PROS: an Efficient Pattern-Driven Compressive Sensing Framework for Low-Power Biopotential-based Wearables with On-chip Intelligence

Nhat Pham[1], Hong Jia[2], Minh Tran[1], Tuan Dinh[3], Nam Bui[4], Young Kwon[2], Dong Ma[5], Phuc Nguyen[6], Cecilia Mascolo[2], and Tam Vu[4]
[1]University of Oxford, [2] University of Cambridge, [3]University of Wisconsin Madison,
[4]University of Colorado Boulder, [5]Singapore Management University, [6]University of Texas at Arlington
nhat.pham@cs.ox.ac.uk,{hj359,ydk21,cm542}@cam.ac.uk,minh.tran@ndcn.ox.ac.uk,
tuan.dinh@wisc.edu,{nam.bui,tam.vu}@colorado.edu,dongma@smu.edu.sg,vp.nguyen@uta.edu

ABSTRACT

While the global healthcare market of wearable devices has been growing significantly in recent years and is predicted to reach $60 billion by 2028, many important healthcare applications such as seizure monitoring, drowsiness detection, etc. have not been deployed due to the limited battery lifetime, slow response rate, and inadequate biosignal quality.

This study proposes PROS, an efficient pattern-driven compressive sensing framework for low-power biopotential-based wearables. PROS eliminates the conventional trade-off between signal quality, response time, and power consumption by introducing tiny pattern recognition primitives and a pattern-driven compressive sensing technique that exploits the sparsity of biosignals. Specifically, we (i) develop tiny machine learning models to eliminate irrelevant biosignal patterns, (ii) efficiently perform compressive sampling of relevant biosignals with appropriate sparse wavelet domains, and (iii) optimize hardware and OS operations to push processing efficiency. PROS also provides an abstraction layer, so the application only needs to care about detected relevant biosignal patterns without knowing the optimizations underneath.

We have implemented and evaluated PROS on two open biosignal datasets with 120 subjects and six biosignal patterns. The experimental results on unknown subjects of a practical use case such as epileptic seizure monitoring are very encouraging. PROS can reduce the streaming data rate by 24X while maintaining high fidelity signal. It boosts the power efficiency of the wearable device by more than 1200% and enables the ability to react to critical events immediately on the device. The memory and runtime overheads of PROS are minimal, with a few KBs and 10s of milliseconds for each biosignal pattern, respectively. PROS is currently adopted in research projects in multiple universities and hospitals.

CCS CONCEPTS

• **Computer systems organization → Embedded systems**; • **Human-centered computing → Mobile devices**.

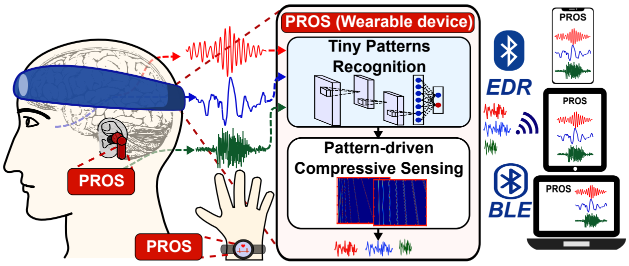

Figure 1: PROS Overview

KEYWORDS

Biosignal, Compressive Sensing, On-chip Intelligence, Edge-AI, Wearable devices, Cyber-Physical systems.

ACM Reference Format:
Nhat Pham, Hong Jia, Minh Tran, Tuan Dinh, Nam Bui, Young Kwon, Dong Ma, Phuc Nguyen, Cecilia Mascolo, and Tam Vu. 2022. PROS: an Efficient Pattern-Driven Compressive Sensing Framework for Low-Power Biopotential-based Wearables with On-chip Intelligence. In *The 28th Annual International Conference On Mobile Computing And Networking (ACM MobiCom '22), October 17–21, 2022, Sydney, NSW, Australia.* ACM, NewYork, NY, USA, 15 pages. https://doi.org/10.1145/3495243.3560533

1 INTRODUCTION

The wearable healthcare market has been experiencing significant growth in recent years, reaching $100 million in 2020 and over $60 billion globally by 2028 [1–3]. It is predicted that healthcare wearable devices will be the next generation of personal telemedicine practice. This is especially important for patients with chronic diseases and after surgery, where constant monitoring is essential to prevent fatalities [4]. However, many wearable-enabled healthcare applications have not been deployed due to limited battery lifetime, slow response rate, and inadequate biosignal quality.

Human biosignals are the key to enabling many healthcare applications. For example, by using facial muscle signals (i.e., electromyography (EMG)), one can monitor the stress level [5, 6] and the eating habit of a user [7–9]. When combining with the brain (i.e., electroencephalogram (EEG)) and eye (i.e., electrooculography (EOG)) signals, one can further supervise the user's emotional states [10, 11], their pain and suffering level [12], or detect emergency events such as epileptic seizures [13], microsleep [14], etc. These healthcare applications often require long-term monitoring of high-fidelity biosignals and the ability to react to emergency events to prevent tragedies quickly.

ACM MobiCom '22, October 17–21, 2022, Sydney, NSW, Australia
© 2022 Association for Computing Machinery.
ACM ISBN 978-1-4503-9181-8/22/10…$15.00
https://doi.org/10.1145/3495243.3560533

The trade-off between signal fidelity, response time, and battery life is a long-standing challenge for wearable devices [15, 16]. In many healthcare applications, the wearable usually takes the role of data collecting device due to their limited energy and computing resources [17]. The collected data are transmitted to nearby mobile devices through wireless communications (e.g., Bluetooth, WiFi) to predict emergency events or upload to users' healthcare providers for further diagnosis. Though maintaining the collected signal fidelity is crucial [18], continuous wireless communication has a high cost on the battery life [19]. E.g., Bluetooth could consume up to several mWs [20], while WiFi could go as high as 10s of mW [21], depending on the data rate. As a result, many healthcare wearables have to reduce signal quality (i.e., by lowering data rate) and increase response latency (i.e., by increasing communication intervals) to improve battery lifetime [19].

In this project, we explore the challenges of building a new event-driven compressive sensing framework, called PROS, that could enable highly energy-efficient wearables for biopotential-based applications. We develop PROS based on the sparsity nature of biosignals and events. Specifically, PROS consists of tiny pattern recognition primitives and a pattern-driven compressive sensing algorithm that work together to significantly reduce transmission rate while maintaining high fidelity signal (Fig. 1). PROS also enables the ability to react to critical events immediately on the device.

Challenges: To realize PROS, we face the following challenges: (1) biosignal events (e.g., seizures, microsleep, pain, etc.) require multimodal sensing channels and a complex algorithm (e.g., machine learning) to detect, which is not feasible on low computing resource wearable; (2) we lack a reliable domain with high sparsity to compress biosignals on the device effectively; (3) low power wearable devices have extremely constrained computing resource, i.e., an MHz microcontroller (MCU) and KBs of system memory, making it challenging to deploy advanced computations without consuming significant energy.

Contributions: To overcome the aforementioned challenges, we make the following contributions:

(1) We identify the pattern primitives of biosignals such as EEG, EOG, and EMG and develop tiny recognition models (TinyPR) for continuous on-chip detection and low-latency responses.
(2) We devise a pattern-driven compressive sensing (PDCS) technique to efficiently compress the captured signal pattern with appropriate wavelet domains, boosting the compression factor and recovered signal quality.
(3) We design a hardware platform and employ optimization techniques in both hardware and OS levels to support advanced signal processing and neural network operations of PROS.
(4) The prototype of PROS is evaluated on two open datasets of 120 subjects. In a practical use case such as epileptic seizure detection, PROS can reduce the data rate by 24X, boost the power efficiency by more than 1200%, and enable real-time responses within 10s of milliseconds while maintaining high fidelity signals.

Potential Applications and Impact: While we currently focus on EEG, EOG, and EMG biosignals and a head-worn form factor in this study, PROS is also applicable for a variety of healthcare wearable devices such as smartwatches, earphones, smart clothes, etc., where achieving high-fidelity biosignal streams, low-latency responses, and long battery life is critical to their applications. To encourage adoption and reproducibility, PROS is available as an open-source project [22] under the LGPLv2 license.

2 OBSERVATIONS

As biosignal events are often intermittent, monitoring them continuously results in wasted energy, computing power, and memory. In this project, we consider events that are associated with EEG, EOG, and EMG, but the proposed solution would be generally applicable to other biosignals in multiple application domains.

Event Sparsity. We observe that the events of interest (e.g., seizures, microsleeps, etc.) are important but rarely happen. Several studies have reported that these events only occur less than 5% of the signal duration [23]. Thus, detecting these events on the device could help to cut a significant amount of energy needed to stream the signals out. However, detecting these events requires multiple signal modalities (i.e., EEG, EOG, EMG, etc.) and a complex algorithm, making it challenging to implement on resource-constrained devices. Our intuition is that we could decompose these complex events into smaller and generic patterns of interest (PoIs). For example, an epileptic seizure waveform could consist of EEG spike/polyspike and slow-wave (focal/generalized non-specific seizures), 3-Hz spike-and-wave discharges (absence seizures), and stiffing and convulsion patterns (tonic-clonic seizures). Similarly, we can decompose a microsleep event into alpha, theta wave, slow eye movements, and muscle contractions patterns on the EEG, EOG, and EMG signals. Thus, it is feasible to detect these patterns directly on the device with an efficient pattern recognition technique.

Signal Sparsity. We also observe that the sparsity property also presents at the signal level. While biosignals are known to be non-sparse in time or frequency domains, they could have sparse representations in other domains (e.g., wavelets). Thus, we do not need all the collected samples to reconstruct the signal. The compressive sensing (CS) theory has been developed to exploit the signal sparsity. It states that the number of signal measurements depends on inherent information contained in the signal and is much lower than the Nyquist rate [19]. The effectiveness of CS relies directly on finding a reliable domain with high sparsity. However, this is still an open challenge for non-stationary biosignals [24].

From these observations, we hypothesize that **by exploiting both event and signal sparsity, the amount of data reduction could be significant**, leading to a highly energy-efficient system. However, we must take great care in designing such a system. With the constrained computing resources of wearable devices, any additional energy spent on complicated algorithms could easily outweigh any benefits from the reduced wireless transmission.

The remaining questions are *(1) How can we develop the pattern detection models so that they can be both accurate and efficient (Sec. 4)? (2) How can we devise a compressive sensing method that could achieve low sampling rate while maintaining high signal fidelity (Sec. 5)?* and *(3) How can we optimize the system to ensure the efficiency of additional computation (Sec. 6)?*

3 PROS SYSTEM OVERVIEW

We design PROS with three objectives, (1) detect signal patterns of interest (PoIs) directly on-chip to eliminate most of the irrelevant signal, (2) compress the detected PoI by using the recognition information to reduce wireless transmission rate further, and (3)

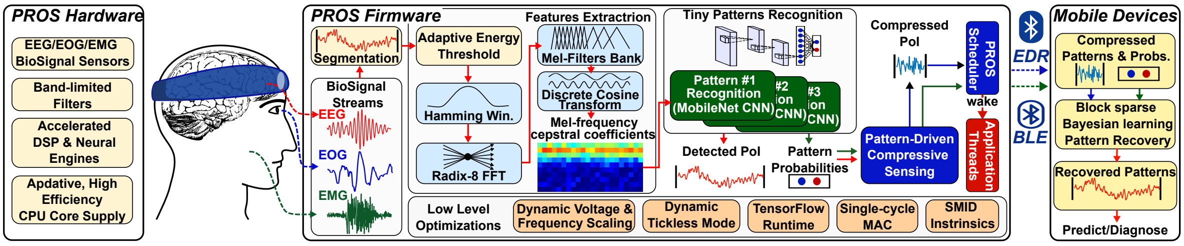
PROS Hardware
EEG/EOG/EMG BioSignal Sensors
Band-limited Filters
Accelerated DSP & Neural Engines
Apdative, High Efficiency CPU Core Supply
PROS Firmware
Segmentation
BioSignal Streams
EEG
EOG
EMG
Features Extractrion
Adaptive Energy Threshold
Hamming Win.
Radix-8 FFT
Mel-Filters Bank
Discrete Cosine Transform
Mel-frequency cepstral coefficients
Tiny Patterns Recognition
Pattern #1 Recognition (MobileNet CNN)
#2
ion
CNN)
#3
ion
CNN)
Detected PoI
Pattern Probabilities
Compressed PoI
Pattern-Driven Compressive Sensing
PROS Scheduler
wake
Application Threads
Low Level Optimizations
Dynamic Voltage & Frequency Scaling
Dynamic Tickless Mode
TensorFlow Runtime
Single-cycle MAC
SMID Instrinsics
EDR
BLE
Mobile Devices
Compressed Patterns & Probs.
Block sparse Bayesian learning Pattern Recovery
Recovered Patterns
Predict/Diagnose

Figure 2: PROS system overview.

optimize hardware and OS operations to enhance system's efficiency. To achieve these goals, we develop three main components for PROS as illustrated in Fig. 2, (a) a firmware framework that detects and compress the PoIs by using our developed tiny embedded machine learning models and a pattern-driven compressive sensing algorithm, (b) a low-power hardware platform that accelerates advanced signal processing and embedded machine learning operations, and (c) a mobile app to recover the compressed PoIs for further processing.

Efficient Features Extraction Pipeline. We design a feature extraction pipeline based on the characteristics of biosignals to compute Mel-frequency cepstral coefficients (MFCCs) from the signals. As the computing resource is highly constrained, we tune the MFCC processing so that the output contains enough vital information of each pattern while being small and lightweight. Additionally, we employ accelerated signal processing methods available on the hardware to boost the processing speed.

Tiny Pattern Recognition Primitives. We develop tiny pattern recognition (TinyPR) primitives to effectively detect patterns of interest (PoIs) from the captured biosignal streams. Each primitive only detects one PoI to ensure its efficiency and flexibility in different applications. Each primitive only needs a few KBs memory and milliseconds of CPU time to operate. We use TensorFlow Lite Microcontroller (TFLM) runtime and vectorized neural operations to push the inference performance.

Pattern-Driven Compressive Sensing. To further reduce the amount of communication, we devise a novel compressive sensing technique to exploit the sparsity property of each PoI. We adaptively apply the optimal compression ratio and wavelet domain transformation based on the pattern recognition information. We use a random binary compression technique to compress the signal with minimal system overhead on the wearable device. To reconstruct the signal, we employ a state-of-the-art Block Sparse Bayesian Learning recovery algorithm combined with pattern recognition information to reduce the required compressed samples by taking advantage of the sparsity of biosignal. As a result, we could achieve a high compression factor and reconstruct the signal with high fidelity on the mobile application.

Hardware and OS Optimizations. To further enhance the processing efficiency, we implement hardware and OS optimizations such as (i) dynamic voltage and frequency scaling, (ii) dynamic tickless mode, and (iii) adaptive energy thresholding. We also develop a scheduler that provides configurations and wakes the application's threads when subscribed PoIs are detected.

PROS Hardware and Mobile Application. We design a low-power hardware platform from the ground up to support PROS. To enable advanced optimizations on the firmware, we equip it with a energy-efficient signal processing and neural network processor and an adaptive, high-efficiency CPU core supply. We also developed a lightweight signal reconstruction algorithm on mobile devices to reconstruct the compressed PoI with high fidelity. The reconstructed PoI could be used for further processing or diagnosis.

4 TINY PATTERN RECOGNITION PRIMITIVES

This section presents our end-to-end pipeline, called TinyPR, for recognizing biosignal patterns. The key contributions of TinyPR are (1) identifying the generic biosignal pattern primitives that are feasible to be efficiently recognized on the low-power hardware and (2) providing a design strategy that can be both accurate and lightweight for those pattern primitives. The developed pattern recognition models can be served as building blocks for biopotential-based applications requiring on-chip pattern recognition. We first highlight key challenges and insights into the design of our framework.

4.1 Key challenges and designs

As per our system requirements, the target recognition model should be highly expressive to detect the biosignal patterns but also resource-efficient for the MCUs' deployment. This expressive-efficiency trade-off poses a critical challenge for our system design.

Detecting biosignal patterns has remained challenging, despite some positive outcomes in preliminary works [25, 26]. Biosignals are highly irregular and heterogeneous [27] due to the complexity and intrinsic properties of biosystems, causing the difficulty for understanding and detecting the interest patterns [25, 28]. For instance, recent works [26, 28] find that most existing approaches are ineffective for learning patterns for clinical analysis and event detection. Besides, the scarcity of interest patterns [29, 30] in biosignals makes the learning even harder: the training data is heavily imbalanced. The resource restriction of MCUs adds another challenge to our design. With limited computing resources in terms of memory, operations, and computation capacity, MCUs require the inference system to have low memory footprints (e.g., a few KBs) and low inference latency.

Existing methods to biosignal learning are mainly based on either the deep learning approach or feature-based machine learning approach [26, 31]. While achieving high recognition performance and being easier to implement on hardware, deep learning models are usually too large for MCUs. On the other hand, simple feature-based learning models are more resource-efficient but not sufficiently and robustly effective at detecting complex patterns [26]. In this work, we propose the combination of the feature-based approach with deep learning: utilizing an informative feature extractor to reduce the burden in learning domain knowledge features. Moreover, we can significantly reduce models' sizes by leveraging quantization techniques without degrading the recognition performance [32].

Subjects variation is also a challenge for biopotential-based systems. While biopotential signals vary among people, our intuition is that they contain similar patterns due to the typical structure of the human body. For example, eye blink signals usually have two opposite consecutive peaks corresponding to the closing and opening phases of the eyelids; alpha brainwaves typically have cyclical or rhythmic changes with a frequency from 8 to 12Hz when the brain neurons become synchronized in a relaxed state. Therefore, our intuition is that if we train the TinyPR models to target common and generic signal patterns of interest, these models could generalize well to unseen subjects. We present the detection performance evaluations of our developed TinyPR models in detail in Sec. 8.

4.2 Pattern Recognition as the Rare Event Detection Problem

Most target patterns rarely occur in biosignals. For instance, seizure events usually account for only 1% in EEG recording data [29]. This results in the highly skewed distribution of training data. Standard methods for event detection and feature selection may not work well with the imbalanced data [33] because they tend to learn features only from the major classes (background signals) and may easily misclassify the minor classes (target patterns).

Therefore, we cast our pattern identification problem as the rare event detection problem [34]. Solving this problem requires adopting either supervised or unsupervised techniques for rare-event detection [33]. The latter requires large models with an enormous amount of unlabeled data, which are not feasible for deploying MCUs. Hence, we focus on the supervision approach to design a more lightweight classification model. In particular, considering target patterns as positive and the rest patterns as negative, the problem becomes a binary classification task. We note that data distribution is highly skewed as positive data is much smaller than negative data. To deal with this issue, we apply SMOTE [35] method to upsample the positive patterns. The next section will present the design of our feature extractor and binary classification model.

4.3 Informative Feature Extraction

Powerful prior knowledge via informative feature extraction can significantly reduce the complexity of recognition models. Mel Frequency Cepstral Coefficients (MFCC), together with Wavelets transform, are the two most common approaches used for extracting biosignal features [36]. Since the computing resource and energy on low-power microcontrollers (MCU) are highly constrained, we only pick the features that are informative while being resource-efficient. MFCC features fit well with these criteria as multiple previous works [36, 37] have proved that MFCC features are reliable in detecting biosignal (EEG/EOG/EMG) events. Furthermore, there are available components in the optimized firmware library, such as ARM-CMSIS, for an efficient implementation. An efficient implementation is critical for low-power MCUs since heavy processing can easily outweigh any benefits of data reduction.

As MFCC is initially used for audio signals, we configure its components to extract useful features from biosignal data. We note that most of the information in biosignals (EEG, EOG, EMG) locate at the low-frequency bands (< 300Hz) [38, 39]. We, therefore, use only ten bands among 39 features of MFCC to extract essential features, further helping reduce the input size of the recognition model. We use Hamming window to slice the signals into slicing frames. Note that sudden chop-off at the frame's edge can lead to a noisy signal because of the sudden amplitude drop. Hence, we gradually drop amplitude near the edge of frames. We apply Discrete Cosine Transform to extract features in the frequency domain and triangular Mel-scale filter banks to transform the signal to Mel-scale power spectrum. Given these features, we can now build an efficient classifier.

Though it is possible to extract meaningful features with autoencoders automatically, it is not efficient on low-power microcontrollers. It has been pointed out in [40] that directly extracting features would be much more energy and computational efficient by taking advantage of the accelerated library of the targeted hardware. Thus, we design our TinyPR models around optimized signal processing and neural operations provided by TinyML frameworks such as TensorFlow Lite Microcontroller [40] and CMSIS-NN [41].

It is also important to note that while MFCC could extract temporal and spectral features well, these features might not be sufficient for all applications. Thus, we envision that PROS serves as an open framework where multiple processing pipelines and pattern recognition models could be developed for various applications.

4.4 Efficient Design for Recognition Model

We build a deep classification model on top of the extracted MFCC features to complete the recognition framework. The resource constraints pose two questions for our design: how to design the best-fit model given particular conditions on memory and power and how to efficiently run the model on MCUs. We wish to achieve these objectives without degrading the recognition performance.

Efficient Architecture. Recent works of TinyML [42], or machine learning for edge devices, provide potential solutions to our problem. TinyML aims to shrink sizeable deep learning models (millions to billions of parameters) into tiny models of a few KBs, mainly by changing the network topology to remove the redundant parameters [43, 44], reducing the input size, or loading only parts of the network to the memory to address the memory bottleneck [45, 46]. However, existing models are not directly applicable for our PROS system because the shrunk models' sizes are still relatively larger than our desiderata, and the designs are primarily specific for image signals instead of biosignals. Therefore, we derive a simple yet powerful architecture for our system based on the recent advances of TinyML [42].

The critical component of our architecture is the block of depthwise convolution (DW-Conv) and pointwise convolution (PW-Conv) [43], which has been proven helpful in multiple resource-aware models, such as MobileNets [43, 44] and MicroNets [47]. DW-Conv is a type of spatial convolution that applies independently on each channel of inputs. PW-Conv uses a 1 × 1 kernel to iterate every point, further linearly combining DW-Conv outputs. Compared to the standard convolution, DW-Conv and PW-Conv require much smaller numbers of parameters, thus being more computationally effective [48]. Also, these operations are supported by the micro deep learning framework TFLMicro [49].

Our architecture consists of a convolutional layer as the input layer, followed by a sequence of DW-PW-Conv blocks, a Dropout

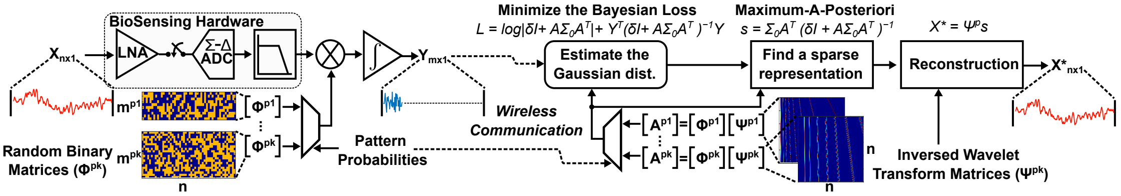
BioSensing Hardware
X_{nx1}
LNA
Σ-Δ ADC
Y_{mx1}
Minimize the Bayesian Loss
$L = log|\delta I + A\Sigma_0 A^T| + Y^T(\delta I + A\Sigma_0 A^T)^{-1}Y$
Maximum-A-Posteriori
$s = \Sigma_0 A^T (\delta I + A\Sigma_0 A^T)^{-1}$
$X^* = \Psi^p s$
Estimate the Gaussian dist.
Find a sparse representation
Reconstruction
X^*_{nx1}
m^{p1}
m^{pk}
$[\Phi^{p1}]$
$[\Phi^{pk}]$
Random Binary Matrices (Φ^{pk})
n
Pattern Probabilities
Wireless Communication
$[A^{p1}]=[\Phi^{p1}][\Psi^{p1}]$
$[A^{pk}]=[\Phi^{pk}][\Psi^{pk}]$
n
n
Inversed Wavelet Transform Matrices (Ψ^{pk})

Figure 3: Pattern-driven Compressing Sensing framework.

layer, and a linear layer. Each DW-PW-Conv block is a stack of a DW-Conv and a PW-Conv with batch normalization and a Relu activation. Under different systems, we control models' sizes by varying the number of DW-PW-Conv blocks and channels' sizes to fit the MCUs' requirements. In particular, we apply the search approach in MobileNetV2 [44] to search for the architecture's configurations achieving the best trade-off of efficiency and recognition accuracy. For the deployment on MCUs, we use the TensorFlow Lite Microcontroller framework [50] to compress the model into the numeric domain, reducing the memory footprint and speeding up the computation.

Post-training Dynamic Range Quantization. To further reduce the model's size for inference, we apply the dynamic range quantization technique [51]. While the floating-point format is used for parameters of most deep learning models to achieve better precision during training, it may be costly to store floating-point numbers, especially on low-memory edge devices. Quantization techniques help solve this issue by converting trained parameters into another number representation. For instance, converting the commonly used float32 format to the int8 format helps save 24 bits. Weights are converted back into float32 format during the inference for better classification performance. We find that the recognition performances in metric-wise are nearly identical to the original ones after this transformation.

Memory complexity. Our final models have only a few KBs in size, highly optimized compared to MobileNetV1 and MobileNetV2 (16.9 MBs and 4 MBs, respectively). We attribute this tremendous compression mainly to the use of an informative MFCC feature set: with small size (10 features (Section 4.3)) and with low dimension (22 dimensions). The input size of classification models reduces from $224 \times 224 \times 3$ (for images in MobileNets) down to 10×22, leading to small numbers of convolution channels and layers required to learn the feature representation. As a result, our smallest models have only nearly 3.5K parameters in total.

Inference with Confidence. Together with producing accurate predictions, an essential requirement for recognition models in practice is to provide the confidence of the prediction. Inspired by the clinical procedure in diagnostics, we impose the confidence level to the pattern recognition result. Together with each classification's output (binary value), our model produces a confidence score representing the certainty of the prediction. This score is generated by thresholding the soft-max scores of the binary classes. The application can choose the threshold to make a trade-off between sensitivity or specificity depending on its requirements.

At this stage, we could eliminate most of the irrelevant signals. However, as we still need to transmit the captured PoI signals, we need to compress the data to reduce the transmission rate further.

5 PATTERN-DRIVEN COMPRESSIVE SENSING

This section discusses the challenges and our proposed Pattern-driven Compressing Sensing (PDCS) technique to reduce the amount of wireless communication in our system. While downsampling is a popular technique to reduce data rate, it has been shown that it could significantly degrade the quality of biosignal analysis and induce higher noise and aliasing [52]. In this study, we employ the compressive sensing (CS) theory as it could avoid signal degradation while requiring minimal system processing and memory overheads, both of which are critical for low-power biopotential-based wearables [53]. It bases on a fundamental assumption that biosignals have sparse representations in a transformed domain such as frequency or time-frequency (e.g., wavelets) [54, 55]. Thus, sampling the signal based on the fastest frequency component based on Nyquist–Shannon theory is redundant [56].

The key contribution of PDCS is the ability to incorporate pattern recognition information to build an efficient data-driven compressive sensing method. Conventionally, the compressive sensing techniques are deployed on low-power devices due to the simplicity of the compression. The performance, however, depends heavily on the choice of the compression ratio and sparse domain basis. Since pattern recognition information was unavailable in previous works [19, 57] due to energy and computational resource constraints, the compression ratio and sparse domain basis are often chosen and tuned offline based on pre-collected data and apply to the whole signal during runtime. It leads to significant variations and inconsistent performance with non-stationary biosignals such as EEG [24]. By enabling energy-efficient on-chip pattern recognition, we can recognize and apply different compression ratios and sparse domain basis for each signal pattern in real-time.

It is also important to note that the merit of PDCS is complementary to TinyPRs. For example, assuming TinyPRs could reduce the transmission rate by M times by eliminating the irrelevant signal and PDCS compress the detected signal by N times on average, we will have the total compression ratio of $M \times N$. Furthermore, the theoretical computational (and energy) cost of PDCS is much lower than running a TinyPR model on the wearable device, i.e., only one matrix-vector multiplication versus a convolutional neural network inference, making the return on investment of PDCS significant.

5.1 PDCS framework design

We design our PDCS framework as illustrated in Fig. 3. PDCS is a digital CS design where we perform compression after digitalization. This design has the advantage that we could use precision, high-rate ADC (e.g., $\Sigma - \Delta$ modulated ADCs [58]) to avoid high-frequency noise and aliasing. PDCS has four important steps as follows.

First, we identify the domain and the transformation basis $\Psi_{n,n}$ where the input signal $X_{n,1}$ has a sparse representation $s_{n,1}$, i.e.,

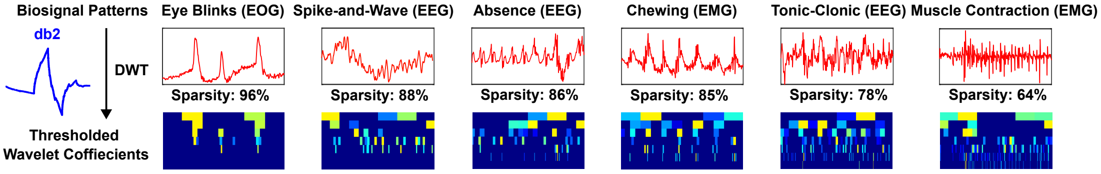

Figure 4: Sparsity variations among patterns in a wavelet (db2) domain.

$X = \Psi s$. As sampling with CS is based on the inherent information contained in the signal rather than its frequency bandwidth, the higher sparsity of the representation s, the less information is presented in the signal. Hence, a lower number of measurements is needed. The sparsity and density are defined as the percentage of zero and non-zeroes values in s, respectively. $Sparsity = 1 - Density$.

Second, we choose an i.i.d random distribution to construct our measurement matrices Φ. We construct multiple Φs for various compression factors (CFs). To ensure the compressed signal can be successfully recoverable, the coherence (μ) between Φ and Ψ, i.e., $\mu = \sqrt{n} * \max_{1 \le i,j \le n} |\langle \Phi_i, \Psi_j \rangle|$, is employed [55]. Lower μ ($1 < \mu < \sqrt{n}$), means more efficient compression. Random measurement matrices such as Gaussian, Bernoulli, Binary, etc. have low μ with any basis [59, 60]. Thus, they are employed as the universal encoders.

Third, on the wearable device, we compress the captured biosignals ($X_{m,1}$) based on pattern recognition information from TinyPR primitives (Sec. 4) and desired CFs ($CF = n/m$) for each pattern (p), i.e., $Y_{m,1} = \Phi^p_{m,n} X_{n,1}$, with m and n are the sizes of the compressed (Y) and the original signal (X), respectively. m should be much smaller than n for the compression to be effective. We transmit Y together with its recognition information to help with the recovery.

Fourth, at the receiver side (e.g., tablets, phones), we find a sparse representation $s_{n,1}$ by minimizing a Bayesian loss function. Using the received pattern recognition information, we dynamically apply different basis functions ($\Psi^p s$) to the Block Sparse Bayesian Learning (BSBL) algorithm to get the optimal results. The original signal is recovered by $\hat{X} = \Psi^p s$.

We tune CF based on the acceptance loss of the recovered signal. The configurations are evaluated on sample datasets to ensure satisfying accuracy. We measure the loss of the CS method by using the Structural SIMilarity index (SSIM) [61]. We employ SSIM in this study since it has better performance on structured signals [61]. The higher SSIM is better. SSIM = 1 means perfect recovery.

5.2 Sparsity variations among patterns

Finding the optimal domain where biosignals have sparse representations is the most crucial task and the most non-trivial one. Previous works on compressive sensing with biosignals show the feasibility of biosignals such as EEG, EOG, EMG to have sparse representations in time-frequency domains such as Gabor, Spline, and Wavelets domains [24, 62–64]. However, as they do not take into account individual signal pattern structure, many studies have reported large variations to the reconstruction accuracy among different channels and trials [24, 65].

Fig. 4 confirms the significant sparsity variations among different biosignal patterns in the same Daubechies 2 (*db*2) wavelet domain. Six biosignal patterns are extracted from an open biosignal dataset [66]. They include (1) eye blink (EOG), (2) spike-and-wave (EEG), (3) absence seizure (EEG), (4) chewing (EMG), (5) tonic-clonic seizure (EEG), and (6) muscle contraction (EMG). We apply the same discrete wavelet decomposition with seven levels. By keeping the recovery similarity index, i.e., SSIM, to be at least 0.9 between the original and recovered, we can find the minimum number of wavelet coefficients that are needed to reconstruct the original signal.

The *db*2 mother wavelet has a high structural correlation with eye blinks patterns. Thus, fewer wavelet coefficients are needed to reconstruct the original signal with only 4% density. On the other hand, the db2 wavelet works poorly with chewing, tonic-clonic seizure, and muscle contraction patterns. Their density is 36, 22, and 15%, respectively. Up to 9X can be observed in the density difference among these biosignals; hence, finding the optimal wavelet domain is a significant challenge that we need to address.

5.3 Optimal wavelet domains search

In this study, we assume that a universal wavelet domain for all the biosignal or even each signal group such as EEG, EOG, or EMG might not exist. However, there exists **an optimal wavelet domain for individual biosignal pattern**. Thus, by knowing the pattern of the interested signal, we could choose the appropriate sparse wavelet domain for each pattern to get the best compression factor. This is not possible in conventional CS systems [24, 57, 63, 67] where we lack the pattern recognition ability from biosignal streams. Hence, we have to trade-off between signal fidelity (i.e., by using the smallest CF) or compression factor (i.e., by accepting the loss with low sparsity patterns). Sec. 4 discuss how we overcome this challenge by capturing pattern information directly on the low-power hardware. The next step is to find the optimal wavelet domain for each biosignal pattern of interest.

There are several quantitative metrics in literature to choose the optimal wavelet domain such as maximum cross correlation [68], mean squared error [69], continuous wavelet coefficients [70], minimum description length [71], etc., that are used for biosignals such as EEG, EOG, EMG, or ECG. They are based on the intuition that the optimal wavelet domain will have the highest similarity between its transformation basis and the input signal [72]. They, however, could not tell us the sparsity of a signal pattern, making it difficult to estimate the compression factors. Furthermore, some studies also point out that similarity-based methods might not always result in optimal wavelet domains [72]. To alleviate this issue, we propose another selection metric called Maximum Sparsity Index (MSI). We define MSI as the maximum percentage of discrete wavelet coefficients that are not significant to reconstruct the signal.

Listing 1 presents our search algorithm. Since there could be an infinite number of wavelet domains [73], we only pick out 70 mother wavelet functions in six families such as Daubechies (db1-15), Coiflet (coif1-5), Fejér-Korovkin (fk4-fk22), Symlet (sym2-15), Biorthogonal Spline (bior1.1-6.8), Reverse B-Spline (rbior1.1-6.8), that are commonly used for biosignals [74–76]. For each mother wavelet function, we apply Discrete Wavelet Transform (DWT) to the input signal (X) with five decomposition levels to get its wavelet coefficients ($coeffs$). The number of decomposition levels is chosen to extract all the frequency information inside the input biosignals [77]. As $coeffs$ is near sparse (i.e., the coefficients that are significantly larger than zero are sparse), we iteratively apply different thresholds to get a sparse representation (s).

Algorithm 1: Optimal wavelet domains search

```
input  : $ssim\_thr$ /*Minimum desired recovery quality*/
         $wavelets\_list$ /*Wavelet domains search space*/
output : $best\_W$ /*a wavelet domain with the highest $MSI$*/
$best\_W \leftarrow None$;
$best\_MSI \leftarrow 0$;
for $W$ in $wavelets\_list$ do
    $MSIs \leftarrow None$
    for $X$ in $signal\_list$ do
        $coeffs \leftarrow DWT(X, W)$
        for $thr$ in $thresholds\_range$ do
            $s \leftarrow thresholding(coeffs, thr)$
            $\hat{X} \leftarrow IDWT(s)$
            if $SSIM(X, \hat{X}) \geq ssim\_thr$ then
                $MSI \leftarrow zeros(s)/len(s)$
                break
        $MSIs.append(MSI)$
    if $best\_MSI < avg(MSIs)$ then
        $best\_MSI \leftarrow avg(MSIs)$
        $best\_W \leftarrow W$
return $best\_W$;
```

We then quantify the quality of the reconstructed signal ($\hat{X}$) from s by applying Inverse Discrete Wavelet Transform (IDWT) and calculating the SSIM index. Only the ones with $SIMM \geq ssim\_thr$ are kept. The $ssim\_thr$ we used for optimal wavelet domain search is 0.9. From our preliminary evaluations, this is sufficient for the signal to maintain its quality similar to the original (as discussed in Sec. 8). Note that this threshold is adjustable depending on the application's requirements. The sparest wavelet representation is the one that has the largest threshold. We calculate MSI by finding the ratio of non-zeroes components in s. The optimal wavelet domain is the one that has the smallest average MSI for all the input signals of the same pattern group. Finally, we repeat the same process to find optimal wavelet domains for all the patterns.

It is important to note that we only use DWT and IDWT to quantify patterns' sparsity, not running them on either the wearable or mobile device. After knowing the optimal domains, we can construct different Φ and Ψ matrices for individual patterns and store them on wearable and mobile devices. However, the conventional compressive sensing theory would require the compressed sample size to be around four-time the density of a sparse representation [60], making it very challenging to work on near-sparse biosignals. E.g., a muscle contraction pattern (Fig. 4) with 36% density will not work as it requires the compressed signal to have 1.44X more samples than the original signal.

5.4 Recovery with Pattern Information and Block Sparse Bayesian Learning

We devise an efficient reconstruction algorithm based on received pattern recognition information and the Block Sparse Bayesian Learning (BSBL) technique [78] as illustrated in Fig. 3. BSBL technique help to address the issue of high compressed sampling rate by taking into account the temporal sparsity and correlation among signal blocks.

To apply the BSBL technique, we consider a window of signal (s) of size N as a series of blocks of size d, i.e.,

$$s = [\ \underbrace{s_1 \dots s_d,}_{s^T[1],\ \text{1st block}}\ \underbrace{s_{d+1} \dots s_{2d},}_{s^T[2]}\ \dots, \underbrace{s_{N-d+1},\ \dots, s_N}_{s^T[N/d]}\ ]^T \qquad (1)$$

A signal with few blocks that are non-zeroes is called a block-sparse signal. This study assumes that the biosignal patterns are block-sparse in their respective optimal wavelet domains.

Each block (s_i) in the signal is modelled as a combination of two multivariant Gaussian distributions, i.e., the noiseless signal $p(s_i; \gamma_i, B_i) \sim N(0, \gamma_i B_i)$, and the noise vector $p(n; \delta) \sim N(0, \delta I)$. γ_i and B_i are the block sparsity control parameter and mutual correlation matrix of the i-th block, respectively. δ is a positive scalar representing the noise and I is the identity matrix. We estimate the parameters γ_i, B_i, and δ, by applying Type-II-maximum likelihood procedure to minimize the following cost function [78, 79],

$$L = log|\delta I + \Phi\Psi\Sigma_0(\Phi\Psi)^T| + Y^T(\delta I + \Phi\Psi\Sigma_0(\Phi\Psi)^T)^{-1}Y \qquad (2)$$

where $\Sigma_0 = diag\{\gamma_1 B_1, ..., \gamma_{N/d} B_{N/d}\}$. In contract to the conventional BSBL technique, we dynamically apply different wavelet basis (Ψ) to the Bayesian learning process based on received pattern information and its optimal wavelet domain. After the learning has converged, we find s by using Maximum-A-Posteriori estimation, i.e., $s = \Sigma_0(\Phi\Psi)^T(\delta I + \Phi\Psi\Sigma_0(\Phi\Psi)^T)^{-1}$. The signal is reconstructed as, $\hat{X} = \Psi s$.

Till this point, we could significantly reduce the wireless transmission rate. However, we might reach the stage where wireless communication is no longer the bottleneck. Thus, we will need to look elsewhere to increase energy efficiency further.

6 HARDWARE AND OS OPTIMIZATIONS

As PROS performs neural network inferences continuously in the background, processing efficiency is critical. We will discuss in this section the hardware and OS optimization techniques that we have adopted from the state-of-the-art to push the processing efficiency of PROS further.

Dynamic Voltage & Frequency Scaling. DVFS technique improves energy efficiency by reducing the operating frequency and voltage of the CPU core based on the workload's demand [80, 81]. We could formulate the energy consumption of a CPU core as, $E_{cpu} = (CV^2 f + VI_{static})T_{run} + VI_{static}T_{sleep}$. where C, V, f, I_{static}, T_{run}, and T_{sleep} are total gate capacitance, operating voltage, switching frequency, static leakage current, running and sleep time, respectively. As switching frequency is directly related to the operating voltage, i.e., $f \propto (V - V_{threshold})^{1.3}$ [81], we can significantly reduce the power consumption by lowering f, which also lowers V. DVFS, however, has a point of diminishing return [82]. When we decrease f, the time required for completing a task (T_{run}) increases, leading to increased static energy consumption due to I_{static}. We confirm this phenomenon on an ARM MCU. As we can observe from Fig. 5, the power efficiency of the CPU core increases up to 30%, i.e., from 7.1 to 9.1 DMIPS/mW (Dhrystone Million Instructions per Second per milliwatt) when we reduce f_{max} from 120 to 26 MHz and V

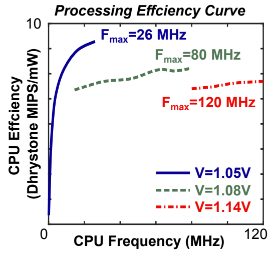

Figure 5: Efficiency Curve.

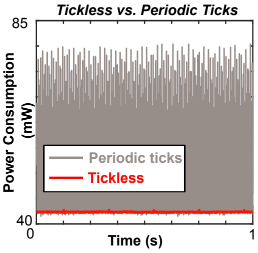

Figure 6: Tickless Sleep.

from 1.14 to 1.05V. However, this scaling is not linear as the static power becomes dominant at lower fs, i.e., the efficiency drop to 0.5 DMIPS/mW with $f = 100kHz$.

To address this issue, we develop a lightweight DVFS algorithm inside the PROS kernel. It is based on the principle that the CPU should run at the maximum frequency of the lowest possible voltage while still meeting the deadline (T_d). The deadline can either be the time where a signal window is returned by the DMA (Direct Memory Access) peripheral or the desired value set by the developer. It follows three steps as follows. First, we set f and V at the maximum values (f_{max}, V_{max}) and run all the background processing (e.g., tinyPR primitives, preprocessing, compressive sensing, etc.) required by the application to measure the CPU time (t_0). Second, we estimate the lowest possible CPU frequency that still meet the deadline, i.e., $f_{min} = \lceil T/t_0 * f_{max} \rceil$. From f_{min}, we can find the lowest possible voltage range (V_{min}) that could support f_{min}. Finally, we set the CPU frequency to the maximum f, supported by V_{min}. This is the optimal frequency for our workload. Depending on the application's workload dynamic, we can run DVFS once at the system startup or run it every scheduling cycle.

Dynamic Tickless Mode. Many OSes such as Linux or FreeRTOS [83] use a global hardware timer generating periodic ticks (e.g., 100 or 1000 ticks per second). This is a nice and simple timebase for OS tasks such as scheduling or synchronizations [84]. However, it negatively impacts low power performance as the CPU is constantly wakened up from its sleep mode every the timer interrupts fires. This leads to a significant energy loss due to constantly waking up. Fig. 6 illustrates the energy consumed by switching back and forth between wake and sleep mode every 1ms will outweigh any energy saved by putting the CPU to sleep.

To address this issue, we employ the dynamic tickless mode (dyntick) [84] for PROS. Dyntick eliminates the periodic timer interrupts when the system is idle. The CPU is put into sleep mode until the next task is ready to run or an interrupt is fired. Since the kernel still needs to wake up when its tasks are ready, we implement a low power timebase (e.g., the real-time clock peripheral on ARM Cortex-M MCUs) that can still run while the CPU is in sleep mode. We set the alarm on this low-power timebase to wake up the CPU when its tasks are ready. We also use it to track how much time the CPU has slept to adjust the kernel timebase. This significantly reduces the energy wasted due to constantly waking up while maintaining the OS kernel's proper operations.

Adaptive Energy Threshold. Our tinyPR primitives (Sec. 4) are powerful tools to recognize PoIs. However, they might be too expensive to run on obvious background signals. Thus, we apply a light-weight adaptive energy threshold method, which is quite effective in eliminating non-stationary background noise in speech recognition systems [85, 86]. After a signal segment of size n has been captured, we calculate its energy by $E = \frac{1}{n}\sum_{i=1}^{n}|x(i)^2|$ and compare with a threshold value. The signal segment with lower energy level is eliminated. We adaptively update the threshold value (λ) based on m previous measurements of background and PoI signals by $\lambda = \alpha\sum_{j=1}^{m}\frac{1}{j}E_{bg} + \beta\sum_{j=1}^{m}\frac{1}{j}E_{PoI}$ [85]. As the definition of background signals varies from one application to another, we will need to adjust α and β accordingly.

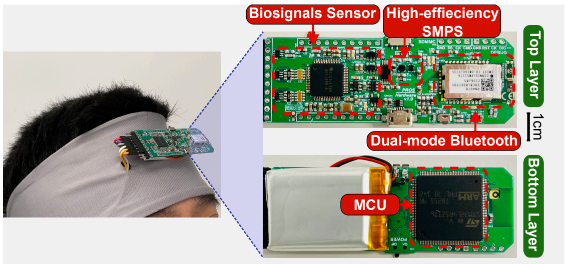

Figure 7: PROS hardware platform.

PROS Abstractions. To provide a friendly interface for application developers, we wrap up all underlying processing procedures with the PROS scheduler. The scheduler provides the application with the interfaces to (1) set up and configure the TinyPR primitives needed by the application, (2) wake up the application threads for real-time responses when a subscribed PoI is detected. It also handles background operations such as running TinyPR primitives and PDCS algorithm.

We wrap the tinyPR models, pre-processing pipeline, and PDCS algorithm as C++ classes and implement the PROS scheduler as a FreeRTOS task. At the initial state, the developer can declare the TinyPR models, confidence threshold, compression ratio, and their mapping to application tasks. During runtime, if the output probability of the positive class is over the defined threshold, the scheduler will notify the subscribed tasks for execution. Direct task-to-task notification of FreeRTOS is employed to ensure efficiency. The notified task could request to access the signal data buffers, but it will need to make a copy before they are overwritten.

7 IMPLEMENTATION

PROS Firmware Framework. We implement PROS based on the FreeRTOS real-time kernel, which provides the base OS functionalities: preemptive task scheduling, dynamic memory management, and synchronizations. We implement additional optimization modules: DVFS and dynamic tickless sleep mode, then integrate them into the FreeRTOS kernel. We train our TinyPR primitives on an Nvidia RTX 3090 GPU and use the TensorFlow Lite Microcontroller to perform inferences on PROS hardware. The neural network operations, MFCC calculation, adaptive energy detector are accelerated by SIMD (Single Instruction Multiple Data) and single-cycle MAC (Multiplication-and-Accumulation) instructions. We use pre-generated binary matrices stored in MCU's FLASH to perform the PDCS algorithm. We also implement the optimal wavelet search algorithm in MATLAB.

PROS hardware and mobile apps. We build a hardware prototype (Fig. 7) to support all the operations of PROS. Specifically, it contains an ARM Cortex-M4F MCU (STM32L4R5, 2MB FLASH, 640KB RAM) with four efficiency modes, accelerated DSP, and neural engines. To support DVFS, we bypass the internal regulator

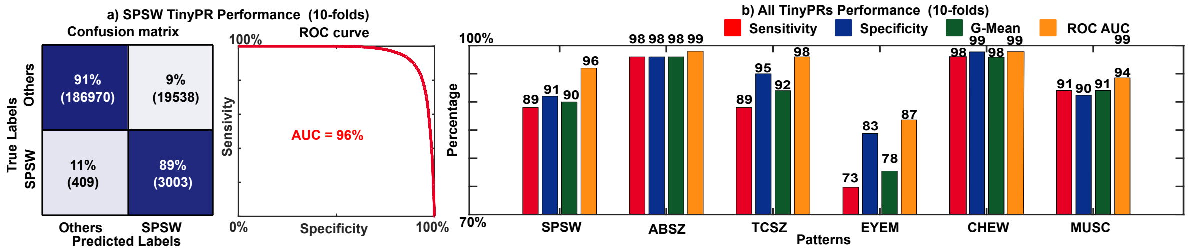

Figure 8: TinyPRs (tiny configuration) performance in 10-folds cross validations.

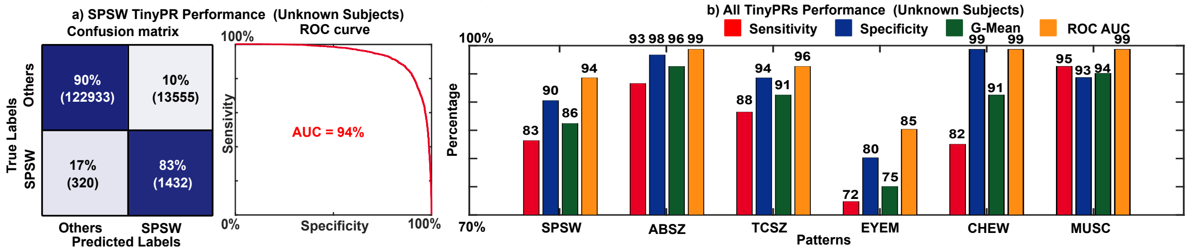

Figure 9: TinyPRs (tiny configuration) performance in unknown subjects evaluations.

with an adaptive Switched Mode Power Supply (SMPS), providing the CPU voltage ranging from 1.00 to 1.35V. We use a dual-mode Bluetooth module (RN4678) to provide wireless communication with Bluetooth EDR (Enhanced Data Rate) and BLE 5 protocols. It supports the throughput up to 48 and 256 kbps with BLE and EDR, respectively. We put together our hardware in a headband form factor. Finally, we deploy the signal reconstruction algorithm to two mobile platforms, i.e., Galaxy S20 and Surface Go 2.

8 EVALUATIONS

8.1 Datasets Preparation.

In this study, we use two open biosignal datasets, namely, TUSZ (Epileptic Seizures EEG events) and TUAG (EOG and EMG events), to develop and evaluate our TinyPR primitives and PDCS algorithms. The collection protocol was approved by Temple University Hospital IRB [87]. From the TUSZ dataset, we pick out a subset of 60 subjects with three important epileptic seizure patterns, i.e., (1) spike-and-sharp-wave (SPSW) patterns, (2) 3-Hertz spike-and-wave discharges (ABSZ), and (3) muscle stiffing and convulsions patterns (TCSZ), which represent focal/generalized non-specific, absence, and tonic-clonic seizures. Similarly, we pick up 60 subjects together with three EOG and EMG patterns, i.e., (1) eyes movements (EYEM), (2) chewing (CHEW), and (3) muscle contractions (MUSC), from the TUAG dataset. The chosen patient data were collected in various clinical settings such as the epilepsy monitoring unit, intensive care unit, emergency rooms, and routine EEG sessions. They contain both inpatients and outpatients with ages from five to 83 years old. As the sampling rate of the datasets varies (from 250 to 1024 Hz), we uniformly resample all the data to 500 Hz, which is the optimal data rate for EEG analysis [52].

We set aside ten subjects from each dataset for the unknown-subjects evaluations. The remaining are split into 10-folds for training (80%), validating (10%), and testing (10%). We then segment raw signal into non-overlapping windows. We use the duration of three seconds to cover sufficient pattern information. For the TUSZ dataset, it results in 2,099,479 data points (background: 2059148, SPSW: 34128, TCSZ: 3651, ABSZ: 2552) for the 10-folds cross validation and 138,340 data points (background: 136292, SPSW: 1752, TCSZ: 220, ABSZ: 76) for the unknown-subjects evaluation. With the TUAG dataset, we have 495,338 data points (background: 440,479, EYEM: 15488, CHEW: 2993, MUSC: 36378) for the 10-folds cross validation and 11,993 data points (background: 9467, EYEM: 99, CHEW: 104, MUSC: 2323) for the unknown-subjects evaluation.

Per each window, we apply the MFCC approach with our parameters' configuration (see Sec. 4.3) to extract the feature set of 10 MFCC features. The window is positive if the target pattern appears and negative, otherwise. With each fold, we apply an oversampling technique, SMOTE [35], to enrich the amount of positive data for the training set. Upsampling is not applied to validation, test, and unknown-subjects sets to keep the original distribution of samples.

8.2 TinyPR Primitives.

Classification metrics. As we cast our TinyPR primitives as binary classification models (Sec. 4). We use four indices of the confusion matrix: true positive (TP) is the number of actual positive segments which are correctly classified; true negative (TN) is the number of the actual negative segments that are correctly classified; false positive (FP) is the number of actual negative segments that are incorrectly classified as positive; false negative (FN) is the number of actual positive segments which are incorrectly classified as negative. With these notions, we define the sensitivity, specificity, and G-Mean scores as follow, $Sens = \frac{TP}{TP+FN}; Spec = \frac{TN}{TN+FP}; G-Mean = \sqrt{Sens * Spec}$. We plot the Receiver Operating Characteristic (ROC) curve to quantify the trade-off between sensitivity and specificity. We also report Area Under the Curve (AUC) as an additional performance metric.

10-folds cross-validation. We use standard Adam optimizer [88] with learning rate of $1e-2$ and $(\beta_1, \beta_2) = (0.5, 0.999)$ for optimization. The classification loss is cross-entropy. We train each model with batch size 32 for 200 epochs. We evaluate the G-Mean score of models on the validation set to select the best model.

We present the results on 10-folds cross-validations in Fig. 8. All the results are from the tiny configuration. Fig. 8a shows an example of a normalized confusion matrix and ROC of the SPSW TinyPR

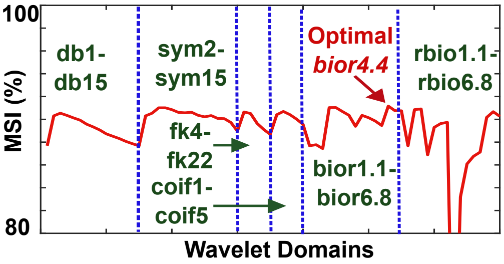

Figure 10: ABSZ optimal domain search.

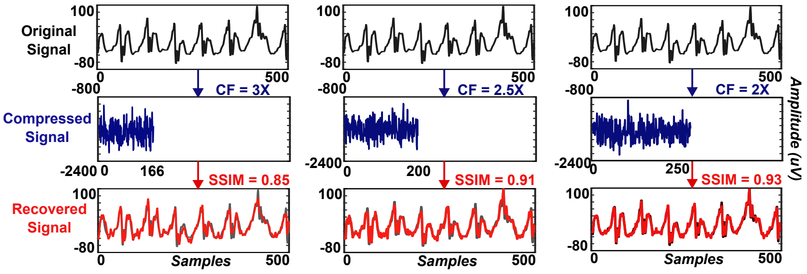

Figure 11: ABSZ recovery quality in *bior*4.4 domain.

primitive. From the confusion matrix, we could see that the TinyPR model can effectively eliminate 91% (specificity) of the irrelevant negative windows while being able to capture 89% (sensitivity) of the positive ones. The G-Mean and AUC scores are 90% and 96%, respectively. Fig. 8b summarizes all the results. Among these, the ABSZ TinyPR has the highest scores with 98% and 99% for G-Mean and AUC, respectively. The EYEM model has the lowest scores (78% G-Mean, 87% AUC) due to low signal amplitudes and large variations between vertical and horizontal movements.

Unknown subjects evaluation. To further evaluate the ability to work on unknown subjects, we use the best TinyPR primitives, chosen by their G-Mean scores, from our 10-fold validations to test on the unknown-subjects set. Fig. 9 presents our results. Fig. 9a shows that our SPSW TinyPR primitive could achieve 90%, 83%, 86%, and 94% of specificity, sensitivity, G-Mean, and AUC, respectively. Similarly, Fig. 9b presents the results of all TinyPR primitives. The EYEM model has the lowest G-Mean and AUC at 75% and 85%, while all other models can achieve more than 85%. These results show the feasibility of our developed TinyPR primitives to work even on people that the models have not encountered.

Model and features sizes. To quantify the effects of model sizes, we evaluate different configurations, such as tiny, small, and medium, on the same SPSW pattern. We see that the training time will converse quicker with larger model sizes, i.e., <50 epochs with the medium configuration versus >100 epochs with the tiny configuration. However, the results do not significantly improve, i.e., 1-2% variation, in the G-Mean score. Similar results are also observed with larger feature sizes, i.e., 20 vs. 10 MFCC features. Since we only need to train the model once, we can afford a longer training time to achieve smaller model and feature sizes. Smaller sizes will significantly reduce the latency and memory footprint during runtime. After quantization, we observe the reduction of 63% with our tiny configuration, i.e., from 11KB (PyTorch) to 5KB (TFLM). The loss after quantization is minimal, with <1% of the G-Mean score.

8.3 Pattern-driven Compressive Sensing.

Optimal wavelet domains search. We conduct the optimal wavelet domains search (Sec. 5.3) for all the patterns on the training dataset. Fig. 10 presents the results for the ABSZ pattern. The results confirm our intuition that the choice of wavelet domain is significantly important. For the ABSZ pattern, the global maxima and minima of average MSI are 91.2% and 61.9% with *bior*4.4 and *rboi*3.1 wavelet domains, respectively. This means there are more than 4X differences between the density of the two domains, making CS unusable with the latter. Interestingly, we observe that several local maxima in each wavelet family have similar results to the global maxima. E.g., the *db*3 domain has an average MSI of 90%, which is only 1% lower. This shows overlapping among wavelet domains, which one might exploit to further improve recovery latency by using simpler wavelet domains. Table 1 summarizes the optimal wavelet domains for all patterns. SPSW has the highest MSI (92.6%) with *bior*6.8 domain while TCSZ has the lowest (71.4%) with *sym*14 due to high frequency and stochastic muscle components.

Table 1: Recovery quality with different CFs.

Pattern	Wavelet Domain	MSI (%)	SSIM with different CFs (w=3s)				
			1.5X	2X	3X	4X	5X
SPSW	bior6.8	92.6	0.99	0.98	0.96	0.94	0.89
ABSZ	bior4.4	91.2	0.99	0.97	0.94	0.89	0.82
TCSZ	sym14	71.4	0.84	0.81	0.57	0.39	0.31
EYEM	sym5	89.6	0.98	0.97	0.91	0.84	0.80
CHEW	bior4.4	84.0	0.93	0.93	0.84	0.78	0.71
MUSC	sym5	79.7	0.92	0.88	0.70	0.60	0.50

To measure the computational cost, we perform the search on a Linux workstation (Core-i7 3.6x8GHz, 128GB RAM). With eight MATLAB parallel workers, the search consumes 6981MB of memory and takes from 38 (CHEW) to 247 (MUSC) hours to finish. As we only need to run the search once, it will not affect the real-time performance during the deployment.

Compression factors tuning. After knowing the optimal wavelet domains, we conduct evaluations to quantify the recovery signal quality with different compression factors (CFs). We run the evaluations on the whole training dataset and note down the CFs and the average SSIMs values in Table 1. For the SPSW pattern (MSI = 92.6%), we can achieve the CF of more than 5X without having the average SSIM drop below 0.85. For patterns with low MSI such as TCSZ (71.4%), we could only achieve the CF of 1.5-2X without deteriorating the recovered signal. We visualize the recovered ABSZ signal quality with different SSIMs in Fig. 11. We could see that when $SSIM \geq 0.85$, the recovered signal looks very similar to the original one. When $SSIM \geq 0.93$, we could not visually spot the differences. This fits with the literature that the recovery starts to be indistinguishable by human eyes when $SSIM \geq 0.92$ [89].

Comparison with previous works. In previous works on compressive sensing (CS) such as [78, 90–92], static and pre-defined CFs and sparse recovery domains (e.g., Discrete Cosine Transform) are used due to the lack a pattern recognition capability on low-power hardware. In [92], three state-of-the-art CS algorithms, namely, DCT-based BSBL-BO [78], DCT-based l_1 [91], and Block-CoSaMP [90] are compared on the EEGLab dataset [93] (32 channels, 80 3-s EEG windows). With a CF of 2X, only DCT-based BSBL-BO achieves a satisfying SSIM of 0.85, while DCT-based l_1 and Block-CoSaMP could only reach SSIMs of 0.45 and 0.48, respectively. In contrast with previous works, PROS enables the ability to

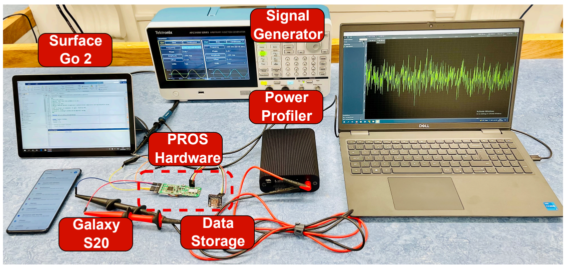

Figure 12: Runtime experiment setup.

recognize signal patterns of interest directly on the sensing hardware. This ability helps the proposed PDCS algorithm to apply optimal CF and sparse recovery domains for individual patterns, leading to more efficient compression.

8.4 Hardware runtime performance.

We deploy the developed TinyPR primitives and PDCS algorithm to our PROS hardware (ARM-Cortex M4F MCU, 2MB FLASH, 640KB RAM, GCC -Ofast) to measure their memory footprints, processing latency, and energy consumption. We also deploy the PDCS recovery algorithm on two mobile devices, i.e., (1) Galaxy S20 (Octa-core 2.2-2.7GHz Cortex-A55) and (2) Surface Go 2 (Dual-core 1.7GHz Intel Pentium) to measure the processing performance. We use the Otti Arc profiler to measure energy consumption with the sampling rate at 4000Hz. Fig. 12 presents our experiment setup.

Memory footprints. With the tiny configuration, each TinyPR primitive only consumes 5KB of FLASH (0.2% of system FLASH) and 30KB of RAM (4.5% of system RAM). The consumed RAM could be dynamically reused if we do not need to run multiple TinyPR primitives concurrently. Our PDCS consumes from 4-30KB of FLASH (0.2-1.4% of system FLASH) to store random CS matrices.

Processing latency. On our PROS hardware, the latency of MFCC calculation, one TinyPR inference, and compressive sensing could be as low as 6, 26, and 1ms when we clock the MCU at 120MHz, respectively. When we clock the MCU at 24MHz, the results are 29, 126, and 4ms. On the mobile devices, we run the recovery algorithm on 2000 different signal samples. The average latency is 50 and 94ms on the Galaxy S20 and Surface Go 2. The results show that PROS can respond to critical events within milliseconds on the hardware. The response time on the phone only depends on wireless communication latency as the additional processing of our recovery algorithm is minimal.

In a real-time setting, biopotential signal windows are continuously buffered and need to be processed every few seconds, e.g., three seconds windows in our evaluation. Our PROS prototype could process the signals in real-time since the whole processing latency on both the wearable and mobile devices could be as low as 88ms. This show the feasibility of PROS in real-time applications.

Energy consumption. We measure the energy consumption of each operation on PROS hardware with and without DVFS. When DVFS is not used, the MFCC calculation, one TinyPR inference, and compressive sensing consume 0.6, 2.4, and 0.1mJ. When DVFS is used, the results are 0.4, 2, and 0.07mJ. Thus, we could see that our DVFS and SMPS could increase the energy efficiency by 30-50%.

Comparison with open-source platforms. We conduct the processing latency (L) and energy consumption (E) measurements of our proposed TinyPR models and PDCS algorithm on an Arduino Nano BLE Sense and a Raspberry Pi Zero v1.3 (Fig. 13). Table 2 summarizes the results. Since Raspberry Pi is not designed for low-power applications, it has the largest overheads. The performance of PROS on both the Arduino and Raspberry could be further improved by optimizing the processing software.

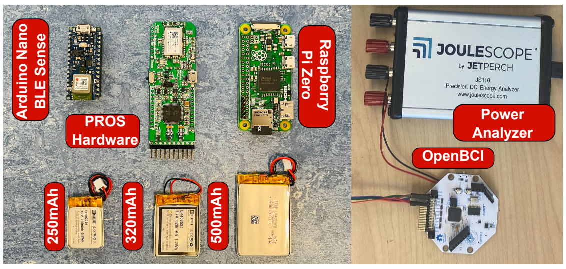

Figure 13: Open-source hardware and biosensing platforms.

Table 2: Open-source platforms evaluations.

Platform	CPU/RAM	TinyPR (L/E)	PDCS (L/E)
Arduino Nano BLE	64MHz/256KB	50ms/4.2mJ	12ms/1.0mJ
Raspberry Pi Zero	1GHz/512MB	92ms/108mJ	4ms/3.6mJ
PROS hardware	120MHz/256KB	26ms/2.4mJ	1ms/0.1mJ

We also measure the power consumption of a commercialized biosensing platform, i.e., OpenBCI. The average consumption while streaming is 146mW. Thus, with 250, 320, and 500mAh LiPo batteries, it could last for 6.3, 8.1, and 12.7 hours, respectively. With PROS hardware and the workload discussed in Sec. 8.5, we could increase the battery life to 84, 107, and 168 hours.

8.5 Epileptic seizures detection use case.

We conducted our experiment to quantify the significance of PROS in detecting epileptic seizures. We choose the seizure detection use case because of its high-fidelity data requirement and challenging local processing on the device.

Previous studies [94, 95] have pointed out that detecting non-motor seizures is non-trivial and requires medical experts to analyze and diagnose the captured signals. Thus, maintaining high-fidelity signals is essential. Furthermore, local seizure detection on wearable devices is challenging when considering the constrained computing resource of low-power microcontrollers. In literature, neural networks such as VGG [96] or ResNet [97] are feasible for detecting seizures with good accuracy. However, such networks are too large to be run on an MCU with limited memory (<1 MB of SRAM) [98]. Even if we can extensively prune the network to run on MCUs, the accuracy will degrade significantly, leading to unusable results) [98]. Thus, sending signals to a nearby offload device is still necessary for further analysis, diagnosis or classification.

In this case study, we focus on focusing on three crucial seizure types, i.e., tonic-clonic, absence, and focal/generalized non-specific seizures. They require three seizure-related patterns, i.e., TCSZ, ABSZ, and SPSW, respectively. To ensure practicality, we use the unknown-subjects dataset (10 subjects, Sec. 8.1). This results in 208,246,500 samples, i.e., 29 hours of data. The CFs for TCSZ, ABSZ, and SPSW are set at 2, 4, and 5X, respectively. Since tonic-clonic seizures have the highest risk of fatality [99], we put TCSZ at the highest priority, followed by ABSZ and SPSW.

The results show that PROS can reduce the number of transmission data by 24X (8,807,370 vs. 208,246,500 samples). Our TinyPR primitives can pick up more than 85% seizure signals and eliminate 86% non-seizure ones. The recovered signal is high-fidelity with the average SSIMs for TCSZ, ABSZ, and SPSW: 0.93, 0.92, and 0.93, respectively. By transmitting all the signals out to a mobile device with Bluetooth EDR, the wearable device consumes 15.7kJ. With PROS, the device only consumes 1.71kJ, giving a boost of 818%. Interestingly, since PROS significantly reduces the transmission data, we could use a lower-rate protocol such as BLE, while it is not possible with the original amount of data throughput. By using BLE and PROS, the device only consumes 1.15kJ in total, boosting the energy efficiency up to 1265%. Thus, with a 500mAh Li-Po battery (Fig. 7), the device could last for a whole week while continuously monitoring seizure events. Finally, PROS could respond to deadly tonic-clonic seizures directly on the device within as low as 32ms. This is especially important when the mobile device might not be available, e.g., during charging or out of communication range. These results show the feasibility that PROS could significantly improve users' experience and even reduce fatalities.

9 RELATED WORKS

Pattern recognition on microcontrollers. Tiny Machine Learning [42], TinyML, provides the solution to bring powerful deep learning models into extremely resource-constrained devices such as microcontrollers. Various approaches [42, 45, 100, 101] have been proposed to optimize and achieve a compact design that satisfies the extreme hardware constraints. MCUNet [45] mitigates the memory bottleneck of the CNN architecture with the patch-based inference approach,while FANN-MCU [100] provides toolkits for building energy-efficient networks on MCUs. Unlike the prior works, our work further considers sparsity and locality properties of biosignals to ameliorate the system efficiency.

Compressive sensing in healthcare. CS has been applied in processing biological data in mobile healthcare and telemonitoring to provide a faster, more accurate, and more energy-efficient system [102, 103]. The application widely includes medical imaging [65, 104, 105], real-time bio-signal processing, and neuroscience applications [102, 106, 107]. Although the benefit of CS in biosignal processing is considerable [108–110], the lack of a reliable sparse domain reduces its effectiveness and creates large variations [111, 112]. We leveraged the advantage of CS and the new ability to detect patterns directly on our hardware design to propose an efficient system for low-power wearable devices.

Low-power wearable platforms. Power consumption is a major concern in any wearable or IoT devices. Thus, it has attracted much attention in recent platforms and OSes [113–116]. Amulet [117], Mindo [118], Convergence [119], RIOT-OS [120], TinyOS [121], FreeRTOS [83] focused on leveraging the event-driven scheduling and low power modes to reduce the energy consumed by an MCU and others high-power components. While existing systems provides hardware or OS optimizations, they have not considers the sparsity of captured signals, which is our main contribution.

Commercialized biosensing platforms. There are several commercialized biosensing platforms on the market, such as Emotiv Epoch [122], Muse [123], and Neurosky MindWave [124]. They are equipped with 250-600 mAh batteries and last from 5 to 9 hours of continuous streaming. Since none of them provides compression or recognition ability, they rely on conventional methods such as downsampling or proprietary wireless protocols to prolong battery lifetime. In our PROS framework, we take advantage of on-chip intelligence and compressive sensing to boost energy efficiency further and enable real-time responses on the devices.

10 DISCUSSIONS

Improving TinyPR performance. The performance of TinyPR primitives directly links to the quality of the training data. Extensive and high-quality training data will result in more accurate pattern recognition models. We envision that the crowdsourcing effort of the community can address the challenge of high-quality training data. The more data we collect, the better TinyPR primitives can be built, and more healthcare wearable applications will be enabled.

Noise and artifacts. As motion, environmental, and muscle artifacts could contaminate the biosignal streams; pre-processing is needed to ensure signal integrity. We assume that motion and environmental noise could be mitigated on the sensing hardware by employing techniques such as active amplifying, sigma-delta modulation, and digital filters as proposed in previous works [125, 126]. For the muscle activities, it is up to the application to decide whether they are signals of interest or unwanted artifacts. Thus, PROS provides a pattern primitive (MUSC) to detect muscle contractions. The application can further process the signal if needed.

Dataset's limitations. In this project, we chose the biosignal datasets that provide practical clinical settings, so we could gauge the effectiveness of PROS on various patients' conditions. The clinical settings, however, are relatively stable and will not reflect all the usage patterns in daily life, such as motion and environmental noise, wearing position, etc. We would love to investigate this further in our future work when real-life data becomes available.

Extending and sharing ability. We envision that PROS serves as a framework where the community could develop support for various biosensors after its release. The amount of potential biosensors and pattern primitives is significant. E.g., facial expressions (EMGs), emotions (electrodermal activity), coughing patterns (acoustic), and many more. Sharing the processing pipeline, such as TinyPR models or PDCS, among multiple applications is another exciting direction we are looking into to enhance the efficiency further.

Other considerations for daily usage. While the main focus of PROS is on battery lifetime, other factors could impact the user's experience. First, wearability is important since monitoring applications such as seizure detection rely on long-term measurements to detect sudden attacks. Second, data privacy is another critical factor. Federated learning could tackle this issue by enabling multiple edge devices to collaborate and build a common model without exchanging local data. Finally, closed-loop control algorithms could be developed between mobile and wearable devices so that PROS could adapt to the changing conditions over time.

11 CONCLUSION

In this study, we propose PROS, an efficient pattern-driven compressive sensing framework for low-power biosensing wearables, by exploiting the sparsity of biosignals. In a practical use case such as epileptic seizures detection, PROS significantly boosts the energy efficiency and enables real-time response to critical events while maintaining high fidelity signal.

ACKNOWLEDGEMENTS

We thank our shepherd and anonymous reviewers for their insightful comments on the manuscript. This research was supported by Oxford DPhil scholarship, the Alfred P. Sloan Fellowship no. FG-2020-13110 (TV), ERC through Project 833296 (EAR), Nokia Bell Labs, and NSF award #2132112.

REFERENCES

[1] Michael Shirer Jitesh Ubrani and Ramon Llamas. *Consumer Enthusiasm for Wearable Devices Drives the Market to 28.4% Growth in 2020, According to IDC.* https://tinyurl.com/snys6tzx.

[2] Fortune Business Insights Pvt. Ltd. *Latest Research 2020: Wearable Medical Devices Market Witness Astonishing Growth at 24.7% CAGR to Reach USD 139,353.6 Million by 2026.* https://tinyurl.com/va7u3vap.

[3] Chris Falkous and Julianne Callaway. *Wearable Technology in Life Insurance.* https://tinyurl.com/3ypnb3de.

[4] Zheng Lou, Lili Wang, Kai Jiang, Zhongming Wei, and Guozhen Shen. Reviews of wearable healthcare systems: Materials, devices and system integration. *Materials Science and Engineering: R: Reports*, 140:100523, 2020.

[5] C-M Tsai, S-L Chou, Elliot N Gale, and Willard D McCall. Human masticatory muscle activity and jaw position under experimental stress. *Journal of oral rehabilitation*, 29(1):44–51, 2002.

[6] Ulf Lundberg, Roland Kadefors, Bo Melin, Gunnar Palmerud, Peter Hassmén, Margareta Engström, and Ingela Elfsberg Dohns. Psychophysiological stress and emg activity of the trapezius muscle. *International journal of behavioral medicine*, 1(4):354–370, 1994.

[7] K Kohyama, L Mioche, and P Bourdio3. Influence of age and dental status on chewing behaviour studied by emg recordings during consumption of various food samples. *Gerodontology*, 20(1):15–23, 2003.

[8] Laurence Mioche, Pierre Bourdiol, Jean-Francois Martin, and Yolande Noël. Variations in human masseter and temporalis muscle activity related to food texture during free and side-imposed mastication. *Archives of Oral Biology*, 44(12):1005–1012, 1999.

[9] Kaoru Kohyama, Laurence Mioche, and JEAN-FRANCOIS MARTIN. Chewing patterns of various texture foods studied by electromyography in young and elderly populations. *Journal of Texture Studies*, 33(4):269–283, 2002.

[10] Xiao-Wei Wang, Dan Nie, and Bao-Liang Lu. Emotional state classification from eeg data using machine learning approach. *Neurocomputing*, 129:94–106, 2014.

[11] Klaus-Robert Müller, Michael Tangermann, Guido Dornhege, Matthias Krauledat, Gabriel Curio, and Benjamin Blankertz. Machine learning for real-time single-trial eeg-analysis: from brain–computer interfacing to mental state monitoring. *Journal of neuroscience methods*, 167(1):82–90, 2008.

[12] J Sarnthein, A Morel, A Von Stein, and D Jeanmonod. Thalamic theta field potentials and eeg: high thalamocortical coherence in patients with neurogenic pain, epilepsy and movement disorders. *Thalamus & Related Systems*, 2(3):231–238, 2003.

[13] Mengni Zhou, Cheng Tian, Rui Cao, Bin Wang, Yan Niu, Ting Hu, Hao Guo, and Jie Xiang. Epileptic seizure detection based on eeg signals and cnn. *Frontiers in neuroinformatics*, page 95, 2018.

[14] Jelena Skorucak, Anneke Hertig-Godeschalk, Peter Achermann, Johannes Mathis, and David R Schreier. Automatically detected microsleep episodes in the fitness-to-drive assessment. *Frontiers in neuroscience*, 14:8, 2020.

[15] Alexander J Casson. Wearable eeg and beyond. *Biomedical engineering letters*, 9(1):53–71, 2019.

[16] Aleksandr Ometov, Viktoriia Shubina, Lucie Klus, Justyna Skibińska, Salwa Saafi, Pavel Pascacio, Laura Flueratoru, Darwin Quezada Gaibor, Nadezhda Chukhno, Olga Chukhno, et al. A survey on wearable technology: History, state-of-the-art and current challenges. *Computer Networks*, 193:108074, 2021.

[17] Shyamal Patel, Hyung Park, Paolo Bonato, Leighton Chan, and Mary Rodgers. A review of wearable sensors and systems with application in rehabilitation. *Journal of neuroengineering and rehabilitation*, 9(1):1–17, 2012.

[18] Bin Hu, Hong Peng, Qinglin Zhao, Bo Hu, Dennis Majoe, Fang Zheng, and Philip Moore. Signal quality assessment model for wearable eeg sensor on prediction of mental stress. *IEEE transactions on nanobioscience*, 14(5):553–561, 2015.

[19] Dharmendra Gurve, Denis Delisle-Rodriguez, Teodiano Bastos-Filho, and Sridhar Krishnan. Trends in compressive sensing for eeg signal processing applications. *Sensors*, 20(13):3703, 2020.

[20] Bluetooth SIG Working Groups. *Bluetooth Core Specification 4.0.* https://tinyurl.com/2e25vsxu.

[21] Fernando Moreno-Cruz, Víctor Toral-López, Antonio Escobar-Molero, Víctor U Ruíz, Almudena Rivadeneyra, and Diego P Morales. trench: ultra-low power wireless communication protocol for iot and energy harvesting. *Sensors*, 20(21):6156, 2020.

[22] *PROS.* https://github.com/PROS-public.

[23] Shelagh JM Smith. Eeg in the diagnosis, classification, and management of patients with epilepsy. *Journal of Neurology, Neurosurgery & Psychiatry*, 76(suppl 2):ii2–ii7, 2005.

[24] Amir M Abdulghani, Alexander J Casson, and Esther Rodriguez-Villegas. Compressive sensing scalp eeg signals: implementations and practical performance. *Medical & biological engineering & computing*, 50(11):1137–1145, 2012.

[25] Li Deng and Dong Yu. Deep learning: methods and applications. *Foundations and trends in signal processing*, 7(3–4):197–387, 2014.

[26] Nagarajan Ganapathy, Ramakrishnan Swaminathan, and Thomas M Deserno. Deep learning on 1-d biosignals: a taxonomy-based survey. *Yearbook of medical informatics*, 27(01):098–109, 2018.

[27] Sander Koelstra, Christian Muhl, Mohammad Soleymani, Jong-Seok Lee, Ashkan Yazdani, Touradj Ebrahimi, Thierry Pun, Anton Nijholt, and Ioannis Patras. Deap: A database for emotion analysis; using physiological signals. *IEEE transactions on affective computing*, 3(1):18–31, 2011.

[28] Thomas M Deserno and Nikolaus Marx. Computational electrocardiography: revisiting holter ecg monitoring. *Methods of Information in Medicine*, 55(04):305–311, 2016.

[29] Nizar Islah, Jamie Koerner, Roman Genov, Taufik A Valiante, and Gerard O'Leary. Machine learning with imbalanced eeg datasets using outlier-based sampling. In *2020 42nd Annual International Conference of the IEEE Engineering in Medicine & Biology Society (EMBC)*, pages 112–115. IEEE, 2020.

[30] Qi Yuan, Weidong Zhou, Liren Zhang, Fan Zhang, Fangzhou Xu, Yan Leng, Dongmei Wei, and Meina Chen. Epileptic seizure detection based on imbalanced classification and wavelet packet transform. *Seizure*, 50:99–108, 2017.

[31] David Belo, João Rodrigues, João R Vaz, Pedro Pezarat-Correia, and Hugo Gamboa. Biosignals learning and synthesis using deep neural networks. *Biomedical engineering online*, 16(1):1–17, 2017.

[32] Amir Gholami, Sehoon Kim, Zhen Dong, Zhewei Yao, Michael W Mahoney, and Kurt Keutzer. A survey of quantization methods for efficient neural network inference. *arXiv preprint arXiv:2103.13630*, 2021.

[33] A Lazarevic, Jaideep Srivastava, and Vipin Kumar. Data mining for analysis of rare events: A case study in security, financial and medical applications. In *Pacific-asia conference on knowledge discovery and data mining*, 2004.

[34] Stijn Luca, Peter Karsmakers, Kris Cuppens, Tom Croonenborghs, Anouk Van de Vel, Berten Ceulemans, Lieven Lagae, Sabine Van Huffel, and Bart Vanrumste. Detecting rare events using extreme value statistics applied to epileptic convulsions in children. *Artificial intelligence in medicine*, 60(2):89–96, 2014.

[35] Nitesh V Chawla, Kevin W Bowyer, Lawrence O Hall, and W Philip Kegelmeyer. Smote: synthetic minority over-sampling technique. *Journal of artificial intelligence research*, 16:321–357, 2002.

[36] A Moura, S Lopez, I Obeid, and J Picone. A comparison of feature extraction methods for eeg signals. In *2015 IEEE Signal Processing in Medicine and Biology Symposium (SPMB)*, pages 1–2. IEEE, 2015.

[37] G. N. Rajesh. Analysis of mfcc features for eeg signal classification. 2019.

[38] Radek Martinek, Martina Ladrova, Michaela Sidikova, Rene Jaros, Khosrow Behbehani, Radana Kahankova, and Aleksandra Kawala-Sterniuk. Advanced bioelectrical signal processing methods: Past, present, and future approach—part iii: Other biosignals. *Sensors*, 21(18):6064, 2021.

[39] Haryong Song, Yunjong Park, Hyungseup Kim, and Hyoungho Ko. Fully integrated biopotential acquisition analog front-end ic. *Sensors*, 15(10):25139–25156, 2015.

[40] Pete Warden and Daniel Situnayake. *Tinyml: Machine learning with tensorflow lite on arduino and ultra-low-power microcontrollers.* O'Reilly Media, 2019.

[41] Liangzhen Lai, Naveen Suda, and Vikas Chandra. Cmsis-nn: Efficient neural network kernels for arm cortex-m cpus. *arXiv preprint arXiv:1801.06601*, 2018.

[42] Partha Pratim Ray. A review on tinyml: State-of-the-art and prospects. *Journal of King Saud University-Computer and Information Sciences*, 2021.

[43] Andrew G Howard, Menglong Zhu, Bo Chen, Dmitry Kalenichenko, Weijun Wang, Tobias Weyand, Marco Andreetto, and Hartwig Adam. Mobilenets: Efficient convolutional neural networks for mobile vision applications. *arXiv preprint arXiv:1704.04861*, 2017.

[44] Mark Sandler, Andrew Howard, Menglong Zhu, Andrey Zhmoginov, and Liang-Chieh Chen. Mobilenetv2: Inverted residuals and linear bottlenecks. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 4510–4520, 2018.

[45] Ji Lin, Wei-Ming Chen, Han Cai, Chuang Gan, and Song Han. Mcunetv2: Memory-efficient patch-based inference for tiny deep learning. *arXiv preprint arXiv:2110.15352*, 2021.

[46] He Li, Kaoru Ota, and Mianxiong Dong. Learning iot in edge: Deep learning for the internet of things with edge computing. *IEEE network*, 32(1):96–101, 2018.

[47] Colby Banbury, Chuteng Zhou, Igor Fedorov, Ramon Matas, Urmish Thakker, Dibakar Gope, Vijay Janapa Reddi, Matthew Mattina, and Paul Whatmough. Micronets: Neural network architectures for deploying tinyml applications on commodity microcontrollers. *Proceedings of Machine Learning and Systems*, 3, 2021.

[48] François Chollet. Xception: Deep learning with depthwise separable convolutions. In *Proceedings of the IEEE conference on computer vision and pattern*

recognition, pages 1251–1258, 2017.

[49] Robert David, Jared Duke, Advait Jain, Vijay Janapa Reddi, Nat Jeffries, Jian Li, Nick Kreeger, Ian Nappier, Meghna Natraj, Tiezhen Wang, Pete Warden, and Rocky Rhodes. Tensorflow lite micro: Embedded machine learning for tinyml systems. In A. Smola, A. Dimakis, and I. Stoica, editors, *Proceedings of Machine Learning and Systems*, volume 3, pages 800–811, 2021.

[50] Marcia Sahaya Louis, Zahra Azad, Leila Delshadtehrani, Suyog Gupta, Pete Warden, Vijay Janapa Reddi, and Ajay Joshi. Towards deep learning using tensorflow lite on risc-v. In *Third Workshop on Computer Architecture Research with RISC-V (CARRV)*, volume 1, page 6, 2019.

[51] Yunhui Guo. A survey on methods and theories of quantized neural networks. *arXiv preprint arXiv:1808.04752*, 2018.

[52] Hongkui Jing and Morikuni Takigawa. Low sampling rate induces high correlation dimension on electroencephalograms from healthy subjects. *Psychiatry and clinical neurosciences*, 54(4):407–412, 2000.

[53] Thales Wulfert Cabral, Mahdi Khosravy, Felipe Meneguitti Dias, Henrique Luis Moreira Monteiro, Marcelo Antônio Alves Lima, Leandro Rodrigues Manso Silva, Rayen Naji, and Carlos Augusto Duque. Compressive sensing in medical signal processing and imaging systems. In *Sensors for health monitoring*, pages 69–92. Elsevier, 2019.

[54] Yaakov Tsaig and David L Donoho. Extensions of compressed sensing. *Signal processing*, 86(3):549–571, 2006.

[55] Mahdi Khosravy, Nilanjan Dey, and Carlos A Duque. *Compressive sensing in healthcare*. Academic Press, 2020.

[56] Robert J II Marks. *Introduction to Shannon sampling and interpolation theory*. Springer Science & Business Media, 2012.

[57] Daibashish Gangopadhyay, Emily G Allstot, Anna MR Dixon, Karthik Natarajan, Subhanshu Gupta, and David J Allstot. Compressed sensing analog front-end for bio-sensor applications. *IEEE Journal of Solid-State Circuits*, 49(2):426–438, 2014.

[58] Pervez M Aziz, Henrik V Sorensen, and J Vn der Spiegel. An overview of sigma-delta converters. *IEEE signal processing magazine*, 13(1):61–84, 1996.

[59] David L Donoho. Compressed sensing. *IEEE Transactions on information theory*, 52(4):1289–1306, 2006.

[60] Emmanuel J Candès and Michael B Wakin. An introduction to compressive sampling. *IEEE signal processing magazine*, 25(2):21–30, 2008.

[61] Zhou Wang, Alan C Bovik, Hamid R Sheikh, and Eero P Simoncelli. Image quality assessment: from error visibility to structural similarity. *IEEE transactions on image processing*, 13(4):600–612, 2004.

[62] Shaou-Gang Miaou and Shu-Nien Chao. Wavelet-based lossy-to-lossless ecg compression in a unified vector quantization framework. *IEEE Transactions on Biomedical Engineering*, 52(3):539–543, 2005.

[63] Selin Aviyente. Compressed sensing framework for eeg compression. In *2007 IEEE/SP 14th workshop on statistical signal processing*, pages 181–184. IEEE, 2007.

[64] Fred Chen, Anantha P Chandrakasan, and Vladimir Stojanović. A signal-agnostic compressed sensing acquisition system for wireless and implantable sensors. In *IEEE Custom Integrated Circuits Conference 2010*, pages 1–4. IEEE, 2010.

[65] Muhammad Ali Qureshi and Mohamed Deriche. A new wavelet based efficient image compression algorithm using compressive sensing. *Multimedia Tools and Applications*, 75(12):6737–6754, 2016.

[66] Neural Engineering Data Consortium. *Temple University EEG Dataset*. https://tinyurl.com/38vjv4u3.

[67] Monica Fira, V Maiorescu, and Liviu Goras. The analysis of the specific dictionaries for compressive sensing of eeg signals. In *Proceedings of the Ninth International Conference on Advances in Computer-Human Interactions, Venice, Italy*, pages 24–28, 2016.

[68] L Yang, MD Judd, and CJ Bennoch. Denoising uhf signal for pd detection in transformers based on wavelet technique. In *The 17th Annual Meeting of the IEEE Lasers and Electro-Optics Society, 2004. LEOS 2004.*, pages 166–169. IEEE, 2004.

[69] Angkoon Phinyomark, Chusak Limsakul, and Pornchai Phukpattaranont. Evaluation of mother wavelet based on robust emg feature extraction using wavelet packet transform. In *Proceedings of ANSCSE 13 13th International Annual Symposium on Computational Science and Engineering*, pages 333–339, 2009.

[70] J Rafiee, MA Rafiee, N Prause, and MP Schoen. Wavelet basis functions in biomedical signal processing. *Expert systems with Applications*, 38(5):6190–6201, 2011.

[71] MASK Khan, TS Radwan, and MA Rahman. Wavelet packet transform based protection of three-phase ipm motor. In *2006 IEEE International Symposium on Industrial Electronics*, volume 3, pages 2122–2127. IEEE, 2006.

[72] Wai Keng Ngui, M Salman Leong, Lim Meng Hee, and Ahmed M Abdelrhman. Wavelet analysis: mother wavelet selection methods. In *Applied mechanics and materials*, volume 393, pages 953–958. Trans Tech Publ, 2013.

[73] Marie Farge. Wavelet transforms and their applications to turbulence. *Annual review of fluid mechanics*, 24(1):395–458, 1992.

[74] Jingwei Too, AR Abdullah, Norhashimah Mohd Saad, N Mohd Ali, and H Musa. A detail study of wavelet families for emg pattern recognition. *International Journal of Electrical and Computer Engineering (IJECE)*, 8(6):4221–4229, 2018.

[75] M Sanjeeva Reddy, B Narasimha, E Suresh, and K Subba Rao. Analysis of eog signals using wavelet transform for detecting eye blinks. In *2010 International Conference on Wireless Communications & Signal Processing (WCSP)*, pages 1–4. IEEE, 2010.

[76] Feifei Qi, Wenlong Wang, Xiaofeng Xie, Zhenghui Gu, Zhu Liang Yu, Fei Wang, Yuanqing Li, and Wei Wu. Single-trial eeg classification via orthogonal wavelet decomposition-based feature extraction. *Frontiers in Neuroscience*, 15, 2021.

[77] Noor Kamal Al-Qazzaz, Sawal Hamid Bin Mohd Ali, Siti Anom Ahmad, Mohd Shabiul Islam, and Javier Escudero. Selection of mother wavelet functions for multi-channel eeg signal analysis during a working memory task. *Sensors*, 15(11):29015–29035, 2015.

[78] Zhilin Zhang and Bhaskar D Rao. Extension of sbl algorithms for the recovery of block sparse signals with intra-block correlation. *IEEE Transactions on Signal Processing*, 61(8):2009–2015, 2013.

[79] Michael E Tipping. Sparse bayesian learning and the relevance vector machine. *Journal of machine learning research*, 1(Jun):211–244, 2001.

[80] Marilyn Wolf. *The physics of computing*. Elsevier, 2016.

[81] Nam Sung Kim, Todd Austin, David Baauw, Trevor Mudge, Krisztián Flautner, Jie S Hu, Mary Jane Irwin, Mahmut Kandemir, and Vijaykrishnan Narayanan. Leakage current: Moore's law meets static power. *computer*, 36(12):68–75, 2003.

[82] Etienne Le Sueur and Gernot Heiser. Dynamic voltage and frequency scaling: The laws of diminishing returns. In *Proceedings of the 2010 international conference on Power aware computing and systems*, pages 1–8, 2010.

[83] Amazon. *FreeRTOS - Real-time operating system for microcontrollers*. https://www.freertos.org/index.html.

[84] Suresh Siddha, Venkatesh Pallipadi, and AVD Ven. Getting maximum mileage out of tickless. In *Proceedings of the Linux Symposium*, volume 2, pages 201–207. Citeseer, 2007.

[85] Qiyue Zou, Xiaoxin Zou, Ming Zhang, and Zhiping Lin. A robust speech detection algorithm in a microphone array teleconferencing system. In *2001 IEEE International Conference on Acoustics, Speech, and Signal Processing. Proceedings (Cat. No. 01CH37221)*, volume 5, pages 3025–3028. IEEE, 2001.

[86] S Gökhun Tanyer and Hamza Ozer. Voice activity detection in nonstationary noise. *IEEE Transactions on speech and audio processing*, 8(4):478–482, 2000.

[87] Iyad Obeid and Joseph Picone. The temple university hospital eeg data corpus. *Frontiers in neuroscience*, 10:196, 2016.

[88] Diederik P Kingma and Jimmy Ba. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*, 2014.

[89] Jeremy R Flynn, Steve Ward, Julian Abich, and David Poole. Image quality assessment using the ssim and the just noticeable difference paradigm. In *International Conference on Engineering Psychology and Cognitive Ergonomics*, pages 23–30. Springer, 2013.

[90] Richard G Baraniuk, Volkan Cevher, Marco F Duarte, and Chinmay Hegde. Model-based compressive sensing. *IEEE Transactions on information theory*, 56(4):1982–2001, 2010.

[91] Daibashish Gangopadhyay, Emily G Allstot, Anna MR Dixon, and David J Allstot. System considerations for the compressive sampling of eeg and ecog bio-signals. In *2011 IEEE Biomedical Circuits and Systems Conference (BioCAS)*, pages 129–132. IEEE, 2011.

[92] Zhilin Zhang, Tzyy-Ping Jung, Scott Makeig, and Bhaskar D Rao. Compressed sensing of eeg for wireless telemonitoring with low energy consumption and inexpensive hardware. *IEEE Transactions on Biomedical Engineering*, 60(1):221–224, 2012.

[93] Arnaud Delorme and Scott Makeig. Eeglab: an open source toolbox for analysis of single-trial eeg dynamics including independent component analysis. *Journal of neuroscience methods*, 134(1):9–21, 2004.

[94] Elisa Bruno, Pedro F Viana, Michael R Sperling, and Mark P Richardson. Seizure detection at home: Do devices on the market match the needs of people living with epilepsy and their caregivers? *Epilepsia*, 61:S11–S24, 2020.

[95] Steven C. Schachter. *Diagnosing Epilepsy*. https://tinyurl.com/yt6ncpse.

[96] Ali Emami, Naoto Kunii, Takeshi Matsuo, Takashi Shinozaki, Kensuke Kawai, and Hirokazu Takahashi. Seizure detection by convolutional neural network-based analysis of scalp electroencephalography plot images. *NeuroImage: Clinical*, 22:101684, 2019.

[97] Afshin Shoeibi, Marjane Khodatars, Navid Ghassemi, Mahboobeh Jafari, Parisa Moridian, Roohallah Alizadehsani, Maryam Panahiazar, Fahime Khozeimeh, Assef Zare, Hossein Hosseini-Nejad, et al. Epileptic seizures detection using deep learning techniques: a review. *International Journal of Environmental Research and Public Health*, 18(11):5780, 2021.

[98] Hongyu Miao and Felix Xiaozhu Lin. Enabling large neural networks on tiny microcontrollers with swapping. *arXiv preprint arXiv:2101.08744*, 2021.

[99] Michael R Sperling. Sudden unexplained death in epilepsy. *Epilepsy currents*, 1(1):21–23, 2001.

[100] Xiaying Wang, Michele Magno, Lukas Cavigelli, and Luca Benini. Fann-on-mcu: An open-source toolkit for energy-efficient neural network inference at the edge of the internet of things. *IEEE Internet of Things Journal*, 7(5):4403–4417, 2020.

[101] Young D. Kwon, Jagmohan Chauhan, and Cecilia Mascolo. Yono: Modeling multiple heterogeneous neural networks on microcontrollers. In *Proceedings of the 21th International Conference on Information Processing in Sensor Networks*, IPSN '22, 2022.

[102] Dharmendra Gurve, Denis Delisle-Rodriguez, Teodiano Bastos-Filho, and Sridhar Krishnan. Trends in compressive sensing for eeg signal processing applications. *Sensors (Switzerland)*, 20:1–21, 2020.

[103] Khalid Abualsaud, Massudi Mahmuddin, Ramy Hussein, and Amr Mohamed. Performance evaluation for compression-accuracy trade-off using compressive sensing for eeg-based epileptic seizure detection in wireless tele-monitoring. In *2013 9th International Wireless Communications and Mobile Computing Conference (IWCMC)*, pages 231–236. IEEE, 2013.

[104] Mahdi Khosravy, Naoko Nitta, Kazuaki Nakamura, and Noboru Babaguchi. Chapter 1 - compressive sensing theoretical foundations in a nutshell, 2020.

[105] José Solaz, José Laparra-Hernández, Daniel Bande, Noelia Rodríguez, Sergio Veleff, José Gerpe, and Enrique Medina. Drowsiness detection based on the analysis of breathing rate obtained from real-time image recognition. *Transportation research procedia*, 14:3867–3876, 2016.

[106] Arnaud Sors, Stéphane Bonnet, et al. A convolutional neural network for sleep stage scoring from raw single-channel eeg. *Biomedical Signal Processing and Control*, 42:107–114, 2018.

[107] Muhammad Zahak Jamal. Signal acquisition using surface emg and circuit design considerations for robotic prosthesis. *Computational Intelligence in Electromyography Analysis-A Perspective on Current Applications and Future Challenges*, 18:427–448, 2012.

[108] Khalid Abualsaud, Massudi Mahmuddin, Mohammad Saleh, and Amr Mohamed. Ensemble classifier for epileptic seizure detection for imperfect eeg data. *Scientific World Journal*, 2015, 2015.

[109] Luisa F. Polania, Rafael E. Carrillo, Manuel Blanco-Velasco, and Kenneth E. Barner. Compressed sensing based method for ecg compression. pages 761–764, 2011.

[110] A. Singh, L. N. Sharma, and S. Dandapat. Multi-channel ecg data compression using compressed sensing in eigenspace. *Computers in Biology and Medicine*, 73:24–37, 6 2016.

[111] Mir Mohsina and Angshul Majumdar. Gabor based analysis prior formulation for eeg signal reconstruction. *Biomedical Signal Processing and Control*, 8(6):951–955, 2013.

[112] Phuong Thi Dao, Anthony Griffin, and Xue Jun Li. Compressed sensing of eeg with gabor dictionary: Effect of time and frequency resolution. In *2018 40th annual international conference of the IEEE engineering in medicine and biology society (EMBC)*, pages 3108–3111. IEEE, 2018.

[113] Robert Simon Sherratt and Nilanjan Dey. Low-power wearable healthcare sensors. *Electronics*, 9(6), 2020.

[114] Toygun Basaklar, Yigit Tuncel, Sizhe An, and Umit Ogras. Wearable devices and low-power design for smart health applications: Challenges and opportunities. In *2021 IEEE/ACM International Symposium on Low Power Electronics and Design (ISLPED)*, pages 1–1, 2021.

[115] Diksha Thakur, Kulbhushan Sharma, and Rajnish Sharma. Ultra low-power low-pass filter design for wearable biomedical applications. In *2021 Devices for Integrated Circuit (DevIC)*, pages 629–632, 2021.

[116] Akira Takeda, Akira Yokosawa, Shintaro Sano, Shunsuke Sasaki, Takeshi Kodaka, Takahiro Tokuyoshi, and Toshiki Kizu. A novel energy-efficient data acquisition method for wearable devices. In *2015 IEEE Symposium in Low-Power and High-Speed Chips (COOL CHIPS XVIII)*, pages 1–3, 2015.

[117] Josiah Hester, Travis Peters, Tianlong Yun, Ronald Peterson, Joseph Skinner, Bhargav Golla, Kevin Storer, Steven Hearndon, Kevin Freeman, Sarah Lord, et al. Amulet: An energy-efficient, multi-application wearable platform. In *Proceedings of the 14th ACM Conference on Embedded Network Sensor Systems CD-ROM*, pages 216–229, 2016.

[118] Chin-Teng Lin, Chun-Hsiang Chuang, Chih-Sheng Huang, Shu-Fang Tsai, Shao-Wei Lu, Yen-Hsuan Chen, and Li-Wei Ko. Wireless and wearable eeg system for evaluating driver vigilance. *IEEE Transactions on biomedical circuits and systems*, 8(2):165–176, 2014.

[119] Elise Saoutieff, Tiziana Polichetti, Laurent Jouanet, Adrien Faucon, Audrey Vidal, Alexandre Pereira, Sébastien Boisseau, Thomas Ernst, Maria Lucia Miglietta, Brigida Alfano, et al. A wearable low-power sensing platform for environmental and health monitoring: The convergence project. *Sensors*, 21(5):1802, 2021.

[120] Emmanuel Baccelli, Cenk Gündoğan, Oliver Hahm, Peter Kietzmann, Martine S Lenders, Hauke Petersen, Kaspar Schleiser, Thomas C Schmidt, and Matthias Wählisch. Riot: An open source operating system for low-end embedded devices in the iot. *IEEE Internet of Things Journal*, 5(6):4428–4440, 2018.

[121] Philip Levis, Samuel Madden, Joseph Polastre, Robert Szewczyk, Kamin Whitehouse, Alec Woo, David Gay, Jason Hill, Matt Welsh, Eric Brewer, et al. Tinyos: An operating system for sensor networks. In *Ambient intelligence*, pages 115–148. Springer, 2005.

[122] *Emotiv brainwear.* https://goo.gl/uagGNX.

[123] *Muse.* https://goo.gl/5zwtcJ.

[124] *NeuroSky MindWave.* https://goo.gl/cEf7fi.

[125] Jiawei Xu, Srinjoy Mitra, Chris Van Hoof, et al. Active electrodes for wearable eeg acquisition: Review and electronics design methodology. *IEEE reviews in biomedical engineering*, 10:187–198, 2017.

[126] Nhat Pham, Tuan Dinh, Zohreh Raghebi, Taeho Kim, Nam Bui, Phuc Nguyen, Hoang Truong, Farnoush Banaei-Kashani, Ann Halbower, Thang Dinh, et al. Wake: a behind-the-ear wearable system for microsleep detection. In *Proceedings of the 18th International Conference on Mobile Systems, Applications, and Services*, pages 404–418, 2020.