2016 IEEE INTERNATIONAL WORKSHOP ON MACHINE LEARNING FOR SIGNAL PROCESSING, SEPT. 13–16, 2016, SALERNO, ITALY

BIOMEDICAL SIGNAL COMPRESSION WITH TIME- AND SUBJECT-ADAPTIVE DICTIONARY FOR WEARABLE DEVICES

Valentina Vadori, Enrico Grisan, Michele Rossi

Dept. of Information Engineering, University of Padova, Italy

Wearable devices allow the seamless and inexpensive gathering of biomedical signals such as electrocardiograms (ECG), photoplethysmograms (PPG), and respiration traces (RESP). They are battery operated and resource constrained, and as such need dedicated algorithms to optimally manage energy and memory. In this work, we design SAM, a Subject-Adaptive (lossy) coMpression technique for physiological quasi-periodic signals. It achieves a substantial reduction in their data volume, allowing efficient storage and transmission, and thus helping extend the devices' battery life. SAM is based upon a *subject-adaptive* dictionary, which is learned and refined at runtime exploiting the time-adaptive self-organizing map (TASOM) unsupervised learning algorithm. Quantitative results show the superiority of our scheme against state-of-the-art techniques: compression ratios of up to 35-, 70- and 180-fold are generally achievable respectively for PPG, ECG and RESP signals, while reconstruction errors (RMSE) remain within 2% and 7% and the input signal morphology is preserved.

Index Terms— wearable devices, signal compression, unsupervised learning, SOM, energy efficiency.

1. INTRODUCTION

Wearable devices are expected to naturally blend into our everyday activities, being exploited to update medical records via the Internet, report sudden variations or allow for prompt intervention [1]. However, since they are *resource constrained* in terms of energy and memory, dedicated algorithms to *optimally manage* the acquired signals are needed. To this end, we advocate compressing the collected time series right on the wearable devices, so that the data to be stored and sent will take a small portion of its original space and the energy required for transmission is decreased.

We design a lossy compression algorithm that allow substantial reductions in the data size of quasi-periodic signals such as electrocardiographic (ECG), photo-plethysmographic (PPG) and respiration (RESP) traces. It builds upon *self-organizing maps* (SOM) [2, 3], which are unsupervised learning neural networks that produce low-dimensional, topology preserving and discretized representations of the input data space. To allow for realtime learning and adaptation, our parametric compressor utilizes the *time-adaptive self-organizing map* (TASOM) [4], which we combine with *vector quantization* (VQ) and *motif extraction* techniques. Our algorithm, called SAM ("Subject-Adaptive coMpression"), learns and adapts a *dictionary* at runtime, right on the subject that wears the device, and uses it for signal compression: the synaptic weights of the TASOM's neurons become progressively and adaptively tuned to approximate the subject's signal distribution, without any prior knowledge upon it. These weights form the codewords of SAM's dictionary.

Several dictionary-based solutions were proposed in the literature, the most notable being Gain-Shape Vector Quantization (GSVQ) [5]. GSVQ however relies on *offline* learning approaches where 1) the dictionary is obtained from precollected datasets and 2) a stream of *residuals* is transmitted to compensate for changes in the signal statistics at runtime. In this paper, SAM is compared against GSVQ and other compression algorithms from the literature involving linear approximations [6], Fourier [7], Wavelet [8] transforms and compressive sensing (CS) [9]. SAM surpasses all of them, achieving remarkable performance, especially at high compression ratios, where the reconstruction error (Root Mean Square Error, RMSE) at the decompressor is kept between 2% and 7% of the input signal's amplitude.

Our main contributions are: 1) An original subject- and time-adaptive dictionary-based compression algorithm, which builds and maintains a dictionary at runtime utilizing self-organizing maps, vector quantization and motif extraction techniques. 2) A thorough performance evaluation of SAM, comparing its performance against that of popular compression algorithms from the literature.

2. PRELIMINARIES ON VECTOR QUANTIZATION FOR SIGNAL COMPRESSION

VQ extends the quantization of scalar random variables to random vectors [10]. Let $\boldsymbol{x} \in \mathbb{R}^m$ be an m dimensional input random vector. A **vector quantizer** is described by: *i)* a set of *decision regions* $\{I_j | I_j \subseteq \mathbb{R}^m\}_{j=1}^{L}$, that represents a partition of $\mathbb{R}^m$, *ii)* a finite set of *codewords* $\mathcal{C} = \{\boldsymbol{y}_j | \boldsymbol{y}_j \in \mathbb{R}^m\}_{j=1}^{L}$ called *codebook* or *dictionary*, where each codeword $\boldsymbol{y}_j$ is assigned a unique index and *iii)* a *quantization rule* $q(\cdot)$: $q(\boldsymbol{x}) = \boldsymbol{y}_j$ if $\boldsymbol{x} \in I_j$. A compression system based on VQ involves an encoder and a decoder. At the

The work in this paper has been supported by the SAMSUNG GRO program and by the University of Padova through the "progetto d'ateneo" IoT-SURF. Any opinions, findings, and conclusions herein are those of the authors and do not necessarily represent those of the funding institutions.

encoder, the samples from the data source are grouped into vectors and each of them is inputted into the VQ, which maps each input vector $\boldsymbol{x}$ onto the best matching codeword, termed $\boldsymbol{y}_{j^*}$, according to the quantization rule. Compression is achieved since the index associated with $\boldsymbol{y}_{j^*}$ is transmitted to the decoder in place of the whole codeword. Because the decoder has exactly the same dictionary stored at the encoder, it retrieves the codeword given its index through a table lookup. The quality of reconstruction is measured by the average distortion between the quantizer input $\boldsymbol{x}$ and the quantizer output $\boldsymbol{y}_{j^*}$. A common distortion measure is the *root mean square error* (RMSE), which is defined as:

$$E[d(\boldsymbol{x}, \boldsymbol{y}_j)] = \left(\sum_{j=1}^{L} \int_{I_j} \|\boldsymbol{a} - \boldsymbol{y}_j\|^2 f_{\boldsymbol{x}}(\boldsymbol{a}) d\boldsymbol{a} \right)^{\frac{1}{2}}, \quad (1)$$

where $\|\boldsymbol{a} - \boldsymbol{y}_j\|$ is the Euclidean distance between vectors $\boldsymbol{a}$ and $\boldsymbol{y}_j$ and $f_{\boldsymbol{x}}(\cdot)$ is the probability density function (pdf) of the random vector $\boldsymbol{x}$. The design of an optimal VQ consists of finding the codebook $\mathcal{C}$ and the partition of $\mathbb{R}^m$ that minimize the average distortion. Once the VQ has been designed, the quantization rule becomes $q(\boldsymbol{x}) = \text{argmin}_{\boldsymbol{y}_j} \|\boldsymbol{x} - \boldsymbol{y}_j\|$, i.e., the selected $\boldsymbol{y}_{j^*}$ is the *nearest codeword* to the input vector $\boldsymbol{x}$.

Linde, Buzo and Gray provided an iterative algorithm to build a VQ that reaches a local optimum of $E[d(\boldsymbol{x}, \boldsymbol{y}_j)]$ [10]. It essentially defines an initial dictionary and proceeds by repeatedly computing the decision regions and improving the codewords until the average distortion falls below a given threshold. Note that this algorithm does not adapt the codebook at runtime. SAM constructs two dictionaries that adapt themselves to the incoming vectors $\boldsymbol{x}$: the former through online adaptation of the prototypes and the second following a batch learning approach, i.e., its prototypes are copied from the ones in the online dictionary as a response to changes in the signal statistics.

3. SELF-ORGANIZING MAPS

The SOM and its time-adaptive version (TASOM) are single layer feed-forward networks having an input layer of source nodes that projects directly onto an output layer of neurons. The SOM provides a structured representation of the input data distribution with the synaptic-weight vectors acting as prototypes. For its output layer, we consider a rectangular lattice $\mathcal{A}$ with L neurons arranged in M rows and M columns. The input space is m-dimensional, i.e., $\mathcal{X} \subset \mathbb{R}^m$ with input vectors $\boldsymbol{x} = [x_1, x_2, \ldots, x_m]^T \in \mathcal{X}$. The SOM input layer has m source nodes, each associated with a single component of the input vector $\boldsymbol{x}$ and each neuron in the lattice is connected to all the source nodes. The links (synapses) between the source nodes and the neurons are weighted, such that the j^{th} neuron is associated with a *synaptic-weight vector* denoted by $\boldsymbol{w}_j = [w_{j1}, w_{j2}, \ldots, w_{jm}]^T \in \mathbb{R}^m$, $j = 1, \ldots, L$, where $L = M^2$ is the total number of neurons in $\mathcal{A}$. Training is unsupervised. Let $\{\boldsymbol{x}(n)\}_{n=0}^N$ be the training set of unlabeled examples (training input patterns), selected at random from $\mathcal{X}$. The learning process proceeds iteratively, from $n = 0$ to $n = N$, where N should be large enough so that self organization develops properly. At iteration n, the n^{th} training input pattern $\boldsymbol{x}(n)$ is presented to the SOM and the following three steps are performed [3]:

Competition. The neurons compete among themselves to be selected as the winning neuron, the one whose synaptic-weight vector most closely matches $\boldsymbol{x}(n)$ according to the Euclidean distance. Its index $i(\boldsymbol{x})$ satisfies:

$$i(\boldsymbol{x}) = \text{argmin}_j \|\boldsymbol{x}(n) - \boldsymbol{w}_j(n)\|, \quad j = 1, \ldots, L\,. \quad (2)$$

Cooperation. The winning neuron $i(\boldsymbol{x})$ identifies the center of a topological neighborhood of cooperating neurons modeled by the function $h_{ij}(n)$. If d_{ij} is the lateral distance between $i(\boldsymbol{x})$ and neuron j in $\mathcal{A}$, then $h_{ij}(n)$ is symmetric around $i(\boldsymbol{x})$ and its amplitude decreases monotonically with increasing lateral distance d_{ij}. Moreover, $h_{ij}(n)$ shrinks over time. In this work, we set $h_{ij}(n) = \exp(-d_{ij}^2/(2\sigma(n)^2))$, where $\sigma(n)$ is the width of the topological neighborhood, exponentially decreasing with increasing time n (see [3]).

Synaptic Adaptation. The synaptic-weight vector $\boldsymbol{w}_j(n)$ of neuron j at time n is changed through the equation:

$$\boldsymbol{w}_j(n+1) = \boldsymbol{w}_j(n) + \eta(n) h_{ij}(n)(\boldsymbol{x}(n) - \boldsymbol{w}_j(n))\,. \quad (3)$$

(3) has the effect of moving the synaptic-weight vector $\boldsymbol{w}_{i(\boldsymbol{x})}$ of the winning neuron $i(\boldsymbol{x})$ (and the synaptic-weight vectors of the neurons in its topological neighborhood, according to $h_{ij}(n)$) toward the input vector $\boldsymbol{x}$. The learning-rate parameter $\eta(n)$ starts at some initial value $\eta(0)$ and then exponentially decreases with increasing time n.

Once the SOM algorithm has terminated, a nonlinear transformation (*feature map*) $\Phi : \mathcal{X} \to \mathcal{A}$ is obtained as $\Phi(\boldsymbol{x}) = \boldsymbol{w}_{i(\boldsymbol{x})}$, where the index $i(\boldsymbol{x})$ is found according to (2). $\Phi(\cdot)$ is a quantization rule as it approximates the input data space $\mathcal{X}$ with the finite set of weights (prototypes) $\boldsymbol{w}_j \in \mathcal{A}$. However, upon completion of the learning phase, the SOM map stabilizes and further learning / adaptation to new input distributions is difficult. In the presence of non-stationary signals, adaptive learning must be employed to update the feature map. This is the reason why in our lossy compression technique we adopt the time-adaptive self-organizing map (TASOM). The TASOM has been introduced in [4] as an extended version of the basic SOM; it works well both with stationary and non-stationary environments, preserving the SOM properties. In a TASOM, each neuron j, $j = 1, \ldots, L$, has a synaptic-weight vector $\boldsymbol{w}_j \in \mathbb{R}^m$ with its own learning-rate $\eta_j(n)$ and neighborhood width $\sigma_j(n)$, which are continuously adapted so as to allow a potentially unlimited training of the synaptic-weight vectors. This feature enables the TASOM to be more flexible and to approximate the input data space distribution as it evolves.

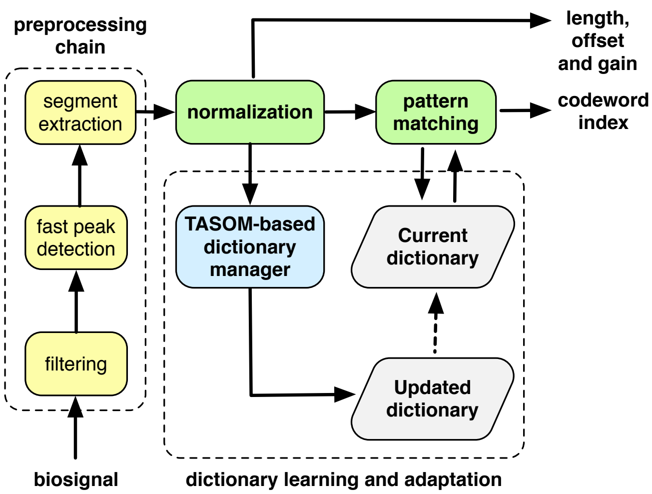

Fig. 1. Diagram of the SAM compression algorithm.

4. SAM: SUBJECT-ADAPTIVE COMPRESSION OF BIOMEDICAL SIGNALS

We leverage the quasi-periodic nature of the considered vital signs to develop a lossy compression technique with subject-adaptive dictionary. First, we identify as *motifs* the sequence of samples between consecutive peaks (which we call *segments*) and we use them to build a *dictionary* that stores typical segments and is maintained consistent and well representative through online updates. The TASOM unsupervised learning algorithm is utilized to construct and manage the dictionary. A diagram of the proposed technique is shown in Fig. 1. The physiological signal is first preprocessed through a third-order Butterworth filter to remove artifacts. Then the fast and simple peak detection algorithm exposed in [11] is employed to locate the signal peaks. The segment extractor splits the signal into segments made up of samples between subsequent peaks and inputs them to the normalization module. Since the segments may have different lengths, linear interpolation resizes the current segment with length $r_{\boldsymbol{x}}(n)$ to a fixed length m. We refer to the resized segment as $\boldsymbol{x}(n)$. The normalization module applies the following transformation to each segment's sample: $x_k(n) = (x_k(n) - e_{\boldsymbol{x}}(n))/g_{\boldsymbol{x}(n)}, k = 1, \ldots, m$, where $e_{\boldsymbol{x}}(n) = \sum_{k=1}^{m} x_k(n)/m$ is the offset and $g_{\boldsymbol{x}}(n) = \sqrt{\sum_{k=1}^{m} x_k(n)^2/m}$ is the gain. The normalized segment feeds the dictionary manager, which uses it to update the dictionary, and the pattern matching module, which returns the best matching codeword from the dictionary and outputs its index. The segment's original length, offset, gain and codeword index are then sent to the receiver in place of the original samples.

The TASOM-based dictionary manager is the key block of SAM. We tune the TASOM paradigm to a communication scenario consisting of a transmitting wearable device and a receiver, such as a PDA or smartphone. At any time instant n, two dictionaries are maintained at the transmitter: the *current dictionary* ($\mathcal{C}^c(n)$), which is used to compress the input signal, and the *updated dictionary* ($\mathcal{C}^u(n)$), which undergoes updating at each time instant through the TASOM algorithm and is maintained to track statistical changes in the input signal's distribution. As for the dictionaries, we consider a TASOM with L neurons. When the compression scheme is activated for the first time, a sufficient number N of signal segments are provided as input to the TASOM to perform a *preliminary training phase*. Such training allows the map to learn the subject signal's distribution. This may be accomplished the first time the subject wears the device. After this, a first subject-specific dictionary is available. It can be used for compression and can also be updated at runtime as more data is acquired. Let assume that time is reset when the preliminary training ends and assume $n = 0$ at such point. Both $\mathcal{C}^c(0) = \{\boldsymbol{c}_1^c(0), \ldots, \boldsymbol{c}_L^c(0)\}$ and $\mathcal{C}^u(0) = \{\boldsymbol{c}_1^u(0), \ldots, \boldsymbol{c}_L^u(0)\}$ are defined as the dictionaries whose codewords $\boldsymbol{c}_*^*(0)$ are equal to the synaptic-weight vectors of the TASOM. At time $n = 0$, we have $\boldsymbol{c}_j^c(0) = \boldsymbol{c}_j^u(0) = \boldsymbol{w}_j(0)$, $j = 1, \ldots, L$. Let also assume that the decompressor at the receiver is synchronized with the compressor, that is, it owns a copy of $\mathcal{C}^c(0)$. From time 0 onwards, for any new segment $\boldsymbol{x}(n)$ ($n = 1, 2, \ldots$) the following procedure is followed:

Algorithm [SAM]:

1) Map $\boldsymbol{x}(n)$ onto the index of the best matching codeword in $\mathcal{C}^c(n)$, i.e., map $\boldsymbol{x}(n)$ onto the index $i_{\boldsymbol{x}}(n)$ such that

$$i_{\boldsymbol{x}}(n) = \text{argmin}_j \, \|\boldsymbol{x}(n) - \boldsymbol{c}_j^c(n)\| \, , \, j = 1, \ldots, L \, . \quad (4)$$

2) Let $d(n) = \|\boldsymbol{x}(n) - \boldsymbol{c}_i^c(n)\|$ be the distance between the current segment and the associated codeword, where we use index i as a shorthand notation for $i_{\boldsymbol{x}}(n)$. Use $\boldsymbol{x}(n)$ as the new input for the current iteration of the TASOM learning algorithm and obtain the new synaptic-weight vectors $\boldsymbol{w}_j(n)$, $j = 1, \ldots, L$.

3) Update $\mathcal{C}^u(n)$ by using the weights obtained in step 2, i.e., setting $\boldsymbol{c}_j^u(n) \leftarrow \boldsymbol{w}_j(n)$ for $j = 1, \ldots, L$.

4) Let $\varepsilon > 0$ be a tuning parameter. If $d(n)/\|\boldsymbol{x}(n)\| > \varepsilon$, then update $\mathcal{C}^c(n)$ by replacing it with $\mathcal{C}^u(n)$, i.e., $\mathcal{C}^c(n) \leftarrow \mathcal{C}^u(n)$ and re-map $\boldsymbol{x}(n)$ onto the index of the best matching codeword in the new dictionary $\mathcal{C}^c(n)$, i.e., map $\boldsymbol{x}(n)$ onto the index $i_{\boldsymbol{x}}(n)$ obtained through (4) using the new dictionary $\mathcal{C}^c(n)$.

5) Send to the receiver the segment's original length $r_{\boldsymbol{x}}(n)$, its offset $e_{\boldsymbol{x}}(n)$, gain $g_{\boldsymbol{x}}(n)$, and the codeword index $i_{\boldsymbol{x}}(n)$. If the current dictionary has been changed in step 4, then also send $\mathcal{C}^u(n)$.

Step 2 makes it possible to always maintain an updated approximation of the input segment distribution at the transmitter. With step 4, we check the validity of the approximation provided by the current dictionary (the one used for compression, which is also known at the receiver). The tunable parameter ε is used to control the signal reconstruction fidelity at the decompressor: if $d(n)/\|\boldsymbol{x}(n)\| \leq \varepsilon$, codeword

$\boldsymbol{c}^c_{i_{\boldsymbol{x}}(n)}(n)$ is assumed to be a suitable representation of the current segment, otherwise $\mathcal{C}^c(n)$ is replaced with the updated dictionary $\mathcal{C}^u(n)$ and the encoding mapping is re-executed. Note that the higher ε, the higher the error tolerance and the lower the number of updates of the current dictionary. On the contrary, a small ε entails frequent dictionary updates: this regulates the actual representation error and also determines the maximum achievable compression efficiency.

At the receiver, the n^{th} segment is reconstructed by picking the codeword with index $i_{\boldsymbol{x}}(n)$ from the local dictionary, performing renormalization of such codeword with respect to offset $e_{\boldsymbol{x}}(n)$ and gain $g_{\boldsymbol{x}}(n)$ and stretching the codeword according to the actual segment length $r_{\boldsymbol{x}}(n)$.

5. NUMERICAL RESULTS

In this section, we compare SAM against popular compression algorithms through the following performance metrics:
Compression Efficiency: is the ratio between the total number of bits required to transmit the full input time series by those required for the transmission of its compressed version (including segment lengths, offsets and gains).
Reconstruction accuracy, represented through the RMSE between the original and the compressed signals and expressed as a percentage of the average segment's peak-to-peak amplitude, $\texttt{p2p}$. Letting $\mathbf{s} = [s_1, s_2, \dots, s_r]^T$ be the vector containing the original signal samples, $\hat{\mathbf{s}} = [\hat{s}_1, \hat{s}_2, \dots, \hat{s}_r]^T$ the reconstructed samples at the decompressor, and r its length, the RMSE is obtained as:

$$\text{RMSE} = \frac{100}{\texttt{p2p}} \sqrt{\frac{\sum_{i=1}^{r} (s_i - \hat{s}_i)^2}{r}} \,.$$

Energy Consumption, evaluated by tracking the number of operations performed by the Micro-Controller Unit (MCU) to run the various algorithms. These, have been translated into the corresponding number of clock cycles and, in turn, into the energy consumption in Joule per bit for a Cortex M4 [12] processor. For the transmission energy, we account for the energy expenditure of a Texas Instruments CC2541 low-energy Bluetooth system-on-chip (SoC) [13]. The considered MCU and radio are widely adopted components for IoT devices. Here, we show the total normalized energy consumption (expressed in Joule per bit), which is obtained as the total energy (Joule) needed to compress and transmit the input times series divided by the number of bits in the original signal.

In all the experiments we set the TASOM parameters as $\sigma_j(0) = \sqrt{2}M$ and $\eta_j(0) = 0.99$, for $j = 1, 2, \dots, L$. In the following, we evaluate SAM on 15 ECG signals from the MIT-BIH arrhythmia database [14]. These are sampled at 360 samples/s and each sample has a resolution of 11 bits. Extracted segments are resized to length $m = 150$. We compare SAM against the following popular compression techniques: Compressive Sensing (CS) [9], Discrete Cosine Transform (DCT) [7], Discrete Wavelet Transform

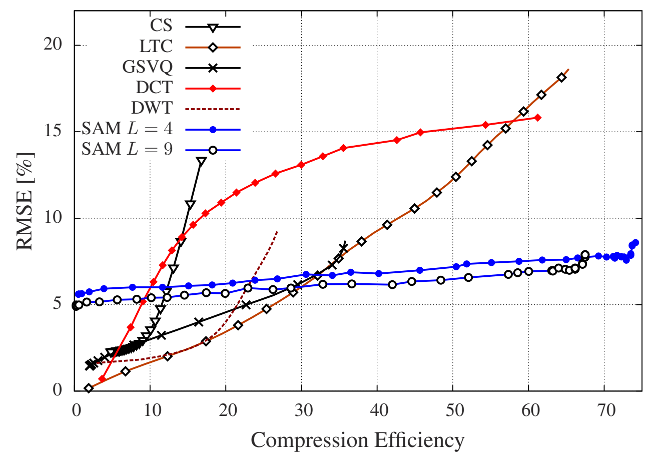

Fig. 2. RMSE *vs* compression efficiency for ECG signals.

(DWT) [8], Gain Shape Vector Quantization (GSVQ) [5] and Lightweight Temporal Compression (LTC) [6].

In Fig. 2, we show the RMSE against the compression efficiency, where both metrics were averaged over MIT-BIH arrhythmia ECG traces. SAM's tradeoff curves are obtained varying ε as a free parameter: small values of ε entail low compression efficiencies, whereas higher efficiencies are obtained for an increasing ε. We note that despite the simplicity of LTC, this is the algorithm of choice at small compression efficiencies, say, smaller than 30. SAM becomes appealing as the compression efficiency increases beyond 30, and presents a slowly increasing RMSE. We found that the compression performance of GSVQ is especially affected by the transmission of the stream of residuals and, for this reasons, GSVQ reaches maximum compression efficiencies of about 35. We underline that their transmission is no longer necessary with SAM, as its dictionary is continuously updated, as opposed to learning it in an offline fashion. This fact leads to a considerable leap in performance, allowing to reap the full benefits of dictionaries, achieving compression efficiencies as high as 70 with just $L = 4$ codewords. From these results, we advocate the use of SAM at high compression efficiencies where it achieves substantial energy savings, whereas LTC may be applied when the desired compression efficiency is small.

In Fig. 3, we look at the total energy consumption (transmission plus compression), expressed in Joule per bit in the original (uncompressed) time series. As a first observation, we see that compression makes it possible to save a substantial amount of energy with respect to sending the signals uncompressed (referred to in the plot as "no compression"). Again, the best algorithms are LTC and SAM: the corresponding curves cross when the RMSE is slightly smaller than 7% and the energy consumption is about $1.37 \cdot 10^{-8}$ Joule/bit. For SAM, with $L = 4$ the energy demand reduces to $5.9 \cdot 10^{-9}$ Joule/bit for the leftmost point in the figure, which corresponds to a further reduction of a factor of 2.3 from the point where the LTC and SAM curves

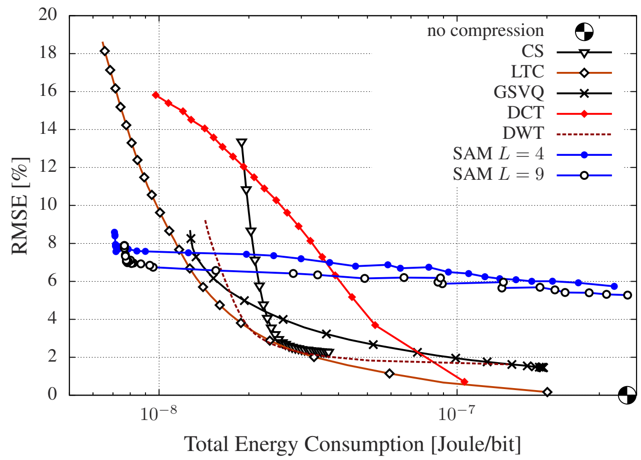

Fig. 3. ECG: normalized total energy consumption.

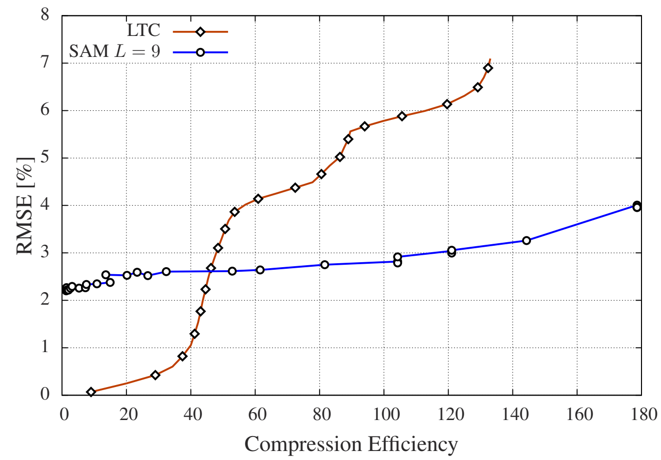

Fig. 5. RMSE *vs* compression efficiency for RESP signals.

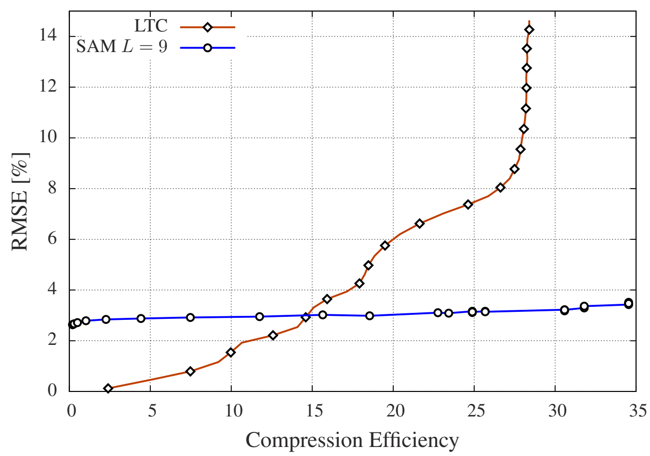

Fig. 4. RMSE *vs* compression efficiency for PPG signals.

intersect. We remark that, although the region where SAM surpasses LTC in terms of energy consumption seems moderate in extent, this region actually encompasses compression efficiencies higher than 30, see Fig. 2, which is the region where SAM is best utilized, providing the highest benefits against all the other approaches.

The compression performance of SAM when applied to PPG signals is presented in Fig. 4. These results have been obtained from PPG traces from the Physionet MIMIC II database [15]. Their sampling rate is 125 samples/s and the resolution is 12 bits per sample. In this plot, we only show SAM and LTC as these are the best performing algorithms, the performance of the remaining ones is similar to that in Fig. 2. We observe that for the considered PPG signals the maximum compression efficiency is much smaller than that achieved with ECG. This fact is due to the lower sampling rate of the PPG traces (125 Hz, as opposed to 360 Hz) which, in turn, implies a smaller number of samples per segment. For PPG, SAM provides a remarkable performance in terms of reconstruction accuracy, delivering an RMSE smaller than 4% across all compression efficiencies. Again, it might make sense to conceive an adaptive algorithm that uses LTC at low compression efficiencies (e.g., smaller than 15), switching to SAM when higher compressions are required.

Figs. 5 show the accuracy and compression performance of LTC and SAM when applied to RESP signals. As for PPG, RESP signals are also taken from the MIMIC II database and are sampled at 125 samples/s with each sample taking 12 bits. Remarkably, we observe that the maximum compression efficiency for RESP is much higher. Here, SAM can compress the input signal up to 180 times, by still providing excellent representation accuracies (RMSE of around 4%). The reason behind this is that RESP is a slow varying signal with respect to ECG and PPG. In fact, one respiration cycle on average contains about five ECG (or PPG) ones. This means that, with SAM more points are approximated through a single codeword within the same RESP cycle and this makes it possible to reach higher compressions (in particular, 5 times more points than with PPG, which is sampled at the same rate and whose segments are normalized to the same length $m = 80$). In this case, the additional energy reduction achievable through SAM, from the point where its energy consumption intersects that of LTC, amounts to a factor of 3.5, although the results are not shown in the interest of space.

In Figs. 6 and 7, we provide some visual insight into the reconstruction accuracy delivered by SAM and LTC for the same RMSE of $\approx 6.7\%$ plotting the original and the reconstructed ECG time series for three representative and full heartbeats. We observe that with SAM an RMSE of about 7% may be considered acceptable and still small and, from Fig. 6, that the TASOM dictionary has better smoothing capabilities than LTC. In fact, besides the usual P, Q, R, S and T peaks of an ECG, in the original ECG trace we also observe a multitude of smaller oscillations which are likely due to measurement artifacts. With SAM, these artifacts are smoothed out in its compressed representations. Another observation is

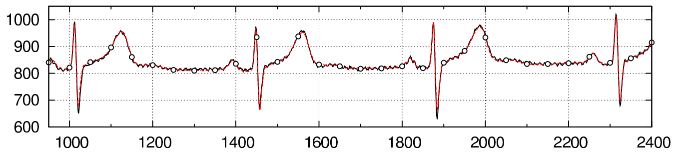

Fig. 6. **SAM:** original (solid line) and reconstructed ECG signals (dashed line with white filled circles). RMSE is 6.71%, compression efficiency is 61.

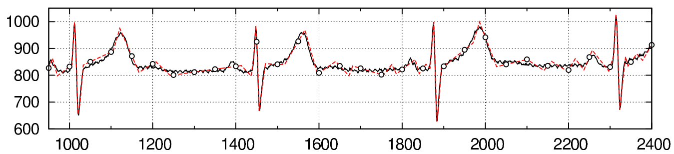

Fig. 7. **LTC:** original (solid line) and reconstructed ECG signals (dashed line with white filled circles). RMSE is 6.71%, compression efficiency is 31.

that the signal reconstructed through SAM looks much more alike the original one, whereas LTC approximates the original signal through line segments and this often provides a morphologically distorted result. SAM's compression efficiency is considerably higher (61 as opposed to 31 for LTC). Finally, we found SOM dictionaries to be excellent in the approximation of recurrent patterns, but for those segment containing rare patterns (anomalies) and still requiring high precision, better RMSE performance is obtained by switching to a different compression algorithm (e.g., DCT), as suggested in [5]. This aspect is left open for further research.

6. CONCLUSIONS

In this paper we have presented SAM, a subject-adaptive lossy compression scheme for wearable devices. SAM exploits a dictionary-based design philosophy, where the dictionary is learned and adapted at runtime to best represent the physiological signals of the subject that wears the device. This is attained using time-adaptive self-organizing maps, which are neural network architectures featuring continuous learning and adaptation capabilities. SAM outperforms other popular compression approaches from the literature, achieving a major reduction in the memory size required by continuous monitoring applications (up to 35-, 70- and 180-fold for PPG, ECG and RESP signals, respectively).

7. REFERENCES

[1] P.J. Soh, G.A.E. Vandenbosch, M. Mercuri, and D.M.M.-P. Schreurs, "Wearable Wireless Health Monitoring: Current Developments, Challenges, and Future Trends," *IEEE Microwave Magazine*, vol. 16, no. 4, pp. 55–70, May 2015.

[2] Teuvo Kohonen, "The Self-Organizing Map," *Proceedings of the IEEE*, vol. 78, no. 9, pp. 1464–1480, Sept. 1990.

[3] Simon Haykin, *Neural Networks: A Comprehensive Foundation*, Prentice Hall, Upper Saddle River, NJ, US, 2nd edition, 1998.

[4] H. Shah-Hosseini and R. Safabakhsh, "TASOM: the time adaptive self-organizing map," in *Proceedings of the International Conference on Information Technology: Coding and Computing*, Las Vegas, NV, US, Mar. 2000, pp. 422–427.

[5] Chia Chun Sun and Shen Chuan Tai, "Beat-based ECG compression using gain-shape vector quantization," *IEEE Transactions on Biomedical Engineering*, vol. 52, no. 11, pp. 1882–1888, Nov. 2005.

[6] T. Schoellhammer, B. Greenstein, M. Wimbrow E. Osterweil, and D. Estrin, "Lightweight temporal compression of microclimate datasets," in *Proceedings of the IEEE International Conference on Local Computer Networks (LCN)*, Tampa, FL, US, Nov. 2004.

[7] V.A. Allen and J. Belina, "ECG data compression using the discrete cosine transform (DCT)," in *Proceedings of Computers in Cardiology*, Durham, NC, US, Oct. 1992, pp. 687–690.

[8] Bashar A. Rajoub, "An Efficient Coding Algorithm for the Compression of ECG Signals Using the Wavelet Transform," *IEEE Transactions on Biomedical Engineering*, vol. 49, no. 4, pp. 355–362, Apr. 2002.

[9] L.F. Polania, R.E. Carrillo, M. Blanco-Velasco, and K.E. Barner, "Compressed sensing based method for ECG compression," in *Proceedings of the IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, Prague, Czech Republic, May 2011, pp. 761–764.

[10] Y. Linde, A. Buzo, and R.M. Gray, "An Algorithm for Vector Quantizer Design," *IEEE Transactions on Communications*, vol. 28, no. 1, pp. 84–95, Jan. 1980.

[11] Mohamed Elgendi, "Fast QRS Detection with an Optimized Knowledge-Based Method: Evaluation on 11 Standard ECG Databases," *PLoS ONE*, vol. 8, no. 9, pp. 1–18, Sep. 2013.

[12] ARM The Architecture for the Digital World, "ARM Cortex-M4 Processor," 2015.

[13] Texas Instruments, "CC2451: 2.4 GHz Low Energy and Proprietary System-on-Chip," 2015.

[14] George B. Moody and Roger G. Mark, "The Impact of the MIT-BIH Arrhythmia Database," *IEEE Magazine of Engineering in Medicine and Biology*, vol. 20, no. 3, pp. 45–50, 2001.

[15] M. Saeed et al., "Multiparameter Intelligent Monitoring in Intensive Care II (MIMIC-II): a public-access intensive care unit database," *Critical Care Medicine*, vol. 39, no. 5, pp. 952–960, May 2011.