Deep Neural Oracle with Support Identification in the Compressed Domain

Deep Neural Oracle with Support Identification in the Compressed Domain

Luciano Prono, *Student Member, IEEE,* Mauro Mangia, *Member, IEEE,* Alex Marchioni, *Student Member, IEEE,* Fabio Pareschi, *Senior Member, IEEE,* Riccardo Rovatti, *Fellow, IEEE,* and Gianluca Setti, *Fellow, IEEE*

***Abstract*—We investigate the advantage of a two-step approach in the recovery of Compressed Sensing (CS) encoded signals in a realistic environment. First, the support of the signal is computed from the compressed measurements exploiting a Deep Neural Network (DNN). Once the support is known, the input signal can be easily recovered by a pseudoinverse operation. We consider a case study involving realistic biomedical signals and a processing architecture based on a limited precision fixed-point arithmetic unit for the implementation of the DNN and the pseudoinverse operation. In this setting, we show that the proposed approach results in a performance improvement of more than 5 dB in terms of average reconstructed signal to noise ratio (ARSNR) compared to CS state-of-the-art approach. This has been possible thanks to two main contributions reported in this paper. The first one is a theoretical investigation of the relationship between the definition of support and both the properties of the input signal and the adopted compression technique. The second one relies on replacing the pseudoinverse operation with a least mean square filter, whose small sensitivity to numerical errors grants advantages in architectures relying on limited precision fixed-point arithmetic units.**

***Index Terms*—Compressed sensing, Biosignal compression, Low-complexity compression, Deep neural networks, Quantization-aware training, Quantized implementation**

I. Introduction

THE INTEREST in the Compressed Sensing (CS) paradigm [1], [2] is mainly due to its peculiarity to enable low-cost signal compression and simultaneously transfer complexity from the encoder to the decoder stage. This aspect reverses common requirements of typical compression schemes, where the most complicated (and power-hungry) stage is the encoder.

This is particularly interesting in many applications including electromagnetic inverse scattering [3], structural health monitoring [4] and image processing [5]. Another area for which CS is particularly useful is the design of Body Area Network (BAN) nodes, which aim to efficiently acquire biosignals. This has already been demonstrated for Electrocardiographic (ECG) [6], [7], Electromyographic (EMG) [8] and Electroencephalography (EEG) [9] signals. CS has also been shown useful for magnetic resonance imaging (MRI) waveforms acquisition [10], where the main advantage stems from reducing the overall acquisition time [11].

Fig. 1. General scheme of an encoder-decoder pair for ECG signals.

As an example, the typical setup for a CS framework specialized for ECG signals is shown in Fig. 1. Such a low-complexity compression scheme requires a decoder stage for which a wide variety of approaches have been presented. One of the first proposed in the literature is *Basis Pursuit* (BP), which performs signal recovery by the solution of a linear programming problem, along with *Basis Pursuit with DeNoising* (BPDN), which takes into account also uncertainty due to noise [12]. More sophisticated decoders [13], [14] exploit assumptions on the class of signal and perturbation to ensure adequate performance when signal reconstruction has to be achieved using hardware with limited computational capability, which is typically available in a BAN gateway. In this setting, iterative algorithm such as the Orthogonal Matching Pursuit (OMP) [15] and the Compressive Sampling Matching Pursuit (CoSaMP) [16] are often used for their low computational complexity [17]. Recently, also the use of Deep Neural Network (DNN) for CS decoding has been investigated, mainly for image or video applications [18]–[21], even if some works targeting biomedical signals can be found in [22], [23].

All mentioned CS frameworks share a low-complexity encoder structure and propose several decoding approaches aiming at reducing as much as possible the number of digital words that represent a single input instance. Reaching this goal, i.e., increasing the compression ratio, further reduces the computational complexity of the encoder since less digital outputs must be both computed and dispatched.

For all these CS decoders, the fundamental hypothesis to allow correct reconstruction is that the class of input signals is *sparse*, i.e., each signal instance, in a proper *sparsity* basis, can be represented by a number of non-null coefficients much lower than the mere signal dimension [24]–[26]. Such a sparsity assumption can be declined in another way by

Manuscript received Mmmm dd, yyyy; revised Mmmm dd, yyyy.

L. Prono is with the Department of Electronics and Telecommunications, Politecnico di Torino, 10129 Torino, Italy (e-mail: luciano.prono@polito.it).

M. Mangia, A. Marchioni and R. Rovatti are with the Department of Electrical, Electronic, and Information Engineering, University of Bologna, 40136 Bologna, Italy, and also with the Advanced Research Center on Electronic Systems, University of Bologna, 40125 Bologna, Italy (e-mail: mauro.mangia2@unibo.it, alex.marchioni@unibo.it, riccardo.rovatti@unibo.it).

F. Pareschi and G. Setti are with the Department of Electronics and Telecommunications, Politecnico di Torino, 10129 Torino, Italy, and also with the Advanced Research Center on Electronic Systems (ARCES), University of Bologna, 40125 Bologna, Italy (e-mail: fabio.pareschi@polito.it; gianluca.setti@polito.it).

introducing the concept of signal *support*, that is the subset of the positions corresponding to the non-null coefficients in the sparse representation. The support identification exploited in [23] paves the way for the definition of an alternative decoding procedure where signal reconstruction is split into two phases: the former employs a DNN to *divine* the support, while the latter uses the divined support to recover the input signal with a simple linear algebra (pseudoinverse) operation.

In this paper, we extend the work in [27], where we showed preliminary results on applying the DNN-based two-step reconstruction proposed in [23] to realistic biomedical signals. Compared to these works, we introduce here several innovative points.

- The support identification is generalized in case of realistic (and therefore non-perfectly sparse) signals. In particular, with respect to [23], [27], the proposed support identification procedure takes into account not only the signal representation in the sparsity basis but also the encoder mechanism and possible sources of noise.
- In case of sparse signal, the proposed decoder mechanism outperforms state of the art CS frameworks in terms of either achieved quality of reconstruction or maximum compression ratio for a fixed quality of service.
- A low-resource implementation of the DNN-based two-step decoder is presented with emphasis on: *i*) limiting the performance degradation in case of fixed point precision arithmetic (for both DNN and pseudoinverse operation); *ii*) optimizing the memory footprint required by both DNN and pseudoinversion. To reach these goals, we adopt quantization-aware techniques in the DNN training and Least Mean Square (LMS) filter instead of direct Moore-Penrose pseudoinverse implementation.

The rest of the paper is organized as follows. In Section II we introduce some basic concepts of CS, including the advantages of separating the CS reconstruction in the two phases of support identification and coefficient computation. Section III addresses the problem of using CS with realistic signals, where the sparsity property is replaced by *compressibility*. In Section IV we discuss the hardware implementation of the proposed approach and we show results. Finally, we draw the conclusion.

II. Compressed Sensing with Support Oracle

Let us consider the input waveform divided in contiguous non-overlapped windows of the same length. Without any loss of generality, we represent windows with vectors x containing n successive Nyquist samples, i.e., $x = (x_0, \dots, x_{n-1})^\top \in \mathbb{R}^n$ where $\cdot^\top$ stands for vector transposition. The CS paradigm is based on the assumption that an orthonormal matrix D exists such that each signal instance can be expressed as $x = D\xi$, where the coefficient vector $\xi = (\xi_0, \dots, \xi_{n-1})^\top$ contains at most $\kappa \ll n$ non-zero entries. In these cases, we say that x is κ-*sparse* and ξ is its sparse representation on the sparsity basis composed by the columns of D. Let us also define the *support* $s = (s_0, \dots, s_{n-1})^\top \in \{0,1\}^n$ of ξ such that $s_j = 1$ if $\xi_j \neq 0$ and $s_j = 0$ otherwise.

In the case of sparse signals, each instance x depends only on a number of scalars that is much smaller than n. This prior is used to define an encoder procedure that compresses x by applying a linear operator that is modeled with a *sensing* matrix $A \in \mathbb{R}^{m \times n}$ with $m < n$. The encoder output is a m-dimensional measurement vector y obtained by projecting x over the rows of A as

$$y = A(x + \nu) \tag{1}$$

where the vector ν includes noise and non-idealities in the system implementation. We define the ratio n/m as *compression ratio* (CR).

The decoder aims at recovering the sparse representation of the input signal ξ, or at least its best approximation $\hat{\xi}$, by leveraging on the sparsity prior. The standard BPDN approach [12], [28] consists of finding the sparsest n-dimensional vector ξ among the infinite solutions of the ill-defined system $y = AD\xi$ by considering the following optimization problem:

$$\hat{\xi} = \arg\min_{\xi \in \mathbb{R}^n} \|\xi\|_1 \ \text{ s.t. } \ \|y - AD\xi\|_2 \leq \tau \tag{2}$$

where $\|\cdot\|_p$ indicates the p-norm of its argument and where $\tau \geq 0$ is proportional to ν. The case $\tau = 0$ defines the BP optimization problem used for the noiseless case. For both BP and BPDN, the reconstructed signal is finally obtained as $\hat{x} = D\hat{\xi}$.

To evaluate the achieved quality of service, one defines the Reconstructed Signal to Noise Ratio (also indicated simply as SNR in [6], [9]), measured in dB, as

$$\text{RSNR} = 20\log_{10}\frac{\|x\|_2}{\|x - \hat{x}\|_2} = \left(\frac{\|x\|_2}{\|x - \hat{x}\|_2}\right)_{\text{dB}} \tag{3}$$

along with the Average RSNR (ARSNR) value

$$\text{ARSNR} = \mathbf{E}\left[\left(\frac{\|x\|_2}{\|x - \hat{x}\|_2}\right)_{\text{dB}}\right] \tag{4}$$

where the $\mathbf{E}[\cdot]$ operator stands for expectation over all possible x.

Reconstruction is possible when the number of measurements m is sufficient and, intuitively, this number is related to the value of κ. CS theory identifies this relationship as $m = \mathcal{O}\left(\kappa \log\left(\frac{n}{k}\right)\right)$ [12], and in practical cases m is often chosen proportional to κ. However, such worst-case theoretical guarantees fail for $m < 2\kappa$ since, when this happens, two κ-sparse signals with non overlapping supports can be potentially mapped in the same measurement vector.

The requirements of the CS framework are not limited to a minimum number of measurements, but also include the proper design of the rows of A, that we define as *sensing sequences*. Most notably, if the elements of a generic sensing matrix row a are drawn as instances of independent and identical distributed (i.i.d.) random variables with zero-mean and unit-variance Gaussian distribution, then ξ can be recovered from y [1], [2], [29] with very high probability. Reconstruction reaches the same level of quality even if the elements of a are instances of i.i.d. antipodal (i.e.$+1/-1$) random variables [29], [30]. Since the latter choice allows an easier computation of y and simple sign inversion can be used instead of full multiplication (with obvious great implementation advantages), we

focus on this class of sensing sequences for the rest of this work. The absence of performance loss compared to real-value matrices A further motivates this choice [30].

This agnostic and general approach can then be specialized to a suitable class of signals in many ways by adopting an adapted CS approach [31]. Among the several approaches that, thanks to adaptation, guarantee better performance with respect to agnostic CS, the state of the art for the design of antipodal sensing matrices is the rakeness-based CS (Rak-CS) [31]–[33]

The Rak-CS approach models the sensing sequences not as instances of i.i.d. variables but as a stochastic process whose correlation matrix $\mathcal{A} = \mathbf{E}[aa^\top]$ is obtained as:

$$\mathcal{A} = \frac{1}{2}\frac{n\mathcal{X}}{\mathrm{tr}(\mathcal{X})} + \frac{1}{2}I_n \tag{5}$$

where I_n is an $n \times n$ identity matrix, and $\mathcal{X} = \mathbf{E}[xx^\top]$ is the correlation matrix of the stochastic process generating input instances.

Roughly speaking, the statistical adaptation of the sensing matrix proposed by Rak-CS is a middle ground between standard CS theory (that suggests i.i.d. based sensing, and for which it is $\mathcal{A} = I_n$) and an over-adapted setting where $\mathcal{A} = \mathcal{X}$. Such an approach has been proved to guarantee good performance also in case of uncommon instances [7].

A. Support Oracle based Decoder

To further formalize the consequence of the sparsity assumption, we recall that no more than κ entries in ξ are non-null, and that they are identified by the position of the elements of s such that $s_j = 1$.

Let us define the operator $\cdot_{|s}$ that, when applied to a n-dimensional vector, selects only the elements corresponding to non-null entries of s, while, when applied to n-column matrices, returns the submatrix composed of the columns whose index corresponds to $s_j = 1$.

As a result, any κ-sparse signal x can be represented by the n-dimensional binary vector s and by $\xi_{|s}$, a non-sparse vector that contains no more then κ real values. As proposed in [23], this notation paves the way to a completely different decoder approach, composed of two consecutive blocks. The first one, indicated in [23] as *support oracle* (SO), is devoted to identify the support and is capable to divine s by looking at the vector y. Then, assuming s is known, by defining $B = AD$, we can observe that

$$y = A(D\xi + \nu) = B\xi + A\nu = B_{|s}\xi_{|s} + A\nu. \tag{6}$$

The fact that $\kappa < m$ makes $B_{|s}$ a *tall* matrix (the number of rows exceeds the number of columns) such that each measurement vector y, ignoring the noise ν, possesses a unique counterimage given by $\xi_{|s} = B^\dagger_{|s}y$, where $\cdot^\dagger$ indicates the Moore-Penrose pseudoinverse operation.

In other words, given (6), to recover the input signal, it is enough a second stage computing

$$\hat{\xi}_{|s} = B^\dagger_{|s}(y - A\nu) = B^\dagger_{|s}y - B^\dagger_{|s}A\nu \tag{7}$$

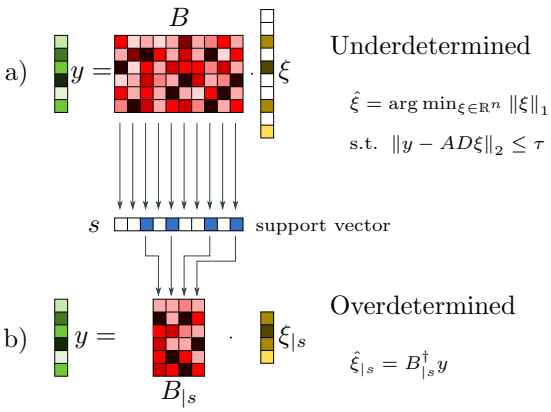

Fig. 2. The original ill-defined underdetermined problem (a) can be transformed in a overdetermined problem (b) by using the support vector s.

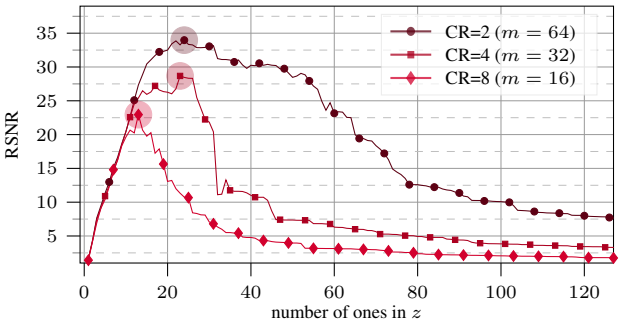

Fig. 3. RSNR as a function of the number of ones in z where the position of the ones in z follows the position of ξ entries with the highest magnitudes. Different profiles refer to $m \times n$ Rak-CS antipodal sensing matrices with $n = 128$ and where the input signal is corrupted by additive Gaussian noise (ISNR=34 dB).

that is a much simpler operation with respect to any CS recovery algorithm. The term $e = B^\dagger_{|s}A\nu$ defines the signal recovery error in the sparse representation.

The advantages introduced by the knowledge of the support vector s in the solving the recovery problem are graphically schematized in Fig. 2.

Since the aim is always the computation of the non-null entries of ξ, if s is unknown, the signal recovery is performed by inverting a wide matrix (which is an ill-defined problem) thus obtaining both null and non-null entries of ξ. Otherwise, assuming that an oracle divining s exists, the recovery problem only focuses on the computation of the non-null entries of ξ such that the recovery stage only performs the (pseudo-)inversion of a tail matrix.

III. Support Oracle for Compressed Signals

As a further important remark, one can note that almost all classes of real signals are only *approximately sparse* since the vector $\xi = D^\top x$ is composed of few entries with magnitude significantly greater than zero while the remaining entries are close to zero. In these cases, signals are not sparse but *compressible*.

As a result, it is not possible to define a support for x by looking at the vector ξ, since any possible definition would

cause rejection of a part of the signal information content. Let us therefore indicate with z a n-size binary vector such that $x = D_{|z}\xi_{|z} + x_d$, where x_d contains signal details of minor interest. With this notation, (7) can be reformulated as

$$\hat{\xi}_{|z} = B^{\dagger}_{|z}[y - A(x_d + \nu)] = B^{\dagger}_{|z}y - B^{\dagger}_{|z}A(x_d + \nu) \qquad (8)$$

where the reconstructed signal is now $\hat{x} = D_{|z}\hat{\xi}_{|z}$ and the reconstruction error in the sparse representation is $e = B^{\dagger}_{|z}A(x_d + \nu)$.

The choice of the support z is fundamental in limiting the error e since, along with A and ν, it defines the maximum achievable performance in terms of RSNR. Increasing the number of ones in z reduces the error contribution due to x_d. On the other hand, reducing the number of ones in z also reduces the reconstruction error due to the relative decrease of the effect due to ν. The optimal support z^* that balances this trade-off can be obtained, in principle, by computing the RSNR for all possible z. Unfortunately, finding z^* implies the solution of a combinatorial optimization problem that requires an exhaustive search over all possible 2^n binary z vectors.

$$z^* = \arg \max_{z \in \{0,1\}^n} \frac{\|x\|_2}{\left\|x - D_{|z}\hat{\xi}_{|z}\right\|_2} \qquad (9)$$

To overcome the impasse, we propose a greedy approach capable of achieving a good approximation of z^* with complexity as low as n steps. First, the entries of ξ are sorted in decreasing order according to their magnitude. Following this order, the z vector, initialized with all zeros, is built iteratively by adding, step by step, a new non-zero element.

At each step, the RSNR is computed. The support z is identified as the vector that achieves the highest RSNR value. Note that the support z that we estimate is actually the extension of the support applied to a compressible signal once it has been compressed by the CS encoder, and depends both on the sensing matrix A and the noise ν. We refer to this as the *Support of Compressed signal* and to the CS decoder based on the pre-computation of z as the *Support Oracle for Compressed signals* (SOC).

To validate this approach, a dataset of synthetic ECSs generated according to [34] as in [23], [27] has been considered. Differently from the approaches in the latter contributions, we consider here the generated ECG as a compressible signal, i.e., we do not impose x_d to be null. Synthetic ECG instances are generated with sample rate $256\,\mathrm{sps}$ and time windows composed of $n = 128$ successive samples. The signal is then corrupted by additive Gaussian noise so that the intrinsic signal-to-noise ratio (ISNR) is set to $34\,\mathrm{dB}$ [35].

In the example of Fig. 3 we have plotted the RSNR as a function of the number of ones in z for an instance of x and ν. The figure shows how both the maximum RSNR and the corresponding z cardinality depend on the number m of rows of A. The figure also highlights the already observed trade-off. When the cardinality of z is low, each new element inserted in the support is associated with a signal component with a large magnitude. This increases the RSNR since the projection of x along the corresponding column of D certainly exceeds the magnitude of the projection of ν on the same column of D. Conversely, when the cardinality of z is high, the additional signal information content brought by the new column of D could be lower than the corresponding noise contribution. Moreover, the RSNR values also depend on the adopted sensing matrix since the higher the compression ratio, the harder the reconstruction.

For each profile in Fig. 3, the highlighted point represents the number of ones in z that maximizes the RSNR. Hence, according to the definition of z and the greedy method we propose, this point corresponds to the support z for a specific compressible signal instance x, a noise vector ν, and a sensing matrix A.

A. Trained Support Oracle for Compressed Signals

All the above considerations about the support oracle decoder for compressed signals are based on the possibility to retrieve the signal support z from the observation of the measurements vector y. To perform this task, we adopt a solution based on a Deep Neural Network (DNN) inspired by [23]. More specifically, the task performed by the DNN is equivalent to n parallel binary classification tasks that estimate the n entries of z.

The structure of the network is reported in Fig. 4, where it is shown that the m-dimensional input layer receives the measurement vector. Then, there are three fully connected hidden layers with $2n$, $2n$ an n neurons respectively and Rectified Linear activation functions (ReLu). Finally, a fully connected output layer with n neurons and sigmoid activation functions produces a vector o with entries in $[0, 1]$. The final estimated support $\hat{z}$ is obtained by applying a threshold $\theta \in [0, 1]$ to o such that $\hat{z}_j = 1$ if $o_j \geq \theta$, and $\hat{z}_j = 0$ otherwise. The CS decoder that adopts this DNN to divine z is named *Trained Support Oracle for Compressed signal* (TSOC).

Compared to the DNN proposed in [23], the matrix A characterizing the encoder stage is not trained along with the rest of the network since the labels used during the training, i.e., the supports z, depend also on the the sensing matrix. In light of that, the training exploits measurement vectors y computed with a fixed sensing matrix A, which is still an antipodal matrix generated according to the Rak-CS approach for the considered class of signals. Note that, A plays the role of a set of hyperparameters that therefore cannot be trained with the parameters characterizing the neural network. Note also that A influences the network architecture since a different number of rows m of the sensing matrix corresponds to a different number of neurons in the input layer.

The DNN parameters set W (including weights and biases for each layer) are trained with a dataset of 2×10^6 signal instances x split in 95% for the actual training and 5% as a test set for performance assessment. For each different matrix A, DNN input-output pairs (y, z) are obtained from both vectors x and randomly drawn noise contributions ν such that $y = A(x + \nu)$. For each value of m, we generate 100 different random candidates A according to the Rak-CS framework. Rak-CS (5) requires the signal correlation matrix $\mathcal{X}$ which is estimated from 5000 signal instances generated especially for

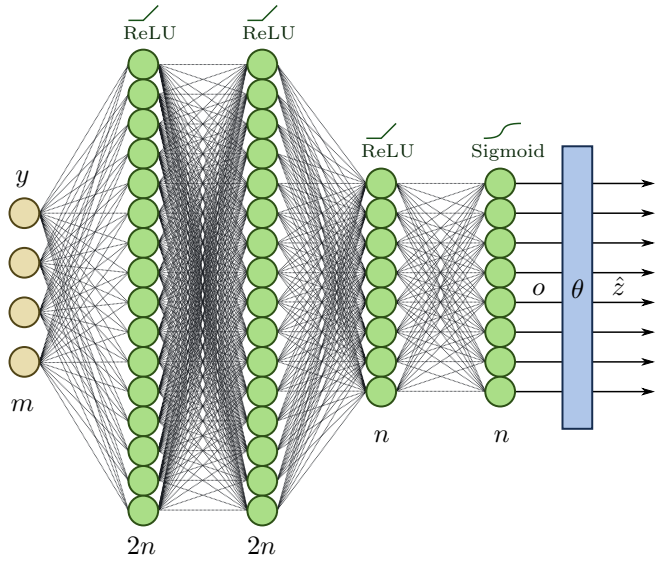

Fig. 4. Structure of the DNN employed to predict the estimated support $\hat{z}$.

TABLE I
NUMBER OF DNN PARAMETERS USED FOR THE TSOC

m	16	20	24	28	32
# param	119 552	120 576	121 600	122 624	123 648
m	36	40	44	48	
# param	124 672	125 696	126 720	127 744	

this purpose. Among these candidates A, the matrix chosen for the encoder stage of the TSOC is the one obtaining the best ARSNR over 1000 signal reconstructions when a BPDN decoder is used[1].

Since m corresponds to both the number of rows in A and the number of neurons in the input layer, the parameters characterizing the first hidden layer are $2nm$ weigths and $2n$ biases. For the case $n = 128$, Tab. I reports the total number of parameters for the adopted DNN settings where m ranges from 16 to 48.

Each of the proposed models is implemented in the TensorFlow framework [37], and the cost function is minimized using stochastic gradient descent with a batch size of 50 instances over 500 epoches and an initial learning rate value equal to 0.1. The minimized cost function is the component-wise clipped cross-entropy between z and o

$$X = -\sum_{j|z_j=1} \mathrm{L}_\epsilon\left(o_j\right) - \sum_{j|z_j=0} \mathrm{L}_\epsilon\left(1-o_j\right) \qquad (10)$$

where $\mathrm{L}_\epsilon(\cdot)$ is a clipped log function defined as $\min\{\log_2(1-\epsilon), \max\{\log_2(\epsilon), \log_2(\cdot)\}\}$ and ϵ is a small value.

Finally, 5000 new instances are used to tune the threshold θ applied to the DNN output o. Resulting values of θ in all the considered settings are close to the middle-range value 0.5, i.e., values in o vectors concentrate close to the two boundary values zero and one.

[1]The BPDN is implementes with the Spectral Projected Gradient for L1 minimization (SPGL1) toolbox [36].

TABLE II
PERFORMANCE OF THE ORACLE IN TERMS OF P, TP, TPR, TNR AND ACC WHERE $\mu(\cdot)$ MEANS AVERAGE OVER THE TEST SET.

m	μ(P)	μ(TP)	μ(TPR)	μ(TNR)	μ(ACC)
16	14.4	12.5	0.885	0.997	0.983
20	17.1	15.4	0.908	0.996	0.983
24	20.3	18.6	0.926	0.994	0.982
28	23.2	21.6	0.933	0.992	0.981
32	25.3	24.0	0.949	0.990	0.981
36	27.0	25.4	0.945	0.989	0.979
40	28.5	26.9	0.947	0.988	0.978
44	29.6	28.0	0.951	0.985	0.976
48	30.3	28.5	0.945	0.985	0.974

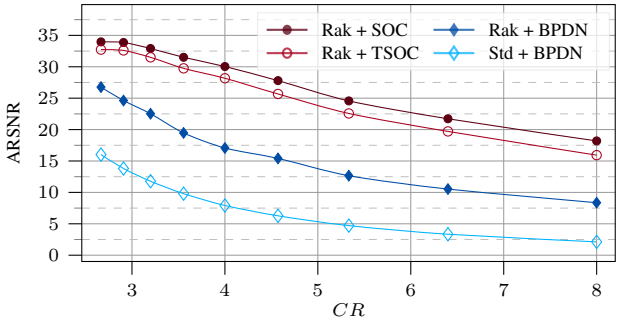

Fig. 5. ARSNR as a function of CR for the proposed TOSC along with the ideal oracle (SOC) compared with the standard decoder (BPDN). The encoder stage follows Rak-CS and the standard CS encoder coupled with BPDN is provided as a reference.

B. Results for ECGs

To assess the performance of the neural network architecture, we take into account different CS settings, each of which considers a matrix A with a different value for m that ranges from $m = 16$ (CR $= 4$) to $m = 48$ (CR $= 2.7$). Therefore, for each setting, we train a different set of parameters. As anticipated in the previous section, the task of the DNN is equivalent to a multi-label classification [38] that considers each label as an independent binary classification problem. More precisely, for each input y, the DNN produces n outputs

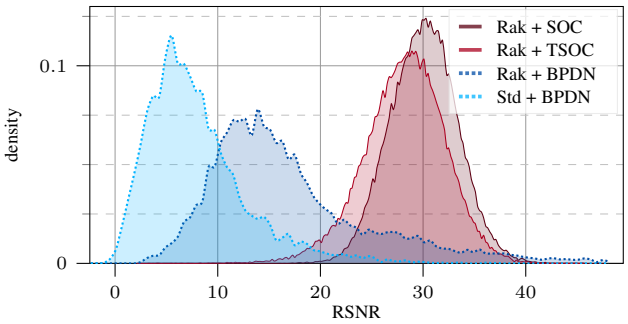

Fig. 6. Probability density functions of RSNR values with CR= 4 ($m = 32$) for the considered system configurations.

that are independently either positive or negative. If the i-th output is positive then $\hat{z}_i = 1$ while a negative outcome for the i-th output implies $\hat{z}_i = 0$.

To assess the capability of correctly estimating z, metrics related to the difference between z and $\hat{z}$ need to be adopted. Let first introduce the metrics for a single binary classification problem before generalize them for the case of n binary classifications performed by the DNN.

If both entries z_i and $\hat{z}_i$ are equal to 1 we mark this classification as a single true positive, while $z_i = \hat{z}_i = 0$ is a single true negative. In case of single miss-classifications we have either a single false positive ($\hat{z}_i = 1$ and $z_i = 0$) or a single false negative ($\hat{z}_i = 0$ and $z_i = 1$). Hence, the overall performance of a DNN prediction can be expressed in terms of the following metrics:

- Positive (P) and Negative (N):

$$\mathrm{P} = \sum_{i=0}^{n-1} z_i, \quad \mathrm{N} = n - \mathrm{P} \tag{11}$$

- True Positive (TP) and True Negative (TN)

$$\mathrm{TP} = \sum_{i=0}^{n-1} z_i \hat{z}_i, \quad \mathrm{TN} = \sum_{i=0}^{n-1} (1 - z_i)(1 - \hat{z}_i) \tag{12}$$

- TP Rate (TPR), TN Rate (TNR) and Accuracy (ACC)

$$\begin{aligned} &\mathrm{TPR} = \mathrm{TP}/\mathrm{P}, \quad \mathrm{TNR} = \mathrm{TN}/\mathrm{N}, \\ &\mathrm{ACC} = (\mathrm{TP} + \mathrm{TN})/n \end{aligned} \tag{13}$$

A summary of the average values for P, TP, TPR, TNR and ACC over the whole test set can be found in Tab. II with the aim of showing the ability of the network to correctly detects the ones in z.

The results in Tab. II show that the number of ones in $\hat{z}$ increases with m, confirming the behaviors of a single instance reported in Figure 3. The difference between P and TP, that is the number of ones wrongly estimated, is roughly constant and less than 2. As a consequence, the TPR tends to increase with m while the TNR slightly decreases. Since in general $\hat{z}$ contains more zeros than ones, the accuracy is dominated by TN so it slightly decreases with m, starting from 0.983 for $m = 16$ to 0.974 for $m = 48$.

Accuracy values very close to one ensure a high overlap between z and $\hat{z}$. Nevertheless, even when $z = \hat{z}$, the decoder still commits an error in the reconstruction as modeled in (8). Thus, the overall performance of TSOC must be evaluated in terms of either ARSNR or RSNR distribution.

Fig. 5 compares the performance in terms of the achieved ARSNR of the proposed TSOC approach with that of SOC (ideal oracle) and BPDN (SPGL1 decoder). All these approaches share the same tuned Rak-CS antipodal sensing matrices A. Further comparison shows BPDN approach coupled with matrices A following the standard CS theory (Std) where -1 and +1 occur with the same probability. Rak + TSOC outperforms Rak + BPDN with a gap of at least $5\,\mathrm{dB}$, while the loss with respect to the ideal oracle (Rak + SOC) never exceeds $2.5\,\mathrm{dB}$. Std + BPDN performance is not even comparable with the ones of the other frameworks.

To provide a further comparison between these approaches, Fig. 6 shows the RSNR distributions in case of CR $= 4$ ($m = 32$). The proposed TSOC, along with the ideal SOC, shows an RSNR variance that does not increase compared to both the already presented Std + BPDN and Rak + BPDB.

IV. Quantization-aware Decoder Architecture

In this section we investigate the implementation of the TSOC-based system in presence of possible hardware limitations, e.g., a limited precision arithmetic unit. The block scheme of the overall system has been depicted in Fig. 7 where we highlight the fact that each digital signal is associated to a number of bits. At the decoder side we can identify two main blocks: *i*) the oracle divining z, *ii*) the reconstructor that uses the oracle output to recover the original waveform.

The representation of each system quantity with a finite number of bits addresses a trade-off between computational burden/memory footprint and capability to correctly reconstruct input signals.

A preliminary investigation on this direction is reported in [39] where authors study the performance loss due to the parameter post-quantization with the two-stage decoder proposed in [23]. They first design the two blocks composing the decoder with full precision and then they simply quantize the entries of both B and D and the DNN parameters.

Here we propose different strategies to limit the loss in performance including quantization-aware techniques, as well as a different approach in the pseudoinverse operation and the quantization of the measurement vector y.

The first issue we address is the quantization of the input of the decoder y. Our analysis assumes that y is quantized by a mid-tread uniform quantizer, with 2^{b_y} levels.

Quantization may come either from the quantization of a measurement vector computed by an analog CS encoder block or from the digital processing of a digital input signal x. In this setting, we need to remember that if n is large enough, each y_i, $i = 0, 1, \ldots, m-1$ can be considered as a zero mean Gaussian distributed random variable, and setting a conversion range that includes all possible values is not possible, or simply not convenient [7], [40].

Being Δ the quantization step, σ_y^2 the variance of the distribution of each value in y, and γ a positive coefficient, we set $2^{b_y}\Delta = 2\gamma\sigma_y$, so that each entry of y is represented with b_y bits and ranges from $y_{\min} = -2^{b_y-1}\Delta = -\gamma\sigma_y$ to $y_{\max} = (2^{b_y-1} - 1)\Delta \approx \gamma\sigma_y$. After quantization, each element of y can be considered a fixed point number belonging to the set $\Sigma_y = \{-1, -1 + \Delta_y, \ldots, 1 - \Delta_y\}$, with $\Delta_y = \Delta/(2\gamma\sigma_y^2)$.

The quantized y values feed both the oracle and the reconstructor. Each of the two structures internally uses parameters that can be quantized to reduce the complexity and memory footprint.

A. The quantized oracle

As a first modification to reduce computational complexity of our DNN structure, we replace the sigmoid function in the output layer with a linear function. Since the sigmoid is

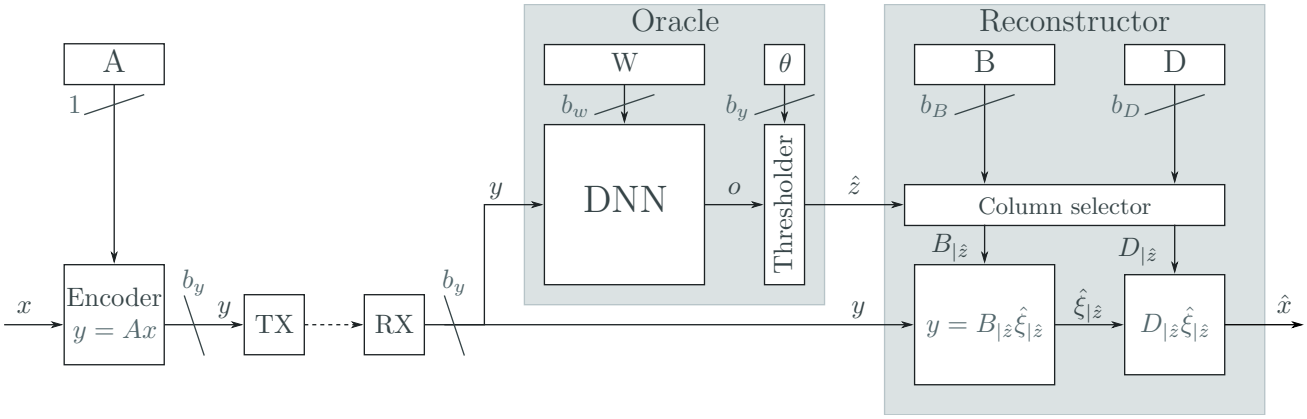

Fig. 7. Block diagram of the overall system including CS encoder, measurements dispatch and the proposed TSOC decoder. The arrows represent digital quantities and for each of them the number of bits is highlighted.

monotone and the output is thresholded, it is sufficient to adapt the threshold value θ.

Then, the parameters of the oracle are quantized to minimize the memory footprint and reduce the resources needed for the support divination. All parameters are encoded with b_w bits and constrained to be in the discrete set $\Sigma_w = \{-1, -1 + \Delta_w, \dots 1 - \Delta_w\}$, where Δ_w is the quantization step chosen be compliant with the adopted fixed point representation.

Moreover, in the hidden layers, all the neurons outputs (activations) are represented with only the b_y most significant bits such that their representation is coherent with the one of the DNN input. As a result, since both activations and parameters are represented in fixed point, integer arithmetic is sufficient to produce the oracle output. In each neuron, the inputs are multiplied by the weights and then summed, therefore, the number of bits required by the arithmetic unit is $b_y + b_w + \log_2(n_L + 1)$ where n_L is the number of neuron inputs that in our case never exceeds $2n$.

Regarding the parameters quantization, a possible choice is reported in [39] where quantization is applied at the end of the training. However, it is possible to adopt strategies during the training that help to reduce the performance loss due to quantization. Here, we investigate some approaches that we group by task:

- limiting the parameters in the set Σ_w so that they are not clipped during quantization;
- limiting the activations in the set Σ_y to avoid overflow during inference;
- updating parameters considering the effect of quantization (quantization-aware training).

The details follow.

1) Limiting the parameters range: We force the parameters to assume values in the set Σ_w with the combination of two methods: "bathtub" regularization and parameter "recycling".

The "bathtub" regularization consists in a regularization term that is added to the cost function (10) and penalizes the parameters that are outside the desired range. We can define the "bathtub" regularization function with its derivative:

$$\frac{\partial R_{\text{bathtub}}(w_{l,i})}{\partial w_{l,i}} = \begin{cases} -1 & \text{for } w_{l,i} < -1 \\ 0 & \text{for } -1 \le w_{l,i} \le 1 - \Delta_w \\ 1 & \text{for } w_{l,i} > 1 - \Delta_w \end{cases} \quad (14)$$

where $w_{l,i}$ is the i-th parameter in the l-th DNN layer. This regularization term affects the cost gradient that is used to update the parameters in the back-propagation algorithm and its effect consists in pushing the parameters value inside the desired set.

The parameter "recycling" is still applied in the training but only at the beginning of each epoch. The parameters that have a values outside the desired set are modifies to a new value uniformly randomly chosen in Σ_w.

2) Limiting the activations in their range: Since both the input and the activations are represented with fixed point, we want to force each neuron to produce an output that is in the set Σ_y. To do so, for all hidden layers, we replace the ReLU with the Saturated ReLU (SReLU) as activation function. SReLU is defined as follows:

$$\text{SReLU}(v) = \begin{cases} 0 & \text{for } v < 0 \\ v & \text{for } 0 \le v < 1 \\ 1 & \text{for } v \ge 1 \end{cases} \quad (15)$$

where v is the weighted sum generated by each neuron.

3) Quantization-aware training: Among the many solutions proposed in the literature, we investigate the results achieved by two techniques, namely fake-quantization [41] and cosine regularization [42].

Fake-quantization [41] suggests to train the network with full precision parameters and to adopt quantized values only during the feed-forward phase. This has the effect of emulating the loss of precision due to quantization.

Cosine regularization [42] is a further regularization term $R_{\text{cosine}}(W)$ added to (10) and defined as follows:

$$R_{\text{cosine}}(W) = -\sum_{l=1}^{L}\sum_{i=1}^{N_l} \frac{\lambda}{2^{b_w}} \cos(2^b w'_{l,i}\pi) \quad (16)$$

where L is equal to the number of layers composing the DNN, N_l is the number of parameter in each layer, $w'_{l,i}$ is

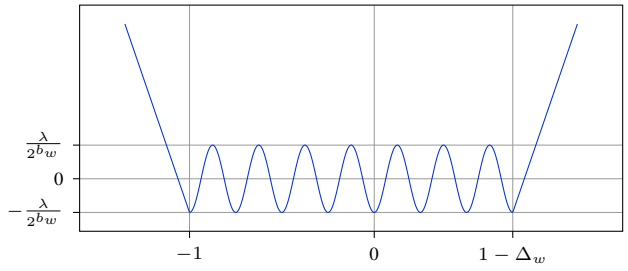

Fig. 8. The regularization function used during training: the "bathtub" regularization works outside the range $[w_{\min}, w_{\max}]$ and keeps the values inside the interval; the cosine regularization pushes the parameters towards the quantizer levels.

$w_{l,i}$ constrained to the set Σ_w, and λ is a parameter that defines the strength of the regularization. As a consequence, the parameters are pushed near the 2^{b_w} values allowed by the quantizer and, therefore, quantization error is reduced. A visual representation of the cosine regularization along with the "bathtub" regularization can be seen in Fig. 8.

B. The LMS-based reconstructor

The final stage retrieves the vector $\xi_{|\hat{z}}$ by solving the equation $y = B_{|\hat{z}}\xi_{|\hat{z}}$. Note that this is a resource-hungry operation.

Indeed, Moore-Penrose pseudo inversion is a computationally expensive operation since it requires a matrix inversion and several square roots operations. As a consequence, devices embedding a floating-point unit are preferred.

The first step adopted to reduce the reconstruction complexity is the quantization of both matrices D and B with respectively b_D and b_B bits. As a result, the memory footprint required for storing them is significantly reduced.

Moreover, since the pseudoinverse operation fundamentally solves the Least Mean Squares problem (LMS), it is possible to compute $\xi_{|\hat{z}}$ as the output of a 1-st order LMS filter [43], [44, Ch. 6]. The LMS filter employs only additions and multiplications, allowing the use of low-power fixed-point arithmetic, not possible with the pseudoinverse approach. Its mechanism is based on a gradient descent algorithm with a fixed amount of iterations q and a learning rate $\eta \ll 1$. A more detailed explanation of the LMS filter algorithm can be found in Appendix A.

C. Architecture design and results

Many hyperparameters need tuning for the final implementation of the overall decoder. They are summarized in Table III. We focus here on the case CR $= 4$. To speed up the process, we split this investigation into two phases. Firstly, we determine the setting characterizing the LMS filter where the oracle is supposed ideal (see App. A for more details). Then, we focus on the tuning of the DNN hyperparameters.

For the LMS setting, we first perform a series of tests with 5000 signal instances, imposing $\hat{z} = z$, i.e., replacing the DNN with the ideal oracle. In practice, $b_B = 9$, the learning rate η of the LMS filter equal to 2^{-6} and $\gamma = 4$ are values that correspond to the maximum ARSNR value.

TABLE III
HYPER-PARAMETERS FOR THE LOW-RESOURCES TSOC IMPLEMENTATION

Decoder input	
b_y	Number of bits used to represent measurements y
$\gamma\sigma_y$	Half range of measurements y
Oracle	
b_w	Number of bits used to represent DNN weights
λ	Strength of cosine regularization
Reconstructor	
b_D	Number of bits used to store matrices D
b_B	Number of bits used to store matrices $B = AD$
q	Iterations of the LMS filter
η	LMS filter learning rate

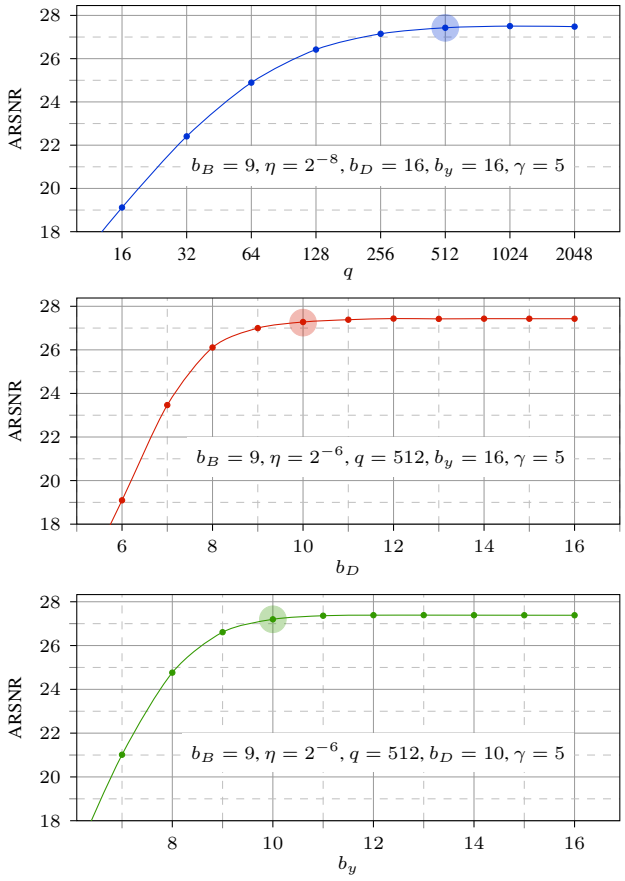

Fig. 9. Performance of the decoder with $\hat{z} = z$, with the variation of the hyper-parameters q, b_D and b_y, which have not an optimum value (i.e. a single value with best performance). The values highlighted are the choices for our implementation test.

Conversely, q, b_D and b_y exhibit profiles that saturate as their values increase, as reported for a few configurations in Fig. 9. In these cases we select the lower value below which the performance starts to degrade. The final values are $q = 512$, $b_D = 10$ and $b_y = 10$.

Once the LMS filter setting is fixed, we consider the overall non-ideal decoder to tune the oracle hyperparameters b_w and λ. We, therefore, perform a DNN training for each hyperparameters configuration. Even considering different values for

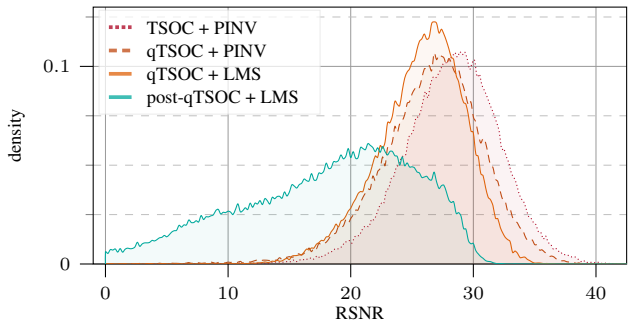

Fig. 10. Probability density functions of RSNR values with CR= 4 ($m = 32$) for the high-precision TSOC and the quantized TSOC (qTSOC) with either the pseudoinverse (PINV) or the LMS filter. The performance in the case of a-posteriori quantization is also shown (post-qTSOC).

λ, the performance is almost constant when b_w is equal or greater than 4, and it significantly degrades only for b_w lower than 4. Since one of the aims is to reduce the memory footprint dedicated to the oracle, we set $b_w = 4$, and for that value, the optimal λ is 10^{-8}.

Therefore, considering $b_w = 4$, we need $495\,\mathrm{kbit}$ to store the DNN parameters (that are 123,648 for $m = 32$), $37\,\mathrm{kbit}$ for the 4096 entries of B ($b_B = 9$ bits each), and $164\,\mathrm{kbit}$ for the 16384 entries of D ($b_D = 10$ bits each). The overall memory footprint of the decoder is $695\,\mathrm{kbit}$, to which we must add $4\,\mathrm{kbit}$ needed for the 32×128 antipodal sensing matrix A in the encoder.

We finally test the whole quantized architecture, with quantized y, W, B and D, along with either the pseudoinverse on a floating-point unit or the LMS filter approach with fixed-point arithmetic. For this final test, we use 9×10^4 ECG windows with the same setup described in Section III. The distribution of the RSNR can be found in Fig. 10, along with the results presented in Section III for the TSOC case (without quantization). The main three cases to consider are therefore the ideal (i.e. high-precision) TSOC with pseudoinverse (PINV) and the two "constrained" cases of the quantized TSOC (qTSOC) with either the pseudoinverse (PINV) or the LMS filter. As it can be seen, the observed variances of each distribution are similar to each other, while the ARSNR for the qTSOC+LMS case is $25.9\,\mathrm{dB}$ and $26.6\,\mathrm{dB}$ for qTSOC+PINV, therefore showing only a slight degradation with respect to the $28.2\,\mathrm{dB}$ characterizing the reference TSOC+PINV setting. To offer an additional reference point, the performance of the case where the DNN parameters are quantized a-posteriori (post-qTSOC), i.e. after the training, is also shown in Fig. 10. Without proper quantization-aware training techniques the performance is greatly degraded, with an ARSNR of $17.7\,\mathrm{dB}$.

Finally, In Fig. 11 it is also possible to see some reconstructed ECG samples in the case of qTSOC with LMS filter along with the original instances.

V. Conclusion

In this paper, we have investigated the application of a two-step decoder, namely, support identification by a DNN and signal reconstruction by an LMS, for a realistic signal

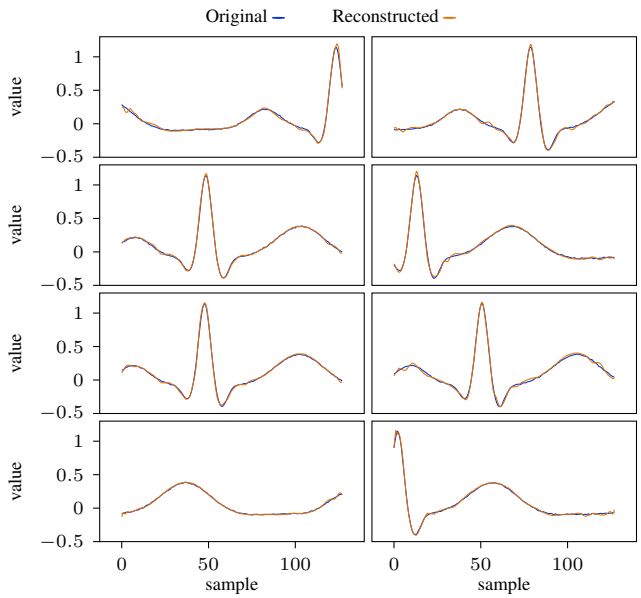

Fig. 11. Samples of the original ECG windows compared with the windows reconstructed using qTSOC + LMS filter with CR = 4 ($m = 32$).

encoded according to CS techniques, assuming a realistic hardware environment. In detail, we have considered support identification for compressed synthetic ECG signals, and we have assumed that the device implementing the decoder is limited to fixed-point arithmetic. Results show that *i*) the two-step approach ensures performances that are much more reliable compared to what obtained to a Rak-CS approach (more than $5\,\mathrm{dB}$ advantage); *ii*) the proposed architecture is robust to quantization error and numerical errors. Moreover, with a low-precision fixed-point arithmetic unit, we achieve results that are a couple of dB lower with respect to those achievable by a standard high-precision implementation.

Nevertheless, the adoption of a DNN in a CS decoder introduces two possible limitations with respect to traditional approaches: *i*) the need for a sufficiently large data set for the neural network training; *ii*) the need for an entire training in case of a change in the sensing matrix.

Appendix A
The LMS filter algorithm

The working principle of the 1-st order LMS filter can be summarized as follows. Given a system $y = Hc$, we have $c \in \mathbb{R}^n$ unknown and $y \in \mathbb{R}^m$ and $H \in \mathbb{R}^{m \times n}$ known. We also define $\hat{c} \in R^n$ as the estimated coefficients vector. The pseudocode of the LMS filter is described in Alg. 1.

In the algorithm, y_j is the j-th value of vector y, $h_{j,*}$ is the j-th row of matrix H and $\eta \ll 1$ is the learning rate coefficient. In line 4, the error ϵ on the forward prediction of the single value y_j is evaluated. Then, in line 5, ϵ is backpropagated to update $\hat{c}$. Both operations are first executed for all the entries of y (and so, for each row of H) and then iterated q times to make the solution converge.

Algorithm 1 LMS filter

Set all the entries of $\hat{c}$ to 0
for $i = 0, 1, \ldots, q-1$ **do**
  for $j = 0, 1, \ldots, m-1$ **do**
    $\epsilon = y_j - h_{j,*} \cdot \hat{c}^T$
    $\hat{c} = \hat{c} + \eta\epsilon \cdot h_{j,*}$
  end for
end for

ACKNOWLEDGMENT

This work was partially supported by the SmartData@PoliTO center, and by the Italian Ministry for Education, University and Research (MIUR) under the program "Dipartimenti di Eccellenza (2018-2022)".

REFERENCES

[1] D. L. Donoho, "Compressed Sensing," *IEEE Trans. on Inf. Theory*, vol. 52, no. 4, pp. 1289–1306, Apr. 2006. doi: 10.1109/TIT.2006.871582

[2] E. J. Candes, J. Romberg, and T. Tao, "Robust uncertainty principles: exact signal reconstruction from highly incomplete frequency information," *IEEE Trans. on Inf. Theory*, vol. 52, no. 2, pp. 489–509, Feb. 2006. doi: 10.1109/TIT.2005.862083

[3] G. Oliveri, P. Rocca, and A. Massa, "A bayesian-compressive-sampling-based inversion for imaging sparse scatterers," *IEEE Transactions on Geoscience and Remote Sensing*, vol. 49, no. 10, pp. 3993–4006, 2011.

[4] H. O. A. Ahmed and A. K. Nandi, "Three-stage hybrid fault diagnosis for rolling bearings with compressively sampled data and subspace learning techniques," *IEEE Transactions on Industrial Electronics*, vol. 66, no. 7, pp. 5516–5524, Jul. 2019. doi: 10.1109/TIE.2018.2868259

[5] H. Shen, X. Li, L. Zhang, D. Tao, and C. Zeng, "Compressed sensing-based inpainting of aqua moderate resolution imaging spectroradiometer band 6 using adaptive spectrum-weighted sparse bayesian dictionary learning," *IEEE Trans. Geosci. Remote Sens.*, vol. 52, no. 2, pp. 894–906, Feb. 2014. doi: 10.1109/TGRS.2013.2245509

[6] D. Gangopadhyay, E. G. Allstot, A. M. R. Dixon, K. Natarajan, S. Gupta, and D. J. Allstot, "Compressed sensing analog front-end for bio-sensor applications," *IEEE J. Solid-State Circuits*, vol. 49, no. 2, pp. 426–438, Feb. 2014. doi: 10.1109/JSSC.2013.2284673

[7] F. Pareschi, P. Albertini, G. Frattini, M. Mangia, R. Rovatti, and G. Setti, "Hardware-Algorithms Co-Design and Implementation of an Analog-to-Information Converter for Biosignals Based on Compressed Sensing," *IEEE Trans. Biomed. Circuits Syst.*, vol. 10, no. 1, pp. 149–162, Feb. 2016. doi: 10.1109/TBCAS.2015.2444276

[8] A. M. R. Dixon, E. G. Allstot, D. Gangopadhyay, and D. J. Allstot, "Compressed sensing system considerations for ECG and EMG wireless biosensors," *IEEE Trans. Biomed. Circuits Syst.*, vol. 6, no. 2, pp. 156–166, Apr. 2012. doi: 10.1109/TBCAS.2012.2193668

[9] M. Shoaran, M. H. Kamal, C. Pollo, P. Vandergheynst, and A. Schmid, "Compact low-power cortical recording architecture for compressive multichannel data acquisition," *IEEE Trans. Biomed. Circuits Syst.*, vol. 8, no. 6, pp. 857–870, Dec. 2014. doi: 10.1109/TBCAS.2014.2304582

[10] M. Lustig, D. L. Donoho, J. M. Santos, and J. M. Pauly, "Compressed sensing MRI," *IEEE Signal Processing Magazine*, vol. 25, no. 2, pp. 72–82, Mar. 2008. doi: 10.1109/MSP.2007.914728

[11] O. Jaspan, R. Fleysher, and M. L. Lipton, "Compressed sensing MRI: a review of the clinical literature," *British Journal of Radiology*, vol. 88, no. 1056, 2015. doi: 10.1259/bjr.20150487

[12] E. J. Candes and T. Tao, "Decoding by linear programming," *IEEE Trans. on Inf. Theory*, vol. 51, no. 12, pp. 4203–4215, Dec. 2005. doi: 10.1109/TIT.2005.858979

[13] H. Zhu, G. Leus, and G. B. Giannakis, "Sparsity-cognizant total least-squares for perturbed compressive sampling," *IEEE Trans. Signal Process.*, vol. 59, no. 5, pp. 2002–2016, May 2011. doi: 10.1109/TSP.2011.2109956

[14] S. Rangan, "Generalized approximate message passing for estimation with random linear mixing," in *2011 IEEE international symposium on information theory proceedings*, Jul. 2011, pp. 2168–2172. doi: 10.1109/ISIT.2011.6033942

[15] J. A. Tropp and A. C. Gilbert, "Signal recovery from random measurements via orthogonal matching pursuit," *IEEE Trans. on Inf. Theory*, vol. 53, no. 12, pp. 4655–4666, Dec. 2007. doi: 10.1109/TIT.2007.909108

[16] D. Needell and J. A. Tropp, "CoSaMP: Iterative signal recovery from incomplete and inaccurate samples," *Applied and Computational Harmonic Analysis*, vol. 26, no. 3, pp. 301–321, May 2009. doi: 10.1016/j.acha.2008.07.002

[17] F. Pareschi *et al.*, "Energy analysis of decoders for rakeness-based compressed sensing of ECG signals," *IEEE Trans. Biomed. Circuits Syst.*, vol. 11, no. 6, pp. 1278–1289, Dec. 2017. doi: 10.1109/TBCAS.2017.2740059

[18] A. Beck and M. Teboulle, "A fast iterative shrinkage-thresholding algorithm for linear inverse problems," *SIAM Journal on Imaging Sciences*, vol. 2, no. 1, pp. 183–202, 2009. doi: 10.1137/080716542

[19] A. Mousavi, A. B. Patel, and R. G. Baraniuk, "A deep learning approach to structured signal recovery," in *3rd annual allerton conference on communication, control, and computing (allerton)*, Sep. 2015, pp. 1336–1343. doi: 10.1109/ALLERTON.2015.7447163

[20] M. Iliadis, L. Spinoulas, and A. K. Katsaggelos, "Deep fully-connected networks for video compressive sensing," *Digital Signal Processing*, vol. 72, pp. 9 – 18, 2018. doi: 10.1016/j.dsp.2017.09.010

[21] J. Zhang and B. Ghanem, "ISTA-Net: interpretable optimization-inspired deep network for image compressive sensing," in *IEEE/CVF conference on computer vision and pattern recognition*, Jun. 2018, pp. 1828–1837. doi: 10.1109/CVPR.2018.00196

[22] B. Sun, H. Feng, K. Chen, and X. Zhu, "A deep learning framework of quantized compressed sensing for wireless neural recording," *IEEE Access*, vol. 4, pp. 5169–5178, 2016. doi: 10.1109/ACCESS.2016.2604397

[23] M. Mangia, L. Prono, A. Marchioni, F. Pareschi, R. Rovatti, and G. Setti, "Deep Neural Oracles for Short-window Optimized Compressed Sensing of Biosignals," *IEEE Trans. Biomed. Circuits Syst.*, vol. 14, no. 3, pp. 545–557, Jun. 2020. doi: 10.1109/TBCAS.2020.2982824

[24] R. G. Baraniuk, E. Candes, R. Nowak, and M. Vetterli (Eds.), "Special issue on compressive sampling," *IEEE Signal Processing Magazine*, vol. 25, no. 2, Mar. 2008.

[25] R. G. Baraniuk, E. Candes, M. Elad, and Y. Ma (Eds.), "Special issue on applications of sparse representation and compressive sensing," *Proceedings of the IEEE*, vol. 98, no. 6, Jun. 2010.

[26] D. Allstot, R. Rovatti, and G. Setti (Eds.), "Special issue on circuits, systems and algorithms for compressed sensing," *IEEE Journal on Emerging and Selected Topics in Circuits and Systems*, vol. 2, no. 3, Sep. 2012.

[27] M. Mangia, A. Marchioni, L. Prono, F. Pareschi, R. Rovatti, and G. Setti, "Low-power ECG acquisition by compressed sensing with deep neural oracles," in *2020 2nd IEEE international conference on artificial intelligence circuits and systems (AICAS)*, Aug. 2020, pp. 158–162. doi: 10.1109/AICAS48895.2020.9073945

[28] E. J. Candès, "The restricted isometry property and its implications for compressed sensing," *Comptes Rendus Mathematique*, vol. 346, no. 9, pp. 589 – 592, 2008. doi: 10.1016/j.crma.2008.03.014

[29] M. Mangia, F. Pareschi, V. Cambareri, R. Rovatti, and G. Setti, *Adapted compressed sensing for effective hardware implementations: A design flow for signal-level optimization of compressed sensing stages*. Springer International Publishing, 2018. ISBN 978-3-319-61372-7

[30] J. Haboba, M. Mangia, F. Pareschi, R. Rovatti, and G. Setti, "A pragmatic look at some compressive sensing architectures with saturation and quantization," *IEEE Journal on Emerging and Selected Topics in Circuits and Systems*, vol. 2, no. 3, pp. 443–459, Sep. 2012. doi: 10.1109/JETCAS.2012.2220392

[31] M. Mangia, F. Pareschi, R. Rovatti, and G. Setti, "Adapted compressed sensing: A game worth playing," *IEEE Circuits Syst. Mag.*, vol. 20, no. 1, pp. 40–60, 2020. doi: 10.1109/MCAS.2019.2961727

[32] M. Mangia, R. Rovatti, and G. Setti, "Rakeness in the design of analog-to-information conversion of sparse and localized signals," *IEEE Trans. Circuits Syst. I, Reg. Papers*, vol. 59, no. 5, pp. 1001–1014, May 2012. doi: 10.1109/TCSI.2012.2191312

[33] M. Mangia, F. Pareschi, V. Cambareri, R. Rovatti, and G. Setti, "Rakeness-based design of low-complexity compressed sensing," *IEEE Trans. Circuits Syst. I, Reg. Papers*, vol. 64, no. 5, pp. 1201–1213, May 2017. doi: 10.1109/TCSI.2017.2649572

[34] P. E. McSharry, G. D. Clifford, L. Tarassenko, and L. A. Smith, "A dynamical model for generating synthetic electrocardiogram signals," *IEEE Trans. on Biom. Eng.*, vol. 50, no. 3, pp. 289–294, Mar. 2003. doi: 10.1109/TBME.2003.808805

[35] Y. Zigei, A. Cohen, and A. Katz, "The weighted diagnostic distortion (WDD) measure for ECG signal compression," *IEEE Trans.*

on Biom. Eng., vol. 47, no. 11, pp. 1422–1430, Nov. 2000. doi: 10.1109/TBME.2000.880093
[36] E. van den Berg and M. P. Friedlander, "SPGL1: A solver for large-scale sparse reconstruction," Dec. 2019.
[37] M. Abadi *et al.*, "TensorFlow: large-scale machine learning on heterogeneous systems," 2015, software available from tensorflow.org.
[38] J. Read, B. Pfahringer, G. Holmes, and E. Frank, "Classifier chains for multi-label classification," *Machine Learning*, vol. 85, no. 3, p. 333, Jun. 2011. doi: 10.1007/s10994-011-5256-5
[39] L. Prono, M. Mangia, A. Marchioni, F. Pareschi, R. Rovatti, and G. Setti, "Low-power fixed-point compressed sensing decoder with support oracle," in *2020 IEEE international symposium on circuits and systems (ISCAS)*, Oct. 2020, pp. 1–5. doi: 10.1109/ISCAS45731.2020.9180502
[40] M. Mangia, F. Pareschi, R. Rovatti, G. Setti, and G. Frattini, "Coping with saturating projection stages in RMPI-based Compressive Sensing," in *2012 IEEE int. Symp. Circuits syst.* IEEE, May 2012, pp. 2805–2808. doi: 10.1109/ISCAS.2012.6271893
[41] V. Peluso and A. Calimera, "Energy-driven precision scaling for fixed-point ConvNets," in *2018 IFIP/IEEE int. Conf. Very large scale integr.*, Oct. 2018, pp. 113–118. doi: 10.1109/VLSI-SoC.2018.8644902
[42] C. Song, B. Liu, W. Wen, H. Li, and Y. Chen, "A quantization-aware regularized learning method in multilevel memristor-based neuromorphic computing system," in *2017 IEEE 6th non-volatile mem. Syst. Appl. Symp.*, Aug. 2017, pp. 1–6. doi: 10.1109/NVMSA.2017.8064465
[43] B. Widrow and M. E. Hoff, "Adaptive switching circuits," in *IRE WESCON conv. Rec.*, vol. 4, 1960, pp. 96–104.
[44] S. O. Haykin, *Adaptive filter theory*, 5th ed. Ontario, Canada: Pearson, 2014. ISBN 978-0-13-267149-1

Luciano Prono received the M.Sc. degree in Electronics Engineering in 2019 from Politecnico di Torino, where he is currently pursuing the Ph.D. degree in Electrical, Electronics and Communication Engineering. His main research interests are compressed sensing, low power deep learning systems and neuromorphic systems.

Mauro Mangia (S'09-M'13) received the B.Sc. and M.Sc. degrees in electronic engineering and the Ph.D. degree in information technology from the University of Bologna, Bologna, Italy, in 2005, 2009, and 2013, respectively. He was a Visiting Ph.D. Student with the Ecole Polytechnique Federale de Lausanne in 2009 and 2012. He is currently a Post-Doctoral Researcher with ARCES, Statistical Signal Processing Group, University of Bologna. His research interests are in nonlinear systems, machine learning, compressed sensing, anomaly detection, Internet of Things, Big Data analytics and optimization.

He was a recipient of the 2013 IEEE CAS Society Guillemin-Cauer Award and of the 2019 IEEE BioCAS Transactions Best Paper Award. He received the Best Student Paper Award at ISCAS2011. He was the Web and Social Media Chair for ISCAS2018.

Alex Marchioni received the B.S. and M.S. degree (with honors) in electronic engineering from the University of Bologna, respectively in 2011 and 2015. In 2018, he joined the Department of Electrical, Electronic, and Information Engineering "Guglielmo Marconi" (DEI) of the University of Bologna as research fellow where he is currently pursuing the Ph.D. degree in Electronics, Telecommunication and Information Technology. His research interests include compressed sensing, Internet of Things, signal processing, and machine learning with focus on dimensionality reduction and anomaly detection.

Fabio Pareschi (S'05-M'08-SM'19) received the Dr. Eng. degree (Hons.) in electronic engineering from the University of Ferrara, Italy, in 2001, and the Ph.D. degree in information technology from the University of Bologna, Italy, in 2007, under the European Doctorate Project (EDITH).

He is currently an Assistant Professor with the Department of Electronic and Telecommunication, Politecnico di Torino. He is also a Faculty Member with ARCES, University of Bologna. His research activity focuses on analog and mixed-mode electronic circuit design, statistical signal processing, compressed sensing, random number generation and testing, and electromagnetic compatibility.

Dr. Pareschi received the Best Paper Award at ECCTD 2005 and the Best Student Paper Award at EMC Zurich 2005 and IEEE EMCCompo 2019. He was a recipient of the 2019 IEEE BioCAS Transactions Best Paper Award. He served as an Associate Editor for the IEEE TRANSACTIONS ON CIRCUITS AND SYSTEMS-PART II from 2010 to 2013. He is currently Associate Editor for the IEEE OPEN JOURNAL OF CIRCUITS AND SYSTEMS.

Riccardo Rovatti (M'99-SM'02-F'12) received the M.S. degree in electronic engineering and the Ph.D. degree in electronics, computer science, and telecommunications from the University of Bologna, Italy, in 1992 and 1996, respectively. He is currently a Full Professor of electronics with the University of Bologna. He has authored approximately 300 technical contributions to international conferences and journals and two volumes. His research focuses on mathematical and applicative aspects of statistical signal processing and on the application of statistics to nonlinear dynamical systems.

He was Distinguished Lecturer of the IEEE CAS Society for the years 2017-2018. He was a recipient of the 2004 IEEE CAS Society Darlington Award, the 2013 IEEE CAS Society Guillemin-Cauer Award and the 2019 IEEE BioCAS Transactions Best Paper Award. He received the Best Paper Award at ECCTD 2005 and the Best Student Paper Award at the EMC Zurich 2005 and ISCAS 2011. He contributed to nonlinear and statistical signal processing applied to electronic systems.

Gianluca Setti (S89,M91,SM02,F06) received a Ph.D. degree in Electronic Engineering and Computer Science from the University of Bologna in 1997. From 1997 to 2017 he has been with the School of Engineering at the University of Ferrara, Italy as an Assitant, Associate and, since 2009 as a Full Professor of Circuit Theory and Analog Electronics. Since December 2017 he is a Professor of Electronics for Signal and Data Processing at the Department of Electronics and Telecommunications (DET) of Politecnico di Torino, Italy. Since 2002 is also a permanent (in kind) faculty member of ARCES, University of Bologna. His research interests include nonlinear circuits, recurrent neural networks, statistical signal processing, electromagnetic compatibility, compressive sensing, biomedical circuist and systems, power electronics, design and implementation of IoT nodes.

Dr. Setti received the 1998 Caianiello prize for the best Italian Ph.D. thesis on Neural Networks. He is also receipient of the 2013 IEEE CAS Society Meritorious Service Award and co-recipient of the 2004 IEEE CAS Society Darlington Award, of the 2013 IEEE CAS Society Guillemin-Cauer Award, the 2019 IEEE Transactions on Biomedical Circuits and Systems best paper award, as well as of the best paper award at ECCTD2005, and the best student paper award at EMCZurich2005, ISCAS2011, PRIME2019 and EMCCOMPO 2019.

He held several editorial positions and served, in particular, as the Editor-in-Chief for the IEEE Transactions on Circuits and Systems - Part II (2006-2007) and of the IEEE Transactions on Circuits and Systems - Part I (2008-2009). He also served in the editorial Board of IEEE Access (2013-2015) and, since of the Proceedings of the IEEE (2015-2018). Since 2019 he served as the first non US Editor-in-Chief of the Proceedings of the IEEE, the flagship journal of the Institute.

Dr. Setti was the Technical Program Co-Chair of NDES2000 (Catania), ISCAS2007 (New Orleans), ISCAS2008 (Seattle), ICECS2012 (Seville), BioCAS2013 (Rotterdam) as well as the General Co-Chair of NOLTA2006 (Bologna) and ISCAS2018 (Florence).

He was a Distinguished Lecturer (2004-2005 and 2014-2015) of the IEEE CAS Society, as well as a member of its Board of Governors (2005-2008), and served as the 2010 CASS President. He held several other volunteer positions for the IEEE and in 2013-2014 he was the first non NorthAmerican Vice President of the IEEE for Publication Services and Products.