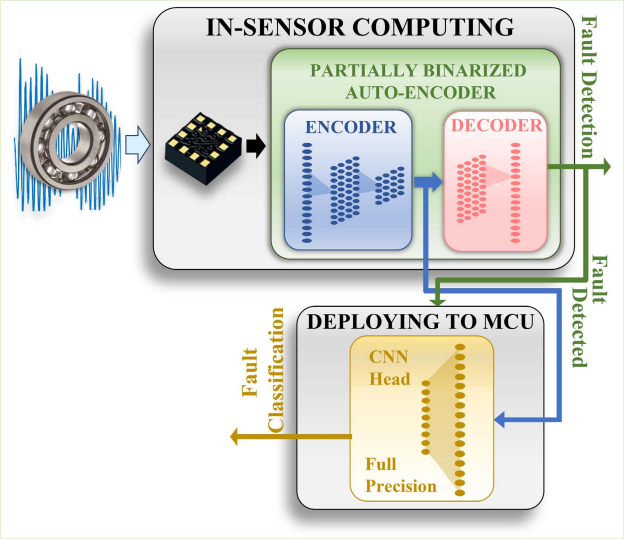
Low-Power Detection and Classification for In-Sensor Predictive Maintenance Based on Vibration Monitoring

Paola Vitolo, *Graduate Student Member, IEEE*, Antonio De Vita, *Student Member, IEEE*, Luigi Di Benedetto, *Senior Member, IEEE*, Danilo Pau, *Fellow, IEEE*, and Gian Domenico Licciardo, *Senior Member, IEEE*

***Abstract*—In this work, a new custom design of an anomaly detection and classification system is proposed. It is composed of a convolutional Auto-Encoder (AE) hardware design to perform anomaly detection which cooperates with a mixed HW/SW Convolutional Neural Network (CNN) to perform the classification of detected anomalies. The AE features a partial binarization, so that the weights are binarized while the activations, associated to some selected layers, are non-binarized. This has been necessary to meet the severe area and energy constraints that allow it to be integrated on the same die as the MEMS sensors for which it serves as a neural accelerator. The CNN shares the feature extraction module with the AE, whereas a SW classifier is triggered by the AE when a fault is detected, working asynchronously to it. The AE has been mapped on a Xilinx Artix-7 FPGA, featuring an Output Data Rate (ODR) of 365 kHz and achieving a power dissipation of 333μ W/MHz. Logic synthesis has targeted TSMC CMOS 65 nm, 90 nm, and 130 nm standard cells. Best results achieved highlight a power consumption of 138μ W/MHz with an area occupation of 0.49 mm^2 when real-time operations are set. These results enable the integration of the complete neural accelerator in the CMOS circuitry that typically sits with the inertial MEMS on the same silicon die. Comparisons with the related works suggest that the proposed system is capable of state-of-the-art performances and accuracy.**



***Index Terms*—Anomaly detection, FPGA, artificial intelligence, autoencoder, classification, in-sensor computing, ultra-low-power.**

I. Introduction

THE without peace evolution of Artificial Intelligence (AI) and Deep Learning (DL) methods in particular [1], is further pushing the development of effective Anomaly Detection (AD) systems as the first step toward the full implementation of Predictive Maintenance (PdM), which has rapidly imposed as one of the main strategies in the heterogeneous context of Industry 4.0 for the huge improvements in the reliability of industrial machines and the significant cost reductions. Although the malfunctions of industrial machines can be attributed to very different causes, involving both electrical and mechanical components, recent studies suggest that mechanical faults are mainly caused by bearing malfunctions, which account for approximately 30-40% of all the sources of failures [2]. Early symptoms of incipient mechanical failures can be detected by sensing weak anomalous vibrations or associated patterns by employing acoustic or inertial sensors, vibrometers and accelerometers, placed on the body of the machines to be monitored, and communicating with a central data processing system. Considering that industrial machines normally operate in noisy environments, DL approaches, by processing the stream of data leaking from the sensors, are much more effective to find the complex correlations between normal and anomalous behaviors than the conventional model and rule-based approaches, mainly during the early stages of the faults when anomalous signals can be very weak [2], [3]. Recent DL approaches are based on Convolutional Neural Networks (CNN), Generative Adversarial Network (GAN),

Manuscript received January 20, 2022; accepted February 19, 2022. Date of publication February 24, 2022; date of current version March 31, 2022. This work was supported by the Italian Ministry of Education, University and Research. The associate editor coordinating the review of this article and approving it for publication was Prof. Ruqiang Yan. *(Corresponding author: Gian Domenico Licciardo.)*

Paola Vitolo, Luigi Di Benedetto, and Gian Domenico Licciardo are with the Department of Industrial Engineering, University of Salerno, Fisciano, 84084 Salerno, Italy (e-mail: pvitolo@unisa.it; ldibenedetto@unisa.it; gdlicciardo@unisa.it).

Antonio De Vita and Danilo Pau are with the System Research and Applications Department, STMicroelectronics, 20092 Milan, Italy (e-mail: antonio.devita@st.com; danilo.pau@st.com).

Digital Object Identifier 10.1109/JSEN.2022.3154479

Recurrent Neural Networks (RNN), etc., each one having its own strengths and weaknesses [1]. However, deploying such networks in PdM systems is very challenging and costly due to the very high computational complexity, memory requirements of those models, and power consumption, which do not meet the needs for always-on monitoring of the apparatus [4], [5]. In this context, Auto-Encoders (AE) [6] are an obvious choice since they combine the use of relatively shallow networks with the possibility of unsupervised training, particularly interesting when the availability of labelled faults data is difficult. Indeed, AE is trained to approximate input data as faithfully as possible at the output. Moreover, if the input data are underlying normal operations of the monitored apparatus, anomalous inputs can be detected by comparing them with the corresponding reconstructed outputs. However, the limitation of the AE is the capability to recognize what it shall be normal, namely, AEs are only able to detect if an anomaly occurs, but they are not capable to classify it [7]. As it will be discussed in the next section, several solutions have been presented in the related works to overcome the above limitation by recurring to conventional and more compact Machine Learning (ML) approaches or to more complex DL, which usually are challenged in finding an acceptable trade-off between the number of physical resources needed by their implementations, the processing speed, and the detection/classification accuracy.

This work presents a new Anomaly Detection and Classification (ADC) system for the PdM of industrial apparatus, which overcomes the above limitations by proposing a new approach to the design a sensor-based edge-AI computing system. The main advantages of the proposed approach are in: a) the continuous monitoring of the vibrations produced by a mechanical apparatus; b) the real-time AD (processing of an input before another input data is available) with very high Output-Data-Rate (ODR); c) the extremely high AD accuracy and the capability to classify anomalies (9 classes have been set by this study); d) a large part of the computational complexity is both close and integrated into the sensors to reduce energy requirements and bandwidth on the communication channels. The system has an original hybrid architecture composed of two intersecting parts: an AD built around a deep convolutional AE and an on-demand Anomaly Classifier (AC), which is triggered only when the AE detects an anomaly, therefore a-synchronously vs the detection. The careful design of the AE, with ultra-low area and energy features, makes viable its integration with the CMOS circuitry typically embodied on the same die of the inertial MEMS, such as A/D, filters, re-samplers, etc., [8], [9], therefore enabling much more effective in-sensor AI. Additionally, the encoder part of the AE operates as a feature extractor for the AC component of the system. Therefore, a CNN for AC has been implemented by pipelining the encoder sub-module of the AE with a classifier, implemented on a low-power microcontroller unit (MCU) and activated on-demand, from its deep sleep state, by the AE. Additional strength points of the proposed AE can be summarized as follows:

- The proposed AE is composed of multilayer NNs for both the encoder and the decoder to achieve a very high detection accuracy.
- A new custom partial-binarization schema has been used for both the encoder and the decoder, to feature binarized weights and non-binarized activations for some selected layers.
- The number of physical resources needed by the AE implementation has been limited thanks to a careful custom HW design rather than by reducing the number of layers [10]–[14].
- This choice takes advantage of the low number of activations of the classifier and enables the possibility of sharing it with multiple AEs integrated into several sensors distributed on the apparatus under monitoring.

As case study, the proposed system has been used for the monitoring of bearings in motors. The Case Western Reserve University (CWRU) public bearing dataset [15] for machine health monitoring has been used for training and validation. The proposed system exhibits state-of-the-art accuracy with a detection accuracy of 99.61% and a classification accuracy up to 94.83% for 9 classes. Implemented on the Xilinx Artix 7 FPGA, the AE counts a total power dissipation of 122 mW (15 mW of dynamic power) operating at the maximum frequency of 45 MHz (i.e. 333 μW/MHz dynamic power), supporting MEMS with an ODR up to 365 kHz, which is one of the most important requirements for real-time monitoring application to high-speed critical manufacturing machines, such as high-speed drills and cutters. Synthesis using TSMC 65 nm LP-HVT CMOS technology gives a power dissipation of 138.62 μW/MHz, an area occupation 0.49 mm^2, and the maximum operating frequency is 230 MHz. To explore the opportunity of embedding the proposed accelerator with the sensor circuitry, to deploy in-sensors computing, analysis with more conservative TSMC 90 nm and 130 nm HVT CMOS technologies have been performed. To our best knowledge, all the achieved results are beyond the state of the art for such systems.

The remaining part of the paper is organized as follows: Section II reports a brief overview of the most recent related works; section III presents the employed models; architecture of the HW design is presented in section IV; implementation results and comparisons with the related works are discussed in section V; section VI concludes the paper.

II. Related Works

To introduce recognition capabilities in AEs, several solutions have been proposed in the recent literature, which usually adopts conventional ML approaches, featuring much less computational complexity than DLs. A large part of the related works is conceived to improve the AE models. In [16] a Feature Distance-Stacked AE (FD-SAE) has been combined with a Support Vector Machine (SVM) to classify 3 anomalies and only when their effects on the bearings are significantly different from the ones during the normal operation.

In [17] a deep generative model based on a Variational-AE (VAE) has been proposed to perform classification with a reduced set of labelled data for training. In [18] wavelet packet denoising and random forests are used to achieve 88.23% classification accuracy in fault diagnosis of rolling bearing and in noisy environment. In [19] a 2D CNN is

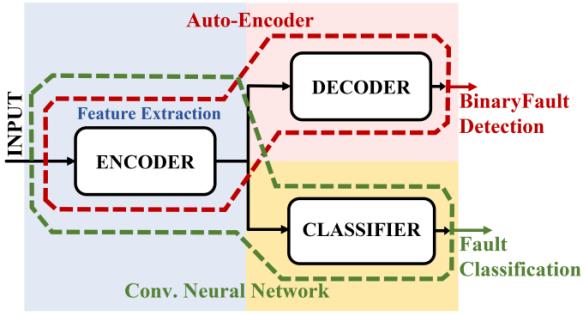


Fig. 1. Schema of the proposed fault diagnosis system. Encoder and decoder are implemented in HW, while the classifier SW runs on a tiny STM32 microcontroller.

proposed to improve classification accuracy of bearing faults by vibration data. However, such models usually require high computational complexity and memory footprint, so that they are not well suited for resource constrained embedded systems. The use of cloud computing is inconvenient when operational critical apparatus must be monitored, due to the low reliability and high latency of remote connections which requires enough bandwidth to guarantee real-time operations; general purpose platforms, using CPUs and GPUs have got silicon sizes, prices and energy costs which are incompatible with the integration into the apparatus to be monitored [5]. Similar limitations affect devoted processors, such as the Xilinx Deep Learning Processor Unit (DPU) core [20], introduced to accelerate CNN inference on FPGAs. Although it is a configurable soft core engine supporting various basic DL features (convolution, max and average pooling, etc.), DPU has been essentially designed for high performance applications and requires too many physical resources and energy budget to be effectively considered for extreme deep edge computing implementations. FPGA designs in [21]–[23] require too many neurons and, hence, too many physical resources and energy budget to be effectively considered for deep edge computing implementations. With the aim to reduce the HW computing resources, [24]–[27] exploit lightweight variants of CNN, based on Tiny-YOLO [24], MobileNet [25] and ShuffleNet [26], but they are still inadequate for HW implementation with a power budget in the order of microwatts. With the purpose to reduce the power dissipation, in [28] a configurable neural custom architecture has been presented, which normally operates with a small, low-precision AE, but it increases the complexity of the network and the computing precision on the detection of anomalies. While this approach could be effective for fault detection, it is not the best choice for PdM, which requires high accuracy when weak, initial signals of early anomalies occur [2], [3]. Therefore, our solution, which is set to design custom neural HW to keep the large part of the computation inside the sensors, appears by far the most viable solution to meet conflicting requirements in terms of availability of physical resources, communication bandwidth, [29] and to support MEMS inertial sensors with high ODR [8], [9].

III. The Underlying Models

The proposed ADC system is schematized in Fig. 1. It is composed of two NNs, the convolutional AE for AD, in turn composed of an Encoder and a Decoder stage, and the CNN (Encoder + Classifier) for fault classification. The encoder also operates as feature extractor for the CNN; therefore, it is shared between the two NNs in the Y-shaped fashion as shown in Fig. 1. The models exploit a partial quantization approach tested for the first time by the authors in [10]–[12] into a Human Activity Recognition (HAR) system. Although the NNs are significantly different in dimensions, topology, and acquiring sensors (tri-axial accelerometers in that case), the partial-binarization approach has been so effective in finding a good trade-off between high recognition accuracy and low-power dissipation of the CNN in HAR that it has also been used for AE. Therefore, all weights of the AE are binarized, namely they are set to be either +1 and -1 and encoded with 1 bit. Consequently, a memory reduction factor of 32:1 can be roughly estimated with respect to a traditional 32 bits floating point coding. Also Multiply-Accumulations (MAC) complexity has been dramatically reduced to much simpler ADD/SUB operators with large gains in area and power consumption. Furthermore, the outputs of the layers that require the largest number of operations or memory requirements have been binarized and Batch Normalization (BN) layers have been introduced to preserve the accuracy [30].

A. Convolutional Auto-Encoder

The proposed AE is shown in Fig. 2, divided into the encoder (2a) and decoder (2b) sequential stages. HNN-based convolutional model has been designed for the encoder and the decoder. This choice enables several significant advantages for the HW implementation of the AE:

- The number of network parameters and the required physical resources of the HNN are drastically lower than those of a comparable fully connected topology.
- The reduced complexity of the operators, and hence their delays, favors the iterative HW design of CONV layers, with an acceptable increment of latency even for sensors with very high ODR.
- The previous advantages also result in a lower occupied area and lower power dissipation with respect to a conventional CNN.

The encoder consists of two convolutional (CONV) and a Max-Pooling (MP) layer, which act as hierarchical extractors and decimation filter, respectively. The CONV layers are composed of 8 channels, with a kernel size of 5, and zero padding. The input window is composed of W1=24 samples. The inputs to the first layer are not binarized, and 16 bits are used for their representation, according to the output of several commercial sensors for PdM [8]. The number of outputs per channel of the first layer is equal to W2 = 20, due to the absence of zero-padding. Thus, the dimensions of the output activations of the CONV1 layer are W2×ch = 20 × 8 (160 samples). CONV1 is followed by a BN layer and binarization, consequently each output can be represented by 1 bit. The activation function used for the binarization is the sign:

$$y = sgn(x) = \begin{cases} -1 & if\ x < 0 \\ +1 & if\ x \geq 0 \end{cases} \quad (1)$$

The outputs have been binarized considering that most of the operations are performed in the next CONV2 layer. CONV2

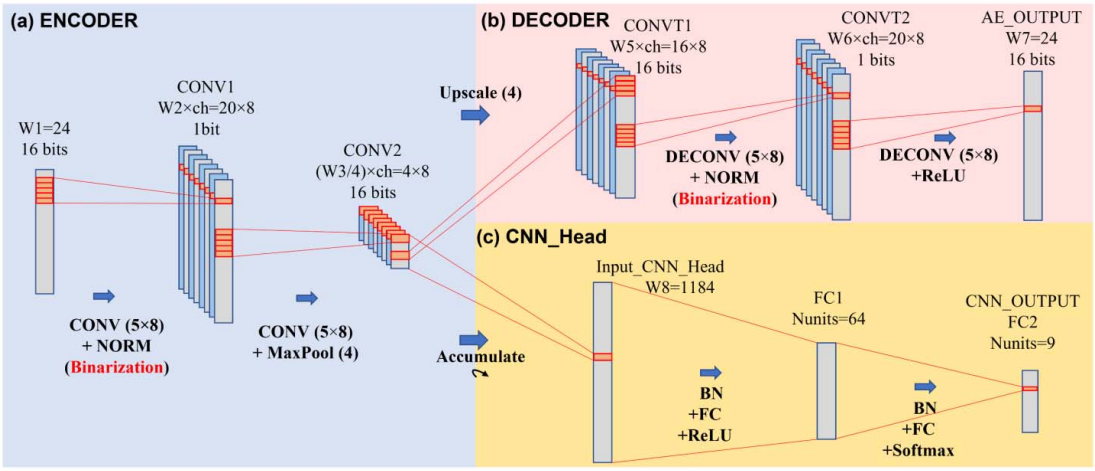


Fig. 2. Model of the proposed AD system, composed of (a) the encoder, (b) decoder and (c) the CNN_Head. The latter, in pipeline with the encoder, composes the CNN classifier.

applies a set of 8 filters of size 8×5, thus the output activations for this layer have dimensions $W3\times ch = 16\times8$ (128 samples). The Max-Pooling has size 4, namely its output activations have dimensions $W3/4\times ch = 4\times8$ (32 samples).

The decoder stage is the mirror image of the encoder. It uses UpSampling layer, which quadruples the size of the input, and the transpose convolutional layer (ConvT), which performs an inverse convolution operation. ConvT consists of a uniform up/down padding of the input and a convolutional operation. Therefore, the input size of ConvT1 is $W5\times ch = 16\times8$, a 4 padding is done resulting in an output size 24×8 and then a convolution with 5 kernel size and 8 channels is used. Its output activations have dimensions $W6\times ch = 20\times8$ (160 samples). ConvT1 is followed by a BN level and binarization, thus each output can be encoded with 1 bit. ConvT2 performs the same padding and convolution as ConvT1, but ReLU is the activation function:

$$y = \mathrm{ReLU}(x) = \begin{cases} 0 & if\ x < 0 \\ x & if\ x \geq 0 \end{cases} \tag{2}$$

Table I reports the complexity of the AE, detailing for each layer the amount of memory to store the parameters and the number of operators. By pursuing the intent to reduce the physical resources for the HW implementation of the system, the encoder also operates as a feature extractor, which, in conjunction with a shallow classifier, composes a CNN capable of classifying faults, sharing a large part of the resources with the AE.

B. CNN for Classification

Fig. 2c represents the CNN_Head module composed of two full-precision Fully Connected (FC) layers and a SoftMax. Together with the encoder of Fig 2a, it composes the CNN for classification, which is then composed of 2 partially binarized CONV layers, 1 Norm layer, 1 MaxPool, 2 FC layers and a SoftMax. FC1 has 64 neurons and (2) as activation function. FC2 has 9 neurons, which correspond to the number of faults that the CNN can classify in this work.

Unlike the AE, an input window of 24 does not embody enough information to disambiguate an anomaly between the various classes, providing an inadequate mean accuracy value of 55%. The input dimensions have been empirically derived for the case study described in the next sub-section, for which the classification accuracy overcomes 90% with an input window of 336 (91.92%) and increases up to a maximum value of 94.83% with an input window of 600. In this last case, the output dimensions of CONV1 and CONV2 layers are $W2\times ch = 596\times8$ (4768 samples) and $W3\times ch = 592\times8$ (4736 samples), respectively.

TABLE I
MEMORY AND OPERATIONS REQUIRED BY THE ADC

	Layer	Parameters [bytes]	Op. per window
ENCODER (24 inputs)	*Conv1*	21	960 ADDs
	Norm1	16	160 SUBs
	Conv2	56	5248 ADDs
	Max Pool	0	128 SUBs + COMP
DECODER (24 inputs)	*Upscale*	0	0
	ConvT1	56	6560 ADDs
	Norm2	16	160 SUBs
	ConvT2	21	984 ADDs
CNN_Head (600 inputs)	*BN*	9,472	4,736 ADDs+4,736 MULTs
	FC1	303,360	75,841 ADDs +75,776 MULTs
	BN	512	256 ADDs+256 MULTs
	FC2	2340	585 ADDs +576 MULTs

The X-CUBE-AI ver. 6.0.0 tool has been used to deploy the proposed CNN_Head in Fig. 2.b on a STM32L476RG microcontroller. The tool automatically converts the pre-trained network into C code, which has been compiled and profiled on the STM32. Considering that a single inference run requires 79,120 MACC operations, each one needing 8.69 clock cycles, at a clock frequency of 80 MHz the total inference time is 8.59 ms only right after an asynchronous anomaly detection, while consuming 120 μA/MHz in run mode. The complexity of CNN_Head in terms of required memory for the parameters and number of arithmetic operations is reported in Table I. Since the CNN_Head required single precision floating point computing resources of the STM32 and lots of memory, then its in-sensor implementation is not justified in practical applications, while it can be conveniently implemented as a

TABLE II
SUBSET OF THE CWRU DATASET USED FOR THE CNN

Fault Location	Diameter [inch]	# Samples	Sample Length
Ball	0.007	293	600
	0.014	293	600
	0.021	293	600
Inner Raceway	0.007	293	600
	0.014	293	600
	0.021	293	600
Outer Raceway	0.007	293	600
	0.014	293	600
	0.021	293	600

TABLE III
MODEL COMPARISONS

		Proposed CNN Partially binarized	Chen [27] 32-bit FP
Op. Required per window	*ADDs*	243,530	2,537,306
	MULTs	81,344	2,366,144
	COMP	3,200	85,232
Memory [bytes]		408,221	1,058,568
Average Accuracy [%]		94.83	99.3

software module on a cheap off-the-shelf MCU or on the processing units equipping almost all the recent FPGAs, working in tandem with the AE. To solve the mismatch between the input windows of the AE (24) and the CNN (600), as well as to achieve a compact ADC system that shares the compact HNN encoder as in Fig. 1, when detecting an anomaly, 24 outputs of the MaxPool layer of the encoder are accumulated into the STM32 embedded RAM by the CNN_Head for 25 times, up to 600 inputs. It is worthwhile to consider that the above arrangement not only facilitates the fine tuning of the classifier after its installation to preserve the accuracy (e.g., in presence of wear of the monitored apparatus) but also enables the time-sharing of the same CNN_head with several sensors equipped with the feature extractor and deployed in different carry positions on the apparatus to be monitored.

C. Case Study and Verification: Bearing Faults Detection and Classification

In order to validate the proposed ADC system, we have used the bearing dataset [15], provided by Case Western Reserve University (CWRU), Cleveland, Ohio, USA. It contains electric motor bearing vibration data sampled at 12 kHz and 48 kHz by using accelerometers attached to the motor housing with magnetic bases. Electro-discharge machining has been used to induce 4 different diameter faults on the rolling elements, the outer races, and the inner races. The motor speeds were set to 1797, 1772, 1750, and 1730 rpm, related to loads of 0, 746, 1492, and 2238 W. The extracted datasets for this work have been of 20,328 samples of 24 data points for the AE and 2637 samples of 600-lenght for the CNN, each one coded with 16 bits. The sample data and the rpm have been of 12 kHz and 1797, respectively. The classified faults are 9 and detailed information is listed in Table II.

The models have been built and trained using TensorFlow and Larq frameworks [31], [32]. The dataset for the AE has been divided into training (75%), validation (12.5%), and test (12.5%) dataset. AE has been trained with batch sizes of 256 for 50 epochs. Since the CWRU dataset also contains anomalous data, they have been exploited for AE training following the algorithm proposed in [33]. To evaluate the model, accuracy and Area Under Curve (AUC) have been calculated on the test dataset. The AUC is a metric for evaluating the performances of the classification model separately from the thresholds set. The AUC scores and accuracy achieved, averaged over 10 trials, are 0.99 and 99.61%, respectively.

For the CNN training, the dataset extracted from that of the CWRU has been divided into training (80%), validation (10%), and test (10%). Categorical cross-entropy has been chosen as loss function and Adam as the optimizer. The number of epochs has been set to 200 and the batch size to 128. The accuracy achieved, averaged over 10 trials, is 94.83%.

Table III compares the proposed classification model with one of the most recent works in the literature [27], in terms of number of operations, memory requirements for parameters and partial outputs, and accuracy. However, considering that the 1D-CNN in [27] outperforms the MobileNet and ShuffleNet V2 in [26], in terms of accuracy and model size, results in Table III can be generalized to the above lightweight networks. Like our proposal, the network in [27] works directly on 1D vibration input signals, without the additional resources required for the Short-Time Fourier Transform (STFT) to convert vibrations into 2D time-frequency signals, as it is usually done in alternative solutions [26]. But nevertheless, our solution requires a memory footprint and a number of operations which are orders of magnitude lower than [27]. On the contrary, partial quantization is primarily responsible for 4.47% reduction of classification accuracy, which anyway remains higher than 94%, and is a very acceptable trade-off for the HW implementation of our system, which aims to prioritize fault detection at the edge.

IV. HARDWARE ARCHITECTURE

The design strategy for the proposed AE HW accelerator has aimed to identify the minimum number of circuital elements, leveraging an extensive use of resource sharing and iterative processing schemas, to meet area and power constraints for in-sensor implementation, while preserving the capabilities of the system to interface with high-ODR sensors in real-time. In this case

Due to partial binarization, only a single bit is required to encode weights and binarized output activations, while a fixed-point coding has been used for non-binarized values to greatly lighten the circuital implementation. A code length of 16 (8.8) bits has been used, which ensures a trade-off between the minimum code length and the resulting accuracy. Thanks to the above quantization choices, the memory required for the convolutional kernels and the partial results could be fitted in internal registers or distributed memories of FPGA, avoiding a slower and more energy-hungry access to higher-level memory. Also, the complexity of the arithmetic circuits has been strongly simplified by the implementation of the quantization schema. In fact, the weight binarization reduces MACC operations in much simpler ADDs/SUBs, so that each CONV/CONVT layer performs the following calculation:

$$\sum_{i=1}^{N} w_i \cdot x_i + b = \pm x_1 \pm x_2 \pm \ldots \pm x_N + b \qquad (3)$$

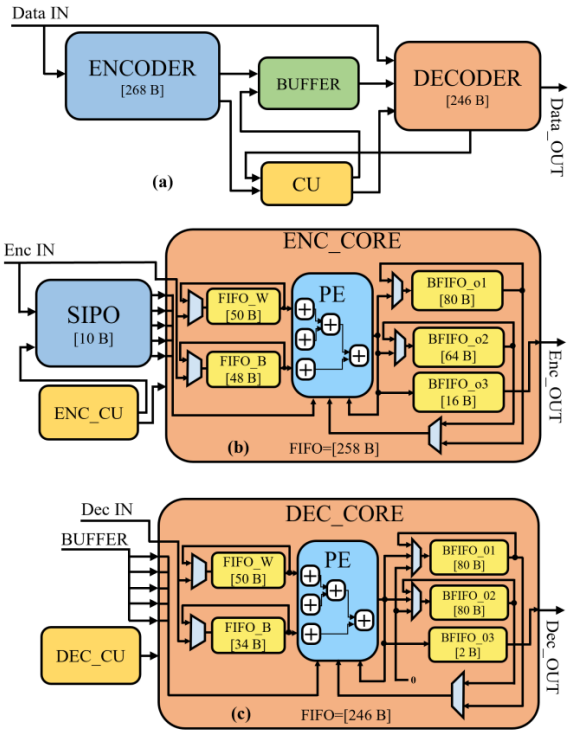


Fig. 3. Block diagram of: (a) the proposed AE Accelerator; (b) the encoder of the proposed AE Accelerator; (c) the decoder of the proposed AE Accelerator.

where w_i and x_i are the weights and the inputs to a certain neuron, respectively, and b is the bias.

Moreover, since the batch normalization is followed by binarization, also the implementation of the BN layer does not require a multiplier. Indeed, given a generic input, x, the output of the batch normalization can be written as $y = \alpha x + \beta = \alpha\left(x + \beta/\alpha\right)$. Since y must be constrained to ± 1 through the *sign* function, there is no need to multiply, and just to know the sign of α and the value $\gamma = \beta/\alpha$, which are encoded by 1 bit and 16 bits, respectively.

A. Implementation of the HW Modules

Fig. 3a schematizes the proposed HW architecture of the AE. It consists of two modules implementing the encoder and the decoder, a FIFO for data buffering between the modules, and a Control Unit (CU) that carefully manages the operations. The encoder and decoder operate asynchronously with each other so that the encoder can accept a new input data without waiting for the decoder to finish its processing, consequently, increasing the maximum Input Data Rate (IDR).

Fig. 3b and 3c show the implementation schemas of the encoder and decoder modules, respectively. Each module is composed of a CU, ENC_CU and DEC_CU, and a core, ENC_CORE and DEC_CORE, which iteratively run to accomplish the operations of the encoder and decoder. The encoder also includes a SIPO to store the data of the receptive field for the first CONV layer. Both cores consist of a PE, multiplexers to select the appropriate inputs for the PE and FIFOs, and embedded memory elements to store all the variables needed to perform the operation locally. In particular, circular buffers based on FIFO have been designed to store the inputs, the partial outputs, and the parameters of the network. Considering that:

- the kernel sizes and the number of channels of each CONV layer are 5 and 8, respectively;
- the weights and bias are encoded by 1 bit and 16 bits, respectively;
- the BN parameters, α and γ, are encoded as the weights and bias, respectively;
- the number of weights, biases, and BN parameters to be stored for the encoder are: CONV1_w+BN1_α+CONV2_w=5 × 8+8+5 × 8×8=400
  CONV1_b+BN1_γ +CONV2_b=8+8+8=24;
- the number of weights, biases, and BN parameters to be stored for the decoder are:
  CONVT1_w+BN2_α+CONVT2_w=5 × 8×8+8+5 × 8=400
  CONVT1_b+BNT1_ γ +CONVT2_b=8+8+1=17;

FIFOs for the encoder have dimensions 400 bits for the weights and 24 × 16 bits for the bias. In turn, the FIFO dimensions for the decoder are 400 bits for the weights and 17 × 16 bits for the bias.

On the other hand, considering that:

- each CONV layer performs a convolution when a number of inputs equal to the kernel size is ready;
- the Max Pool layer must wait a number of inputs equal to its size;

an iterative architecture has been devised that does not wait for all the data to be available, but only those necessary for the start of processing. For example, CONV1 processing starts as soon as 5 of 24 input data are available. This reduces the SIPO dimension of 79%, from 24 × 16 bits to 5 × 16 bits, and FIFOs for the partial outputs of CONV1, CONV2, MaxPool, CONVT1 and CONVT2 layers, having dimensions 40 × 16 bits, 32 × 16 bits, 8 × 16 bits, 40 × 16 bits, and 40×16 bits, respectively, with an overall reduction of 74%. Therefore, ENC_CORE and DEC_CORE embed 268 bytes and 246 bytes of FIFO memories, respectively.

The FIFOs related to weights and biases must be initialized by using an external data stream during the setup phase, and, subsequently, they are set as circular buffers by the CU for all the rest of the time. The memory elements for the partial results, on the other hand, operate as FIFOs when they are written and as circular buffers when they are the inputs for subsequent operations.

Fig. 4 shows the implementation schema of the PE, used for both the encoder and the decoder. The PE consists of a 3-level adder tree, with 16-bit (8.8) fixed-point coding. As shown in the equation in the inset of Fig. 4, it performs a dot product between two vectors and the sum with the bias or the previous partial result. The vectors D_{IN} and w have both length 5 and w has binarized elements. A1, A2 and A3 of the adder tree are 16-bits adders with circuitry for binarized weights multiplication. A4 and A5 are 16-bits adders. The output of the adder tree is iterated through the mux "RES," to calculate the convolution (3) for each layer. The activations functions (1) and (2) have been implemented

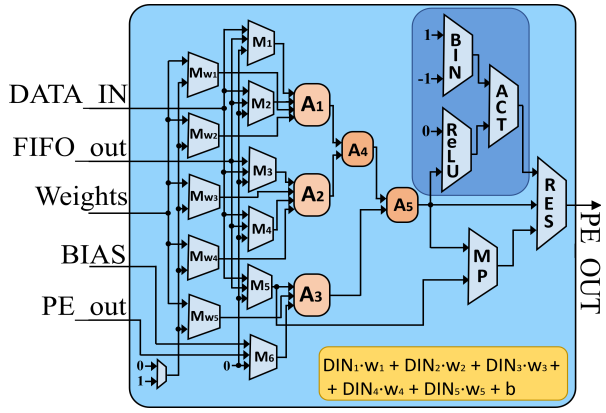


Fig. 4. Implementation schema of the processing element (PE).

by the muxes "BIN" and "RELU," fed by the adder tree and selected by the mux "ACT" and "RES" when the dot product calculation is complete. Additionally, the PE performs the necessary comparisons for the Max Pooling layer by means of the "MP" mux, and the decoder performs padding and upscale operations, by appropriately adding 8 zeros or the 8 previous data in BFIFO_o1 and BFIFO_o2 in Fig. 3c.

B. Dynamic Generation of the Layers

To reduce HW resources and perform all the operations necessary to execute the layers by reusing the single PE present in each module, control logics, ENC_CU and DEC_CU, have been implemented to manage the inputs and outputs of the PE and to correctly store the partial results in the FIFOs. The CUs have been realized by using Finite State Machines (FSMs), which have a state for each layer, in turn featuring a sub-state for each dot operation. For example, to generate CONV1 of Fig. 2, the control logic provides two substates, each of which runs in a single clock cycle: one that manages the dot product and one that performs BN and binarization. In particular, considering Fig. 3, in the first substate the five input data stored in SIPO and the five weights stored in the FIFO_W are multiplied and then added to the bias stored in FIFO_B. In the second substate, the previous result and the γ parameter of BN, stored in FIFO_B, are added and then the binarization is performed taking into account the α parameter of BN, stored in FIFO_W, and the result is stored in BFIFO_o1. In this way, the first output of CONV1, related to one channel of Fig. 2, is calculated. To obtain the results of the other channels, the two aforementioned substates are iterated a number of times equal to the number of channels, 8. Therefore, to obtain an output for each channel of CONV1, which includes the operations of convolution, BN, and binarization on 5 input data, $2 \times 8{=}16$ clock cycles are required. Once the operations for CONV1 have been performed, the control logic uses the circuitry of Fig. 4 to obtain the results of the other layers. In particular, 72, 32, 72 and 9 clock cycles are required for CONV2, MaxPool, CONVT1 and CONVT2, respectively. Therefore, considering that one clock cycle is required to pass from one state to another, ENC_CORE requires 123 clock cycles to process an input sample and provides one output each 4 inputs because of the last Max Pooling layer of size 4. In turn, DEC_CORE is asynchronously fed by the encoder through the buffer, it employs 9 to 83 clock cycles to provide an output, depending on whether padding or upscaling must be performed.

Therefore, the ADC system can process new input data after 123 clock cycles and elaborates 16 input samples (required for a fault detection) in a minimum of 3632 clock cycles.

It is worthwhile to observe that since ENC_CORE and DEC_CORE work in parallel, the maximum ODR sensor supported by the system is limited only by the processing time of the encoder.

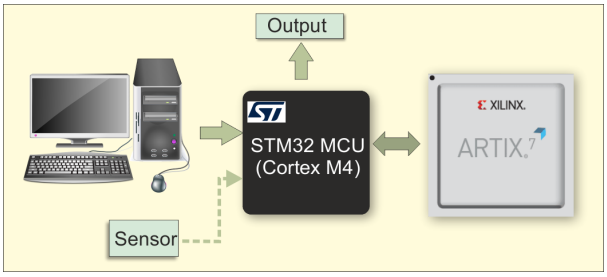


Fig. 5. Schema of the test board of the proposed system. A PC can be used as data source for training and a sensor for on-line operations.

V. Synthesis and Implementation Results

The proposed design has been implemented on the Xilinx Artix-7 (xc7a35tfgg484-1) FPGA [34] by using the Vivado IDE suite. Fig. 5 shows the schema of the test board of the proposed system. It is composed of a small Digilent CMOD A7-35T, equipped with the FPGA used to implement the AE; the STM32F401RE MCU [35] has been used to implement the classifier, according to the proposed HW/SW hybrid schema, and also to manage the data transfer from the data source to the FPGA by using a custom, simplified SPI-like interface. The data source can be either the PC, used to store the test dataset, or the sensor such as the X-NUCLEO-IKS01A1 [36], equipped with the LSM6DSO IMU. Moreover, synthesis results targeting TSMC 130 nm, 90 nm, and 65 nm CMOS standard cells by using the Cadence toolchain have been reported to evaluate the in-sensor integration cost with the other logic currently mapped onto MEMS sensor circuitry.

A. FPGA

Table III reports details of the FPGA implementations of our design, compared with the most recent related works [21], [37]. Since the performances of the design results from the trade-off between the capability to manage sensors with high ODR, the power consumption and the number of mapped physical resources, a new Figure-of-Merit (FoM) has been introduced to make straightforward the comparisons. Therefore, ODR-on-Power-and-Resources (OPR) has been defined as:

$$OPR = \frac{ODR}{Tot.Power \times Resources} \tag{4}$$

where ODR is the one supported in real-time, *Resources* is the number of physical resources for the implementation. It could be either the occupied area of the design (for ASICs), or the total number of LUTs, registers and macros mapped on FPGAs.

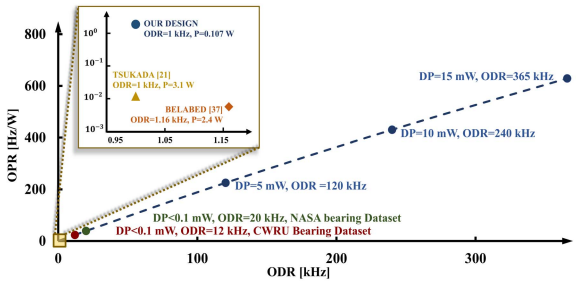


Fig. 6. OPR vs. ODR on FPGA. The linear trend is due to the significant quiescent power dissipation of the FPGA compared to the dynamic power (DP). The inset shows the detail around ODR = 1 kHz to highlight comparisons with [21] and [37].

TABLE IV
FPGA RESULTS AND COMPARISON

	Belabed.[37]	Tsukada[21]	Proposed AE
HW Classifier	Yes	No	No
Accuracy [%]	-	-	**99.61**
AUC	-	0.952	**0.99**
HW platform	Zynq 7020	Zynq 7020	Artix 7
Freq. [MHz]	100	100	**45**
ODR @ Freq [kHz]	1.16	1	**365**
Dyn. Power [μW/MHz]	22360	-	**333**
Total Power @ODR [W]	2.4	3.1	**0.122**
# of LUTs	38840	13725	**2449**
# of LUTRAMs	1916	-	**211**
# of FFs	43630	12342	**2319**
# of BRAMs	28	155	**0**
# of DSPs	46	72	**0**
OPR [Hz/W]	0.0059	0.0123	**1.96**

Fig. 6 shows the OPR of our design in terms of the ODR values, up the maximum supported ODR, while the inset presents the detail around ODR = 1 kHz to highlight comparisons with [21] and [37]. Considering that our system obtains an AUC = 0.99, quite higher than the only one accuracy metric provided by [21] and [37], the curve in the Fig. 6 clearly shows the advantages of our AE with respect to the related works. Indeed, the proposed implementation requires 2449 LUTs and 2319 FFs, which are order of magnitudes lower than the other designs, although it does not exploit DSPs and BRAMs, to make the implementation results as platform independent as possible. Moreover, although the maximum frequency of 45 MHz is lower than the operative frequency of the alternatives, the maximum supported ODR sensor of our design is 365 kHz, namely more than two orders of magnitude higher than the counterparts. Such a high ODR value enables several highly demanding PdM applications whose high-speed mechanical parts must be monitored in real-time (such as high-speed drills and cutters) even with the use of ultrasonic MEMS microphones [9], [38] (e.g., pressure leaks, bearing condition, gear meshing, pump cavitation, etc.). It is worthwhile to note that if we lower the operating frequency to set the ODR to 1 kHz, like the alternatives in Table IV, the power consumption of our proposal, 107 mW, remains significantly lower. With reference to Application Processing Units (APU) built with FPGAs, that from some years have become very attractive to setup highly customizable platforms [39], an interesting solution is the Xilinx DPU [20] to implement high performance NNs, including GoogLeNet, ResNet and MobileNet, on Xilinx Zynq SoC devices. The DPU IP provides some possible configurations regarding the DSP slice, LUT, block RAM, UltraRAM, the number of DPU cores, the convolution architecture, etc., to meet various types of constraints. However, even with the smallest convolution architecture, B512, as shown in Table V, the FPGA resources used by the DPU core on the Ultrascale+ ZCU102 are 36,458 LUT, 41,744 FF, 77.5 BRAM, and 124 DSP, with a power consumption of 5.718 W [20], to which memory and a program running on the APU must be provided to handle interrupts, data transfers and storage of input, temporary and output data, resulting in significantly greater overall resources than to our project.

TABLE V
COMPARISON WITH XILINX DPU [20]

	DPU core (B512) [20]	Proposed AE
Power[W]	5.718	**0.122**
# of LUTs	36458	**2449**
# of FFs	41744	**2319**
# of BRAMs	77.5	**0**
# of DSPs	124	**0**

Fig. 7 shows the implementation breakdown of the AE on the FPGA, in terms of utilization and power consumption. Results show that about 56% of mapped LUTs and 43% of power consumption are used for the PEs and CUs of the encoder and decoder. About 89% of FFs are occupied by the FIFOs, whose quantity is the actual limiting factor of the proposed design.

B. Standard Cells

In Table VI comparisons of synthesis results with 130 nm (the one usually adopted to manufacture the most commercially available MEMs sensors), 90 nm and 65 nm have been reported. CMOS technologies with High Voltage-Threshold (HVT) have been selected for the noteworthy difference of leakage power dissipation, which could be a serious problem when the design is used at low frequencies [10]. For ease of comparison, also the 65 nm General Purpose (GP) technology has been included. When estimated with Cadence Joules feed with SAIF files, the HVT cells reduce leakage power by 77% at the cost of a maximum frequency reduction of about 10%. Maximum operating frequencies, ranging from 100 to 250 MHz, enable ODR values in the MHz range. Such values are much higher than those typically required by current applications, and they give an estimation of the scaling capability of the proposed design with perspective sensor technologies. Table VI also reports the total power dissipation when the ODR is set at 20 kHz, namely at an AE operation frequency of 2.46 MHz. It is worthwhile to note the large increment of the static power dissipation due to leakage of the 65 nm GP technology with respect to the HVT counterpart. Comparisons with the state of the art are not that simple due to the limited number of works targeting low power integrated implementations and the lack of data reported by other papers. However, from the recent literature, we have considered the data of the ADEPOS design in [28] and [40], since it is one of the rare works that target low-power AE-based system with a 65 nm CMOS technology, although it is not a quite fair comparison. Indeed, to reduce the power dissipation from a maximum value of 744 μW to a stand-by power dissipation of 12 μW, ADEPOS completely activates

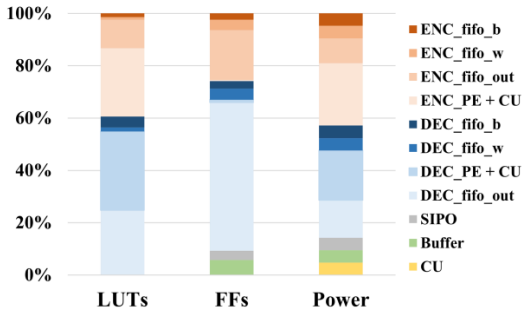


Fig. 7. Breakdown of the AE on FPGA.

TABLE VI
STANDARD CELLS SYNTHESIS COMPARISONS

	Proposed AE				Bose [28]
Technology	130 nm HVT	90 nm LP HVT	65 nm GP	65 nm LP HVT	65 nm
Acc [%]		99.54			100
Max Freq. [MHz]	100	150	250	200	-
Max Sensor ODR [MHz]	0.81	1.21	2	1.63	-
Dyn. Power [μW/MHz]	599	210	120	138	-
Total Power @ODR=20kHz [μW]	1474	519	1512	341	744/12*
Area [mm²]	3.54	1.30	8.51	0.49	0.57
OPR @ODR=20kHz [Hz μW⁻¹mm⁻²)]	3.83	29.64	1.55	120	47.16/2924*

**Active phase/inactive phase*

only after that an anomalous data has been detected, dynamically increasing its complexity and accuracy only during about 1% of the lifetime [34], based on tests with the dataset in [41] and an ODR of 20 kHz. Depending on the activation phase, it achieves OPR = 47.16 Hz $\mu W^{-1}mm^{-2}$ (2924 when inactive).On the contrary, our design works with a constant high accuracy level, well-suited for PdM, which requires the identification of the initial very weak signal of anomalies, with a power dissipation of 341 μW when the operation frequency is set for an ODR = 20 kHz and an occupied area of 0.49 mm^2, resulting in OPR = 120 Hz $\mu W^{-1}mm^{-2}$. Those results, to our best knowledge, overcome the state of the art in term of performances for these kinds of systems.

VI. CONCLUSION

This article proposes a HNN-based model for fault diagnosis and a custom neural HW accelerator for in-sensor computing targeting PdM applications. The custom HW in-sensor accelerator works synergistically with an MCU to offer both ultra-low power AD and highly accurate classification with a high performance/cost ratio. Future work will investigate the application of this system in other application contexts and the possibility of managing multi-sensor fusion data.

REFERENCES

[1] Y. Ran, X. Zhou, P. Lin, Y. Wen, and R. Deng, "A survey of predictive maintenance: Systems, purposes and approaches," 2019, *arXiv:1912.07383*.

[2] S. Zhang, S. Zhang, B. Wang, and T. G. Habetler, "Deep learning algorithms for bearing fault diagnostics—A comprehensive review," *IEEE Access*, vol. 8, pp. 29857–29881, 2020.

[3] J. Yang, Y. Yang, and G. Xie, "Diagnosis of incipient fault based on sliding-scale resampling strategy and improved deep autoencoder," *IEEE Sensors J.*, vol. 20, no. 15, pp. 8336–8348, Aug. 2020.

[4] S. R. Saufi, Z. Ahmad, M. S. Leong, and M. H. Lim, "Challenges and opportunities of deep learning models for machinery fault detection and diagnosis: A review," *IEEE Access*, vol. 7, pp. 122644–122662, 2019.

[5] M. Capra *et al.*, "Hardware and software optimizations for accelerating deep neural networks: Survey of current trends, challenges, and the road ahead," *IEEE Access*, vol. 8, pp. 225134–225180, 2020.

[6] J. Zhai, S. Zhang, J. Chen, and Q. He, "Autoencoder and its various variants," in *Proc. IEEE Int. Conf. Syst., Man, Cybern. (SMC)*, Miyazaki, Japan, Oct. 2018, pp. 415–419.

[7] M. Ribeiro, M. Gutoski, A. E. Lazzaretti, and H. S. Lopes, "One-class classification in images and videos using a convolutional autoencoder with compact embedding," *IEEE Access*, vol. 8, pp. 86520–86535, 2020.

[8] *Ultra-Low-Power High-Performance 3-Axis Accelerometer With Digital Output for Industrial Applications*, DocID027668 Rev 2, STMicroelectronics, Italy, 2015.

[9] *Ultrasound Behavior and Guidelines of Analog MEMS Microphone IMP23ABSU*, IMP23ABSU Rev 2, STMicroelectronics, Italy, 2020.

[10] A. De Vita, A. Russo, D. Pau, L. D. Benedetto, A. Rubino, and G. D. Licciardo, "A partially binarized hybrid neural network system for low-power and resource constrained human activity recognition," *IEEE Trans. Circuits Syst. I, Reg. Papers*, vol. 67, no. 11, pp. 3893–3904, Nov. 2020.

[11] A. D. Vita, D. Pau, C. Parrella, L. D. Benedetto, A. Rubino, and G. D. Licciardo, "Low-power HWAccelerator for AI edge-computing in human activity recognition systems," in *Proc. 2nd IEEE Int. Conf. Artif. Intell. Circuits Syst. (AICAS)*, Aug. 2020, pp. 291–295.

[12] A. D. Vita, D. Pau, L. D. Benedetto, A. Rubino, F. Petrot, and G. D. Licciardo, "Low power tiny binary neural network with improved accuracy in human recognition systems," in *Proc. 23rd Euromicro Conf. Digit. Syst. Design (DSD)*, Aug. 2020, pp. 309–315.

[13] G. D. Licciardo, C. Cappetta, L. Di Benedetto, A. Rubino, and R. Liguori, "Multiplier-less stream processor for 2D filtering in visual search applications," *IEEE Trans. Circuits Syst. Video Technol.*, vol. 28, no. 1, pp. 267–272, Jan. 2018.

[14] G. D. Licciardo, C. Cappetta, L. Di Benedetto, and M. Vigliar, "Weighted partitioning for fast multiplierless multiple-constant convolution circuit," *IEEE Trans. Circuits Syst. II, Exp. Briefs*, vol. 64, no. 1, pp. 66–70, Jan. 2017.

[15] *Case Western Reserve University (CWRU) Bearing Data Center*. Accessed: Oct. 2020. [Online]. Available: https://csegroups.case.edu/ bearingdatacenter/pages/welcome-case-western-reserve-universitybearing-data-center-website.

[16] M. Cui, Y. Wang, X. Lin, and M. Zhong, "Fault diagnosis of rolling bearings based on an improved stack autoencoder and support vector machine," *IEEE Sensors J.*, vol. 21, no. 4, pp. 4927–4937, Feb. 2021.

[17] S. Zhang, F. Ye, B. Wang, and T. G. Habetler, "Semi-supervised bearing fault diagnosis and classification using variational autoencoder-based deep generative models," *IEEE Sensors J.*, vol. 21, no. 5, pp. 6476–6486, Mar. 2021.

[18] Z. Wang, Q. Zhang, J. Xiong, M. Xiao, G. Sun, and J. He, "Fault diagnosis of a rolling bearing using wavelet packet denoising and random forests," *IEEE Sensors J.*, vol. 17, no. 17, pp. 5581–5588, Sep. 2017.

[19] R. Magar, L. Ghule, J. Li, Y. Zhao, and A. B. Farimani, "FaultNet: A deep convolutional neural network for bearing fault classification," *IEEE Access*, vol. 9, pp. 25189–25199, 2021.

[20] Y. Lei, Q. Deng, S. Long, S. Liu, and S. Oh, "An effective design to improve the efficiency of DPUs on FPGA," in *Proc. IEEE 26th Int. Conf. Parallel Distrib. Syst. (ICPADS)*, Dec. 2020, pp. 206–213.

[21] M. Tsukada, M. Kondo, and H. Matsutani, "A neural network-based on-device learning anomaly detector for edge devices," *IEEE Trans. Comput.*, vol. 69, no. 7, pp. 1027–1044, Jul. 2020.

[22] J. Maria, J. Amaro, G. Falcao, and L. A. Alexandre, "Stacked autoencoders using low-power accelerated architectures for object recognition in autonomous systems," *Neural Process. Lett.*, vol. 43, no. 2, pp. 445–458, Apr. 2016.

[23] M. Coutinho, M. Torquato, and M. Fernandes, "Deep neural network hardware implementation based on stacked sparse autoencoder," *IEEE Access*, vol. 7, pp. 40674–40694, 2019.

[24] W. Fang, L. Wang, and P. Ren, "Tinier-YOLO: A real-time object detection method for constrained environments," *IEEE Access*, vol. 8, pp. 1935–1944, 2020.

[25] W. Yu and P. Lv, "An end-to-end intelligent fault diagnosis application for rolling bearing based on MobileNet," *IEEE Access*, vol. 9, pp. 41925–41933, 2021.
[26] H. Liu, D. Yao, J. Yang, and X. Li, "Lightweight convolutional neural network and its application in rolling bearing fault diagnosis under variable working conditions," *Sensors*, vol. 19, no. 22, pp. 1–20, 2019.
[27] C. C. Chen, Z. Liu, G. Yang, C. C. Wu, and Q. Ye, "An improved fault diagnosis using 1D-convolutional neural network model," *Electron.*, vol. 10, no. 1, pp. 1–19, 2021.
[28] S. K. Bose *et al.*, "ADEPOS: A novel approximate computing framework for anomaly detection systems and its implementation in 65-nm CMOS," *IEEE Trans. Circuits Syst. I, Reg. Papers*, vol. 67, no. 3, pp. 913–926, Mar. 2020.
[29] Z. Zou, Y. Jin, P. Nevalainen, Y. Huan, J. Heikkonen, and T. Westerlund, "Edge and fog computing enabled AI for IoT–An overview," in *Proc. IEEE Int. Conf. Artif. Intell. Circuits Syst. (AICAS)*, Hsinchu, Taiwan, Mar. 2019, pp. 51–56.
[30] E. Sari, M. Belbahri, and V. Partovi Nia, "How does batch normalization help binary training?" 2019, *arXiv:1909.09139*.
[31] M. Abadi. (2015). *TensorFlow: Large-Scale Machine Learning on Heterogeneous Distributed Systems*. [Online]. Available: https://www.tensorflow.org/
[32] *LARQ*. Accessed: Jun. 2021. [Online]. Available: https://docs.larq.dev/larq/
[33] Y. Yamanaka, T. Iwata, H. Takahashi, M. Yamada, and S. Kanai, "Autoencoding binary classifiers for supervised anomaly detection," in *Trends in Artificial Intelligence* (Lecture Notes in Computer Science) vol. 11671. Springer, 2019, pp. 647–659.
[34] (Feb. 2018). Xilinx. *7 Series FPGAs Data Sheet: Overview*. XC7A35T-1CPG236C datasheet. [Online]. Available: https://www.xilinx.com/support/documentation/data_sheets/ds180_7Series_Overview.pdf
[35] (Jan. 2015). STMicroelectronics. *STM32F401xD STM32F401xE*. STM32F401RE Datasheet. [Online]. Available: https://www.st.com/resource/en/datasheet/stm32f401re.pdf.
[36] (May 2015). STMicroelectronics. *X-NUCLEO-IKS01A1 Motion MEMS and Environmental Sensor Expansion Board for STM32 Nucleo*. [Online]. Available: https://www.st.com/resource/en/datasheet/x-nucleo-iks01a1.pdf
[37] T. Belabed, M. G. F. Coutinho, M. A. C. Fernandes, V. Carlos, and C. Souani, "Low cost and low power stacked sparse autoencoder hardware acceleration for deep learning edge computing applications," in *Proc. 5th Int. Conf. Adv. Technol. Signal Image Process. (ATSIP)*, Sep. 2020, pp. 1–6.
[38] C. Murphy. *Choosing the Most Suitable Predictive Maintenance Sensor*. Analog Devices. [Online]. Available: https://www.analog.com/en/technical-articles/choosing-the-most-suitable-predictive-maintenance-sensor.html#
[39] G.-D. Licciardo and M. Costagliola, "An H. 264 encoder for real time video processing designed for spear customizable system-on-chip family," in *Proc. IEEE Int. Conf. Signal Process. Commun.*, Dubai, United Arab Emirates, Nov. 2007, pp. 824–827.
[40] B. Kar, P. K. Gopalakrishnan, S. K. Bose, M. Roy, and A. Basu, "ADIC: Anomaly detection integrated circuit in 65-nm CMOS utilizing approximate computing," *IEEE Trans. Very Large Scale Integr. (VLSI) Syst.*, vol. 28, no. 12, pp. 2518–2529, Dec. 2020.
[41] *NASA Dataset*. Accessed: Oct. 2020. [Online]. Available: https://ti.arc.nasa.gov/tech/dash/groups/pcoe/prognostic-data-repository

Paola Vitolo (Graduate Student Member, IEEE) was born in Sarno, Italy, in 1992. She received the B.Sc. *(cum laude)* and M.Sc. *(cum laude)* degrees in electronic engineering from the University of Salerno, Fisciano, Italy, in 2017 and 2021, respectively, where she is currently pursuing the Ph.D. degree with the Department of Industrial Engineering. Her current research activities are on integrated low-power hardware accelerators for artificial neural networks.

Antonio De Vita (Student Member, IEEE) was born in Avellino, Italy, in 1992. He received the B.Sc. and M.Sc. *(cum laude)* degrees in electronic engineering and the Ph.D. degree in microelectronics from the University of Salerno, Fisciano, Italy, in 2015, 2017, and 2021, respectively. He joined STMicroelectronics as a Research and Development Digital Design Engineer in 2021. His current research activities concern digital architectures for machine learning.

Luigi Di Benedetto (Senior Member, IEEE) received the B.Sc. and M.Sc. *(cum laude)* degrees in electronic engineering and the Ph.D. degree in solid state electronics from the University of Salerno, Fisciano, Italy, in 2006, 2009, and 2013, respectively. In 2013, he was a Visiting Scientist with the Fraunhofer IISB and Friedrich-Alexander-Universität Erlangen-Nürnberg, Erlangen, Germany. Since 2013, he has been a Research Fellow, and since 2018, he has been an Assistant Professor in Electronic with the Department of Industrial Engineering, University of Salerno. His main research interests include the modeling, simulation, and development of high-power electronic devices based on wide bandgap semiconductor and design of VLSI systems.

Danilo Pau (Fellow, IEEE) graduated in electronic engineering from the Politecnico di Milano in 1992. Since 1991, he has been with STMicroelectronics, Italy, as a System Researcher. He worked on HDMAC hardware design and MPEG2 video memory reduction, then on video coding and transcoding, next on embedded 3D and VG graphics, and computer vision with handcrated algorithms. Currently, his focus is on the development of tools to bridge deep learning frameworks with resource constrained applications on micro-controllers and sensors. He currently serves IEEE Region 8 Action for Industry focused on the internship initiative. He is also a member of the Machine Learning, Deep Learning, and AI in the CE (MDA) Technical Stream Committee of the IEEE Consumer Electronics Society (CESoc).

Gian Domenico Licciardo (Senior Member, IEEE) received the electronic engineering degree from the University of Naples Federico II in 2002 and the Ph.D. degree in information engineering from the University of Salerno, Italy, in 2006. From 2007 to 2018, he was an Assistant Professor in Electronic at the University of Salerno, and he joined the Department of Industrial Engineering of the same university as an Associate Professor in 2018. He currently supervises the research activities in the circuital electronic fields, teaches digital electronics to bachelor's, master's, and Ph.D. students of the electronic engineering courses, and serves as a Coordinator for the IEEE Student Branch of the University of Salerno. He published several international papers about his main research interests which span from the modeling, simulation, and characterization of electron devices to the design of digital VLSI systems for signal processing. He is an associate editor of several international journals published by IEEE and Springer.