Deep Neural Network Accelerator Based on Tiled RRAM Architecture," in *Proceedings of the IEEE International Electron Devices Meeting (IEDM)*, San Francisco, CA., Dec. 2019, pp. 14–4.

[160] A. Shoeb, H. Edwards, J. Connolly, B. Bourgeois, T. Treves, and J. Guttag, "Patient-specific Seizure Onset Detection," in *Proceedings of IEEE Engineering in Medicine and Biology Society Conference (EMBC)*, 2004.

[161] M. A. Bin Altaf, C. Zhang, and J. Yoo, "A 16-Channel Patient-Specific Seizure Onset and Termination Detection SoC With Impedance-Adaptive Transcranial Electrical Stimulator," *IEEE Journal of Solid-State Circuits*, vol. 50, no. 11, pp. 2728–2740, 2015.

[162] J. Yoo, L. Yan, D. El-Damak, M. A. B. Altaf, A. H. Shoeb, and A. P. Chandrakasan, "An 8-channel scalable eeg acquisition SoC with patient-specific seizure classification and recording processor," *IEEE journal of solid-state circuits*, vol. 48, no. 1, pp. 214–228, 2012.

Mostafa Rahimi Azghadi (S'07–M'14–SM'19) completed his PhD in Electrical & Electronic Engineering at The University of Adelaide, Australia, earning the Doctoral Research Medal, as well as the Adelaide University Alumni Medal. He is currently a senior lecturer in the College of Science and Engineering, James Cook University, Townsville, Australia, where he researches low-power and high-performance neuromorphic accelerators for neural-inspired and deep learning networks for a variety of applications including automation, precision agriculture, aquaculture, marine sciences, and medical imaging. His research has attracted over $0.7M in funding from national and international resources.

Dr. Rahimi was a recipient of several national and international accolades including a 2015 South Australia Science Excellence award, a 2016 Endeavour Research Fellowship, a 2017 Queensland Young Tall Poppy Science award, a 2018 JCU Rising Star ECR Leader fellowship, and a 2019 Fresh Science Queensland finalist. Dr Rahimi is a senior member of the IEEE and a TC member of Neural Systems and Applications of the circuit and system society. He serves as an associate editor of *Frontiers in Neuromorphic Engineering* and *IEEE Access*.

Corey Lammie (S'17) is currently pursuing a PhD in Computer Engineering at James Cook University (JCU), where he completed his undergraduate degrees in Electrical Engineering (Honours) and Information Technology in 2018. His main research interests include brain-inspired computing, and the simulation and hardware implementation of Spiking Neural Networks (SNNs) and Artificial Neural Networks (ANNs) using ReRAM devices and FPGAs. He has received several awards and fellowships including the intensely competitive 2020-2021 IBM international PhD Fellowship, a Domestic Prestige Research Training Program Scholarship, and the 2017 Engineers Australia CN Barton Medal awarded to the best undergraduate engineering thesis at JCU. Corey has served as a reviewer for several IEEE journals and conferences including IEEE Transactions on Circuits and Systems and the IEEE International Symposium on Circuits and Systems (ISCAS).

Jason K. Eshraghian (M'19-S'16) is a Post-Doctoral Researcher at the Department of Electrical Engineering and Computer Science, University of Michigan in Ann Arbor. He received the Bachelor of Engineering (Electrical Electronic) and the Bachelor of Laws degrees from The University of Western Australia, Perth, WA, Australia in 2016, where he also completed his Ph.D. Degree. From 2015 to 2016, he was a Research Assistant at Chungbuk National University, South Korea. He is a member of the IEEE Neural Systems and Applications Committee. His current research interests include neuromorphic computing and spiking neural networks. Jason was awarded the 2019 IEEE Very Large Scale Integration Systems Best Paper Award, and the Best Paper Award at the 2019 IEEE Artificial Intelligence Circuits and Systems Conference for his work in neuromorphic vision.

Melika Payvand (M'19) is a research scientist at the Institute of Neuroinformatics, University of Zurich and ETH Zurich. She received her M.S. and Ph.D. degree in electrical and computer engineering from the University of California Santa Barbara in 2012 and 2016 respectively. Her research activities and interest is in exploiting the physics of the computational substrate for online learning and sensory processing. She is part of the scientific committee of the Capocaccia workshop for neuromorphic intelligence, is serving as a technical committee member of Neural Systems, Applications and Technologies in Circuits and System society and as a technical program committee for International Symposium on Circuits and Systems (ISCAS). She is a guest editor of Frontiers in Neuroscience and is the winner of the best neuromorph award of the 2019 Telluride neuromorphic workshop.

Elisa Donati (M'14) received the B.Sc, MSc degree in Biomedical Engineering from University of Pisa, Pisa, Italy (cum laude), and the Ph.D. degree in BioRobotics from the Sant'Anna School of Advanced Studies, Pisa, Italy in 2016. Currently, she is a Senior Scientist at the Institute of Neuroinformatics, University of Zurich and ETHZ where she is training as a neuromorphic engineer. Her research interests include how to interface neurorobotics and neuromorphic engineering for building smart and wearable biomedical devices. In particular, she is interested in designing VLSI systems for prosthetic devices, such as adaptive neuromorphic pacemakers. Another recent application includes a neuromorphic processor for controlling upper limb neuroprosthesis. She is investigating how to process EMG data to extract features to produce motor commands by using spiking neural networks. Dr Donati serves as an associate editor of Frontiers in Neuromorphic Engineering and she is a TC member of Neural Systems and Applications of the circuit and system society and of the Biomedical circuit and system society. As member she is part of the commission that is organizing the 2nd IEEE international conference on artificial intelligence circuits and systems.

Bernabe Linares-Barranco (M'90-S'06-F'10) received the B. S. degree in electronic physics in June 1986 and the M. S. degree in microelectronics in September 1987, both from the University of Seville, Sevilla, Spain. From September 1988 until August 1991 he was a Graduate Student at the Dept. of Electrical Engineering of Texas A&M University. He received a first Ph.D. degree in high-frequency OTA-C oscillator design in June 1990 from the University of Seville, Spain, and a second Ph.D. degree in analog neural network design in December 1991 from Texas A&M University, College-Station, USA. Since June 1991, he has been a Tenured Scientist at the "Instituto de Microelectrónica deSevilla", (IMSE-CNM-CSIC) Sevilla, Spain, which since 2015 is a Mixed Center between the University of Sevilla and the Spanish Research Council (CSIC). From September 1996 to August 1997, he was on sabbatical stay at the Department of Electrical and Computer Engineering of the Johns Hopkins University. During Spring 2002 he was Visiting Associate Professor at the Electrical Engineering Department of Texas A&M University, College-Station, USA. In January 2003 he was promoted to Tenured Researcher, and in January 2004 to Full Professor. Since February 2018, he is the Director of the "Insitituto de Microelectrónica de Sevilla". He has been involved with circuit design for telecommunication circuits, VLSI emulators of biological neurons, VLSI neural based pattern recognition systems, hearing aids, precision circuit design for instrumentation equipment, VLSI transistor mismatch parameters characterization, and over the past 25 years has been deeply involved with neuromorphic spiking circuits and systems, with strong emphasis on vision and exploiting nanoscale memristive devices for learning. He is co-founder of two start-ups, Prophesee SA (www.prophesee.ai) and GrAI-Matter-Labs SAS (www.graimatterlabs.ai), both on neuromorphic hardware. He is an IEEE Fellow since January 2010.

Giacomo Indiveri (SM'19) is a dual Professor at the Faculty of Science of the University of Zurich and at Department of Information Technology and Electrical Engineering of ETH Zurich, Switzerland. He is the director of the Institute of Neuroinformatics (INI) of the University of Zurich and ETH Zurich. He obtained an M.Sc. degree in electrical engineering and a Ph.D. degree in computer science from the University of Genoa, Italy. He was a post-doctoral research fellow in the Division of Biology at Caltech and at the Institute of Neuroinformatics of the University of Zurich and ETH Zurich. He was awarded an ERC Starting Grant on "Neuromorphic processors" in 2011 and an ERC Consolidator Grant on neuromorphic cognitive agents in 2016. His research interests lie in the study of neural computation, with a particular focus on spike-based learning and selective attention mechanisms. His research and development activities focus on the full custom hardware implementation of real-time sensory-motor systems using analog/digital neuromorphic circuits and emerging memory technologies.