A Low-Power Processor With Configurable Embedded Machine-Learning Accelerators for High-Order and Adaptive Analysis of Medical-Sensor Signals

Kyong Ho Lee, *Student Member, IEEE*, and Naveen Verma, *Member, IEEE*

***Abstract*—Low-power sensing technologies have emerged for acquiring physiologically indicative patient signals. However, to enable devices with high clinical value, a critical requirement is the ability to analyze the signals to extract specific medical information. Yet given the complexities of the underlying processes, signal analysis poses numerous challenges. Data-driven methods based on machine learning offer distinct solutions, but unfortunately the computations are not well supported by traditional DSP. This paper presents a custom processor that integrates a CPU with configurable accelerators for discriminative machine-learning functions. A support-vector-machine accelerator realizes various classification algorithms as well as various kernel functions and kernel formulations, enabling range of points within an accuracy-versus-energy and -memory trade space. An accelerator for embedded active learning enables prospective adaptation of the signal models by utilizing sensed data for patient-specific customization, while minimizing the effort from human experts. The prototype is implemented in 130-nm CMOS and operates from 1.2 V–0.55 V (0.7 V for SRAMs). Medical applications for EEG-based seizure detection and ECG-based cardiac-arrhythmia detection are demonstrated using clinical data, while consuming 273 μJ and 124 μJ per detection, respectively; this represents 62.4× and 144.7× energy reduction compared to an implementation based on the CPU. A patient-adaptive cardiac-arrhythmia detector is also demonstrated, reducing the analysis-effort required for model customization by 20×.**

***Index Terms*—Active learning (subject-specific adaptation), biomedical electronics, machine learning (artificial intelligence), medical signal processing, support vector machine (SVM).**

I. Introduction

UNPRECEDENTED technologies have recently emerged that make it possible to sense [1], [2] and acquire [3]–[5] physiologically indicative patient signals within low-power and small-scale devices. Although the signals, such as electrocardiograms (ECGs), electroencephalograms (EEGs), etc., are relevant in a wide range of clinical applications, high-value medical devices require extracting specific medical information from these. The devices envisioned include closed-loop therapeutic systems, prosthetic systems, sensors for continuous medical decision support, etc. [6].

The challenge is that the physiological signals available through low-power sensors are extremely complex to analyze. Fig. 1 illustrates the challenges. The first issue is that the specific states of interest must be detected in the presence of numerous background physiologic variances; for adequate

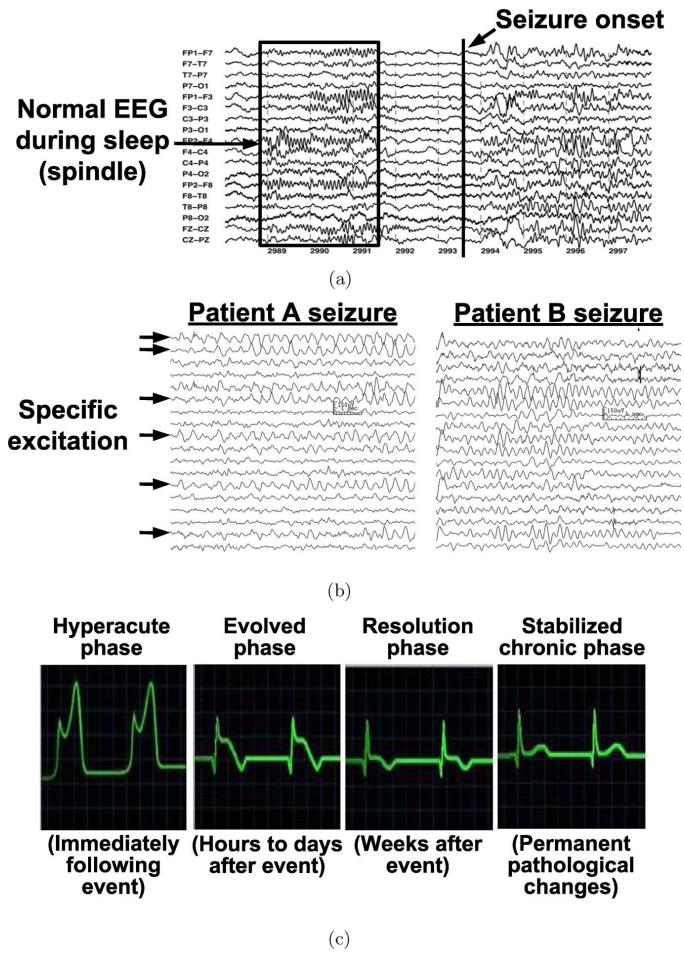


Fig. 1. Challenges with physiological signal analysis. (a) High-order models are required to distinguish targeted physiological states from background activity. (b) The manifestation of targeted states are different from patient-to-patient. (c) Physiological changes over time (particularly following acute events) result in signal dynamics.

Manuscript received November 30, 2012; revised February 24, 2013; accepted February 25, 2013. Date of publication April 03, 2013; date of current version June 21, 2013. This paper was approved by Guest Editors Antonio Liscidini and Doug Smith. This work was supported in part by the SRC, NSF, and a Qualcomm Innovation Fellowship.

The authors are with the Electrical Engineering Department, Princeton University, Princeton, NJ 08544 USA (e-mail: kyonglee@princeton.edu, nverma@princeton.edu).

Color versions of one or more of the figures in this paper are available online at http://ieeexplore.ieee.org.

Digital Object Identifier 10.1109/JSSC.2013.2253226

specificity, this implies the need for high-order signal models. Fig. 1(a) shows EEG signals expressing the onset of a seizure; but, also shown is normal EEG associated with sleep (known as spindle). The two bursts resemble each other, requiring that the physical waveforms be adequately modeled in order to distinguish between them. While analytical representations are emerging (e.g., that attempt to model seizure dynamics from the neuronal-circuit level [7]), these generally do not model the physical waveforms to the high level required for clinical applications. The second challenge is that, across clinical applications, the expression of the targeted states within the signals can be highly variable from patient to patient [8]. As an example, Fig. 1(b) shows the seizure EEG from two patients. The two cases illustrate distinct excitations, both spectrally and spatially over the channels, implying the need for patient-customized models. The third challenge is that patient physiology changes over time, particularly in response to acute events. Fig. 1(c) shows the ECG of a patient following a myocardial infarction (heart attack); over a period of weeks to months, tissue in the heart muscles evolves, causing changes in the ECG before settling with permanent pathological characteristics [9]. This implies that signal models may also need to track these dynamics for use in chronic devices.

Data-driven approaches can be used to create models based on physical sensor data rather than analytical methods. In particular, powerful frameworks for data-driven and adaptive modeling have emerged from the domain of machine learning. These raise capabilities for overcoming the challenges mentioned above. The critical issue, however, is that with little prior consideration for enabling these methods in low-power systems, the computations involved are energy intensive. While low-power medical processors have recently been reported, they have either focused on traditional DSP [10], [11], or have not enabled the programmability required across applications and/or across patient cases while also supporting high-order modeling and analysis frameworks [12]–[17]. We, thus, present a flexible general-purpose processor *specialized for high-order machine-learning signal-analysis functions* for a wide range of clinical applications. The processor includes a CPU for programmable computations as well as configurable machine-learning accelerators to support various kernels (details are described in Section III). The design features the following advances:

- A flexible accelerator is integrated with a general-purpose CPU. The CPU enables programmable feature extraction for a range of physiological signals and clinical applications, and the accelerator, through hardware configurability, enables a large space of algorithmic-performance versus energy- and memory-usage tradeoffs.
- Dynamic model adaptations are enabled in the background through a hardware accelerator for embedded active learning. This enables scalable patient-model customization by minimizing the burden on human experts. An in-place algorithm for active learning enables hardware sharing for kernel functions by minimizing the overhead of context switches in the presence of real-time detection.
- Medical applications and algorithms are developed and demonstrated (using clinical datasets), showing the use of the processor for continuous, patient-adaptive monitoring within a network (via a Bluetooth radio interface).

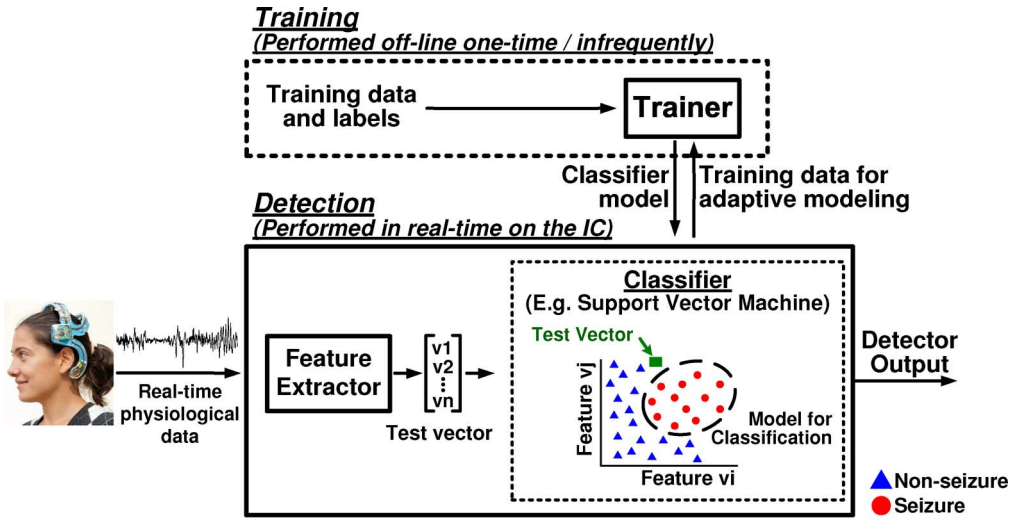
The paper is structured as follows. Section II analyzes the structure and energy of machine-learning algorithms. We focus primarily on data-driven classification in order to identify an architecture that exploits hardware specialization to addresses the energy-efficiency and computational-flexibility needs. Section III discusses the microarchitecture and circuit details of the processor, providing extended details of the design in [18]. Section IV presents the measurement results as well as application demonstrations using real medical data. Finally, Section V provides conclusions.

II. Analysis for a Hardware-Specialized Processor

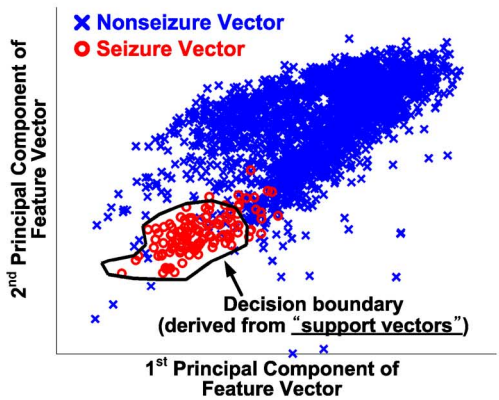
Given the range of machine-learning frameworks for modeling and analysis, this section considers their suitability in medical applications. Specifically, the issues of constructing models from medical data and enabling algorithms for clinical applications are considered. This motivates our focus on discriminative frameworks. This section then analyzes the energy and computational programmability required in discriminative algorithms. As a kernel, we focus on support vector machines (SVMs) [19] which are a data-driven classification framework. We notice that the algorithms offer structure that substantially separates the need for energy efficiency from the need for computational flexibility; this suggests an accelerator-based architecture for processor specialization. Given the importance of adapting the models for patient-specific and temporal variabilities, the related challenges are described, motivating the need for hardware support of embedded active learning in the classification algorithms.

A. Discriminative versus Generative Frameworks

Broadly, two categories of machine-learning frameworks exist: discriminative and generative. The key difference is that, by observing a signal, discriminative frameworks attempt to model *a specific target variable* associated with a process, whereas generative frameworks attempt to model *the underlying process more broadly*, potentially making numerous variables accessible within an algorithm. The challenge with generative frameworks is that substantial training data can be needed to create adequate models [20]; this is of particular concern when models are being dynamically constructed and rapid convergence is desired (as in the clinical applications considered in this work). Limited medical data, especially for rare pathophysiologic events, thus makes embedded generative models potentially less viable. On the other hand, the focus on specific variables makes algorithms using and constructing discriminative models potentially more robust. In fact, many medical devices being considered today are interested in decoding targeted states for specific actuation-control and prosthetic functions [6]; these can be well addressed by discriminative frameworks. This work thus focuses on enabling various discriminative algorithms along with algorithms for adaptive model construction.



(a)



(b)

Fig. 2. (a) SVM framework in an EEG-based seizure-detection example. The trainer generates a classifier model from previous observations. Real-time detection occurs in two steps: feature extraction and classification. (b) SVMs form a decision boundary from support vectors, which are sampled from the edge of the data distributions (only the principal components are shown).

B. Energy Analysis

The SVM is a supervised machine-learning framework for discriminative classification that has gained popularity due to its computational efficiency and robustness [19]. Supervised machine-learning implies that a human expert must be involved during a training phase to provide classification labels for training data. The data and labels are then used to construct a classification model. The involvement of a human expert is the typical protocol accepted for training medical devices [8], [21], [22]. Fig. 2(a) shows the two aspects involved, namely infrequent training and real-time detection. Detection occurs continuously on the sensed data, making its energy the primary concern. The processor presented in this work thus focuses primarily on detection, as well as continuous sensor-data acquisition for offline model adaptations. In a typical medical algorithm, detection can be further divided into two aspects: feature extraction and classification. Feature extraction is an important step that improves detection by explicitly representing the input signal by the biomarkers of the medical variable of interest [20]. It is thus closely tied to the application and the medical signals involved. Fig. 2(b) shows the details of SVM classification. Labeled feature vectors derived from the training data are plotted (in the figure, dimensionality reduction via principal component analysis is employed only to enable visualization). The SVM selects a reduced number of training feature vectors, called the *support vectors*, from the edges of the class-data distributions in order to represent an optimal decision boundary for classification (i.e., that maximizes the geometric distance to the support vectors). The corresponding classification function is then given by the following (where $\vec{x}$ is the real-time feature vector being classified, $\vec{sv}_i$ is a support vector, K is a kernel transformation used to enhance the decision-boundary flexibility, and α and b are modeling parameters):

$$f(x) = \sum_{i=1}^{N} \alpha_i K(\vec{sv}_i, \vec{x}) - b. \quad (1)$$

In order to direct our hardware specialization efforts, the real-time execution energy of several medical applications has been profiled. The typical result is represented by the measurements shown in Fig. 3, which correspond to an ECG-based cardiac-arrhythmia detection application (profiled on an MSP430 processor). The challenge is that,

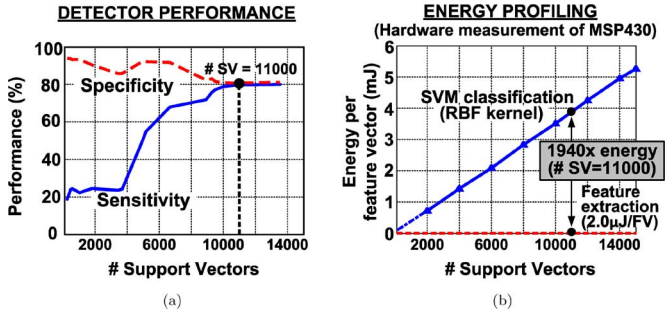


Fig. 3. Performance and energy of representative machine-learning algorithm (ECG-based cardiac arrhythmia detection using real medical data from [34]). (a) High-order models are required to achieve high accuracy, but for strong classification kernels (i.e., RBF kernel is used in example shown). (b) The classification energy scales with the model complexity (note, while the performance appears to saturate at $\#\mathrm{SV} = 11000$ for the data shown, additional support vectors may be selected by the training algorithm, which optimizes an objective function to derive the decision boundary).

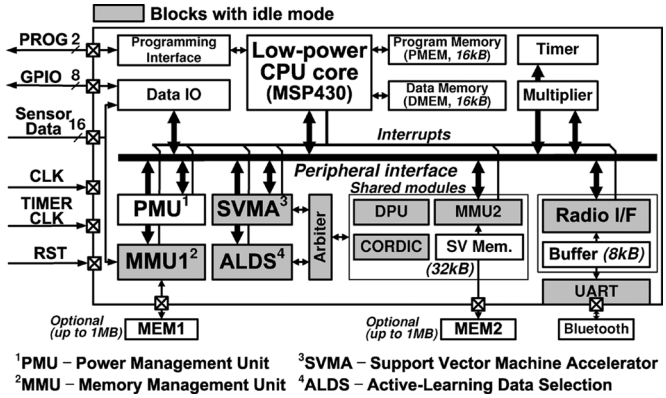


Fig. 4. Processor architecture with machine-learning accelerators.

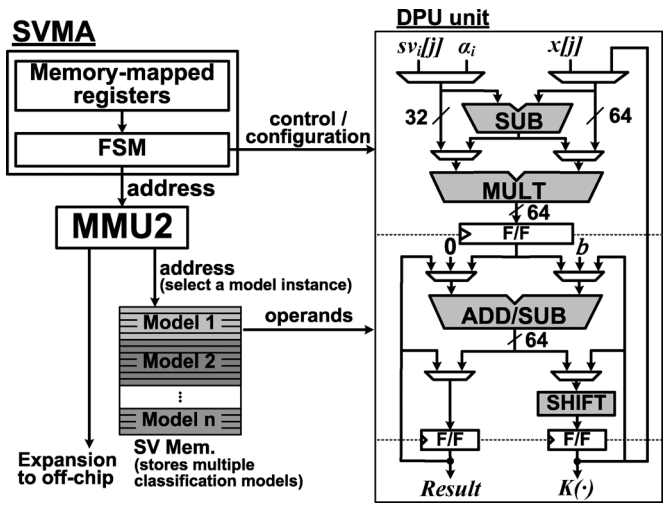


Fig. 5. SVMA is configurable via memory-mapped registers to enable various classifier structures and kernels. The FSM controls the configurable data path unit (DPU) to realize the computations.

although the data-driven classification framework enables construction of high-order models, the energy to apply those models dominates for strong classifiers. Fig. 3(a) shows that a large number of support vectors is required to yield high performance [sensitivity represents the true-positive rate (i.e., $true\_positive/(true\_positive + false\_negative)$) while specificity represents the true-negative rate (i.e., $true\_negative/(true\_negative + false\_positive)$)]; however, Fig. 3(b) shows that, using a strong classification kernel (a radial-basis function (RBF) kernel is used in the example shown), energy scales proportionally, making it dominate by orders of magnitude over feature extraction (a similar result is observed for EEG-based seizure detection, with the energy also dominating over instrumentation and digitization [23]). A key insight, however, is that feature-extraction computations require flexibility, since they are closely tied to the application, as discussed above. On the other hand classification with high-order models requires computational efficiency, but the computations can be substantially fixed due to the kernels involved (Section III-B discusses important configurability requirements for classification). The separation of flexibility and efficiency requirements implies that an accelerator-based architecture can significantly reduce energy while enabling a broad range of clinical applications.

C. Active Learning for Model Adaptation

A key benefit of an efficient data-driven modeling framework for medical sensor applications is that with patient-specific data being acquired continuously by the sensors, patient customization of the embedded models is possible. The challenge is that in a supervised-learning framework, patient-by-patient customization requires effort from experts, limiting its scalability. Active learning is an approach wherein the most informative instances are chosen from a training pool to reduce the labeling effort [24], [25]. Active learning has recently shown to enable efficient model customization with minimal labeling effort from human experts in clinical applications including arrhythmia detection [26] and seizure detection [27]. In the processor presented, we incorporate support for a modified active-learning framework wherein data can instead be continuously assessed *as it is being sensed* to downselect to a highly reduced set of optimal instances which can then be sent to clinical experts to enable scalable customization over a network. Section III-C describes the hardware support for this.

III. Processor Architecture and Circuits

Fig. 4 shows an architecture block diagram of the processor. While the accelerators address the energy-intensive modeling and classification computations, the CPU provides top-level configuration (of the accelerators) as well as programmable feature-extraction computations. This enables support over a wide range of applications. The machine-learning accelerators include a support vector machine accelerator (SVMA) and an active-learning data selection (ALDS) accelerator. The SVMA can be configured for various classification algorithms via various energy-scalable kernels. The ALDS enables background support for embedded active learning of signal models by optimizing interactions with remote clinical experts. Both the SVMA and the ALDS require computations over vectors in a kernel-transformed space (as described below). They thus share a specialized data-path unit (DPU) and CORDIC engine; an arbiter controls access to these modules during real-time operation. Aside from computational blocks, the architecture includes a radio interface (with 8-kB local buffer) to enable control of an off-chip Bluetooth module that establishes a

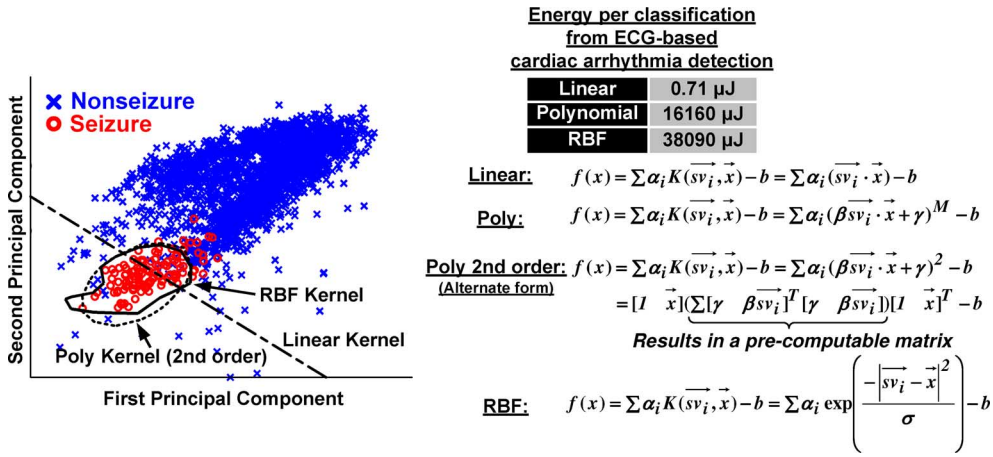


Fig. 6. The choice of specific kernel function strongly impacts the classification energy and memory requirements. For the polynomial kernel, the alternate formulation, which achieves linearization by transforming vectors to matrices, is also shown [23].

communication link with gateway devices (e.g., smart phone), so that the monitoring and model-adaptation processes can occur interactively with clinical experts over a wide-area network. The architecture also includes a power-management unit (PMU) that enables idle-mode clock gating of all shaded blocks through software, as well as memory-management units (MMU1, MMU2) that enable background control for continuous data logging and updates to the adaptive models. The subsections below describe the roles and details of the various blocks.

A. Programmable CPU

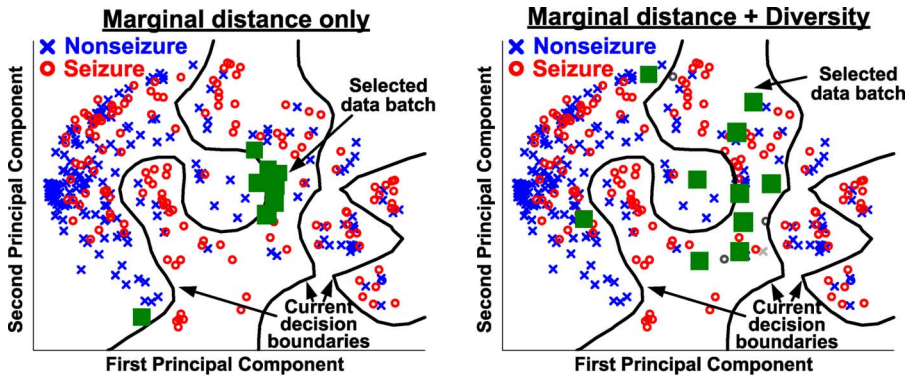
The CPU is based on an MSP430-compatible instruction set and has 16 kB of program memory and 16 kB of data memory. User programs are loaded to the program memory (PMEM) through a dedicated programming interface, and are executed upon upload. The peripherals, including all accelerators, are accessed through a specialized peripherals interface via memory-mapped status and configuration registers. The eight general-purpose I/O (GPIO) are treated as a peripheral and serve as the interface to sensor data in a typical application. A peripheral timer and multiplier unit are included to facilitate the programmable feature-extraction computations.

B. Configurable Support Vector Machine Accelerator (SVMA)

Fig. 5 shows the modules involved in computing SVM classification. The SVMA is a finite state machine that provides operands and model data to the DPU (by controlling the SV memory interface). Kernel functions, thus computed, can be configured via memory-mapped control registers. The SVMA enables programmable partitioning of the local 32-kB SV memory to instantiate multiple classifiers with various kernel functions (as described below). The classifier instances can also be combined in structured ways to realize various multi-class algorithms [28] and ensemble classifiers (e.g., classifier trees and adaptive boosting, wherein multiple weak classifiers are combined to form a strong classifier [29]). As an example, multiple models can be loaded into the SV memory, and, through memory-mapped control registers, the user program can initiate classifier-kernel computations corresponding to each model. The computational outputs from each classifier can then be retrieved from memory-mapped data registers and combined within the user program to synthesize the final decision function and algorithm.

An important aspect with data-driven methods is that the performance of various kernel functions depends strongly on the characteristics of the application data. This has critical implications since the choice of kernel functions substantially affects the computational energy and memory requirements. It is thus necessary to enable SVMA configurability both over the kernel functions, and over different formulations for some kernel functions, so that various points within a performance-versus-energy/-memory space are supported as preferred across different applications. As an example, Fig. 6 considers a seizure-detection application, showing the effective decision boundary realized by SVM kernels involving three different transformation functions K [linear, polynomial, radial-basis function (RBF)]. While the kernels yield increasing levels of strength (i.e., flexibility), the actual strength required depends on how the data is distributed in the feature space; for instance, in the case shown, the polynomial kernel yields comparable separability to the RBF kernel (and in fact linear kernels are found to be sufficient for some cases of patient data [23]). However, as shown by the measured energy for each kernel (from simulation of the CPU), the required energy varies by orders of magnitude.

Similarly, alternate formulations of the kernel functions are also beneficial for enabling different energy trade-offs. In [23], a formulation is presented for polynomial kernels that explicitly overcomes energy scaling with respect to the number of support vectors. This is achieved by transforming feature vectors into matrices in order to linearize quadratic polynomials, thus enabling a factorization that permits precomputation over all the support vectors. Though substantial energy savings are demonstrated, particularly for medical applications, the transformation from vectors to matrices exacerbates energy scaling with respect to the feature-vector dimensionality [23]. Consequently, the optimal choice of formulation once again depends on the application characteristics, further underscoring the need for configurability in the SVMA. Table I shows simulation results of the SVMA (for two representative applications), sampling the scalability range in performance versus cycle count and memory



(a) (b)

Fig. 7. Batch data (a) without diversity metric, leading to clustered selections, and (b) with diversity metric, leading to coverage over large regions of the feature space.

TABLE I
SIMULATED SVMA CYCLE-COUNT/MEMORY REQUIREMENTS AND PERFORMANCE BASED ON REAL PATIENT DATA (FROM [34]), SHOWING THE SCALABILITY ACHIEVED

Seizure detection†					
Kernel	Performance			Cycle count (kcycles)	SV memory required (kB)
	Sensitivity	Latency	False Alarm		
RBF	100%	4.8 sec	1.2 /day	29.6	16.5
Poly (2nd order)	100%	4.4 sec	2.4 /day	24.9	16.2
Alternate Poly				9.7	9.4
Linear	100%	15.0 sec	18.0 /day	0.20	0.19
Arrhythmia detection††					
Kernel	Performance			Cycle count (kcycles)	SV memory required (kB)
	True Pos.	True Neg.	False Pos.		
RBF	75.9%	90.3%	25.4%	1341	640.0
Poly (2nd order)	74.6%	89.0%	28.1%	1684	1094.4
Alternate Poly				1.8	1.9
Linear	57.1%	90.6%	30.4%	0.09	0.08

† Support vectors required for RBF, poly and linear kernels are 169, 166 and 72, respectively.
†† Support vectors required for RBF, poly and linear kernels are 14246, 24363 and 14246, respectively.
Sensitivity: (# seizures detected) / (# total seizures) x 100
Latency: average delay of detector after electrographic onset
False Alarm: # false alarms in a day
Derived from [8]

that is enabled by incorporating configurability in the kernel functions and kernel-function formulations (for polynomial kernels). The amount of support-vector memory required is also reported to highlight the memory-usage dependence on the kernels. The DPU circuit required to support this range of configurability is discussed in Section III-D.

C. Active-Learning Data Selection (ALDS) Accelerator

Active learning involves selecting the optimal data instances in a training pool in order to reduce the analysis and labeling effort required by an expert during supervised model construction. For medical-sensor applications, we modify the concept to improve the scalability of model customization during dynamic model adaptation. Our approach enables algorithms wherein a seed model constructed offline from population-level patient-generic data is initially used by the processor. The processor then assess the sensed data to choose the optimal instances to send to clinical experts [26]. In this process, a pool size is specified by a user-defined epoch of sensed data, and the batch size, which we aim to make much smaller, is defined as the number of selected instances from within the epoch. The batch data is transmitted to a clinical expert, who assigns training labels to construct a refined model. This is then sent back to be uploaded onto the device. This process occurs over the network through the use of a dedicated hardware interface to an off-chip Bluetooth radio (which can communicate with gateway devices for wide-area-network connectivity). The process then iterates to achieve a desired level of model convergence.

The key aspect of the embedded active-learning approach is continuous assessment of the sensed data. The processor incorporates hardware support to compute two primary metrics for data assessment; due to the large number of CPU cycles that these would require (see below), an accelerator is integrated for background computation. The two metrics are the marginal distance (m) and the diversity (d) of the data instances. Since in an SVM, feature vectors near the decision boundary are most likely to form the support vectors, the marginal distance is used to represent the proximity of data to the current decision boundary (in the kernel-transformed feature space). This is equivalent to the magnitude of the SVM decision function; in algorithms where classification is simultaneously being performed on the sensed data, this metric is readily available. In addition to a marginal-distance metric, the diversity metric has recently been shown to substantially improve model convergence [30]. It aims to choose batch data that explicitly covers large regions of the feature space. For illustration, Fig. 7 considers data from a seizure-detection application, showing the initial SVM decision boundary as well as the distribution of pool data and batch data, selected using an m-only [Fig. 7(a)] as well as an m-and-d [Fig. 7(b)] criterion.

The challenge with the diversity metric is that pool data under consideration must be assessed with respect to other instance in the selected batch. The number of compute cycles required thus scales with both the pool and batch sizes, which can be very high if large pools are desired (to maximize data reduction) and large batches are desired (to amortize communication and analysis/labeling overheads). Fig. 8(a) shows simulation results measuring the number of compute cycles required if the CPU were used. This motivates the dedicated ALDS accelerator to enable background computations for active learning. Fig. 8(b) shows the computational flow of the ALDS. While m comes directly from the SVM computation, d is computed iteratively over the data in the pool and the selected batch data. d requires the kernel-transformed computation shown in Fig. 8(b). The overall score for selecting a data instance is then derived by combining m and

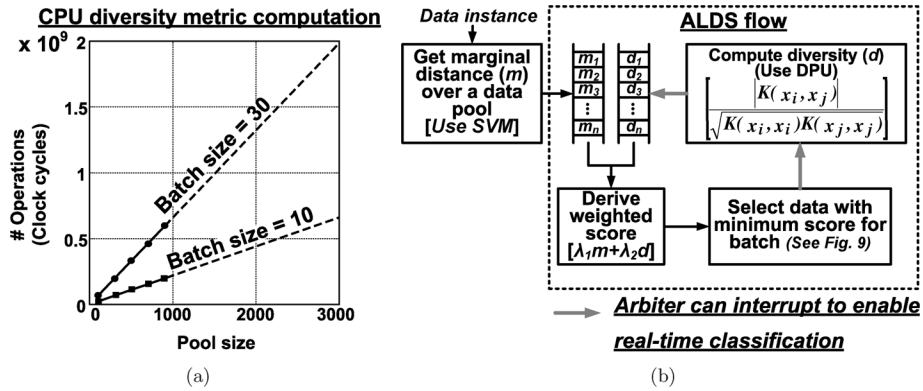


Fig. 8. (a) Simulation of batch-selection algorithm on CPU, illustrating the need for background computation via an accelerator, and (b) ALDS accelerator computation flow.

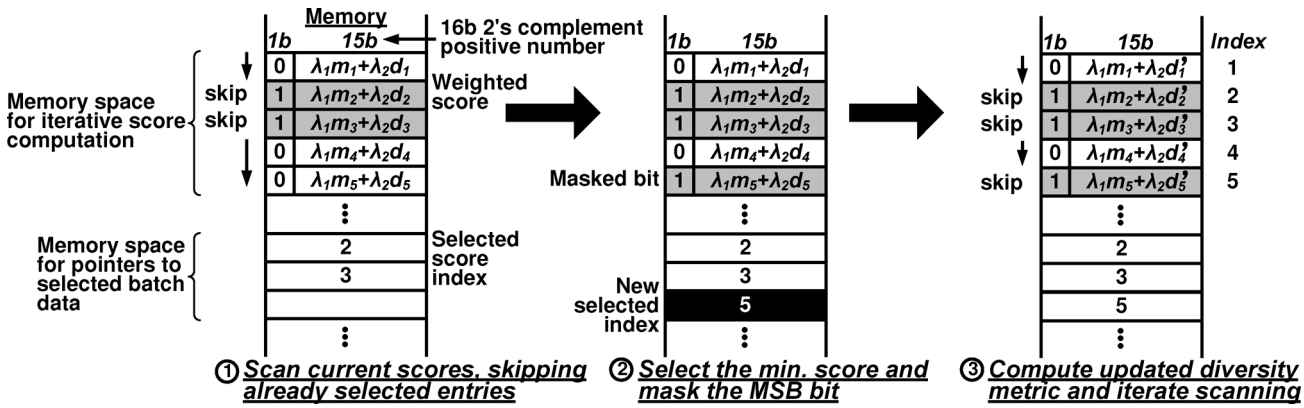


Fig. 9. Weighted metrics are stored as 15 b, and the MSB bit is used to indicate previously selected data points to be skipped in subsequent iterations.

d using programmable weights (λ_1 and λ_2). After computing the scores associated with data instances, the ALDS scans the memory space to find the minimum score, thus selecting an instance for batch data. The ALDS uses the most-recently selected data instance (feature vector $\vec{x}_j$) to calculate d for every remaining instance in the pool ($\vec{x}_i$).

Details of the implementation are shown in Fig. 9. The ALDS calculates weighted scores and stores them in the SV memory. The in-place implementation exploits the fact that the computed metrics (m and d), weights (λ_1 and λ_2), and, therefore, the combined scores are all positive. The reduced precision thus required allows the MSB of the corresponding memory location to be used as a marker to indicate selected batch instances. The marked instances are then omitted by the ALDS during subsequent iterations where further instances are selected. Following each iteration, the selected instance is marked by its MSB, and its index is written to memory space dedicated for pointers to the selected batch data.

Since the DPU and CORDIC (described below) are specialized for the kernel computations and transformations required over feature vectors, they are reused by the SVMA and ALDS (i.e., for computing d). However, since ALDS computations occur in the background while SVMA computations are required for real-time sensor-data analysis, the ALDS computation flow in Fig. 8(b) can be interrupted (under the control of the arbiter in Fig. 4).

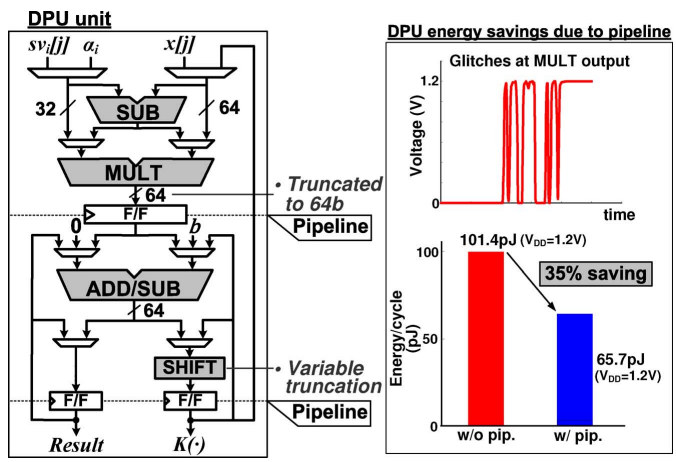


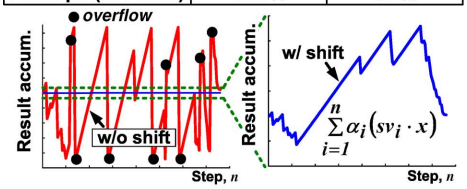
Fig. 10. DPU employs two-stage pipeline to reduce glitch propagation, simulations show the energy benefit of the pipeline architecture.

D. Data-Path Unit (DPU)

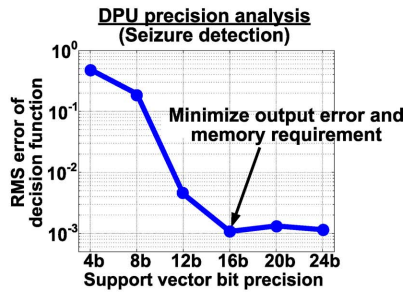
Fig. 10 shows the details of the DPU. The DPU performs the arithmetic required over feature vectors for the SVMA and ALDS modules. The element-wise operands as well as SVM-model parameters ($\vec{sv}_i$, α_i, and b) are applied in a specialized two-stage pipeline. Flip-flop insertion in the pipeline mitigates active-glitching and leakage energy (i.e., by reducing the critical-path delay [31]). The results from transistor-level simula-

DPU dynamic range scaling due to shifting

Classification performance of seizure detection		
	True Pos.	True Neg.
Floating pt.	93.8%	100%
Fixed pt. (w/ shift)	93.8%	100%
Fixed pt. (w/o shift)	68.8%	8.2%



(a)



(b)

Fig. 11. (a) In-line truncation of shifting prevents overflow yielding comparable performance to a floating-point implementation. (b) 16-b support-vector precision minimizes memory requirements and classification error.

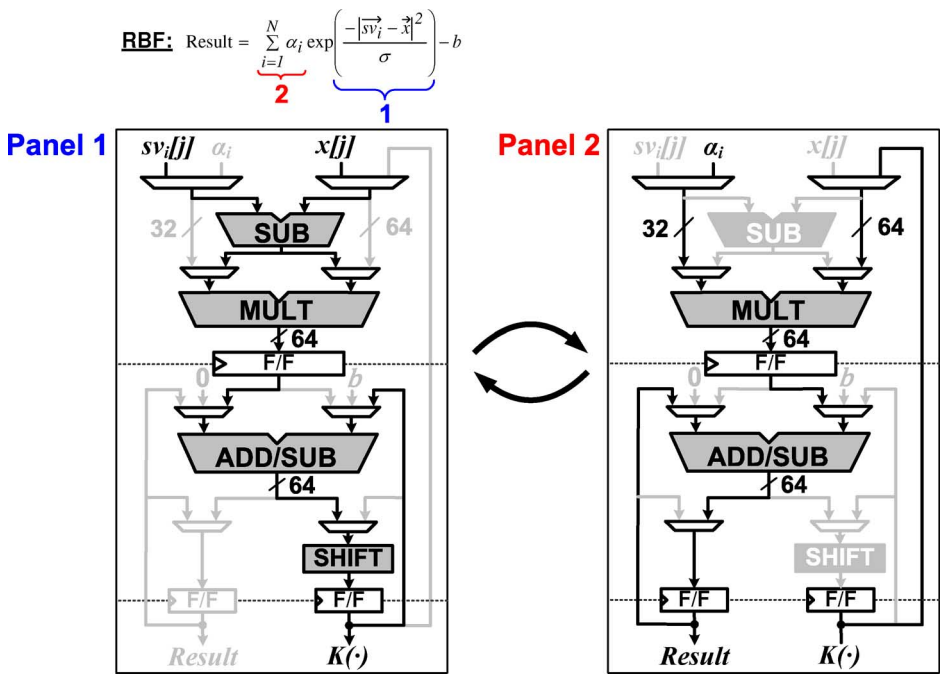


Fig. 12. Computation of the decision function with the RBF kernel. Accumulation over each support vector and accumulation over all support vectors is achieved in registers at the second stage of the pipeline.

tions are shown, illustrating the glitching that would otherwise propagate through the stages; simulations estimate that 35% energy savings (at $V_{DD} = 1.2$ V) are achieved by eliminating glitch propagation.

To minimize energy, fixed-point computations are used for the various kernel functions and model data supported. However, we found that optimizing the computational precision and avoiding computational overflow posed a critical challenge. To overcome this, the DPU employs variable truncation via an in-line SHIFT module. This enables any Q-format to balance the precision and dynamic-range requirements. A barrel shifter in the SHIFT module enables truncation of intermediate results for decision-function computations over large support-vector sets and large feature-vector dimensionalities. Bit-true simulations (results shown in Fig. 11(a)) indicate that this enables performance near that of a floating-point implementation; without this, simulations exhibit computational overflow even with small vector dimensionalities and sets. Another factor is the precision of the support vectors; this critically impacts both the computational hardware, but also the required size of the local SV memory. Fig. 11(b) shows analysis from simulation, sweeping the support-vector bit precision in order to characterize the RMS error of the classification kernel function.

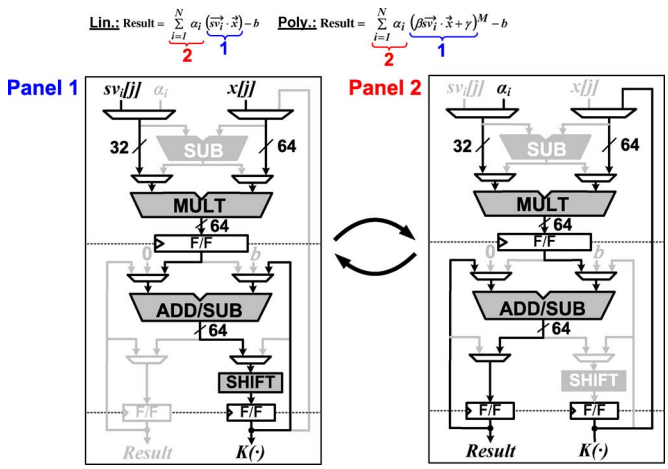


Fig. 13. Computation of the decision function with linear and polynomial kernels.

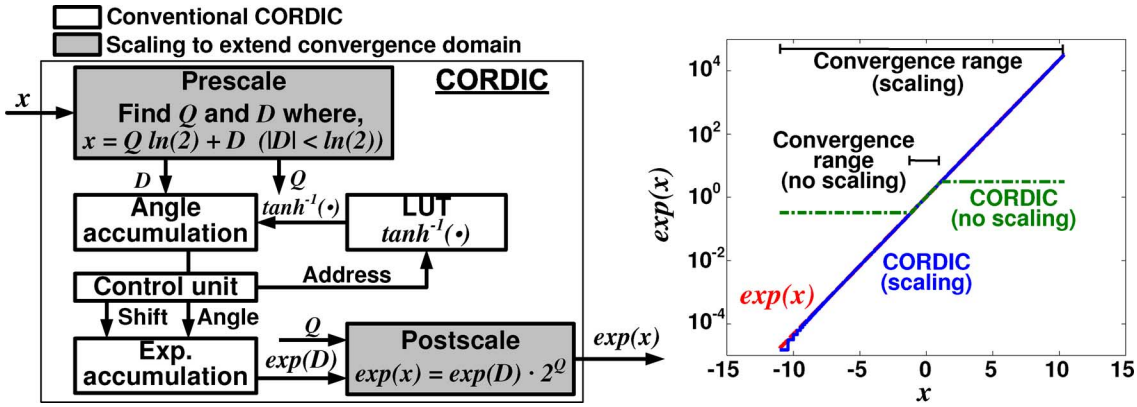
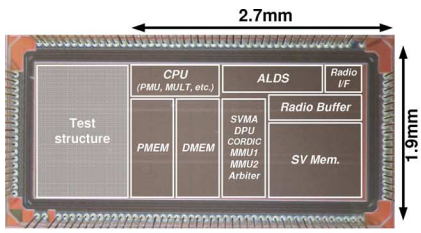


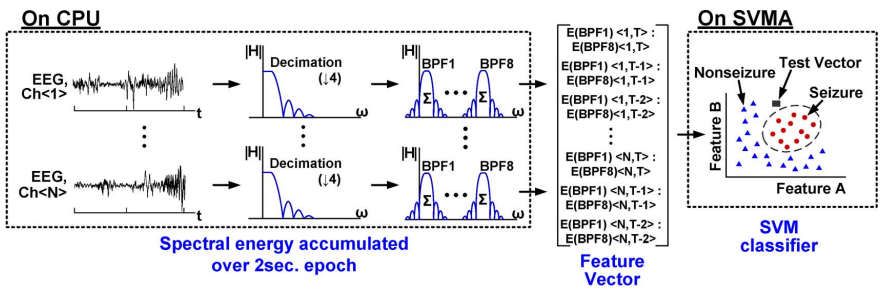
Fig. 14. CORDIC implementation (left) and the resulting region of convergence with and without scaling (right).



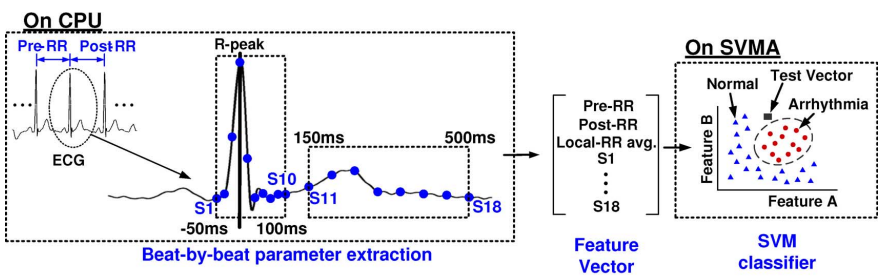
Chip summary	
Energy per clock (@ 0.6V)	• CPU: 14.8 pJ • PMEM/DMEM: 10.1 pJ • SVMA: 28.0 pJ • SV Mem.: 17.3 pJ • ALDS: 25.7 pJ • Radio buffer: 4.9 pJ • Radio I/F: 6.3 pJ
Memory size	16kB (PMEM) 16kB (DMEM) 32kB (SV Mem.) 8kB (Radio buffer)
Logic size	371k NAND2 eq. gates
Technology	IBM 130nm LP CMOS
Voltage	0.55-1.2V (logic), 0.70-1.2V (SRAM)

Comparison with the state of the art					
	Kwong [10]	Ashouei [11]	Shoeb[12]	Yoo [17]	This work
Application	General	General	EEG	Eye open/close detection	General
Programmable feature computation	Y	Y	N	N	Y
Classifier accelerator	Threshold based	None	None	SVM (Linear)	SVM (Lin., Poly., RBF)
Patient-specific model adaptation	N	N	N	N	Y

Fig. 15. Die photo and prototype IC summary.



(a)

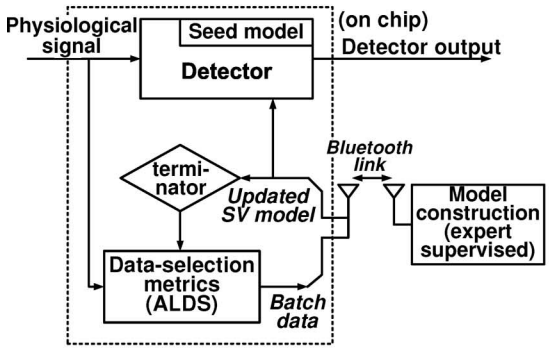


(b)

Fig. 16. (a) EEG-based seizure detection extracts the energy from eight different frequency bins from each EEG channel in a 2-s epoch, and three epochs are combined to form a feature vector. (b) ECG-based arrhythmia detection samples 18 data points in each ECG beat along with the R-to-R information. Feature extraction runs on CPU and the SVMA executes SVM classification.

Analysis was performed over several applications, with the results from a seizure detector shown; 16-b precision was chosen for the support vectors since this yields sufficient accuracy, while beyond this the accuracy tends to saturate.

The various computational paths through which the DPU implements the selectable kernel functions and formulations are shown in Figs. 12 and 13. The RBF kernel computation in Fig. 12 primarily requires three computations: subtract-square accumulation, exponentiation, and weighted summation over the support vectors. The subtract-square accumulation is executed at $K(\cdot)$ through the path in Panel 1; the elements in vectors $\vec{sv}_i$ and $\vec{x}$ are sequentially loaded by the SVMA



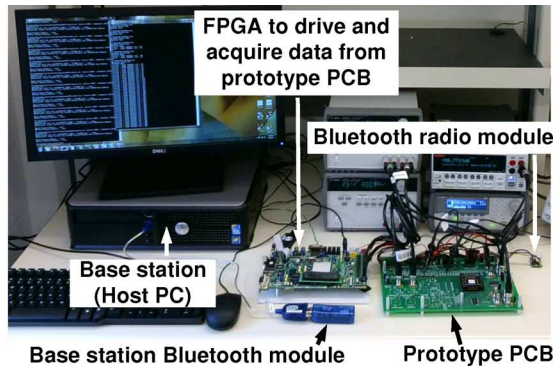


Fig. 17. Active learning block diagram (left) and demonstration setup (right).

and applied to the SUB and MULT modules to calculate the squared distance between the vectors. The ADD/SUB module is then used to accumulate the result in $K(\cdot)$. The accumulation process proceeds, controlled by the SVMA, over all dimensions of the vectors. Following this, exponentiation is implemented via the CORDIC (described in Section III-E), and the result, as part of a weighted summation over the support vectors (with coefficients α_i), is stored in ***Result*** via the paths shown in Panel 2. Following summation over all support vectors, ***Result*** gives the final classification output. Similarly, for polynomial and linear kernels, the primary computations are dot-product and weighted summation over the support vectors. These are computed through the paths shown in Fig. 13 at $K(\cdot)$ and ***Result***, respectively. Unlike the RBF kernel, polynomial and linear kernels, do not required the SUB module (i.e., since dot-products, rather than vector distances, are computed).

E. CORDIC

An embedded CORDIC engine is used for hardware-efficient computation of the exponentiation function required in the RBF kernel. The limitation with conventional CORDIC architectures is a narrow range of convergence, as shown in Fig. 14. To extend the range, we employ a pre-scale and post-scale scheme via the computation flow shown [32]. The pre-scale represents the argument as a quotient (Q) and a remainder (D), which explicitly lies within the convergence range. The post-scale then derives the correct value through simple shifting operations. As shown, the effective convergence range is thus substantially increased with minimal hardware complexity.

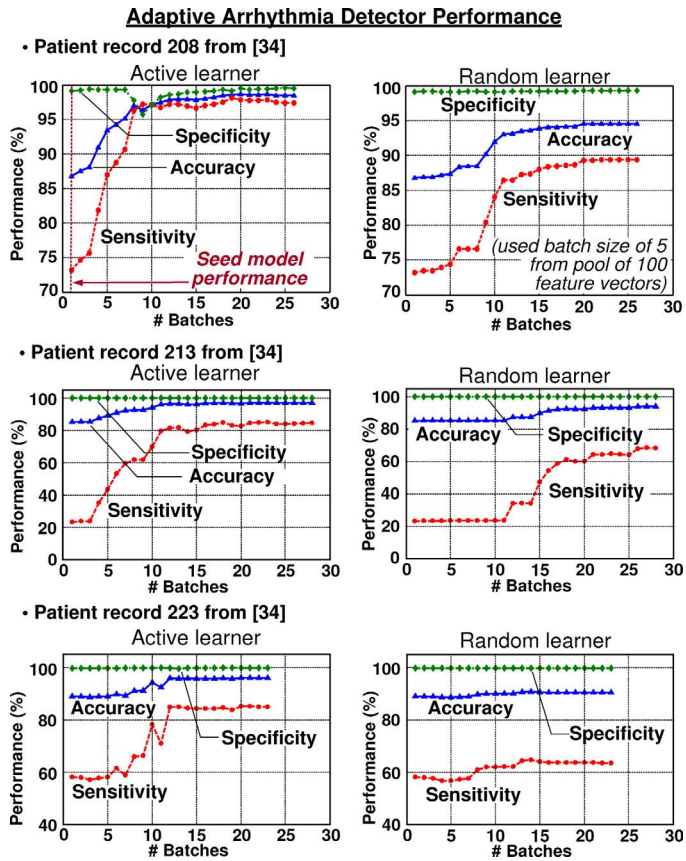


Fig. 18. Active learning performance measured from the chip showing the benefit of the ALDS approach.

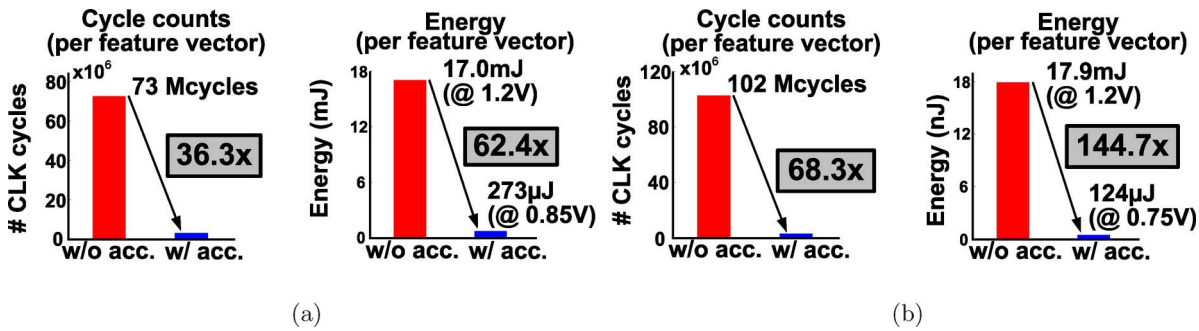
IV. Measurement and Demonstration

The processor is implemented in 130-nm LP CMOS from IBM (die photo shown in Fig. 15). It operates from 1.2–0.55 V (0.7 V for SRAMs), and the measured energy/cycle for the CPU, SVMA, and ALDS modules (including their SRAMs) is provided in the summary table shown. The processor enables algorithms employing high-order data-driven signal models and patient-adaptive capabilities, and it can be used for a range of biomedical sensor applications thanks to programmable feature extraction (supported by the CPU) and configurable classification and modeling kernels (supported by the accelerators).

We have implemented and tested several medical-sensor applications including a seizure detector (based on the 18-channel EEG algorithm in [8]), an arrhythmia detector (based on the single-channel ECG algorithm in [33]), and a patient-adaptive arrhythmia detector (based on the single-channel ECG features suggested in [33]). All tests are performed using clinical patient data from the CHB-MIT and MIT-BIH databases [34], [35], with expert-annotated EEG and ECG recordings, respectively.

A. EEG-Based Seizure Detection

Fig. 16(a) shows a block diagram of the seizure-detection application. The CPU performs feature extraction according to user software, and the SVMA performs classification configured by user software. The EEG signals are acquired from various scalp locations. Each EEG channel is downsampled by a factor of four using a decimation filter, and the band-limited signal components in eight different bins (centered at 0, 3, 6, 9, 12, 15,



(a) (b)

Fig. 19. Cycle count and energy savings for (a) seizure detection and (b) arrhythmia detection.

18, and 21 Hz) are extracted using bandpass FIR filters. The energy of each component is then accumulated in a 2-s window. This gives eight features per channel. The features from three consecutive windows are then combined to capture transitional variances of the EEG. The final resulting feature vector thus has a dimensionality of $8 \times 3 \times$ N (where N corresponds to the number of EEG channels).

B. ECG-Based Arrhythmia Detection

Fig. 16(b) shows the block diagram of the ECG-based arrhythmia-detection application. As in seizure detection, feature extraction occurs on the CPU, and classification occurs in the SVMA. The algorithm uses morphology features of the ECG. Two windows are defined for each beat segment, corresponding to 1) -50 ms to $+100$ ms, and 2) $+150$ ms to $+500$ ms, with respect to the R-peak in the ECG QRS complex. Ten evenly spaced samples (S1–S10) are extracted from the first window, and eight evenly spaced samples (S11–S18) are extracted from the second window, giving 18 samples from each ECG beat segment. Additionally, three R-peak-to-R-peak measures are used to form the final feature vector. Pre-RR measures the R-to-R distance between the current beat and the previous beat, while post-RR measures the R-to-R distance between the current beat and the next beat. Local-RR is the average R-to-R distance of 10 local ECG beats.

C. Patient-Adaptive Arrhythmia Detection

The block diagram of the patient-adaptive arrhythmia-detection application is shown in Fig. 17 along with the test setup. The same features as shown in Fig. 16(b) are used. The adaptive detector is seeded with a population-level patient-generic model derived from the collective records gathered retrospectively from multiple patients. The algorithm then continuously selects batch data from incoming patient-specific ECG sensing (as described in Section III-C) and transmits these to a base station. A new model is then generated and provided back to the device for upload. This process continues iteratively until model convergence is achieved. To permit lab demonstration, a batch size of 5 is selected from a pool of 100, yielding $20\times$ data reduction; however, in practice much larger pools would be used for greater data reduction.

In the test setup, the prototype PCB consists of the prototype IC and a Bluetooth module. The IC's radio interface controls the Bluetooth module to communicate with a PC base station, which functions as a clinical server. The FPGA board shown is used to drive and acquires data from the prototype IC. This test setup was in fact used for all three application demonstrations.

Fig. 18 shows measured results from the application. As shown, active learning gives substantial performance improvement, iteratively achieved over the patient-generic model. The results from a random-learner are also shown, which does not use the computed metrics but rather selects batch data randomly from the incoming sensor data. The superior performance of the active learner highlights the benefit of the ALDS approach.

D. Application Energy Measurements

To demonstrate the energy and cycle-count savings within applications, we utilize the RBF kernel. Although the SVMA can be configured to support various other kernels as preferred, the RBF is an energy-limiting kernel and is most general due to the high-level of flexibility it provides. Fig. 19 shows the measured cycle counts and energy for the applications, comparing the case without using the accelerators (i.e., CPU-only) to the case using the accelerators. The accelerators reduce the total cycle counts by $36.3\times$ and $68.3\times$ for the seizure and arrhythmia detectors, respectively. This allows real-time operation by the processor at a reduce V_{DD} (without the accelerators, the applications cannot run in real time even at full 1.2 V V_{DD}). The resulting energy savings are $62.4\times$ and $144.7\times$, with a total energy per classification of 273 μJ (@0.85 V, 2 MHz) and 124 μJ (@0.75 V, 1.5 MHz), respectively.

V. Conclusion

Low-power physiological-signal recording technologies have emerged for advanced medical applications enabled by low-power devices. For many envisioned applications, however, it is critical to extract clinically valuable outputs from the acquired physiological signals. The challenge is that analyzing physiological signals requires high-order model due to the complex nature of the underlying processes, and patient-specific models are often needed since the manifestations of targeted states are highly variable. Machine learning offers promising tools to address both challenges, but the computations involved are not well handled by traditional DSP, and thus the high-order models required for accurate analysis dominate energy consumption.

In this work, we propose a biomedical processor with configurable machine-learning accelerators for low-energy and real-time detection algorithms. Supporting various classification kernel functions, a configurable SVMA module enables a wide tradeoff space for classification energy and memory

usage versus performance. An ALDS module enables online active learning to minimize the burden on clinical experts for patient-specific model adaptation.

The proposed architecture reduces the energy of two representative applications (i.e., EEG-based seizure detection and ECG-based arrhythmia detection) by 62× and 145×. This enables μW-level power consumption and real-time execution of the applications.

ACKNOWLEDGMENT

The authors would like to thank MOSIS for IC fabrication.

REFERENCES

[1] J. Viventi, D.-H. Kim, L. Vigeland, E. S. Frechette, J. A. Blanco, Y.-S. Kim, A. E. Avrin, V. R. Tiruvadi, S.-W. Hwang, A. C. Vanleer, D. F. Wulsin, K. Davis, C. E. Gelber, L. Palmer, J. V. der Spiegel, J. Wu, J. Xiao, Y. Huang, D. Contreras, J. A. Rogers, and B. Litt, "Flexible, foldable, actively multiplexed, high-density electrode array for mapping brain activity *in vivo*," *Nat. Neurosci.*, vol. 14, no. 12, pp. 1599–1605, Dec. 2011.

[2] D.-H. Kim, N. Lu, R. Ghaffari, and J. A. Rogers, "Inorganic semiconductor nanomaterials for flexible and stretchable bio-integrated electronics," *Annu. Rev. Biomed. Eng.*, vol. 14, pp. 113–128, Aug. 2012.

[3] R. F. Yazicioglu, P. Merken, R. Puers, and C. V. Hoof, "A 60 μ W 60 nV/$\sqrt{\text{Hz}}$ readout front-end for portable biopotential acquisition systems," *IEEE J. Solid-State Circuits*, vol. 42, no. 5, pp. 1100–1110, May 2007.

[4] T. Denison, K. Consoer, W. Santa, A.-T. Avestruz, J. Cooley, and A. Kelly, "A 2 μ W 100 nV/rtHz chopper-stabilized instrumentation amplifier for chronic measurement of neural field potentials," *IEEE J. Solid-State Circuits*, vol. 42, no. 12, pp. 2934–2945, Dec. 2007.

[5] N. Verma, A. Shoeb, J. Bohorquez, J. Dawson, J. Guttag, and A. P. Chandrakasan, "A micropower EEG acquisition SoC with integrated feature extraction processor for a chronic seizure detection system," *IEEE J. Solid-State Circuits*, vol. 45, no. 4, pp. 804–816, Apr. 2010.

[6] A. Csavoy, G. Molnar, and T. Denison, "Creating support circuits for the nervous system: Considerations for "brain-machine" interfacing," in *Proc. IEEE Symp. VLSI Circuits*, Jun. 2009, pp. 4–7.

[7] P. Suffczynski, S. Kalitzin, and F. H. L. da Silva, "Dynamics of non-convulsive epileptic phenomena modeled by a bistable neuronal network," *Neuroscience*, vol. 126, no. 2, pp. 467–484, 2004.

[8] A. Shoeb and J. Guttag, "Application of machine learning to epileptic seizure detection," in *Proc. Int. Conf. Mach. Learn.*, Jun. 2010.

[9] D. C. Dugdale, M. A. Chen, and D. Zieve, 2012, Post Myocardial Infarction, [Online]. Available: http://www.nlm.nih.gov/medlineplus/ency/imagepages/18030.htm

[10] J. Kwong and A. P. Chandrakasan, "An energy-efficient biomedical signal processing platform," in *Proc. ESSCIRC*, 2010, pp. 526–529.

[11] M. Ashouei, J. Hulzink, M. Konijnenburg, J. Zhou, F. Duarte, A. Breeschoten, J. Huisken, J. Stuyt, H. de Groot, F. Barat, J. David, and J. V. Ginderdeuren, "A voltage-scalable biomedical signal processor running ECG using 13 pJ/cycle at 1 MHz and 0.4 V," in *Proc. IEEE Int. Solid-State Circuits Conf. Dig. Tech. Papers*, 2011, pp. 332–334.

[12] A. Shoeb, D. Carlson, E. Panken, and T. Denison, "A micropower support vector machine based seizure detection architecture for embedded medical devices," in *Proc. IEEE Eng. Med. Biol. Soc. Conf.*, Sep. 2009, pp. 4202–4205.

[13] S. Raghunathan, S. K. Gupta, H. S. Markandeya, P. P. Irazoqui, and K. Roy, "Ultra low-power algorithm design for implantable devices: Application to epilepsy prostheses," *J. Low Power Electron. Appl.*, vol. 1, no. 1, pp. 175–203, May 2011.

[14] V. Karkare, S. Gibson, and D. Markovic, "A 130 μW, 64-channel neural spike-sorting DSP chip," *IEEE J. Solid-State Circuits*, vol. 46, no. 5, pp. 1214–1222, May 2011.

[15] M. Chae, W. Liu, Z. Yang, T. Chen, J. Kim, M. Sivaprakasam, and M. Yuce, "A 128-channel 6 mW wireless neural recording IC with on-the-fly spike sorting and UWB transmitter," in *Proc. IEEE Int. Solid-State Circuits Conf. Dig. Tech. Papers*, 2008, pp. 146–147.

[16] T.-C. Chen, W. Liu, and L.-G. Chen, "128-channel spike sorting processor with a parallel-folding structure in 90 nm process," in *Proc. ISCAS*, 2009, pp. 1253–1256.

[17] J. Yoo, L. Yan, D. El-Damak, M. B. Altaf, A. Shoeb, H.-J. Yoo, and A. Chandrakasan, "An 8-channel scalable EEG acquisition SoC with fully integrated patient-specific seizure classification and recording processor," in *Proc. IEEE Int. Solid-State Circuits Conf. Dig. Tech. Papers*, 2012, pp. 292–294.

[18] K. H. Lee and N. Verma, "A 1.2–0.55 V general-purpose biomedical processor with configurable machine-learning accelerators for high-order, patient-adaptive monitoring," in *Proc. ESSCIRC*, 2012, pp. 285–288.

[19] V. N. Vapnik, *The Nature of Statistical Learning Theory*. New York, NY, USA: Springer, 1995.

[20] N. Verma, K. H. Lee, and A. Shoeb, "Data-driven approaches for computation in intelligent biomedical devices: A case study of EEG monitoring for chronic seizure detection," *J. Low Power Electron. Appl.*, vol. 1, no. 1, pp. 150–174, Apr. 2011.

[21] J. Volkmann, J. Herzog, F. Kopper, and G. Deuschl, "Introduction to the programming of deep brain simulators," *Movement Disorders*, vol. 17, pp. S181–S187, 2002.

[22] R. Sarpeshkar, C. Salthouse, J.-J. Sit, M. W. Baker, S. M. Zhak, T. K.-T. Lu, L. Turicchia, and S. Balster, "An ultra-low-power programmable analog bionic ear processor," *IEEE Trans. Biomed. Eng.*, vol. 52, no. 4, pp. 711–727, Apr. 2005.

[23] K. H. Lee, S.-Y. Kung, and N. Verma, "Improving kernel-energy trade-offs for machine learning in implantable and wearable biomedical applications," in *Proc. IEEE Int. Conf. Acoust., Speech, Signal Process.*, May 2011, pp. 1597–1600.

[24] D. Angluin, "Queries and concept learning," *Mach. Learn.*, vol. 2, pp. 319–342, 1988.

[25] E. B. Baum, "Neural net algorithms that learn in polynomial time from examples and queries," *IEEE Trans. Neural Netw.*, vol. 2, no. 1, pp. 5–19, Jan. 1991.

[26] K. J. Jang, G. Balakrishnan, Z. Syed, and N. Verma, "Scalable customization of atrial fibrillation detection in cardiac monitoring devices: Increasing detection accuracy through personalized monitoring in large patient populations," in *Proc. IEEE Eng. Med. Biol. Soc. Conf.*, Aug. 2011, pp. 2184–2187.

[27] G. Balakrishnan and Z. Syed, "Scalable personalization of long-term physiological monitoring: Active learning methodologies for epileptic seizure onset detection," *J. Mach. Learn. Res.*, vol. 22, pp. 73–81, 2012.

[28] E. Shih and J. Guttag, "Reducing energy consumption of multi-channel mobile medical monitoring algorithms," in *Proc. 2nd Int. Workshop Syst. Netw. Support for Healthcare and Assisted Living Environments*, Jun. 2008.

[29] R. E. Schapire, "A brief introduction to boosting," in *Proc. 16th Int. Joint Conf. Artificial Intell.*, 1999.

[30] K. Brinker, "Incorporating diversity in active learning with support vector machine," in *Proc. Int. Conf. Mach. Learn.*, Aug. 2003.

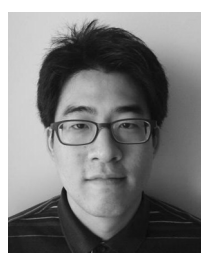
[31] M. Seok, D. Jeon, C. Chakrabarti, D. Blaauw, and D. Sylvester, "A 0.27 V 30 MHz 17.7 nJ/transform 1024-pt complex FFT core with super-pipelining," in *Proc. IEEE Int. Solid-State Circuits Conf. Dig. Tech. Papers*, 2011, pp. 342–344.

[32] J. A. Walthur, "A unified algorithm for elementary functions," in *Proc. AFIP Spring Joint Comput. Conf.*, 1971, pp. 379–385.

[33] P. de Chazal, M. O'Dwyer, and R. B. Reilly, "Automatic classification of heartbeats using morphology and heartbeat interval features," *IEEE Tran. Biomed. Eng.*, vol. 51, no. 7, pp. 1196–1206, Jul. 2004.

[34] [Online]. Available: http://www.physionet.org,Physionet.

[35] A. L. Goldberger, L. A. N. Amaral, L. Glass, J. M. Hausdorff, P. C. Ivanov, R. G. Mark, J. E. Mietus, G. B. Moody, C.-K. Peng, and H. E. Stanley, "PhysioBank, PhysioToolkit, and PhysioNet: Components of a new research resource for complex physiologic signals," *Circulation*, vol. 101, no. 12, pp. e215–e220, 2000.



Kyong Ho Lee (S'10) received the B.S. degree from the Korea Advanced Institute of Science and Technology (KAIST), Daejeon, in 2004 and the M.S. degree from Stanford University, Stanford, CA, USA, in 2009, both in electrical engineering. He is currently working toward the Ph.D. degree at Princeton University, Princeton, NJ, USA.

His research interests include ultra-low energy circuit design with an emphasis on biomedical applications employing machine learning techniques and machine learning algorithms for high energy efficiency. He is a co-recipient of Qualcomm Innovation Fellowship (QInF) 2011.

Naveen Verma (M'05) received the B.A.Sc. degree in electrical and computer engineering from the University of British Columbia, Vancouver, BC, Canada, in 2003 and the M.S. and Ph.D. degrees in electrical engineering from the Massachusetts Institute of Technology, Cambridge, MA, USA, in 2005 and 2009, respectively.

Since July 2009, he has been an Assistant Professor of Electrical Engineering at Princeton University, Princeton, NJ, USA. His research focuses on ultra-low-power integrated circuits and systems with an emphasis on sensing applications. Of particular importance is the use of emerging devices for the creation of functionally diverse systems and the use of advanced signal-analysis frameworks for low-power inference over embedded signals. On the circuit level, his focus spans low-voltage digital logic and SRAMs, low-noise analog instrumentation and data-conversion, and integrated power management.

Prof. Verma is corecipient of 2008 ISSCC Jack Kilby Award for Outstanding Student Paper, and 2006 DAC/ISSCC Student Design Contest Award. He is recipient of the Alfred Rheinstein Junior Faculty Award at Princeton and the 2013 NSF CAREER award.