Design of Energy-efficient Sensing Systems with Direct Computations on Compressively-sensed Data

Mohammed Shoaib

A Dissertation

Presented to the Faculty

of Princeton University

in Candidacy for the Degree

of Doctor of Philosophy

Recommended for Acceptance

by the Department of

Electrical Engineering

Advisers: Niraj K. Jha and Naveen Verma

September 2013

Abstract

The aim of this thesis is to explore the energy limits that can be achieved by signal-processing systems when they explicitly utilize signal representations that encode information efficiently. Compressive sensing is one method that enables us to efficiently represent data. The challenge, however, is that in compressive sensing, signals get substantially altered due to the random projections involved, posing a challenge for signal analysis. Moreover, due to the high energy costs, reconstructing signals before analysis is also often infeasible. In this thesis, we develop methodologies that enable us to directly perform analysis on embedded signals that are compressively sensed. Thus, our approach helps potentially reduce the energy and/or resources required for computation, communication, and storage in sensor networks.

We specifically focus on transforming linear signal-processing computations so that they can be applied directly to compressively-sensed signals. We show that this can be achieved by solving a system of linear equations, where we solve for a projection of the processed signals as opposed to the processed signals themselves. This opens up two approaches: (1) when the projection matrix is the random projection matrix used in compressive sensing, where we show that the linear equations can be solved with a least-squares approximation, and (2) when the projection matrix is an auxiliary matrix, where we show that the equations become underdetermined, allowing us to obtain either high-accuracy or low-energy solutions based on two designer-controllable knobs. We study our methodologies through information metrics, validating their generality, and through application to biomedical detectors, utilizing clinical patient data. Through a prototype IC implementation, we also demonstrate a hardware architecture that exploits the two knobs for power management. Further, we also explore options for hardware specialization through architectures based on custom-instruction and coprocessor computations. We identify the limitations in the former and propose a co-processor based platform, which exploits parallelism in computation as well as voltage scaling to operate at a subthreshold minimum-energy

point. We show that the optimized coprocessor reduces the computational energy of an embedded signal-analysis platform by over three orders of magnitude compared to that of a low-power processor with custom instructions alone.

Acknowledgments

I thank the almighty for his uncountable blessings in my life. This dissertation would not have been possible without the help of so many people in so many ways. Most important were my two amazing advisors. I would like to express my sincere gratitude to both of them. Prof. Naveen Verma was the one who made me walk the first steps as a researcher. His enthusiasm to explore interesting areas is unequivocal. He has been open-minded and has always encouraged me to develop and pursue original ideas. It is also his guidance that has been foremost in shaping my Ph.D. work. My co-advisor, Prof. Niraj K. Jha, has been another tremendous source of support. He has been patient with me at all times. Although I have learnt immensely from his technical expertise, I still have so much to learn from his kind disposition and superior discipline.

I have also been fortunate to have Dirk Robinson and Guotong Feng as my mentors during my internship at Ricoh. I would like to thank them and everyone else in the digital optics research team: Kathrin Berkner and David Stork for their support and feedback, and Ken Gudan for shooting hoops and catching frisbees with me. I am especially indebted to Dirk for introducing me to Pawan Baheti, who has become a great friend. Pavan is the one who connected me with Harinath Garudadri and Gene Marsh at Qualcomm. I would like to thank them all for providing me internship opportunities. Special thanks to Somdeb Majumdar for being an excellent mentor and for taking time to provide feedback on this thesis. I also thank everyone in the Peanut research team for their support and feedback. The friends I made during these days, especially Jun Ke, Muhammad Shumail, Mayank Saraf, and Nikhil Kundargi, have been wonderful. Thank you all.

The years I spent at Princeton have got me attached to its vibrant community. I cannot thank my friends and everyone on and off the campus enough for filling my graduate life with pleasant memories. Associating with several organizations on campus has been an enriching experience. Sohaib Sultan and his family have been extremely kind and generous. I would like to thank them for helping me feel at home, especially during my initial

years. Thanks also to all members of the Religious Life Program for their companionship. I would like to thank Arjun Vijaykumar, Parthav Desai, Harish Balasubramanian, Sreechakra Goparaju, Sushant Sachdeva, Divjyot Sethi, Arnab Sinha, Prateek Mishra, and all Association of South Asians at Princeton juniors as well as Pehchaan juniors for making my stay at Princeton even more enjoyable. I would also like to thank the Princeton Cricket Club members - Niket Agarwal, Chiranjeev Kalra, Jehangir Amjad, Zeerak Ahmed, and the entire Pakistani gang for the wonderful times we had. Especially memorable will be our game against University of Pennsylvania in Spring 2011. I would also like to thank Carol Porter, Dean Redman, Jeff Himpele, and all Teagle seminar colleagues at The McGraw Center. Also, a very special thanks to Dr. Venkatesan Perry and his family for their care and support throughout my stay at Princeton.

I would like to thank Muzaffer Simsir, Chunxio Li, Chun-Yi Lee, Jun-Wei Chuah, Ting-Jung Lin, Meng Zhang, Sourindra Chaudhuri, Aoxiang Tang, Chia-Chun Lin, Xianmin Chen, Debajit Bhattacharya, and Arsalan Mohsen Nia for their companionship. Special thanks to Ajay Bhoj for his caring friendship. I would also like to thank all other graduate students whom I have missed in this list for their support and friendship. The faculty in the Electrical Engineering and Computer Science Departments have been a phenomenal source of knowledge and guidance. I would like to thank all of them. I also thank the Electrical Engineering staff, Stacey Weber, Sarah Braude, Colleen Conrad, Lisa Lewis, and Lori Bailey, for all their help.

I am grateful to Qualcomm and Princeton for supporting my research through Ph.D. fellowships for a total of three years. This work was also supported in part by the Gigascale Systems Research Center and in part by NSF grants.

Finally, my heartfelt thanks to my beloved parents, my dearest wife Nawsheen, and my wonderful brother Adnan. I would not have been able to accomplish anything without their unwavering love and support.

To my parents and family who have supported me throughout my education.

Contents

List of Tables

List of Figures

Chapter 1

Introduction

Through computation and communication capabilities, sensor networks can exploit distributed resources and information sources, enabling powerful and scalable monitoring systems. Frameworks that substantially advance these individual capabilities have recently emerged. In this dissertation, we investigate how these frameworks can be united in resource-constrained nodes to directly impact sensing systems.

1.1 Need for Energy-efficient Sensing Systems

The ability to provide advanced assessment of data over a large number of signal channels will be of critical value in next-generation sensor networks. The challenge in many compelling applications is that the physical systems involved are too complex to adequately model at an analytical level. Data-driven techniques are emerging as an extremely powerful approach for overcoming this challenge [1]. Analysis of physiological signals, in order to interpret specific, clinically-relevant states, is a representative example that can lead to biomedical sensors of high clinical value. With the increasing availability of data in the medical domain, both through low-power sensing technologies and in the form of electronic records, methods based on supervised machine learning can provide extremely powerful frameworks for deriving accurate signal models from the data.

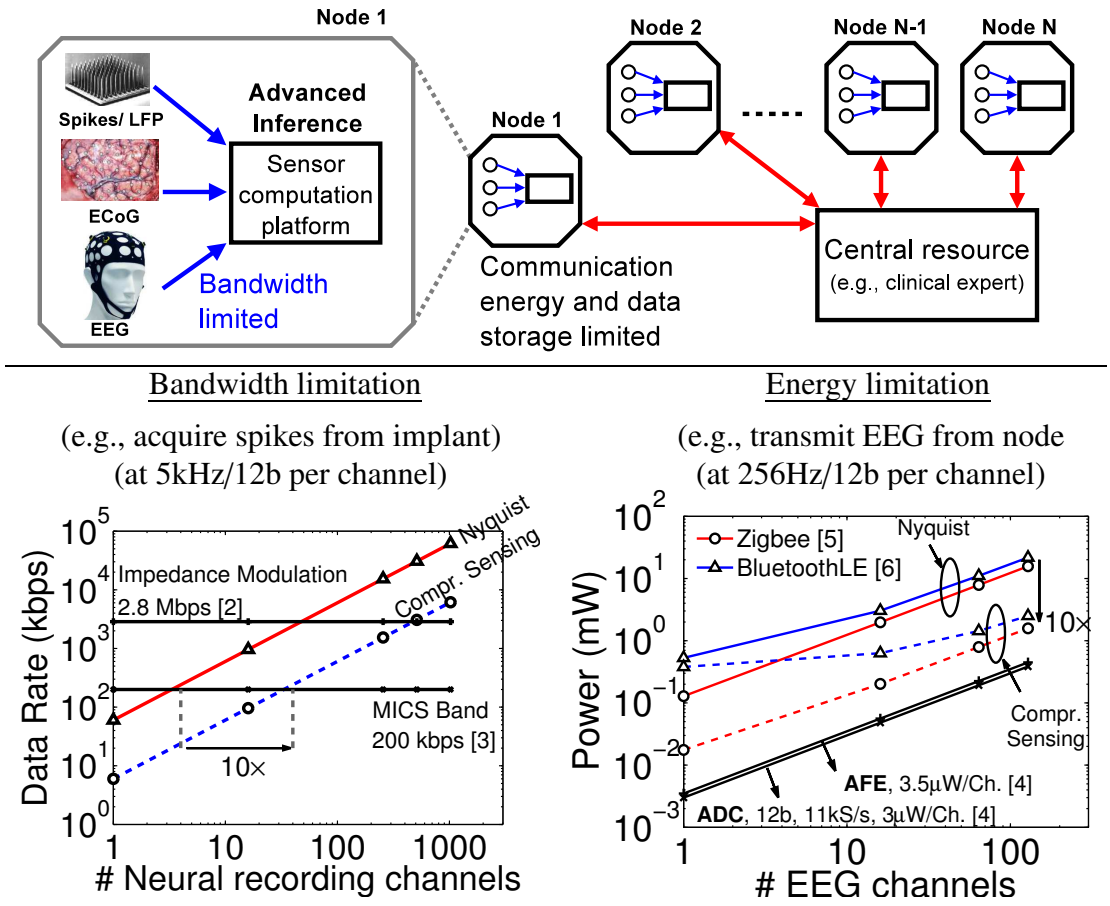


Figure 1.1: Bandwidth and energy limitation of systems for physiological signal acquisition. Nyquist sampling imposes strict limitations, necessitating over an order of magnitude reduction in the data, which can potentially be achieved through efficient signal representations, such as compressive sensing. (ECoG: electrocorticogram, LFP: local field potential).

The problem, however, is that in distributed sensor networks, communication and storage of data pose a critical limitation. Fig. 1.1 illustrates the bottlenecks. First, the signals of interest may be distributed and/or only invasively accessible. Taking neuronal spikes as a representative example, in order to transmit these signals from an energy-constrained implant to an external computational platform, transcutaneous or MICS-band links are typically utilized [2, 3]. However, Fig. 1.1 (bottom left) shows the strict bandwidth limitations faced for even moderate channel counts. Second, in a low-power sensor network, data storage on a node and communication (to centralized resources or gateway devices) face capacity and energy limitations. For instance, in Fig. 1.1 (bottom right), the power re-

quired to transmit data in a electroencephalogram (EEG) acquisition system is shown to dramatically dominate that for instrumentation and data conversion [4–6].

Efficient methods for representing the signals that dramatically reduce the data could thus play a critical role in such systems, particularly as they scale. A powerful concept that has recently shown great potential is that many physical signals of interest exhibit sparsity when transformed to some secondary domain. Though representations based on sparsity are not necessarily optimal, they have shown to be much more efficient on many levels than the original signals. Thus, as shown in Fig. 1.1, sparsity based signal representations (such as those employing compressive sensing [7, 8]) can help alleviate some of the network constraints faced by sensor networks.

Although sparsity-based transforms enable efficient representation of data, they pose a new set of challenges for machine-learning frameworks, which are required for data-driven signal analysis. In this thesis, we attempt to overcome these challenges and examine how a framework for efficient signal representation (that is based on sparsity) can be united with a framework for advanced signal analysis (that is based on supervised machine learning). In particular, we investigate the possibility of using signal-processing functions within data-driven methods that enable efficient representations based on sparsity to be used *throughout the system* to significantly reduce system energy. To this end, we explicitly develop methodologies for processing signal representations based on sparsity, and analyze the impact and trade-offs that are introduced on computational energy as a result.

1.2 Challenges posed by Sparsity-based Signal Representations for Signal Analysis

The way that physical signals exist in nature is not optimal from the perspective of embedded systems that aim to process the encoded information. By reducing the amount of data required to represent information, frameworks based on sparsity have enabled orders-

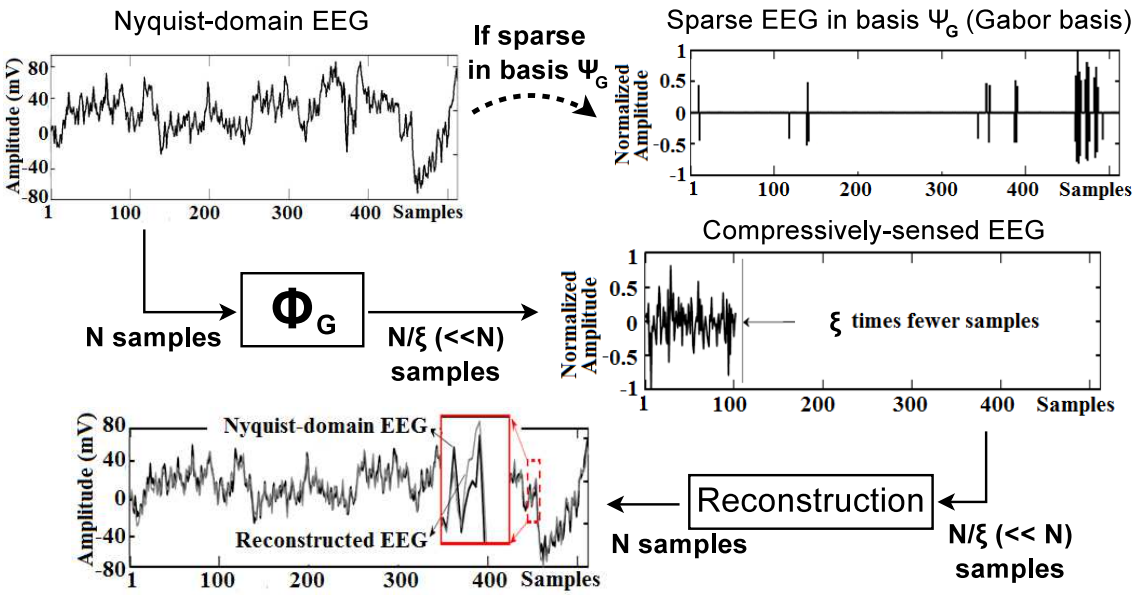


Figure 1.2: Nyquist-domain EEG is sparse in the Gabor basis, enabling substantial compression; although accurate reconstruction is possible, it is computationally intensive, motivating signal analysis directly using the compressed signal.

of-magnitude improvement in efficiency in data-conversion and communication subsystems [9, 10]. However, a critical challenge is that these transformations greatly alter the signal features, preventing us from directly performing the originally-intended processing on the resulting signals. As an illustration, Fig. 1.2 shows an embedded signal [EEG from a human] acquired from a sensor. The signal is represented both in the time domain (through Nyquist sampling) and transformed to an alternate domain using a Gabor basis $\mathbf{\Psi_G}$, *i.e.*, where it exhibits sparsity. While this enables an efficient representation through $\xi\times$ ($\xi \gg 1$) fewer data samples (derived using a random-projection transformation $\mathbf{\Phi_G}$ that is incoherent with the Gabor basis [11]), as shown, we see that the resulting signal is substantially altered at the time of measurement. To perform the original signal processing, one option would be to reconstruct the signal before analysis (as shown at the bottom). However, in the signal-representation frameworks such as compressive sensing, signal reconstruction is extremely energy-intensive [12], making it impractical.

Since our approach in this thesis enables us to exploit sparsity-based efficient signal representations throughout the system, it helps to not only circumvent reconstruction costs, but to reduce system energy when data being processed by the system explicitly represents information in a more efficient manner. The sparsity-based signal representation we focus on is compressive sensing. This has gained popularity, particularly in low-power nodes, because of its potential for very low energy compression as well as the range of signals to which it can be effectively applied [7, 8]. The methodology we propose will thus enable us to perform direct signal-processing computations on *compressively-sensed signals* in a supervised machine-learning framework. Further, we study hardware architectures that are capable of performing signal analysis on ultra-low-power sensor nodes while extensively exploiting data in the network through the efficient signal representation of compressive sensing. Through comprehensive experiments, we also show that our approach enables the signals to be reconstructed accurately so that network resources can selectively analyze them in the Nyquist domain. This, for instance, has high value in biomedical applications where the monitoring regimen may involve offline analysis by clinical experts.

1.3 Thesis Contributions

The motivation behind this thesis is the growing need for low-energy systems that also enable substantial analysis of embedded signals for high-value applications. We aim to explore whether the use of efficient signal representations based on sparsity can provide a strong path to achieving this goal. We identify several aspects to this problem including the practical question of energy consumption. The overview below outlines the main features of our work and how each component contributes to an understanding of energy-accuracy trade-offs in the system.

[1]. A methodology for processing signal representations based on sparsity that impacts the energy and accuracy of signal processing systems.

For the sake of quantitative analysis, the signal representation framework that we focus on is compressive sensing. This enables the use of non-square random-projection matrices to derive compressed representations. We develop a new methodology for transforming linear signal-processing operations for *direct application to compressed representations*. We show that this can be achieved by solving a linear system of equations. To solve these over-determined system of equations, we first use a least-squares approximation and study the energy-accuracy trade-offs introduced by our formulation [13, 14]. However, we also find that by introducing an auxiliary matrix into the formulation, several useful design options become available [15]. First, it becomes possible to relax the over-determined system of equations in order to achieve an exact solution; second, it becomes possible to scale the dimensionality of the resulting matrix transformation, yielding an energy-versus-accuracy knob for processing on compressed representations. Thus, in our study, we develop metrics for accuracy based on generalized signal-processing systems, and analyze the energy-accuracy trade-offs enabled by our auxiliary-matrix based formulation.

[2]. Quantitative evaluation of the performance of inference stages in signal-processing systems based on sparsity.

Though, initially, we use directed metrics to evaluate the accuracy of compressed signals within the system, we extend our study to investigate how such signals perform in the context of inference stages [13–15]. Since inference stages specifically rely on the information encoded in the signals, we find that they enable promising architectures for systems based on the signal representations envisioned. Beyond this practical implication, our work suggests another impact on energy-accuracy trade-offs. By relating application performance to the information in the embedded signals, inference stages quantitatively provide a metric to establish the accuracy needs of the signals. This gives both an intuitive and quantitative

meaning to the energy-accuracy trade-offs. It thus helps us understand the energy limits that might be approached in an application.

[3]. Demonstration of compressed-domain signal-processing systems through prototype integrated circuit implementations.

We explore the impact of the proposed methodology on practical systems. Based on measurements from a prototype integrated circuit (IC) implementation, we study new architectures that exploit our methodology towards embedded power management enabled by operating in the compressed domain [16, 17]. We show that in the compressed domain, application-level performance can scale substantially with computational energy. Further, since, our target platforms face additional needs for dramatic power reduction, through post-layout simulations of a second IC implementation, we also analyze domain-specific machine-learning computations in order to study architectures and circuits that selectively trade flexibility for energy efficiency to achieve the required power savings. Based on our investigations, we propose a coprocessor-based architecture for energy-efficient signal analysis on embedded sensor nodes [18, 19].

[4]. Demonstration and quantitative validation of the proposed methodology through application case studies in the biomedical domain.

In order to demonstrate our methodology and evaluate the energy and performance of a resulting system, we apply it to the following applications from the biomedical domain.

Epileptic seizure detection using EEGs. We investigate the impact of both the least-squares ([13,14]) and the auxiliary-matrix based solutions ([15–17]) on a seizure detection application. In this application, analyzing EEG signals locally can help reduce the network data [4]. However, systems for local analysis still need to collect raw data from multiple distributed sensors. This can potentially limit the number of measurement channels based on the available communication-energy budget. Moreover, since algorithms analyze

multiple EEG channels simultaneously, data buffering is necessary on the local processing platform. By simultaneously alleviating the storage, communication, and computational energy requirements, a system with end-to-end efficient signal representations can greatly benefit this application.

Neural prosthesis using neuronal spikes. We apply the auxiliary-matrix based methodology to a neural prosthesis system [15]. Neural prosthesis presents critical system challenges that can be substantially addressed by efficient data representations [20–22]; it thus serves as another useful demonstration vehicle. In particular, the signals themselves must be acquired from an implant that faces severe energy constraints [23, 24] and must be transmitted *via* a transcutaneous communication link that faces strict bandwidth constraints [2]. Due to its serial nature, transmission requires data buffering over all sensing channels, thus also imposing storage limitations on the implant. Once received by a stage worn on the head, the signals must be processed, still with minimal energy due to size and weight limitations. Finally, the neural information must be decoded for prosthetic control.

Cardiac arrhythmia detection using electrocardiograms (ECGs). High computational energy is a major barrier for embedded signal analysis. Although compressed-domain processing enables significant reductions in computational energy, we find that it can lead to modest increases in the modeling complexity of data-driven algorithms. Moreover, in certain applications like arrhythmia detection, the required modeling complexity for signal analysis can be extremely high to begin with. These applications form a suitable basis for further VLSI optimizations. Thus, we analyze the arrhythmia-detection application to identify the computational-energy bottlenecks and simulate an IC implementation to study optimizations at the circuit and architecture levels [18, 19, 25]. We show that our platform can be generalized to allow selective flexibility for embedded signal analysis in other applications. We demonstrate that the resulting energy reductions from this approach can augment the energy savings achievable through compressed-domain signal analysis.

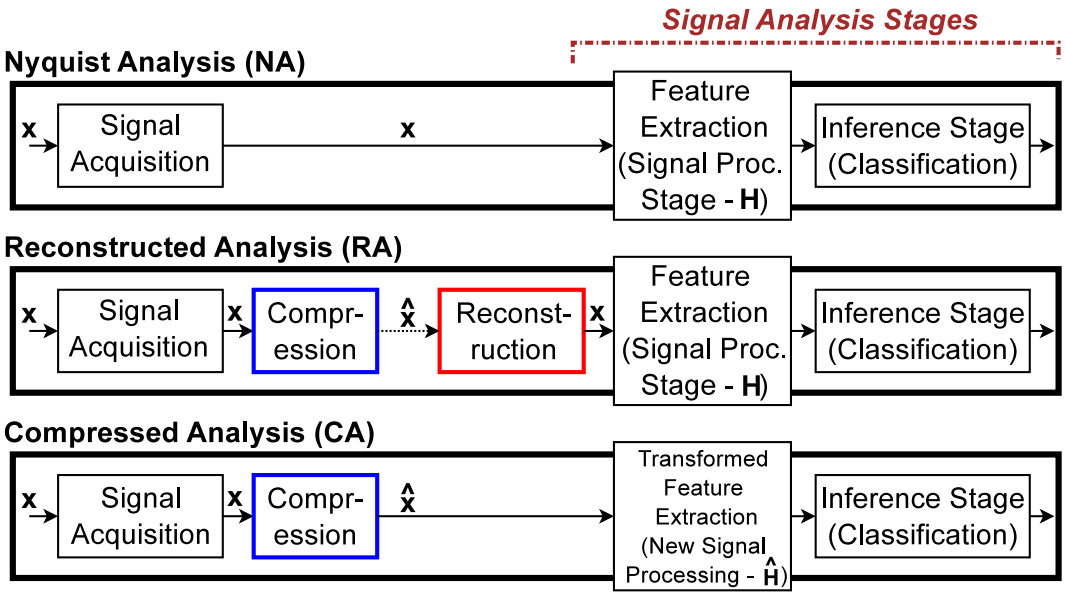


Figure 1.3: System models used for evaluation: Reconstruction is bypassed in CA to provide significant savings in computational energy.

1.4 System Models for Evaluation

In this thesis, we use three system models for comparisons. Fig. 1.3 shows the generalized system approaches. First, Nyquist analysis (NA) is defined as the usual approach wherein the embedded signals are time-domain representations obtained through Nyquist sampling. Second, reconstructed analysis (RA) is defined as an alternate approach wherein a compressed signal representation is initially received, but is then reconstructed before processing (which is represented by matrix transformation $\mathbf{H}$). RA corresponds to the system model most commonly used with frameworks like compressive sensing [10,26]. Third, compressed analysis (CA) is defined as our targeted approach wherein the end-to-end embedded signals are representations based on compressive sensing. Since our methodology aims to explicitly enable energy-accuracy trade-offs in signal processing, comparisons between CA and RA will enable us to isolate the impact on accuracy due to these trade-offs from the impact on accuracy due to compression of the initial signal. Fig. 1.3 also shows the processing stages used in each of the three approaches. Embedded signals are

first processed by a feature-extraction stage. Then, the extracted features are used to develop classification models and perform classification *via* an inference stage. Note that in CA, we need to derive *new signal-processing operations* ($\hat{\mathbf{H}}$) that enable us to obtain a representation of the targeted features (with minimal distortion errors) directly from the compressively-sensed signals. Thus, CA completely avoids signal reconstruction.

Scope of the Thesis

In this thesis, we derive $\hat{\mathbf{H}}$ such that it achieves processing on random projections wherein the inner product of the outputs is preserved with respect to the inner-product of the outputs obtained from Nyquist-domain processing. Our methodology is thus directly applicable to supervised linear classifiers where an output score $\mathbf{s}$ is derived using the inner product between a feature vector (FV) $\vec{\mathbf{x}}$ and a decision model vector $\vec{\mathbf{w}}$ as follows:

$$\mathbf{s} = f\left(\vec{\mathbf{x}} \cdot \vec{\mathbf{w}}\right) \tag{1.1}$$

where f is a function that converts the inner product of two vectors into the desired output. Examples of linear classifiers, which rely the inner-product computation, are (Fisher's) linear discriminant analysis, logistic regression, maximum-likelihood estimation, perceptron, and support-vector machine (SVM) classifiers [27]. Our methodology also applies to unsupervised algorithms such as K-means clustering where the inner-product between the input FV and cluster centroids is used as a decision function. Our methodology, however, does not apply to classifiers that do not explicitly involve the inner-product computation.

1.5 Organization of the Thesis

The thesis is organized as follows. In Chapter 2, we present background on sparse-signal representations and compressive sensing. We also discuss related work in compressed-

domain signal analysis. In Chapter 3, we present the compressed-domain processing methodology using a least-squares approximation. We apply our methodology to an epileptic seizure-detection application. To validate our approach, we also provide a mathematical rationale and simulated performance results. Next, in Chapter 4, we present an improved methodology that employs an auxiliary matrix. We apply this methodology to a neural prosthesis system and show that it enables two strong knobs for controlling energy and accuracy. In Chapter 5, we take advantage of these knobs to study the benefits at a hardware level in the seizure-detection system of Chapter 3. We show through a prototype IC implementation that the auxiliary matrix provides an ability for computational power management. In Chapter 6, we explore architecture- and circuit-level optimizations to accommodate high-order data-driven models for signal analysis. We show that such optimizations can significantly help reduce the computational energy of embedded systems for arrhythmia detection. Finally, we conclude in Chapter 7.

Chapter 2

Background and Related Work

As described in the previous chapter, efficient representations of data that exploit sparsity can play a critical role in overcoming the storage and communication-energy limitations in distributed sensor networks. In this chapter, we first describe how to find sparse representations of a signal in Sec. 2.1. In Sec. 2.2, we then present background on compressive sensing, which is a framework that takes advantage of signal sparsity in a secondary basis. We show that a characteristic of compressive sensing is that it enables very low-energy compression at the cost of high-complexity signal reconstruction. In Sec. 2.3, we then present related work that exploits compressive sensing in sensor systems. We also highlight the need for signal analysis in such systems. Finally, in Sec. 2.4, we describe related methods that have explored some theoretical aspects of signal analysis performed directly with compressively-sensed signals.

2.1 Sparse Signal Representations

A signal is said to exhibit sparsity when a linear combination of a small number of elementary signals can account for most or all information of the signal [28]. A large body of research is focused on the search for sparse representations of signals [28,29]. The problem of finding a sparse representation of an N-dimensional signal $\mathbf{x}$ simplifies to determining

an $N \times K$-dimensional overcomplete dictionary matrix $\mathbf{\Psi}$ and a K-dimensional coefficient vector $\mathbf{s}$ such that the representation of $\mathbf{x}$ is either exact $\mathbf{x} = \mathbf{\Psi s}$, or approximate, $\mathbf{x} \approx \mathbf{\Psi s}$. This process is also known as sparse coding.

Determining the sparsest representation, however, is an NP-hard problem [30]. To achieve a tractable solution, most algorithms assume the dictionary matrix to be known. In this case, if $N > K$ and $\mathbf{\Psi}$ is a full-rank matrix, an infinite number of solutions exist for $\mathbf{s}$ requiring us to set suitable constraints. The solution with the fewest number of non-zero coefficients is an appealing representation. Thus, a sparse representation can be derived as the solution to the following optimization problem:

$$\underset{\mathbf{s}}{\operatorname{argmin}} \| \mathbf{s} \|_0 \text{ subject to } \| \mathbf{x} - \mathbf{\Psi s} \|_2^2 < \epsilon \tag{2.1}$$

where ϵ is the approximation error in the representation. Recently, there has also been interest in determining the sparsity basis in parallel using a finite set of representative training data [31]. Results have shown that when compared to fixed bases, learned dictionaries are often superior in finding the sparsest representation [30, 31].

2.2 Data Compression in Sensor Networks

In this section, we provide an overview of data compression algorithms used to compress the raw data in sensor networks. We then focus on compressive sensing [7, 8], which exploits the sparsity of signals to construct efficient representations. We find it suitable for our study due to a key attribute: it has the potential to realize compressed representations with very low energy. Moreover, the large compression factors and broad applicability of compressive sensing also yield the potential for substantially impacting the bandwidth, storage, and communication-energy limitations in large-scale sensor networks, thus enabling more data to be exploited throughout the network. This makes it even more relevant to low-power embedded systems [10, 32–35].

Data compression could either be lossless or lossy. Lossless compression is an encoding process that exploits data redundancy for compression. The original data are reconstructible without any loss of information. In general, the entropy of the source that generates the data is an upper bound on the compression factor. Thus, although lossless compression provides accurate reconstruction, it typically allows only small compression factors [36]. In [37], Strydis *et al.* provide a good survey of lossless compression algorithms in the context of medical sensor networks. In contrast, lossy compression algorithms have the benefit of achieving higher compression factors at the cost of reduced reconstruction accuracy. Since, higher compression factors are desirable and reconstruction error is permissible in our applications, we consider lossy compression in detail. Sensor signals can be compressed in the time domain or transform domain [36]. Next, we provide an overview of these approaches for lossy compression.

Compression in the time domain. Most of the time-domain lossy data-compression techniques utilize interpolation and prediction algorithms. These techniques attempt to compress data by examining a succession of neighboring samples. Although time-domain techniques provide substantial compression factors, a key limitation of these approaches is error propagation. As a result, any misprediction can impact subsequent data samples. This limitation makes such algorithms less robust for wireless networks where channel fading and other artifacts can lead to significant data errors [36]. A widely used algorithm for time-domain compression is differential pulse-code modulation (DPCM). The main idea in DPCM is to determine the input signal samples by linear prediction. The current signal sample, $x(n)$, is estimated from the past samples by using either a fixed or adaptive linear predictor as follows.

$$\hat{x}(n) = \sum_{k=1}^{N} a_k \, x(n-k) \tag{2.2}$$

where $\hat{x}(n)$ is an estimate of $x(n)$ at time n, and $\{a_k\}$ is the predictor weight. The error sequence [*i.e.*, $e(n) = \hat{x}(n) - x(n)$] is quantized by using a nonuniform quantizer. The quan-

tizer outputs are then coded by assigning variable-length codewords to the error sequence according to the frequency of occurrence, *e.g.*, by using Huffman codes [36]. DPCM has recently found applicability to biomedical signals [38]. Owing to the quasi-periodic nature of cardiac data, it has been shown to be particularly useful for compressing ECG signals [39, 40]. A compression factor of 7.8 and a reconstruction signal-to-noise ratio (SNR) of 29.1 dB has been reported for ECG signals sampled at 500 Hz [40]. Some other popular time-domain compression techniques are differential linear predictive coding where signals are analyzed to create filter coefficients that best match the input segment [38], zero-order interpolation such as amplitude-zone time-epoch coding where signals are modeled as a combination of flat regions and slopes [41], polynomial-predictive compression where the signals are considered as a combination of high-order polynomials [42,43], *etc*. A survey of time-domain compression algorithms is presented in [38, 44–46]. The compression factor in these methods is limited by the amount of quantization error and permissible code length. Moreover, most of these algorithms rely on specific signal morphologies and are limited in their applicability to a broad range of signals. Transform-domain methods provide an alternate way for data compression.

Compression in the transform domain. The key idea in transform-domain compression/coding is to divide the signal into transformed components and judiciously allocate bits in the transform domain (often after discarding near-zero coefficients) [36, 47]. Although these methods provide higher compression factors and a robust alternative to time-domain methods, they are often computationally intensive since they first require transforming data before compression. Linear transforms are mostly used in sensor networks. In these methods, the input signal, $\mathbf{x}$, is first divided into blocks of data and each block is processed as follows:

$$\mathbf{v} = \mathbf{A}\mathbf{x} \tag{2.3}$$

Table 2.1: Comparison of different lossy compression methods in sensor networks.

Compression method	Advantages	Disadvantages
1) Time-domain compression	· Low computational complexity	· Low compression factors · Error propagation · Narrow applicability
2) Transform-domain compression	· High compression factors · No error propagation · Broad applicability	· High computational complexity

where **v** is the transformed sequence and **A** is the matrix representing the linear transform. A variety of matrices is used to transform digital waveforms to the frequency domain including Fourier transform matrix [48], Karhunen-Loeve transform (KLT) matrix [49] and discrete cosine transform (DCT) matrix [48]. The transform matrix **A** can also allow multi-resolution filtering on the time-frequency scales [38]. Compression after wavelet transform is the most widely used approach for time-frequency coding [50, 51]. Due to the matrix multiplications involved, transform-domain coding methods suffer from high computational costs. Although optimized algorithms and architectures can help alleviate their complexity, their applicability to severely energy-constrained sensor nodes is still limited [48, 52–54]. Transform-domain methods, however, permit large compression factors. For instance, using ECG signals, a compression factor of 24 is reported for KLT-based coders [55]. Since signal recording conditions and noise levels vary from study to study, a thorough comparison of the compression methods is very difficult to make. However, transform-domain methods typically yield much higher compression factors than time-domain compression methods [38]. Table 2.1 summarizes the benefits and limitations of time-domain and transform-domain compression techniques in sensor networks.

Exploiting sparse signal representations can potentially obviate the limitations of both time- and transform-domain techniques. Sparse coding is a special class of data compression methods. It is used to compress data that are represented sparsely either in the time domain or the transform domain. A range of algorithms exists for sparse coding [56–58].

A survey of different sparse coding algorithms is also presented in [59]. Compressive sensing is one such technique that exploits the sparsity of signals *indirectly*. It allows for compression of signals in the time domain without prediction or interpolation [52]. Since signals often exhibit sparsity in some basis, compressive sensing applies to a broad-range of signals. It permits large compression factors and unlike transform-domain methods, the complexity of compression is also very low since it involves simple addition and subtraction operations [32]. The cost, however, is the increased computational complexity for reconstruction. We show ahead in Sec. 2.3 that this asymmetry of compressive sensing is acceptable for the considered sensor systems. Next, we present details of compressive sensing.

Exploiting Sparsity for Low-energy Compression: Compressive Sensing

The theory of compressive sensing relies on the principles of uncertainty and incoherence between two bases. This characteristic has generated much interest due to its implication on generating efficient signal representations without prior knowledge of the signal structure. This attribute is also the one that provides broad applicability to compressive sensing in application areas such as imaging [34, 35, 60], video coding [61], channel estimation [62], genotyping [63], ultrasonic systems [64], active/passive radar [65, 66], and surface metrology [67]. The details of the compressive-sensing framework follow.

For any N-sample signal $\mathbf{x}$, which can be represented as $\mathbf{\Psi s}$, where $\mathbf{\Psi}$ is the sparse dictionary and $\mathbf{s}$ is a vector of K-sparse coefficients, we can use a measurement matrix $\mathbf{\Phi}$ to transform $\mathbf{x}$ to a set of M $[O\{Klog(N/K)\} < M \ll N]$ compressed samples (denoted by $\hat{\mathbf{x}}$) in the time domain as follows:

$$\hat{\mathbf{x}}_{\mathbf{M\times 1}} = \mathbf{\Phi}_{\mathbf{M\times N}}\, \mathbf{x}_{\mathbf{N\times 1}}. \tag{2.4}$$

The compression factor $\xi = N/M$ quantifies the amount of compression achieved by the projection. For accurate recovery of $\mathbf{x}$ from $\hat{\mathbf{x}}$, $\mathbf{\Phi}$ needs to be incoherent with $\mathbf{\Psi}$ and satisfy the restricted isometry property (RIP), which is defined as follows:

$$(1 - \delta_s) \parallel \mathbf{x} \parallel_2^2 \leq \parallel \mathbf{\Phi x} \parallel_2^2 \leq (1 + \delta_s) \parallel \mathbf{x} \parallel_2^2 \tag{2.5}$$

where $\delta_s < 1$ is the isometry constant. An $M \times N$ dimensional matrix $\mathbf{\Phi}$ whose entries are *i.i.d.* samples from the uniform distribution $U(+1, -1)$ or from the normal distribution $N(0, 1)$ is often maximally incoherent with $\mathbf{\Psi}$ and satisfies the RIP [7,68]. Deriving $\mathbf{\Phi}$ from $U(+1, -1)$ also leads to low-energy compression on the sensor node since the projection is reduced to simple additions and subtractions.

Although sensing can incur very little energy, the reconstruction of $\mathbf{x}$ from $\hat{\mathbf{x}}$ can be costly. We see from Eq. (2.4) that $\hat{\mathbf{x}}$ is underdetermined (*i.e.*, knowing only $\hat{\mathbf{x}}$ and $\mathbf{\Phi}$, there are an infinite number of possible solutions for $\mathbf{x}$ and, hence, for $\mathbf{s}$). However, since $\mathbf{x}$ is sparse in $\mathbf{\Psi}$, the sparsest solution for $\mathbf{s}$ is often the correct solution with high probability. One common approach used to determine the sparse solution is to solve the following convex optimization problem:

$$\text{minimize } \parallel \mathbf{s} \parallel_1 \text{ subject to } \hat{\mathbf{x}} = \mathbf{\Phi\Psi s}, \tag{2.6}$$

The reconstructed signal is then given by $\mathbf{x_R} = \mathbf{\Psi s}^*$, where $\mathbf{s}^*$ is the optimal solution to Eq. (2.6). The $l1$-norm above serves as a proxy for sparseness as it penalizes small coefficients so that the optimization drives them to zero. Although Eq. (2.6) requires only a small number of measurements ($M \ll N$) to enable accurate recovery, even with the most efficient approach, the complexity of solving the optimization problem can be prohibitive on typical power-constrained platforms, such as sensor nodes [12,69–71]. Fig. 2.1 shows that even at $\xi = 3\times$, reconstruction costs over four orders of magnitude more operations than compression. Recently, there have been attempts to optimize the reconstruction

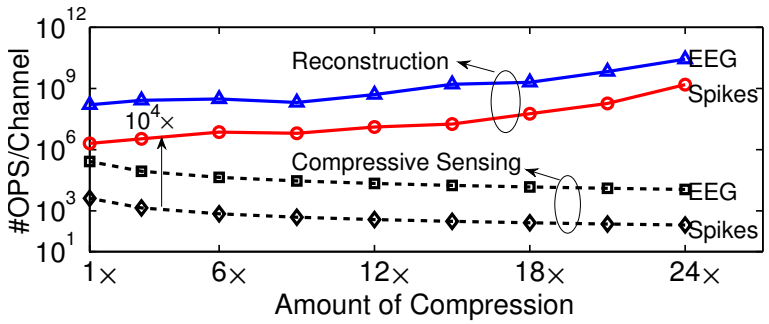


Figure 2.1: Reconstruction can cost over four orders of magnitude more operations than compression (number of operations were estimated using Lightspeed [77]).

process for hardware implementation. For instance, implementations that accelerate the reconstruction process using parallel computing or GPGPU [72, 73] have been explored. In [72], Borghi *et al.* compare the performance of reconstruction algorithms on an IBM cell processor, NVidia GPU, and Intel dual-core processor. In [73], Andrecut implements an algorithm that is an approximation to matching pursuit for signal reconstruction [74]. For 512-dimensional vectors, these GPU implementations achieve decompression results on the order of tens of milliseconds. Their major bottleneck is memory bandwidth from the main memory to the GPU and back. There have also been FPGA- and ASIC-based implementations for accelerating reconstruction [75, 76]. At a compression factor of 4×, these implementations demonstrate reconstruction of 128-dimensional signals in 24 μS and 10 μS, respectively. The reconstruction delays, however, increase exponentially with larger data vectors and at higher amounts of data compression or signal sparsity. Thus, due to the large energy/delay overheads involved, reconstruction is still unviable in many low-power applications.

2.3 Sensor Systems for Exploiting Sparsity

Given its potential for enabling low-energy compression, compressive sensing has also begun to be exploited in sensor networks [32]. Fig. 2.2 illustrates the network-level model enabled by compressive sensing. On the left, the figure shows the Nyquist-domain picture

where signals are sampled and compressed using nonlinear transformation functions on a sensor node before storage/transmission for remote retrieval. Decompression of the data enables recovery and analysis of the original signal at the receiver. The related trade-offs in the compressive-sensing approach are shown on the right. An N-sample data vector $\mathbf{x}$ is compressed to $M \ll N$ samples either using random projections after sampling at the Nyquist rate [10,26,78] or through analog-to-information (A2I) conversion [9,79–82]. The A2I conversion process involves three stages: demodulation, filtering, and uniform sampling. A sensed signal is first modulated by a psuedo-random sequence of ± 1's in the analog domain. The modulated signal is then processed by a low-pass filter and finally sampled at a lower Nyquist rate using a traditional ADC. Both A2I conversion and random projections enable low-energy data compression on a sensor node at the cost of high-complexity reconstruction at the receiver. This is exactly the trade-off required for a network of ultra-low-power sensor nodes (*e.g.*, such as those described in Sec. 1.1, which employ ECoG, LFP, EEG, and spike measurements). In such applications, the sensor nodes typically face severe energy constraints whereas the reconstruction process (performed at a base station) is not energy-constrained. As a result, sensor systems that employ compressive sensing have been able to efficiently serve as nodes for acquiring and transmitting signals to a base station, which performs signal analysis after reconstruction [10, 26, 33, 83–86].

The basic network model of Fig. 2.2, however, drastically limits the system functions that can be implemented on devices, particularly as many sensing applications progress towards the need for advanced embedded signal analysis. As a representative example, consider the medical domain where devices are striving to provide high value on-node analysis of physiological signals. This approach has enabled several new possibilities. For instance, out-patient monitoring networks in the future promise comprehensive healthcare delivery over large populations and potentially diverse disease states [87]; deep-brain stimulators [88, 89] offer unprecedented modalities for delivering therapy to patients affected by neurological conditions, ranging from Parkinson's disease to epilepsy; neural prosthe-

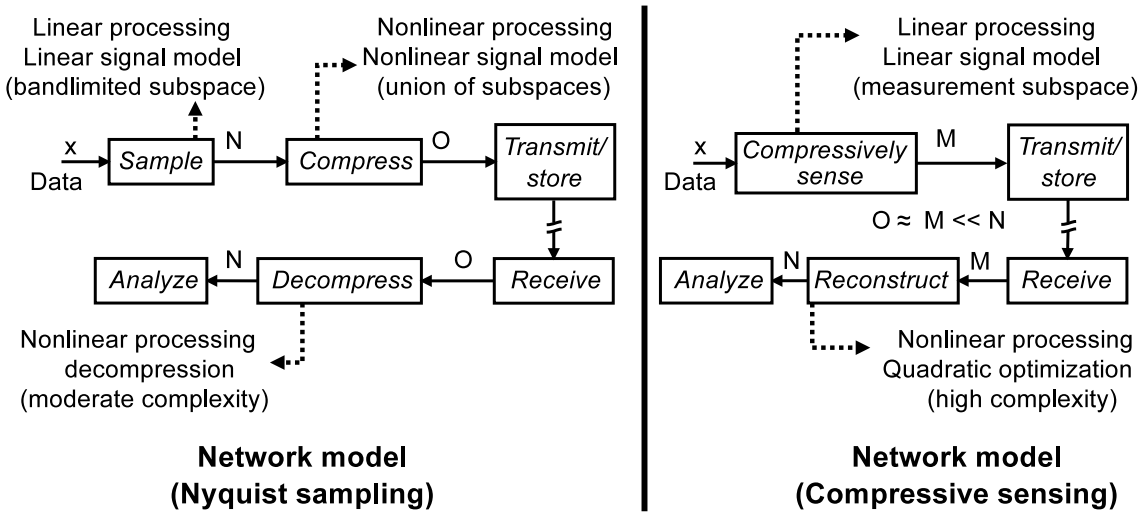


Figure 2.2: Comparison of network models (including data flow) of Nyquist sampling and compressive sensing.

sis (*i.e.,* brain-machine interface) [90–92] is increasingly being employed to restore motor functions in disabled patients. The central need, as devices in such systems aim to provide actionable outputs [91], is the ability to detect specific physiological states of interest in real-time from embedded signals that are available through minimally invasive and low-power sensors.

The need for embedded on-node signal analysis motivates the preferred network model shown in Fig. 2.3. In the Nyquist-domain picture shown in Fig. 2.3(a), signals are analyzed on the sensor node to distill out the most informative data instances, which are then compressed for subsequent storage or transmission to a base station. This approach thus has the potential to significantly reduce storage/communication energy requirements of the sensor node [93].

Similarly, the preferred network model in the compressed domain is shown in Fig. 2.3(b). Ideally, we expect to be able to directly analyze compressively-sensed signals on the sensor node before storage/transmission. Directly analyzing the compressed data can save us computational energy in two ways: (1) by avoiding the costly reconstruction process, and (2) by processing fewer data samples on the node. However, as described in

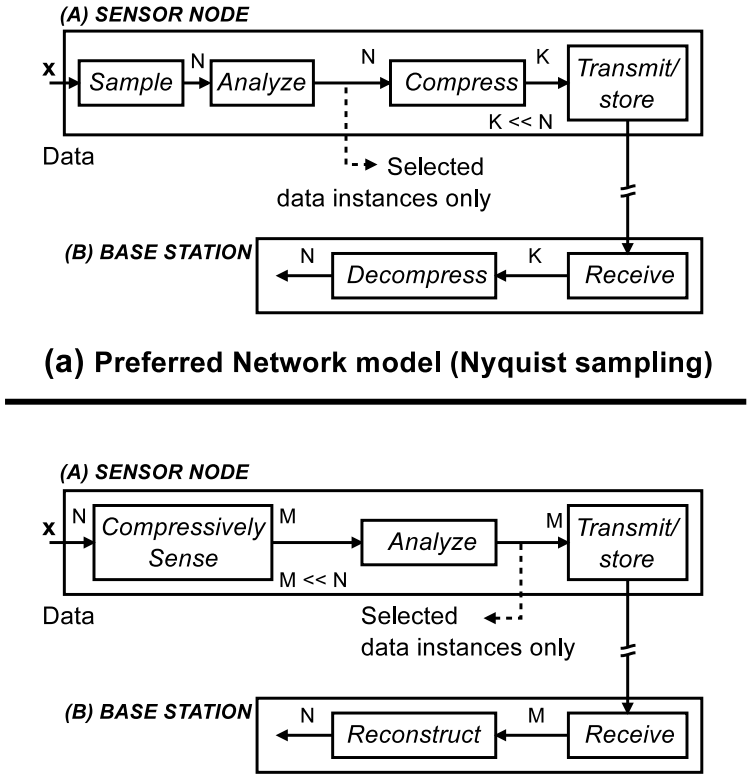


Figure 2.3: Preferred network models (including data flow) of Nyquist sampling and compressive sensing.

Sec. 1.2, the problem faced by such approaches is that embedded signals get substantially altered due to the random projections in compressive sensing. Consequently, sensor systems that employ compressive sensing are often restricted to the network model of Fig. 2.2, where signal analysis, which has to begin with reconstruction, is delegated to base stations with substantially relaxed energy constraints [10, 26, 33, 83–86]. To the best of our knowledge, this dissertation is the first in demonstrating practical implementations of sensor systems that directly analyze compressively-sensed data, thereby enabling the preferred network model of Fig. 2.3(b).

Before we look at processing compressed signals, we note that performing signal analysis on embedded platforms is a challenging task even in the Nyquist domain. This is due to (1) the severe platform constraints and (2) the complex computations involved. For in-

stance, in the biomedical domain, the full functionality of sensor nodes must be achieved at very low-power levels (*e.g.*, 1-10 mW for wearable devices and 10-100 μW for implantable devices [94–96]). Next, we discuss related work in Nyquist-domain signal analysis on sensor nodes.

Embedded signal analysis in the Nyquist domain: Algorithms and architectures

In the recent literature, low-power platforms for processing data in the Nyquist domain have emphasized an optimization of signal-processing computations, largely based on synthesis and analysis operations [97–99]. However, analyzing data in emerging applications raises issues that cannot be handled by signal processing alone. For instance, physiological signals in the biomedical domain pose two essential challenges: (1) the signal correlations to clinically-relevant states are often too complex to model based on physiology, and (2) the precise correlations are hard to isolate in the presence of normal physiologic activity. Since the acquired signals themselves represent the good and bad activity, a potential key to accurate signal detection could thus lie in the ability to develop models based on representative or exemplary data. In fact, data-driven modeling techniques are emerging as a powerful alternative to signal-processing algorithms that help overcome the mentioned challenges [100–102]. This, in large part, has been prompted by the recent large-scale availability of data in the medical domain as well as the development of machine-learning techniques that are capable of exploiting large amounts of data to model specific correlations and then use the models within a decision function [103]. Furthermore, data-driven modeling often involves probabilistic methods for model construction and can thus also tolerate substantial variations due to background physiologic activity [104–106].

In contrast to the synthesis and analysis operations employed in signal-processing algorithms, machine-learning methods focus on modeling and detection by exploiting potentially rich data that are available. In practice, although modeling based on exemplary data can be done offline, detection, through the application of a model, is performed in real-time

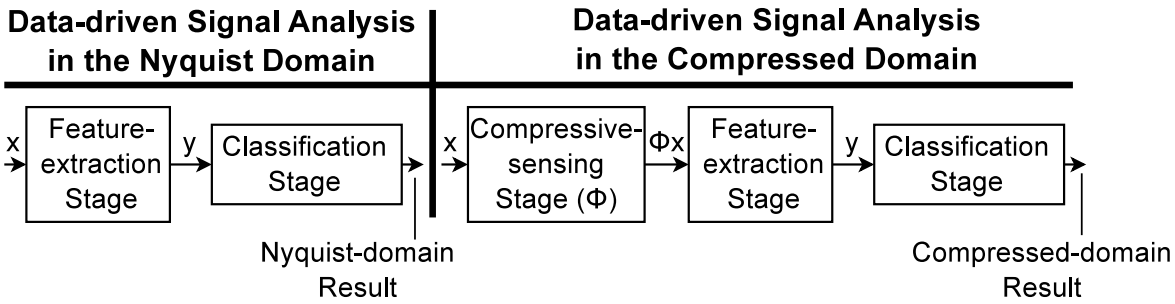


Figure 2.4: Data-driven signal analysis involves two stages of computation: feature extraction and classification.

for signal analysis. Further, as shown on the left of Fig. 2.4, real-time data-driven signal analysis in the Nyquist domain involves two stages of computation: feature extraction and classification. The corresponding computations in the compressed domain are shown on the right.

With regards to Nyquist-domain implementations in the literature, chronic patient monitoring devices are beginning to exploit data-driven techniques, but have thus far been limited in their ability to incorporate the complete computation [91,93,107]. It has been shown, for instance, that local feature-extraction computations can be performed on the signal, reducing the communication data so that computationally-intensive data-driven classification can be performed on a separate device [93].

In contrast, we show implementations of sensor systems that incorporate the complete computation both in the Nyquist and compressed domains. We also show the applicability of our platforms to medical detection problems in the embedded-systems space. We thus investigate the advantages of using efficient signal representations throughout the system, most notably for signal analysis based on data-driven methods. The key aspect for our platforms is a methodology for compressed-domain signal analysis. Next, we consider related work in this direction.

2.4 Sensor-signal Analysis (with Representations based on Sparsity)

Given that signal representations based on sparsity promise to conserve signal information, analysis on the resulting representations should be possible despite the signal alterations (*e.g.*, those in Fig.1.2). This thesis utilizes this concept to develop methodologies for signal processing. Relating back to the compressed-domain signal analysis picture shown on the right of Fig. 2.4, the key idea is to transform computations in the compressed domain so that the result of signal analysis is (nearly) the same as that in the Nyquist domain. This can be achieved in three ways: (1) by transforming the compressive-sensing stage, (2) by transforming the feature-extraction stage or (3) by transforming the classification stage. Prior work has made some progress on a theoretical level by transforming the compressive-sensing stage and the classification stage (for specific discriminative and generative classifiers). There have also been explorations along a fourth path where none of the computations are transformed and the result of signal analysis in the compressed domain is compared directly with that of the Nyquist domain. We, on the other hand, transform computations in the feature-extraction stage. We not only explore this direction at a theoretical level but also study its implications through practical system demonstrations. Next, we present details of related work along the other three paths as well as describe the benefits of transforming feature extraction instead.

2.4.1 Not Transforming any Stage

Previous work has demonstrated the theoretical aspects of performing signal analysis in the compressed domain without transforming any computations. There have also been some specific application-level case studies. Random projections used in compressive sensing preserve the inner-product between data vectors [108–110]. This property has been a cornerstone in most of the empirical results. For example, by directly using compressively-

sensed data, it has been shown that the accuracy of classifying text documents with K-means clustering is retained up to compression factors of 3× [111], and the accuracy of isolating genes with SVM classifiers is retained up to compression factors of 4× [112].

Theoretical work, however, has focused more on deriving error bounds for classifiers operating on compressed data. The error bounds help us understand the compression limits for accurate signal analysis without transforming any computation in Fig. 2.4. These approaches consider both discriminative and generative classifiers.

Theoretical methods with discriminative classifiers

In [113], Calderbank *et al.* derive an error bound for learning discriminative models using an SVM classifier with linear kernels [114–116]. They show that if the high-dimensional data points have a sparse representation in some linear basis, then it is possible to train a soft-margin SVM classifier on a low-dimensional random projection of that data while retaining a classification performance that is comparable to that achieved by working in the original data space. They show that performance degradation occurs at a rate of $O(\sqrt{\delta_s})$ where $\delta_s < 1$ is the isometry constant used in Eq. (2.5). The authors also mention that if the data are sampled with large compression factors, δ_s becomes large, and as a result, the SVM becomes a weak learner, requiring more training samples for accurate modeling. By using synthetic data, they demonstrate that with approximately 3× fewer data samples, the SVM generalization error is less than 5%. Generalization error is a measure of how well a learning algorithm generalizes to unseen data.

Other explorations with discriminative models consider regression [117–119] and nearest neighbor classifiers [120]. In [117], Zhou *et al.* consider the problem of estimating β using a vector of noisy observations $\mathbf{y}$ and control variables $\mathbf{x}$ in the compressed linear-regression model $\mathbf{y} = \mathbf{\Phi x}\beta + \mathbf{\Phi}\epsilon$, where ϵ is standard noise with a known distribution, and $\mathbf{\Phi}$ is the random-projection matrix satisfying RIP. The authors provide an analysis of the LASSO estimator built from these compressed data, and discuss a property called sparsis-

tence, *i.e.*, the number of random projections needed to recover β (with high probability) when it is K-sparse. They show that the probability of correctly estimating β (with empirical risk that is below the value in the Nyquist domain) is retained up to compression factors of 20× and for sufficiently sparse β, it can be retained even up to 120×.

In [118], Maillard *et al.* expand upon the results in [117], where they show that with $M = O(\sqrt{K})$ measurements, it can cost only $O(NK^{3/2})$ elementary operations to compute the least-squares regression function in the compressed domain. Using synthetic data, the authors show that the estimation error is $O(logK/\sqrt{K})$. They also claim that this is competitive with the best methods known in the Nyquist domain.

Another paper showing similar results is for nearest-neighbor classifiers by Indyk and Motwani [120], which exploits the notion of *locally-sensitive hashing*. The authors present extensive theoretical proofs showing the retention of accuracy in compressed-domain nearest-neighbor classification.

Theoretical methods with generative classifiers

Modeling of compressively-sensed data through generative methods has also been explored. In [121], Durrant *et al.* study the performance of Fisher linear discriminant analysis on compressively-sensed data. The authors demonstrate a tight upper bound on the average misclassification error over the random choice of the projection matrix. They also present a simple demonstration using Nyquist-domain data vectors of dimensionality $N = 100$. They show that the classification error decreases exponentially as the number of compressed samples M approaches N. Their results for classifying data from 10 classes show that at compression factors of 10, 4, and 2×, the empirical error rates of misclassification were 52, 22, and 5%, respectively. For good generalization, the authors suggest that the required projection dimension (M) grows logarithmically with the number of classes. They also show that their error bound tightens in a natural way as the number of training examples increases.

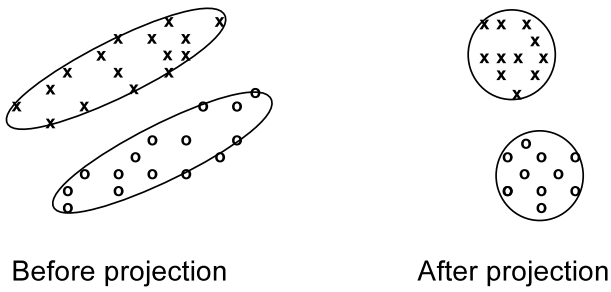


Figure 2.5: The effects of random projection: the dimension is drastically reduced while the clusters remain well-separated and become more spherical. Figure adapted from [122].

There has also been work on estimating a mixture of Gaussians by directly using compressively-sensed data [122, 123]. Besides inner-product preservation, these approaches exploit another key property of random projections called the *concentration of measure*, which states that if data clusters in the Nyquist domain are highly eccentric (that is, far from spherical), random projection will make them more spherical; spherical distributions make it easier for algorithms to classify the data instances [124, 125]. This property is also illustrated in Fig. 2.5. Since clusters of high eccentricity present an algorithmic challenge, modeling randomly projected data can potentially provide a natural way of data separation, significantly impacting the performance of algorithms such as those that estimate a mixture of Gaussians. For instance, in [122], Dasgupta presents theoretical error bounds that show that expectation maximization (EM) with compressively-sensed data consistently yields models of quality (log-likelihood on a test set) comparable to or better than those found by EM in the Nyquist domain. Through experimental results, where 50 Gaussians were fitted to optical character recognition data ($N = 256$), the author shows that the detection error is just over 4% at compression factors of 6×.

The methods described so far consider classifying compressed data without any feature extraction. There has also been some work on studying the limits of specific feature-extraction computations performed directly on compressively-sensed data [126, 127]. In [126], principal component analysis (PCA) is considered in detail where the Rayleigh-Ritz theory is extended to the special case of highly eccentric distributions (the first eigenvector

is significantly larger than the others). The paper demonstrates accurate PCA computations on compressively-sensed data up to compression factors of nearly 5×.

Practical classifiers are limited in their ability to construct decision models. By not transforming any computations, the obtainable performance at a given amount of compression is thus restricted by the problem complexity and the available dataset. Increasing the separability of the data by transforming computations can help simplify the process of building decision models, thereby enabling accurate classifier performance at higher compression factors. Thus, transforming computations helps us gain more control over the algorithmic performance. Next, we consider related work in this direction.

2.4.2 Transforming the Sensing Stage

We showed in the previous section that comparing the performance of compressed-domain analysis with that of the Nyquist domain can help us quantify error bounds and understand the relationship between compression factor and the performance of data-driven models. By transforming computations in the compressed domain, however, we can potentially do better, achieving performance much closer to the Nyquist domain, perhaps at even higher compression factors. One approach towards this goal involves transforming computations in the compressive-sensing stage. In this section, we show that although transforming the sensing stage can help achieve accurate classification, the signals cannot be reconstructed since the new sensing matrices do not necessarily satisfy RIP and the ±1 structure of the random-projection matrices is disrupted, leading to higher costs for data compression. Fig. 2.6 illustrates the concept of transforming the sensing matrices. The key idea is to derive new mapping functions that preserve low-dimensional geometric structure in the compressed data so that the performance of a subsequent classifier can be retained. These mappings could then be used instead of the random projections to reduce the dimensionality of the data. An important concept that enables us to find these mappings is that of a *manifold*.

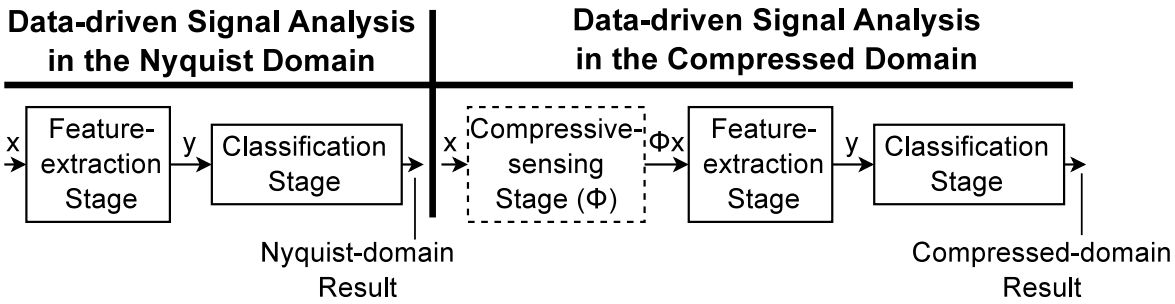


Figure 2.6: Transformation of computations in the sensing stage.

A manifold is a topological space that captures the essential structure of high-dimensional data with fewer dimensions. In other words, data that possess merely K intrinsic degrees of freedom can be assumed to lie on a K-dimensional manifold in the high-dimensional ambient space. The claim is that projections on to this manifold are sufficient for accurate classification [128]. The problem of *manifold learning* thus focuses on identifying the manifold model, given high-dimensional training data. Once identified, any point on a manifold can be represented using essentially K pieces of information derived through a *new mapping function* (similar to the random projection used in compressive sensing). Several manifold learning algorithms exist that help us determine the mapping function, such as Isomap [129], locally linear embedding [130], multiscale projections [131], and Hessian eigenmaps [132]. Some approaches [133], [128], however, do not derive a new mapping but, instead, study the generalization error in manifold learning when we use random projections for dimensionality reduction. These latter approaches are similar to the methods presented in the previous section.

Another interesting approach is presented in [134], where the authors incorporate the discrimination power of a classifier into the optimization problem of Eq. (2.1) for sparse representation. The idea in this case is different than compressive sensing since it aims to directly exploit a sparse representation for signal classification. The objective is to find a K-sparse representation $\mathbf{s}$ of an observed signal $\mathbf{x}$ that is as discriminative as possible between the different signal classes. Thus, this approach involves deriving a new dictionary matrix

$\mathbf{\Psi}$ that efficiently represents $\mathbf{x}$, while simultaneously allowing classification directly with the sparse representation of $\mathbf{x}$. The details of this approach are as follows.

Given a set of D signals in a signal matrix $\mathbf{X} = [\mathbf{x_1}, \mathbf{x_2}, \ldots, \mathbf{x_D}]$ with the corresponding representation in dictionary $\mathbf{\Psi}$ as $\mathbf{S} = [\mathbf{s_1}, \mathbf{s_s}, \ldots, \mathbf{s_D}]$, of which D_i samples are in class C_i, $1 \leq i \leq C$. The mean m_i and variance σ_i^2 for class C_i can then be computed in the sparse space as follows:

$$m_i = \frac{1}{D_i} \sum_{\mathbf{s} \in C_i} \mathbf{s}, \quad \text{and} \quad \sigma_i^2 = \frac{1}{D_i} \sum_{\mathbf{s} \in C_i} \| \mathbf{s} - m_i \|_2^2 \tag{2.7}$$

The mean of all samples is defined as $m = \frac{1}{D} \sum_{i=1}^{D} \mathbf{s_i}$. The Fisher discriminant ratio can then be defined as:

$$F(\mathbf{S}) = \frac{S_D}{S_V} = \frac{\| \sum_{i=1}^{D} D_i (m_i - m)(m_i - m)^T \|_2^2}{\sum_{i=1}^{D} \sigma_i^2} \tag{2.8}$$

The *difference between sample means* S_D can be interpreted as the inter-class distance and *sum of variance* S_V similarly can be interpreted as the interclass scatter. Maximizing $F(\mathbf{S})$ thus implies improved classifiability of the data. Fisher's criterion is motivated by the intuitive idea that the discrimination power is maximized when the spatial distribution of different classes is as far away as possible and the spatial distribution of samples from the same class is as close as possible [135].

In [134], the optimization problem of Eq. (2.1) is modified to incorporate $F(\mathbf{S})$, resulting in the following objective function:

$$J(\mathbf{S}, \mathbf{\Psi}, \lambda_1, \lambda_2) = F(\mathbf{S}) - \lambda_1 \sum_{i=1}^{D} \| \mathbf{s_i} \|_0 - \lambda_2 \sum_{i=1}^{D} \| \mathbf{x_i} - \mathbf{\Psi s_i} \|_2^2 \tag{2.9}$$

where λ_1 and λ_2 are positive scalar weighting factors chosen to adjust the tradeoff between the discrimination power, sparsity, and sparse approximation error. The authors also present an algorithm for maximizing $J(\mathbf{S}, \mathbf{\Psi}, \lambda_1, \lambda_2)$, which ensures a sparse representation for direct classification.

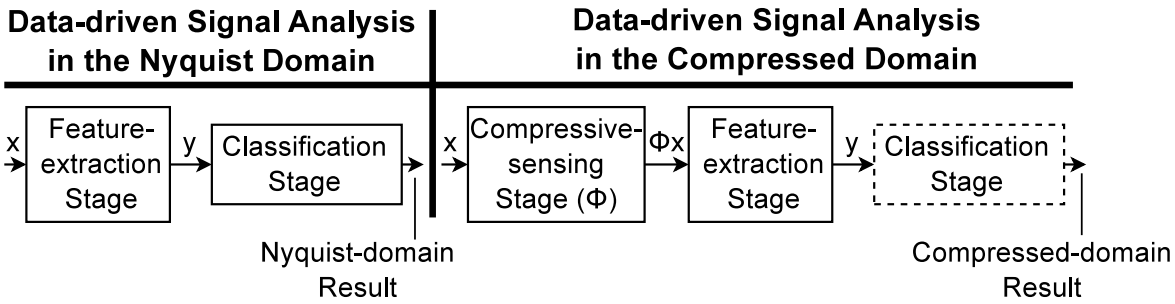


Figure 2.7: Transformation of computations in the classification stage.

A downside of the methods presented in this section is clearly that the ±1 structure of the projection matrix ($\mathbf{\Phi}$) could be compromised, thereby potentially increasing the energy costs of compression. These approaches may thus be undesirable in energy-constrained networks, such as those discussed in Sec. 1.1.

2.4.3 Transforming the Classification Stage

Since tampering with the sensing matrix can potentially increase the energy for data compression, an alternative approach is to transform the classification computations as shown in Fig. 2.7. In this section, we present related work in this direction. We show that existing methods attempt to derive new classifiers, which are not necessarily data driven.

In [136], Davenport *et al.* provide high-probability bounds (over the choice of the random projection matrix $\mathbf{\Phi}$) on the detection, classification, and filtering of signals from few measurements when the *set of potential signals is known a priori*. We present details of their approach for filtering in Sec. 2.4.4 and for detection/classification below. The authors consider signal classification performance for a single test point using the Neyman-Pearson (NP) detector. They transform computations in the Nyquist-domain detector so that it can act directly upon compressively-sensed data. Their approach is from the perspective of signal detection. The details of their formulation follow.

The authors begin by examining the problem of distinguishing between two hypotheses

$$\mathcal{H}_0 \quad : \quad \mathbf{y} = \mathbf{\Phi n} \tag{2.10}$$

$$\text{and } \mathcal{H}_1 \quad : \quad \mathbf{y} = \mathbf{\Phi}(\mathbf{x} + \mathbf{n}) \tag{2.11}$$

where $\mathbf{x}$ is an N-dimensional signal, $\mathbf{n} \sim N(0, \sigma^2 I_N)$ is *i.i.d.* Gaussian noise with variance σ^2, and $\mathbf{\Phi}$ is the random projection matrix used in compressive sensing. Observe that the hypotheses $\mathcal{H}_0$ and $\mathcal{H}_1$ have the probability density functions $f_0(y)$ and $f_1(y)$, respectively, which are defined as follows

$$f_0(y) \quad = \quad \frac{exp\left[-\frac{1}{2}\mathbf{y^T}\left(\sigma^2\mathbf{\Phi\Phi}\right)^{-1}\mathbf{y}\right]}{\left|\sigma^2\mathbf{\Phi\Phi^T}\right|^{1/2}(2\pi)^{M/2}} \tag{2.12}$$

$$\text{and } f_1(y) \quad = \quad \frac{exp\left[-\frac{1}{2}(\mathbf{y}-\mathbf{\Phi x})^\mathbf{T}\left(\sigma^2\mathbf{\Phi\Phi}\right)^{-1}(\mathbf{y}-\mathbf{\Phi x})\right]}{\left|\sigma^2\mathbf{\Phi\Phi^T}\right|^{1/2}(2\pi)^{M/2}}. \tag{2.13}$$

The NP-optimal decision rule is to compare the ratio $\Lambda(y) = f_1(y)/f_0(y)$ to a threshold η, *i.e.*, the following likelihood ratio test [135]

$$\Lambda(y) = \frac{f_1(y)}{f_0(y)} \underset{\mathcal{H}_0}{\overset{\mathcal{H}_1}{\gtrless}} \eta.$$

By taking logarithms, the authors obtain the following simplified ratio test (with a new threshold γ)

$$\mathbf{y^T}\left(\mathbf{\Phi\Phi^T}\right)^{-1}\mathbf{\Phi x} \underset{\mathcal{H}_0}{\overset{\mathcal{H}_1}{\gtrless}} \sigma^2\log(\eta) + \frac{1}{2}\mathbf{x^T\Phi^T}\left(\mathbf{\Phi\Phi^T}\right)^{-1}\mathbf{\Phi x} := \gamma. \tag{2.14}$$

Thus, they transform the Nyquist-domain NP detector and define a new compressed-domain detector as follows

$$t \; := \; \mathbf{y^T}\left(\mathbf{\Phi\Phi^T}\right)^{-1}\mathbf{\Phi x}. \tag{2.15}$$

The authors provide error bounds for this detector and show that the performance degradation is $O(\sqrt{N/M})$. They highlight the relationship between the detector performance

and the SNR of the input signal and show that at sufficient SNR values (above 25 decibel), the detection rate can be 100% at compression factors of up to 25×. Since, distinguishing between two signals $\mathbf{\Phi(x_0 + n)}$ and $\mathbf{\Phi(x_1 + n)}$ is equivalent to distinguishing $\mathbf{\Phi(x_0 + n) - \Phi(x_0) = \Phi(n)}$ from $\mathbf{\Phi(x_1 - x_0 + n)}$, they also generalize the above formulation to a binary classification problem.

Transforming computations in a maximum-likelihood classifier (MLC) has been explored in [137]. The authors show that classifying a compressively-sensed signal amounts to correlating it with a compressed (or smashed) representation of a template. Drawing a parallel with a matched filter used in the Nyquist domain for correlation, they call their compressed-domain approach the *smashed filter*. The details of their formulation follow.

Suppose an N-dimensional signal $\mathbf{x}$ belongs to one of P classes $C_i, i = 1, \ldots, P$ and we let hypothesis $\mathcal{H}_i$ signify that the signal $\mathbf{x}$ belongs to class C_i, and further assume that class C_i contains a single *known signal* $\mathbf{z_i}$. With noisy measurements $\mathbf{y = x + n}$, the MLC classifies signals according to which class has the maximum class-conditional likelihood as follows

$$\underset{i=1,\ldots,P}{\operatorname{argmax}}\ p\left(\mathbf{y}|\mathcal{H}_i\right) \tag{2.16}$$

where $p\left(\mathbf{y}|\mathcal{H}_i\right)$ is the probability distribution for the measured signal under hypothesis $\mathcal{H}_i$. Assuming an additive white Gaussian noise model with variance σ for $\mathbf{n}$, $p\left(\mathbf{y}|\mathcal{H}_i\right)$ can be denoted as

$$p\left(\mathbf{y}|\mathcal{H}_i\right) = \frac{1}{(2\pi\sigma)^{N/2}} e^{-\frac{1}{2\sigma}\| \mathbf{y}-\mathbf{z_i} \|_2^2}. \tag{2.17}$$

Given the above probability distribution, Eq. (2.16) simplifies to the following

$$\underset{i=1,\ldots,P}{\operatorname{argmin}} \| \mathbf{y} - \mathbf{z_i} \|_2^2, \tag{2.18}$$

which is the familiar nearest-neighbor problem. The above equation essentially implies correlating $\mathbf{y}$ with known templates $\mathbf{z_i}$ using a matched filter. A class assignment is then

made to $\mathbf{y}$ based on the maximum correlation value. Similarly, the authors show that in the compressed domain (*i.e.*, when $\mathbf{y} = \mathbf{\Phi x}$), the MLC simplifies to the following

$$\underset{i=1,\ldots,P}{\operatorname{argmin}} \| \mathbf{y} - \mathbf{\Phi z_i} \|_2^2. \tag{2.19}$$

The optimization problem above again represents correlation of $\mathbf{y}$, but this time with a compressed (smashed) version of $\mathbf{z_i}$. In the paper, the authors also generalize the above classifier to the case when template $\mathbf{z_i}$ is derived from a parameterized function $f(\theta)$, where θ is the control parameter. They demonstrate that the required number of measurements for accurate classification grows sublinearly with the number of classes. With σ in the 0.001 – 0.02 range, they demonstrate that for an image classification problem, their compressed-domain classifier can retain accuracies equal to the Nyquist domain up to compression factors of approximately 100×.

Although transforming computations in the classification stage can help retain the performance of compressed-domain signal analysis up to very high compression factors, the classification models in previous work are not data driven. Further, transformations are proposed for only specific classifiers. Besides, these approaches do not consider the feature-extraction stage, which is critical to improving the performance of detectors in several application domains.

2.4.4 Transforming the Feature-extraction Stage

Our study aims to investigate the potential of practical sensing and signal-processing systems that are based on representations derived from random projections. This raises two distinct aspects. First, system energy becomes a primary driver of our methodologies and analyses (as opposed to approaches in Sec. 2.4.2 that transform the sensing stage). Second, our methodologies attempt to broaden the scope of this idea to other signal-processing operations (as opposed to approaches in Sec. 2.4.3 that transform specific classification com-

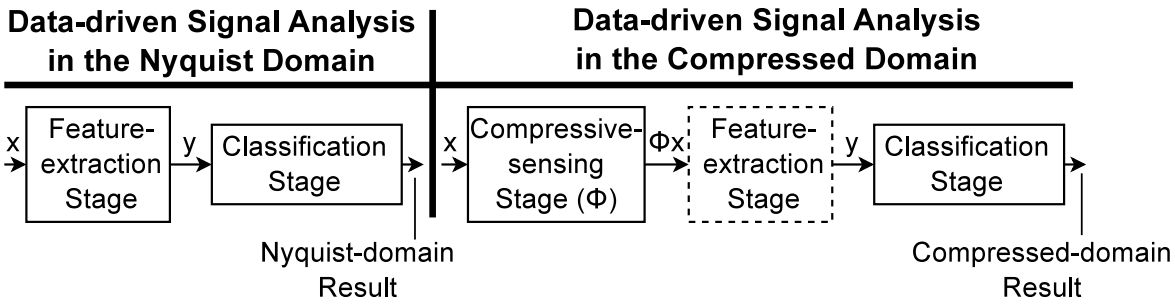


Figure 2.8: Transforming of computations in the feature-extraction stage.

putations). As shown in Fig. 2.8, our methodology focuses on transforming computations in the feature-extraction stage, which is important for enabling specific signal-analysis capabilities in applications. For instance, in medical sensors, discriminative analysis is used in many applications; however, typically, the algorithms require extracting specific signal features, which correspond to biomarkers for analysis [50, 102, 138–140]. Feature extraction is also a vital computation in several other application domains such as computer graphics [141], character recognition [142], medical imaging [143], image understanding [144], pattern recognition [145], and face detection [146]. Feature extraction involves signal-processing operations, which need to be performed before classification and cannot be addressed by previous work. Our investigations in this thesis suggest that a methodology to address all forms of linear signal processing is possible, while making energy-accuracy trade-offs explicit.

There are few attempts in the literature to address the problem of signal processing in the compressed domain. One peripherally related approach is presented in [136] where the authors perform signal filtering in the compressed domain. Their approach, however, is from the perspective of detection theory. Assuming knowledge about the desired signal, they demonstrate a formulation that allows them to remove interference from a measured signal in the compressed domain. The authors formalize this problem as follows

$$\mathbf{y} = \mathbf{\Phi}(\mathbf{x} + \mathbf{n}) \tag{2.20}$$

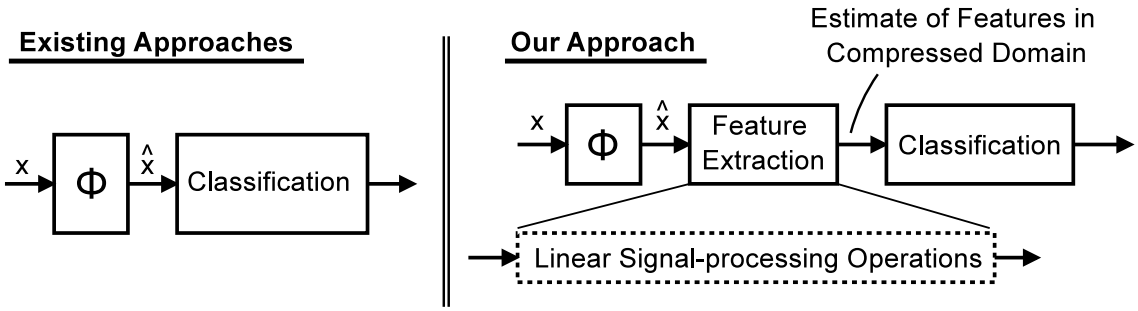


Figure 2.9: Unlike other approaches in the literature, we transform feature-extraction computations to the compressed domain. This approach coupled with a robust classifier results in an end-to-end performance comparable to the Nyquist domain.

where **y**, **x**, and **n** are the measured, desired, and interference signals, respectively. The authors only consider the case where either the interfering signal or the signal of interest lives in a known subspace. The key idea in this case is to use a filtering matrix **P** to project the measurements **y** into a subspace orthogonal to the interference, and thus eliminate the contribution of the interference to the measurements. Suppose $\mathbf{S_N}$ is the subspace of the interference signal **n** (which is assumed to be known) and $\boldsymbol{\Upsilon}$ is an orthogonal basis for $\mathbf{S_N}$. The authors show that the projection operator

$$\mathbf{P} = \mathbf{I} - (\boldsymbol{\Phi}\boldsymbol{\Upsilon})(\boldsymbol{\Phi}\boldsymbol{\Upsilon})^{\dagger} \tag{2.21}$$

where $(\boldsymbol{\Phi}\boldsymbol{\Upsilon})^{\dagger}$ represents the pseudo-inverse of $\boldsymbol{\Phi}\boldsymbol{\Upsilon}$, represents the interference-removing filter in the compressed domain. Clearly, the limitation of this approach is generalizability beyond the interference removal problem. Moreover, in a practical scenario, where signals are acquired from sensor nodes that monitor physical systems, it is not always possible to know anything beforehand about the characteristics of the desired or interfering signal. This restricts the applicability of the above approach even for filtering.

In this thesis, we transform linear signal-processing operations through a generalized formulation. We demonstrate our results for biomedical systems, where feature extraction results in the extraction of biomarkers. Biomarkers are features in a sensed signal

that are determined to have some correlation with the physiological states of interest in a clinical application. The extraction and representation of these signal features is thus a critical aspect of medical detectors [102]. Fig. 2.9 illustrates our objective. While most approaches referenced in this chapter have focused on the performance of various classifiers after processing input vectors through a random projection (see Fig. 2.9 on the left), our approach focuses on estimating specific features directly from compressed input signals. Note that we do not perform signal reconstruction at any stage. Rather, we transform linear signal-processing operations (*e.g.*, FIR filtering, wavelet transforms, *etc.*) so that they can be applied directly to compressed-signal representations. The resulting feature estimates are coupled with a data-driven classifier for accurate end-to-end performance. Next, we present details of our methodology and as a case study apply it to a system for epileptic seizure detection.

Chapter 3

Transformation to Compressed Domain with Least-squares Approximation

Storage, energy, and bandwidth constraints in low-power medical sensor networks raise the need for retaining efficient signal representations throughout the network. In order to achieve this goal, in this chapter, we present a methodology that transforms signal-processing functions to the compressed domain. Our approach thus enables signal analysis without reconstruction. We derive a projection of the processed signals using a least-squares approximation and exploit the properties of random projections to estimate signal features. Since mutual information is key to accurate classification of data, we also present a validation of our approach based on the analysis of mutual information in the estimated signal features. We show that the resulting estimates retain information up to high compression factors. By taking advantage of a flexible data-driven classifier, our approach thus enables accurate end-to-end performance. Further, we show that as the number of samples in the original sensed signal increases, the error in the feature estimates diminishes. We demonstrate our methodology with a specific focus on a continuous seizure-detection system based on EEG sensing, which employs spectral-energy features as biomarkers. The compressed-domain system we propose addresses two simultaneous clinical requirements:

(1) the acquisition of EEG signals for offline analysis by clinical experts, and (2) the accurate online detection of seizures for patient charting and eventual closed-loop therapeutic-stimulation protocols.

3.1 Introduction

Due to tremendous advances in medical sensors, a wide range of physiologically-indicative signals has become accessible through low-power recording modalities. This has led to the possibility of creating systems that can both detect clinically-important events on the sensor node and continuously acquire signals for offline analysis by clinical experts.

The combination of offline expert analysis and real-time on-sensor detection is becoming the preferred way to manage chronic diseases such as epilepsy [147]. Systems are thus beginning to emerge that can perform real-time sensing of EEG and detection of seizures [78, 93, 147]. However, such systems do not support acquisition, transmission or storage of waveforms for expert analysis. Fig. 3.1 illustrates the challenges associated with retaining waveforms in such systems. Since storage is limited in typical on-scalp sensors, waveforms are transmitted to an intermediate gateway device, which, in turn, transmits them for remote analysis. The figure (at the bottom) shows that the energy requirements for communicating waveforms from the sensor to a gateway device over a Zigbee or a bluetooth radio can be severe even for moderate channel counts [4–6].

Efficient signal compression, such as that enabled by compressive sensing, can help alleviate these constraints (as also illustrated in Fig. 3.1). Relating back to Fig. 1.3, some researchers have presented RA systems that exploit compressive sensing to only acquire waveforms for remote analysis [10, 148]. Others have proposed NA systems that can only do signal analysis or real-time detection [4]. The CA methodology we develop in this chapter enables efficient signal representations based on compressive sensing to be ex-

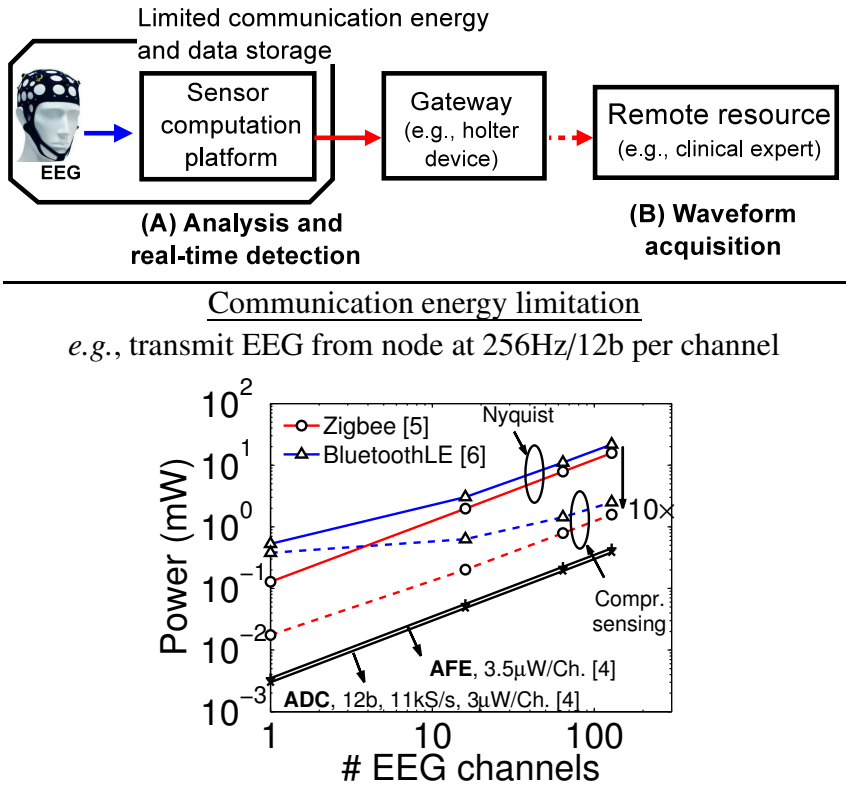


Figure 3.1: Energy limitations posed by waveform transmission. Zigbee and Bluetooth energies are shown for raw EEG transmission and compressed EEG transmission. Analog-front-end (AFE) and analog-to-digital conversion (ADC) energies [4] are also shown for comparison.

ploited throughout the system, which makes chronic EEG waveform acquisition possible *while simultaneously enabling on-node signal analysis*.

3.2 Background

In this section, we present a survey of some algorithms for seizure detection in the Nyquist domain along with background on SVM classifiers.

3.2.1 Nyquist-domain Algorithms for Seizure Detection

Several approaches for seizure detection have been proposed in the literature. Detection with probabilistic modeling has been explored in [149] and [150]. Detection using auto

regression and maximum-likelihood estimators has also been explored [151, 152]. In this work, we focus on a state-of-the-art patient-specific seizure onset detector, which has the structure of linear processing and energy extraction followed by classification for seizure detection. Linear processing implies that we are able to multiply the EEG vectors by a matrix to derive the signal features. Our approach is thus extensible to seizure-detection methods, which employ other linear processing functions, such as discrete wavelet transform (DWT), finite-impulse response (FIR) filters, Fourier transform, *etc.* Our approach, however, does not apply to feature-extraction methods, which cannot be represented as a linear matrix operation, *e.g.,* probabilistic modeling and auto-regression.

It has been shown that EEG spectral energy (derived after linear FIR filtering) can serve as a biomarker that indicates the onset of a seizure [147]. We thus focus on an algorithm that employs spectral-energy features for seizure detection [147]. We refer to this algorithm, henceforth, as the baseline Nyquist-domain algorithm for seizure detection. Table 3.1 shows a comparison of the baseline algorithm with several others from the literature. The performance of the detectors is characterized using the metrics of detection latency, sensitivity, and specificity. Latency refers to the delay between an expert-identified electrographic onset and the detector's recognition of seizure onset. By definition, this latency is a non-negative number and ideally close to zero. Sensitivity refers to the percentage of test seizures identified by a detector and is desired to be close to 100%. The specificity is represented by the number of false alarms per hour, which refers to the number of times, over the course of an hour, that the detector declares the onset of seizure activity in the absence of an actual seizure. Ideally, the number of false alarms should be close to zero. Table 3.1 shows that spectral-energy features with an SVM can achieve improved accuracy with a low detection latency.

Table 3.1: Comparison of Nyquist-domain methods for seizure detection.

Algorithm	Data (H, S, P)[1]	Features + Detection Method	Performance		
			Specificity	Latency	FA/hr.
[153]	(652, 126, 28)	DWT + probabilistic	78%	9.85 sec.	0.86
[154]	(29.7, 47, 12)	Spatio-temporal + NN	100%	9.35 sec.	0.03
[155]	(1360, 91, 57)	DWT + SVM	96%	1.60 sec.	0.45
[147]	(558, 148, 21)	Spectral energy + SVM	96%	4.59 sec.	0.15

[1]**H,S,P:** Hours, #Seizures, and # Patients; **NN:** Nearest Neighbor; **FA/hr.:** False alarms/ hr.

3.2.2 Support-vector Machine Algorithm

A range of inference algorithms has been shown to be effective for deriving models from complex datasets associated with physical systems [115, 156–158]. The SVM algorithm has emerged as an important candidate [159], particularly in biomedical applications. The SVM algorithm consists of two phases: training and classification. Training results in a detection model consisting of a set of vectors, called *support vectors* (SVs), which distinguish positive data classes from the negative ones. The SVM sequential minimal optimization algorithm is described in [160, 161] and in the excellent tutorial by Chris Burges [116]. The following provides an overview of the training phase of the algorithm. Given n vectors $\vec{\mathbf{x}_\mathbf{i}}$, $i \in \{1, 2 \ldots n\}$ of training data, each with dimensionality D_{SV}, and corresponding labels y_i, the algorithm finds a subset of training vectors that effectively separates the data into classes indicated by labels y_i. This subset, denoted by $\vec{\mathbf{sv}}_j$, $j = 1, \ldots, N_{SV}$, constitutes the N_{SV} SVs. The SVs are such that the distance between them and the nearest vectors in different classes are maximized [114]. Finding the SVs is a quadratic minimization problem with the following objective function [161]:

$$W(\alpha) = \frac{1}{2}\left[\sum_{i=1}^{n} \alpha_i y_i \sum_{j=1}^{n} \delta_j y_j K(\vec{\mathbf{x}_\mathbf{i}}, \vec{\mathbf{x}_\mathbf{j}})\right] - \sum_{i=1}^{n} \alpha_i,$$

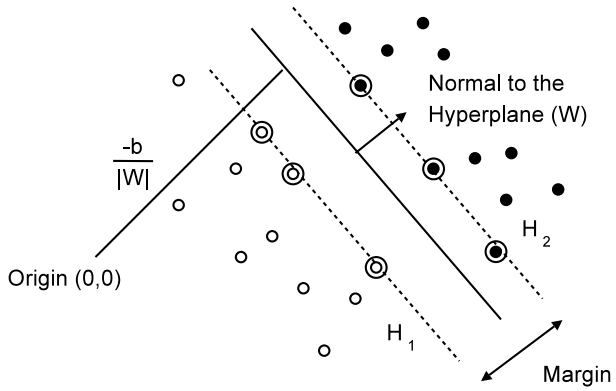


Figure 3.2: Hyperplanes for the separable case of an SVM classifier. The SVs are circled. W is normal to the minimum-margin hyperplane and $|b|/||W||$ is the perpendicular distance from the hyperplane to the origin.

subject to the constraint $\sum_{i=1}^{n} \delta_i y_i = 0$. Once all δ_i are determined, the SVs $\mathbf{s\vec{v}_i}$ can be calculated using only those $\mathbf{\vec{x}_i}$'s for which the weights δ_i are non-zero. $K(\mathbf{\vec{x}_i}, \mathbf{\vec{x}_j})$ is the kernel function that transforms data vectors ($\mathbf{\vec{x}_i}$ and $\mathbf{\vec{x}_j}$) into a secondary space and often comprises nonlinear operations. One of the most common kernels is the radial basis function (RBF), which is of the form:

$$K(\mathbf{\vec{x}_i}, \mathbf{\vec{x}_j}) = e^{-\gamma||\mathbf{\vec{x}_i} - \mathbf{\vec{x}_j}||^2},$$

where γ is a parameter that controls how aggressively the model fits the training data. Other possible kernel functions are: (i) polynomial homogeneous: $K(\mathbf{\vec{x}_i}, \mathbf{\vec{x}_j}) = (\mathbf{\vec{x}_i} \cdot \mathbf{\vec{x}_j})^d$, (ii) polynomial inhomogeneous: $K(\mathbf{\vec{x}_i}, \mathbf{\vec{x}_j}) = (\mathbf{\vec{x}_i} \cdot \mathbf{\vec{x}_j} + \beta)^d$, and (iv) hyperbolic tangent: $K(\mathbf{\vec{x}_i}, \mathbf{\vec{x}_j}) = \tanh(\kappa \mathbf{\vec{x}_i} \cdot \mathbf{\vec{x}_j} + c)$, where c, d, β, and κ are training parameters. Model construction with an SVM thus involves selecting data vectors in a feature space that separate positive-class instances from the negative-class ones. The nonlinear kernel transformation enables this separation with more flexibility in a secondary subspace. A visualization of the SVM training algorithm is provided in Fig. 3.2.

In a practical SVM implementation for embedded systems, although training can be done offline, classification, through the application of the SV model, must be performed in real-time for signal analysis. The actual classification computation is shown below [for the

RBF and polynomial transformation kernels]:

$$\text{Data Class} = sgn\left[\sum_{i=1}^{N_{SV}} K\left(\vec{\mathbf{x}} \cdot \vec{\mathbf{sv_i}}\right) \alpha_i \delta_i - b\right] \tag{3.1}$$

$$\text{where } K\left(\vec{\mathbf{x}}, \vec{\mathbf{sv_i}}\right) = \begin{cases} exp(-\gamma||\vec{\mathbf{x}} - \vec{\mathbf{sv_i}}||^2) & \text{RBF kernel} \\ F(\vec{\mathbf{x}} \cdot \vec{\mathbf{sv_i}} + \beta)^d & \text{Poly. kernel} \end{cases}$$

Here, *sgn*[] is the *signum function*, $\vec{\mathbf{x}}$ is the FV to be classified, and $\vec{\mathbf{sv_i}}$ is the i^{th} SV (b, d, α_i, β, γ, and δ_i are training parameters).

3.3 Overview of the Proposed Approach

As mentioned in the previous chapter, signal-analysis algorithms typically base their decision rules on key features extracted from the signals *via* signal-processing functions; this is particularly true for medical detectors, where the features correspond to physiological biomarkers. These algorithms then use a classifier to perform inference over the extracted features. Powerful classification frameworks exist in the domain of machine learning that can construct high-order and flexible models through data-driven training. In many such frameworks, the classification step utilizes a distance metric (*e.g.*, 2-norm or inner product) between feature vectors [162, 163]. In certain cases, the distance metric may also be invoked within the feature extraction step; for instance, to extract spectral-energy biomarkers from physiological signals (such as the EEG). In this chapter, we demonstrate our results for the latter case using a seizure-detection application, where clinical studies have shown that EEG spectral energy (derived using the inner product between feature vectors after linear FIR filtering) can serve as a biomarker that indicates the onset of a seizure [147]. In fact, spectral features are generic biomarkers for neural field potentials and are relevant

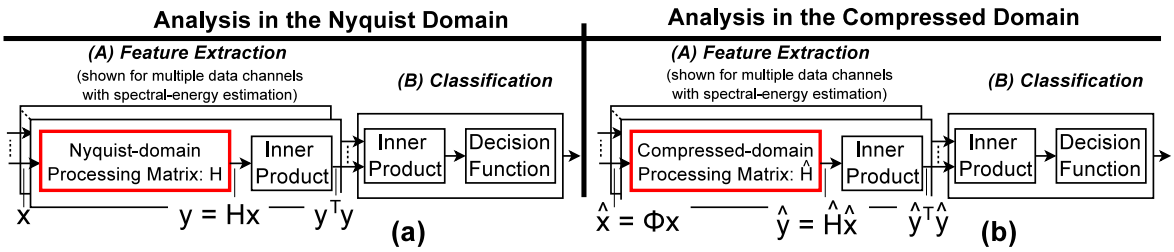


Figure 3.3: Transforming any signal-processing function, which can be represented as a matrix $\mathbf{H}$, into an equivalent operator $\hat{\mathbf{H}}$ in the compressed domain. This needs a solution for a projection of $\mathbf{y}$, which preserves the inner product of vectors.

for a broad range of neurological applications (*e.g.*, brain-machine interfaces [91], sleep disorders [164], *etc.*).

Connecting the above concepts with the multi-channel system shown in Fig. 3.3, in the Nyquist domain [Fig. 3.3(a)], an N-dimensional signal $\mathbf{x}$ from each data channel is multiplied with an $N \times N$ matrix operator $\mathbf{H}$ to perform linear signal processing to derive a feature vector $\mathbf{y}$. The inner product between feature vectors can then be used to derive the spectral-energy features from each data channel [as shown in Fig. 3.3(a), spectral-energy features can be denoted by the inner product $\mathbf{y^T y}$]. Extending this to an analysis in the compressed domain [Fig. 3.3(b)], we now aim to process compressed representations of the input signal, namely $\hat{\mathbf{x}} = \mathbf{\Phi x}$, where $\mathbf{\Phi}$ represents the $\frac{N}{\xi} \times N$ ($\xi \gg 1$) random-projection matrix used for compressive sensing. We thus seek to find a matrix transformation $\hat{\mathbf{H}}$ that leads to a representation of a signal that has been suitably processed as intended, but that is derived *by directly using* $\hat{\mathbf{x}}$.

When applied to the baseline seizure-detection system, our approach focuses on *transforming* signal-processing computations in order to enable extraction of specific spectral-energy features directly from compressed input signals (*i.e.*, our aim is to achieve $\hat{\mathbf{y}}^\mathbf{T}\hat{\mathbf{y}} \approx \mathbf{y^T y}$). We show first that mutual information is preserved in the computed estimate of the spectral features even at high compression factors, and second that the SVM enables high-performance detection as long as mutual information is preserved. Despite the flexibility offered by the SVM, however, in Sec. 3.5, we show that *regularization* is essential in or-

der to extract the energy estimates; regularization refers to the use of a transformation that specifically yields a random projection of the processed signal (we describe the requirements on the random projection in Sec. 3.5). We also show that the error of the estimates is substantially reduced as the number of samples in the original sensor signal increases. Our specific contributions in this chapter are as follows:

- We describe a methodology for transforming linear signal-processing operations into the compressed domain using a least-squares approximation. We demonstrate our methodology on a seizure detector based on spectral-energy features extracted from compressively-sensed EEG. Our methodology enables a system for local signal analysis and efficient signal acquisition. Previous work on analyzing compressively-sensed signals (described in Chapter 2) has investigated theoretical bounds for classifiers presented with compressed vectors. Our approach focuses on transforming computations, enabling key signal (spectral-energy) features to be extracted that correspond to the biomarkers for detection. This is critical to improving the performance of medical-detection algorithms [4, 102].

- We show that, in general, EEG signals can be represented by a small set of basis vectors. This representation enables us to apply the Johnson-Lindenstrauss (JL) lemma [110], with a quantitative understanding of the accuracy limits. As a result we can extract the energy of processed signals directly from their compressed representations.

- We mathematically validate the proposed approach, emphasizing the importance of the transformations used to process the compressed signals. We show that this leads to statistical characteristics in the matrices that are critical for deriving the energy estimates.

- We show that the absolute error of the computed features can be very low for large compression factors when the number of samples in the sensor signal is high. Thus, we investigate the scaling limits of our approach based on mathematical bounds.

- We analyze the hardware implications of implementing a detector using the proposed approach. The hardware operations required are compared to optimized Nyquist-domain methods, where optimizations, such as folded FIR implementations, are possible.

The rest of this chapter is organized as follows. In Sec. 3.4, we introduce the baseline Nyquist-domain seizure detector and describe the mathematics behind the proposed approach for transforming the detector. We provide an analytical rationale for the proposed approach in Sec. 3.5. In Sec. 3.6, we present experimental results. In Sec. 3.7, we study the reconstruction error of compressively-sensed EEG. In Sec. 3.8, we describe the potential of our approach for yielding high performance in applications involving raw sensor signals, which require a large number of samples. In Sec.3.9, we provide a hardware implementation analysis. Finally, we conclude in Sec. 3.10.

3.4 Seizure Detection Using Compressively-sensed EEG

In this section, we first describe the baseline Nyquist-domain seizure-detection algorithm we use in our study. The baseline algorithm employs spectral-energy features and an SVM classifier. Then, we specifically formulate feature-extraction processing in terms of matrix multiplications. This permits transformation to the compressed domain using the random projection matrix $\mathbf{\Phi}$, as we describe below.

Fig. 3.4 illustrates the baseline Nyquist-domain seizure detection algorithm, which relies on patient-specific classifier training; this has been shown to substantially improve seizure-detection performance for scalp-recorded EEG [147]. A two-second epoch of each EEG channel is processed using eight band-pass filters (BPFs) with passbands of 0-3 Hz,

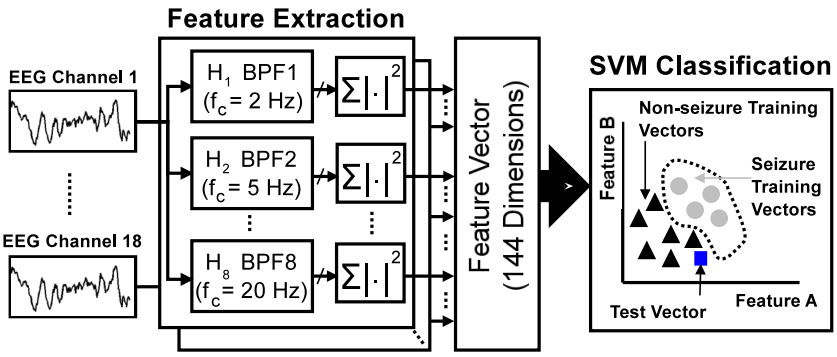


Figure 3.4: The baseline epileptic-seizure detection algorithm employing spectral-analysis feature extraction and SVM classification.

3-6 Hz, ..., 21-24 Hz. The energy from each filter is then represented by summing the squared value of the output samples to form a FV, which is then used for classification by an SVM.

The baseline detector is validated on 558 hours of EEG data from 21 patients (corresponding to 148 seizures) in the CHB-MIT database [165]. For every patient, up to 18 channels of continuous EEG is processed using eight BPFs, leading to an FV dimensionality of 144. The Nyquist-domain detector has been demonstrated to achieve an average latency, sensitivity, and specificity of 4.49 sec., 96.03%, and 0.1471 false alarms per hour, respectively [147].

In the baseline algorithm, suppose we let each epoch of Nyquist-sampled EEG from channel $j \in [1, 18]$ be denoted by an N-dimensional vector $\mathbf{x_j} \in \mathbb{R}^N$ and the eight BPFs by matrices $\mathbf{H_i} \in \mathbb{R}^{N \times N}$, $i \in [1, 8]$. As illustrated in Fig. 3.5(a), a filtering operation based on $\mathbf{H_i}$ can then be formulated as a linear multiplication given by

$$\mathbf{y_{ij}} = \mathbf{H_i x_j}, \tag{3.2}$$

where $\mathbf{y_{ij}} \in \mathbb{R}^N$ represents the filtered signal of N samples. To form the FV, the energy in each frequency band defined by $\mathbf{H_i}$ must then be computed. This can be achieved *via* the following operation:

$$y_{ij} = \mathbf{y_{ij}^T y_{ij}}, \tag{3.3}$$

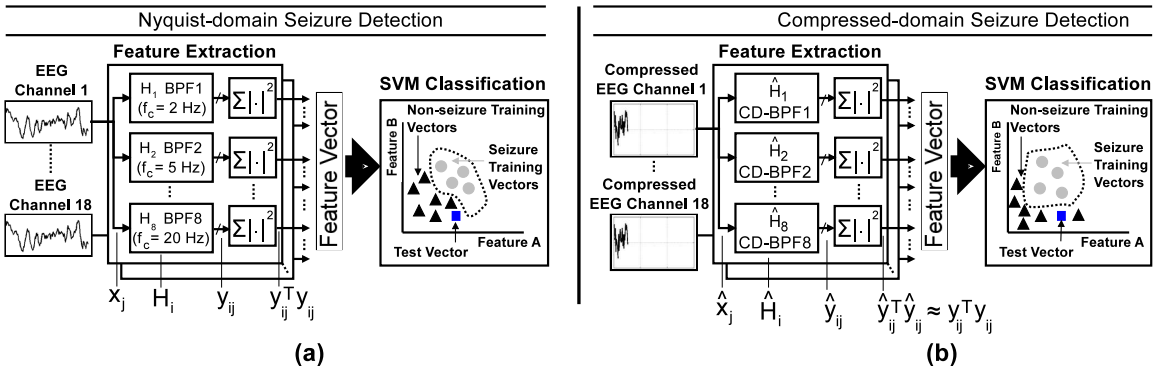


Figure 3.5: The seizure detection algorithm involves feature extraction and classification using an SVM. In the compressed domain, we transform the Nyquist-domain BPFs $\mathbf{H_i}$ to the compressed-domain BPFs $\hat{\mathbf{H}}_\mathbf{i}$.

where y_{ij} forms one dimension of the resulting FV. This operation is thus performed up to the maximum FV dimensionality of 144.

The formulation above makes possible the compressed-domain analysis system shown in Fig. 3.5(b). In this system, a corresponding matrix $\hat{\mathbf{H}}_\mathbf{i}$ is constructed such that the desired filtered signal can be represented by $\hat{\mathbf{y}}_\mathbf{ij}$, which is derived directly from the compressed input signal $\hat{\mathbf{x}}_\mathbf{j}$. The energy of $\mathbf{y_{ij}}$ can then be estimated from $\hat{\mathbf{y}}_\mathbf{ij}$ by exploiting a property of random projections. Each $\hat{\mathbf{H}}_\mathbf{i}$ thus effectively forms a *compressed-domain band-pass filter* (CD-BPF).

Suppose an epoch of compressively-sensed EEG from channel $j \in [1, 18]$ is represented by the M-sample signal $\hat{\mathbf{x}}_\mathbf{j} \in \mathbb{R}^\mathrm{M}$ with $M \ll N$ (for substantial data compression). The signal $\hat{\mathbf{x}}_\mathbf{j}$ can then be related to the Nyquist-domain signal $\mathbf{x_j}$ by

$$\hat{\mathbf{x}}_\mathbf{j} = \mathbf{\Phi x_j}, \tag{3.4}$$

where $\mathbf{\Phi}$ is the random projection matrix used for compressive sensing. The number of compressed samples M determines $\xi = N/M$. Next, we present methodologies to extract the signal features directly from $\hat{\mathbf{x}}_\mathbf{j}$ and estimate the spectral energy.

3.4.1 Feature-extraction Formulation

In the compressed-domain system of Fig. 3.5(b), the M-sample random projection of the filtered signal $\hat{\mathbf{y}}_{\mathbf{ij}}$ (which is derived from the CD-BPFs $\hat{\mathbf{H}}_{\mathbf{i}}$ that directly use $\hat{\mathbf{x}}_{\mathbf{j}}$) can be represented as follows:

$$\begin{aligned} \hat{\mathbf{y}}_{\mathbf{ij}} &= \hat{\mathbf{H}}_{\mathbf{i}}\hat{\mathbf{x}}_{\mathbf{j}} \\ &= \hat{\mathbf{H}}_{\mathbf{i}}\boldsymbol{\Phi}\mathbf{x}_{\mathbf{j}}. \end{aligned} \tag{3.5}$$

It may seem that one should thus attempt to derive an $\hat{\mathbf{H}}_{\mathbf{i}}$ that would yield $\hat{\mathbf{y}}_{\mathbf{ij}} = \mathbf{y}_{\mathbf{ij}}$, but we will see ahead in Sec. 3.5 that solving this equation for such an $\hat{\mathbf{H}}_{\mathbf{i}}$ would lead to a poor estimate of the FV. Instead, $\hat{\mathbf{y}}_{\mathbf{ij}}$ can appropriately represent the energy of $\mathbf{y}_{\mathbf{ij}}$ if it is chosen to be the random projection of $\mathbf{y}_{\mathbf{ij}}$, given by

$$\hat{\mathbf{y}}_{\mathbf{ij}} = \boldsymbol{\Phi}\mathbf{y}_{\mathbf{ij}}. \tag{3.6}$$

In Sec. 3.4.2, we will show a justification for this choice of $\hat{\mathbf{y}}_{\mathbf{ij}}$, but proceeding further, Eq. (3.6) leads to the following relationship [from Eqs. (3.2) and (3.5)]:

$$\hat{\mathbf{H}}_{\mathbf{i}}\boldsymbol{\Phi}\mathbf{x}_{\mathbf{j}} = \boldsymbol{\Phi}\mathbf{H}_{\mathbf{i}}\mathbf{x}_{\mathbf{j}}, \tag{3.7}$$

Then, in order to determine $\hat{\mathbf{H}}_{\mathbf{i}}$, we need to solve the set of linear equations represented by the following matrix relationship:

$$\hat{\mathbf{H}}_{\mathbf{i}}\boldsymbol{\Phi} = \boldsymbol{\Phi}\mathbf{H}_{\mathbf{i}} \Leftrightarrow \mathbf{H}_{\mathbf{i}}^{\mathbf{T}}\boldsymbol{\Phi}^{\mathbf{T}} = \boldsymbol{\Phi}^{\mathbf{T}}\hat{\mathbf{H}}_{\mathbf{i}}^{\mathbf{T}}. \tag{3.8}$$

However, with $M \ll N$, matrix $\hat{\mathbf{H}}_{\mathbf{i}}^{\mathbf{T}}$ corresponds to $M \times M$ unknowns constrained by $N \times M$ equations. Such a system with fewer unknowns than equations is considered overdetermined and has no exact solution. We can, however, solve Eq. (3.8) in the least squares

sense as follows:

$$\underset{\hat{\mathbf{H}}_\mathbf{i}}{\operatorname{argmin}} \|\mathbf{H}_\mathbf{i}^\mathbf{T}\mathbf{\Phi}^\mathbf{T} - \mathbf{\Phi}^\mathbf{T}\hat{\mathbf{H}}_\mathbf{i}^\mathbf{T}\|_2^2. \tag{3.9}$$

To solve the least-squares problem in Eq. (3.9), we set the derivative of the Euclidean-norm to zero. This approach of solving overdetermined systems is also known as the solution with normal equations [166]. From Eq. (3.9), we thus have

$$\begin{aligned}\|\mathbf{H}_\mathbf{i}^\mathbf{T}\mathbf{\Phi}^\mathbf{T} - \mathbf{\Phi}^\mathbf{T}\hat{\mathbf{H}}_\mathbf{i}^\mathbf{T}\|_2^2 &= \left(\mathbf{H}_\mathbf{i}^\mathbf{T}\mathbf{\Phi}^\mathbf{T} - \mathbf{\Phi}^\mathbf{T}\hat{\mathbf{H}}_\mathbf{i}^\mathbf{T}\right)^\mathbf{T}\left(\mathbf{H}_\mathbf{i}^\mathbf{T}\mathbf{\Phi}^\mathbf{T} - \mathbf{\Phi}^\mathbf{T}\hat{\mathbf{H}}_\mathbf{i}^\mathbf{T}\right) \\ &= \mathbf{\Phi}\mathbf{H}_\mathbf{i}\mathbf{H}_\mathbf{i}^\mathbf{T}\mathbf{\Phi}^\mathbf{T} - \mathbf{\Phi}\mathbf{H}_\mathbf{i}\mathbf{\Phi}^\mathbf{T}\hat{\mathbf{H}}_\mathbf{i}^\mathbf{T} - \hat{\mathbf{H}}_\mathbf{i}\mathbf{\Phi}\mathbf{H}_\mathbf{i}^\mathbf{T}\mathbf{\Phi}^\mathbf{T} + \hat{\mathbf{H}}_\mathbf{i}\mathbf{\Phi}\mathbf{\Phi}^\mathbf{T}\hat{\mathbf{H}}_\mathbf{i}^\mathbf{T}.\end{aligned} \tag{3.10}$$

To minimize the squared error in Eq. (3.9), we take the derivative of the right hand side of Eq. (3.10) with respect to $\hat{\mathbf{H}}_\mathbf{i}$ and set it to zero. We thus get

$$-\left(\mathbf{\Phi}\mathbf{H}_\mathbf{i}\mathbf{\Phi}^\mathbf{T}\right)^\mathbf{T} - \mathbf{\Phi}\mathbf{H}_\mathbf{i}^\mathbf{T}\mathbf{\Phi}^\mathbf{T} + 2\mathbf{\Phi}\mathbf{\Phi}^\mathbf{T}\hat{\mathbf{H}}_\mathbf{i}^\mathbf{T} = 0 \tag{3.11}$$

From this, we obtain

$$\begin{aligned} 2\mathbf{\Phi}\mathbf{\Phi}^\mathbf{T}\hat{\mathbf{H}}_\mathbf{i}^\mathbf{T} &= 2\mathbf{\Phi}\mathbf{H}_\mathbf{i}^\mathbf{T}\mathbf{\Phi}^\mathbf{T} \\ \implies \hat{\mathbf{H}}_\mathbf{i}\mathbf{\Phi}\mathbf{\Phi}^\mathbf{T} &= \mathbf{\Phi}\mathbf{H}_\mathbf{i}\mathbf{\Phi}^\mathbf{T} \qquad (3.12) \\ \text{Thus,} \quad \hat{\mathbf{H}}_\mathbf{i} &= \mathbf{\Phi}\mathbf{H}_\mathbf{i}\underbrace{\mathbf{\Phi}^\mathbf{T}\left(\mathbf{\Phi}\mathbf{\Phi}^\mathbf{T}\right)^{-1}} \\ &= \mathbf{\Phi}\mathbf{H}_\mathbf{i}\mathbf{\Phi}_\mathbf{R}^\dagger, \qquad (3.13)\end{aligned}$$

where $\mathbf{\Phi}_\mathbf{R}^\dagger \in \mathbb{R}^{N\times M}$ is called the Moore-Penrose right pseudo-inverse of $\mathbf{\Phi}$. The solution for $\hat{\mathbf{H}}_\mathbf{i}$ in Eq. (3.13) thus represents the best-fit hyperplane for the system of equations given in Eq. (3.8). This least-squares solution for each of the CD-BPFs $\hat{\mathbf{H}}_\mathbf{i}$ can also be directly written in a closed form from Eq. (3.8) by multiplying both sides of the equation by $\mathbf{\Phi}^\mathbf{T}\left(\mathbf{\Phi}\mathbf{\Phi}^\mathbf{T}\right)^{-1}$. Next, we describe how $\hat{\mathbf{y}}_\mathbf{ij}$, when chosen to be a random projection of $\mathbf{y}_\mathbf{ij}$ [as in Eq. (3.6)], can be used to estimate the energy of $\mathbf{y}_\mathbf{ij}$.

3.4.2 Spectral-energy Extraction from the Random Projection

The desired spectral energy can be derived in the Nyquist domain using the inner-product operation shown in Eq. (3.3). An important aspect of the proposed approach is that a corollary from the JL lemma [110] states that inner products are preserved under random projections. Eq. (3.6) allows us to exploit this property to approximate the inner product $(\mathbf{y}_{ij}^{\mathbf{T}}\mathbf{y}_{ij})$ of the filtered EEG signal by inner product $\left[\left(\mathbf{\Phi y}_{ij}\right)^{\mathbf{T}}\mathbf{\Phi y}_{ij}\right]$ of its random projection, hence justifying the choice of $\hat{\mathbf{y}}_{ij} = \mathbf{\Phi y}_{ij}$.

Lemma 1. *(Johnson-Lindenstrauss) Let $\epsilon \in (0,1)$ and $B \subset \mathbb{R}^N$ be a set of B_c vectors with $M = O[logB_c/(\epsilon^2\ log\epsilon)] \ll N$. Then, $\exists$ a mapping $T : \mathbb{R}^N \rightarrow \mathbb{R}^M$ such that for all $\mathbf{u}, \mathbf{v} \in B$*

$$(1-\epsilon)\left\|\mathbf{u}-\mathbf{v}\right\|_2^2 \leq \left\|f(\mathbf{u})-f(\mathbf{v})\right\|_2^2 \leq (1+\epsilon)\left\|\mathbf{u}-\mathbf{v}\right\|_2^2. \tag{3.14}$$

The JL lemma makes a compelling statement about the preservation of structure in a subspace of $\mathbb{R}^{\mathrm{N}}$. It states that although the absolute data values get affected due to a linear transformation T, the structure in the data is preserved in a lower-dimensional subspace. Thus, if we are only interested in the pairwise distances among a set of vectors up to a factor of $(1 \pm \epsilon)$, $\epsilon \ll 1$ and do not care about the actual vectors, we can embed the vectors in a lower-dimensional (M) subspace through projection T. As mentioned in the previous chapter, this property has also been the cornerstone of a large body of research including algorithms that exploit manifold structure in machine learning [112, 133, 167, 168]. A number of useful results can be derived from Eq. (3.14). One important corollary, which we will take advantage of, is the inner-product preservation property of random projections.

Corollary 1. *(JL corollary: Inner-product preservation) Let $\epsilon \in (0,1)$ and $B \subset \mathbb{R}^N$ be a set of B_c vectors with $M = O[logB_c/(\epsilon^2\ log\epsilon)] \ll N$. Further, let $\mathbf{\Omega} \in \mathbb{R}^{M\times N}$, where each entry is sampled i.i.d. from a Gaussian $N(0,1)$ or from $U(+1,-1)$. Then $\forall\ \mathbf{y_1}, \mathbf{y_2} \in B$,*

$$P\left(|\mathbf{y_1}\cdot\mathbf{y_2} - (\mathbf{\Omega y_1})\cdot(\mathbf{\Omega y_2})/M| > \epsilon\right) \leq 4e^{-(\epsilon^2-\epsilon^3)M/4}. \tag{3.15}$$

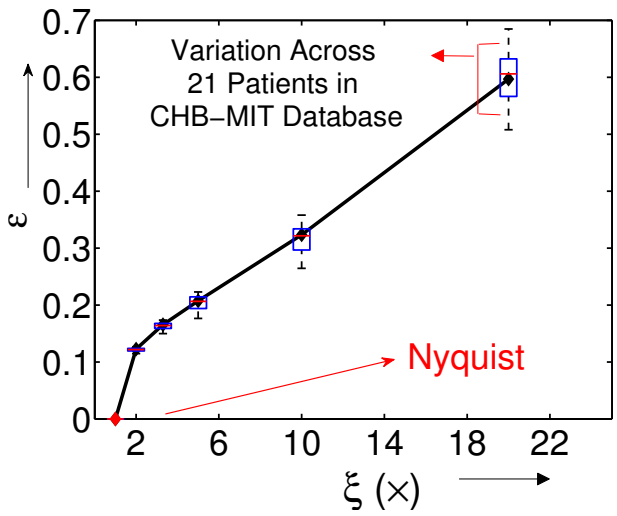


Figure 3.6: The average absolute error (ϵ) in the inner product of the projected epochs normalized to the corresponding Nyquist-domain inner products $\mathbf{y}_{\mathbf{ij}}^{\mathbf{T}}\,\mathbf{y}_{\mathbf{ij}}$ is $\sim$ 0.32 for ξ = 10×.

Note that in the corollary above, the original dimension (N) of the data is irrelevant as far as the reduced dimension (M) goes; it is the number of points [or the cardinality (B_c) of B] that is important.

It is postulated that the properties of the JL lemma are valid for small values of B_c (*i.e.*, if B is a subset of $\mathbb{R}^N$) [110]. For the seizure-detection application, we will show below that EEG data can be adequately represented by a small number (~95) of basis vectors. We thus need to preserve the pairwise distances only for this small basis set. We can therefore take advantage of the JL corollary by setting $\mathbf{\Omega}$, $\mathbf{y_1}$, and $\mathbf{y_2}$ equal to $\sqrt{M}\mathbf{\Phi}$, $\mathbf{y_{ij}}$, and $\mathbf{y}_{\mathbf{ij}}^{\mathbf{T}}$, respectively. We see from Eq. (3.15) that the spectral energy derived from $\mathbf{y_{ij}}$ [as in Eq. (3.3)] is thus preserved under the random projection $\mathbf{\Phi}$ [as in Eq. (3.6)] up to a scaling factor of $\sqrt{M}$ and error ϵ.

Fig. 3.6 shows the average value of the normalized absolute error ϵ across 21 patients in the CHB-MIT database (note that although the inner-product error has both positive and negative values, we study the absolute value of the error normalized to the corresponding Nyquist-domain inner products $\mathbf{y}_{\mathbf{ij}}^{\mathbf{T}}\,\mathbf{y}_{\mathbf{ij}}$). We observe an error of 0.12 in the inner products at ξ = 2×, which increases to 0.32 at ξ = 10× and 0.59 at ξ = 20×. Since the error depends

only on M, we will see ahead in Sec. 3.8 that the error diminishes for large compression factors ξ as the number of samples N in the original sensor signal increases. This, for example, corresponds to an increase in the signal sampling rate or an increase in the length of the epoch. Even for the signal parameters considered, though the error may seem substantial, we will show in Sec. 3.6.1 that the mutual information of the resulting features is preserved, leading to high end-to-end performance for the seizure detector.

Using the rationale above, each dimension of the FV is thus formed from the random projection of the processed signal according to the following relationship:

$$\hat{f}_{ij} \approx f_{ij} = \hat{\mathbf{y}}_{\mathbf{ij}}^{\mathbf{T}} \, \hat{\mathbf{y}}_{\mathbf{ij}}, \tag{3.16}$$

where $\hat{f}_{ij}$ is one dimension of the FV in the compressed domain. As in the Nyquist-domain case, this operation is performed with eight filters ($\hat{\mathbf{H}}_{\mathbf{i}}$) and 18 EEG channels ($\hat{\mathbf{x}}_{\mathbf{j}}$), leading to a total FV dimensionality of 144.

Representation of the Filtered EEG by a Small Basis Set. To invoke Corollary 1 for signal $\mathbf{y_{ij}}$ and its transpose $\mathbf{y}_{\mathbf{ij}}^{\mathbf{T}}$, we argued that the cardinality (B_c) of a basis set, which is required to represent all the filtered EEG vectors (corresponding to all 2 sec. epochs of a patient), is small. On an average, EEG recordings for 23 hrs. (totaling $\sim 40,000$ epochs) are available from one patient in the CHB-MIT EEG database [165]. Thus, in the Nyquist domain, after processing 18 channels with eight BPFs, there are a total of about $E = 6$M filtered EEG vectors [as obtained from Eq. (3.2)] for each patient. Since this number increases as more data are observed, it can potentially invalidate the use of the JL lemma for the estimate of Eq. (3.16), which is applicable to only a small number of basis vectors B_c [from Lemma 1, $B_c = O(\epsilon^{-M\epsilon^2})$ where M is the number of samples in the compressed signal and ϵ is the average normalized absolute error in the inner products]. In this section, we will show that the JL lemma is still valid for the seizure-detection application. For each

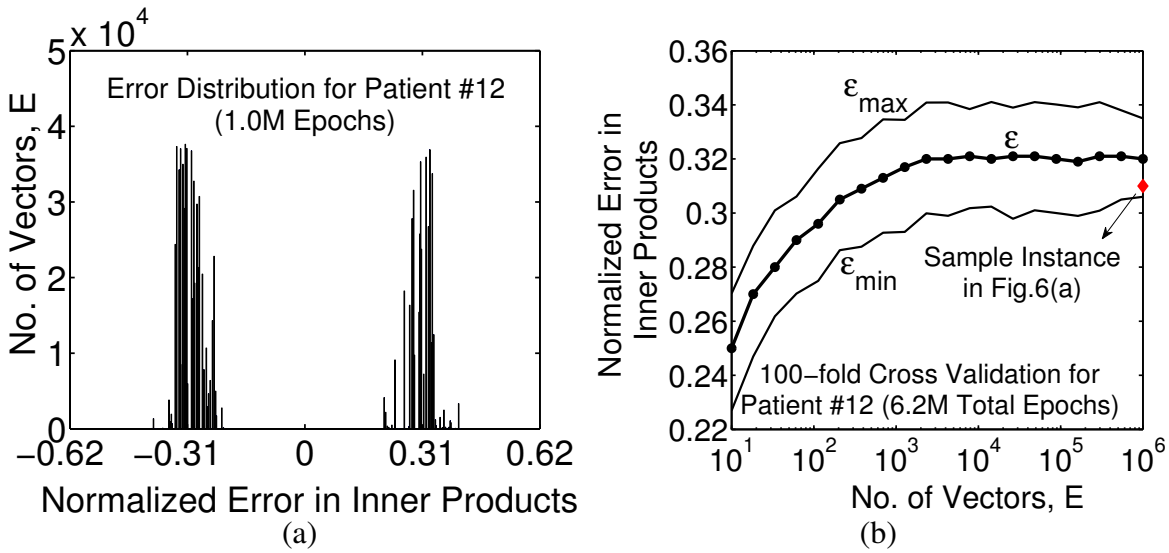


Figure 3.7: At $\xi = 10\times$ for Patient #12: (a) the normalized inner-product error has both positive and negative values. The normalized absolute error has $\epsilon_{mean} = 0.31$ and variance $< 3\%$ of the mean. (b) With a 100-fold cross-validation, we observe that ϵ saturates after $\sim 1k$ filtered EEG vectors.

patient, there is only a small set of basis vectors (< 175) underlying E, and the number of such vectors saturates, remaining constant with the number of data epochs.

First, we illustrate the result with a case study using Patient #12. This patient has a total of $E = 6.2$M filtered EEG vectors obtained from 24 hrs. of recorded data. Fig. 3.7(a) shows the distribution of the normalized error in the inner products using a subset of 1M filtered EEG vectors. The error values are computed at $\xi = 10\times$ and normalized to the corresponding Nyquist-domain inner products $\mathbf{y_{ij}^T}\,\mathbf{y_{ij}}$. The figure shows that the actual normalized error in the inner products has both positive and negative values. For this particular patient, the mean value of the normalized absolute error (ϵ_{mean}) is 0.31. To determine the true average of the error (ϵ) for 1M filtered EEG vectors, we compute ϵ_{mean} [as shown in Fig. 3.7(a)] for 100 different subsets of 1M vectors chosen randomly from the 6.2M superset. We then evaluate ϵ to be the average value of the hundred ϵ_{mean} estimates. This process is called a 100-fold cross-validation. For 1M filtered EEG vectors of Patient #12, the average error ϵ at $\xi = 10\times$ is plotted in Fig. 3.7(b) along with the deviation in the error around ϵ. Here, the experiment is repeated as the number of filtered EEG vectors in the subset is increased, re-

sulting in the profile shown. We observe that the average value of the normalized absolute error ϵ remains roughly constant beyond $\sim 1k$ vectors even though the JL lemma suggests that ϵ should increase with the number of vectors in the subset under consideration. To analyze this behavior, we proceed to determine the *true set of basis vectors* underlying the filtered EEG epochs.

Suppose, for each patient, we denote all filtered epochs E observed up to time $2E$ sec. by $\mathbf{y_{ijk}}$, where $i \in [1, 8]$ is the filter number, $j \in [1, 18]$ is the channel number, and $k \in [1, E]$ is the epoch number. Further, suppose we are able to represent each $\mathbf{y_{ijk}}$ using a set of B basis vectors denoted by $\{\mathbf{b_p}\}$, $p = 1, \ldots, B$. We can write the following relationship between the column vectors $\mathbf{y_{ijk}}$ and $\mathbf{b_p}$:

$$\mathbf{y_{ijk}} = \sum_{p=1}^{B} \alpha_p \mathbf{b_p}, \tag{3.17}$$

where α_p is a scalar weight applied to basis vector $\mathbf{b_p}$. The inner product of $\mathbf{y_{ijk}}$ with itself is given by

$$\mathbf{y_{ijk}^T y_{ijk}} = \left(\sum_{\mathbf{p=1}}^{\mathbf{B}} \alpha_\mathbf{p} \mathbf{b_p} \right)^\mathbf{T} \sum_{p=1}^{B} \alpha_p \mathbf{b_p} = \sum_{p=1}^{B} \sum_{q=1}^{B} \alpha_p \alpha_q \mathbf{b_p^T b_q}. \tag{3.18}$$

Thus, the inner product of $\mathbf{y_{ijk}}$ with itself can be represented as a linear combination of the pairwise inner products of vectors in $\{\mathbf{b_p}\}$. Hence, if the number of basis vectors B remains constant (and reasonably small at < 256 to achieve $\epsilon = 0.3$ at $\xi = 10\times$) with increasing values of E, the inner product of any vector $\mathbf{y_{ijk}}$ would be approximately preserved under random projections according to the JL corollary.

In order to compute the basis vector set $\{\mathbf{b_p}\}$ for E epochs, we employ the singular value decomposition (SVD) of the matrix of all filtered EEG vectors $[\mathbf{y_{ij1}}, \mathbf{y_{ij2}}, \ldots, \mathbf{y_{ijE}}]$. Suppose we denote this high-dimensional matrix of N rows and E columns by $\mathbf{Y}$. SVD enables decomposition of $\mathbf{Y}$ into orthogonal matrices $\mathbf{U}$, $\mathbf{V}$, and a diagonal matrix $\mathbf{D}$ such that

$$\mathbf{Y} = \mathbf{U D V^T}. \tag{3.19}$$

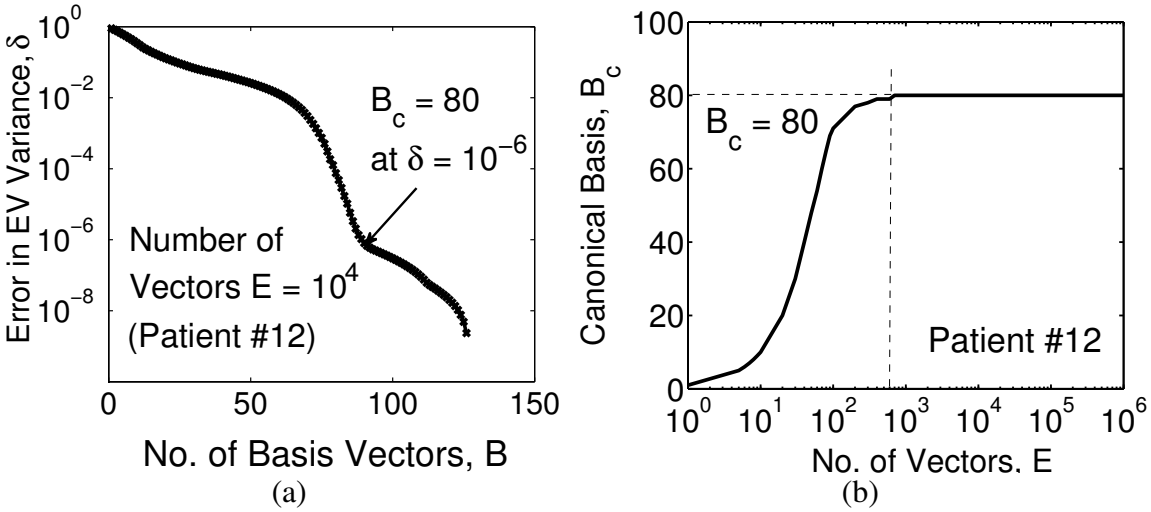


Figure 3.8: A small set of basis vectors underlies the filtered EEG data. For Patient #12: (a) $B_c = 80$ is achieved at $\delta = 10^{-6}$, (b) the canonical basis set (B_c) saturates after 680 data instances.

Along the principal diagonal, matrix $\mathbf{D}$ has entries $\beta_e \leq 1,\ e \in [1, E]$, which are known as the singular values of the matrix $\mathbf{Y}$. The diagonal entries are ordered (*i.e.*, $\beta_1 \geq \beta_2 \geq \ldots \geq \beta_E$) and represent weights for the columns of $\mathbf{V^T}$ [see Eq. (3.19)]. The column vectors in $\mathbf{V^T}$ thus form an orthonormal basis that spans the E epochs in $\mathbf{Y}$. These are also known as the principal components of $\mathbf{Y}$ and represent the eigenvectors of the covariance matrix $\mathbf{F^T Y}$. The cummulative sum of the E weights captures full variance in the eigenvalues (EVs). However, when $\sum_{e=1}^{B_c} \beta_e \approx \sum_{e=1}^{E} \beta_e$, a basis (of cardinality B_c) for $\mathbf{Y}$ is established. Since the canonical basis comprises the first B_c columns of $\mathbf{V^T}$, the residual error (δ) in the EV variance is computed as

$$\delta = \sum_{e=B_c+1}^{E} \beta_e \left(\sum_{f=1}^{E} \beta_f \right)^{-1}, \tag{3.20}$$

Although δ should be zero to capture the full variance in the data, for some small δ, the number of basis vectors B_c required to represent the E filtered EEG vectors can be determined. For example, Fig. 3.8(a) sets a threshold of $\delta = 10^{-6}$ to obtain $B_c = 80$ with $E = 10^4$ filtered EEG vectors (results are for Patient #12). Again using $\delta = 10^{-6}$, Fig. 3.8(b) shows the cumulative increase in the number of canonical basis vectors required as we increase the number of filtered EEG vectors. We observe that a value of $B_c = 80$ is required to

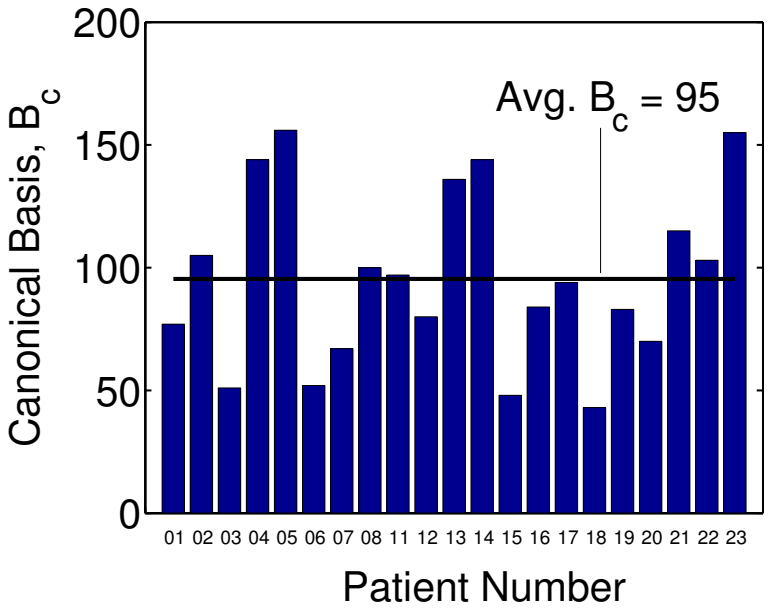


Figure 3.9: A small set of basis vectors underlie the filtered EEG data. $B_c = 95$, on an average, for all patients in the database is achieved at $\delta = 10^{-6}$.

represent all $E = 6.4$M filtered EEG vectors. Similarly, Fig. 3.9 shows the value of B_c required for all patients in the CHB-MIT database (for a threshold of $\delta = 10^{-6}$), illustrating that generally, a small set of basis vectors is required (with an average value of $B_c = 95$ for the patients in the database) to represent a large number of filtered EEG vectors. This behavior offers a rationale for the profile observed in Fig. 3.7(b), and justifies the use of Corollary 1 in Sec. 3.4.2.

3.5 Basis for Regularization Approach

In the methodology presented in the previous section, we aim to find CD-BPFs (defined by $\hat{\mathbf{H}}_\mathbf{i}$) that yield a random projection of the filtered signal $\hat{\mathbf{y}}_\mathbf{ij}$ [see Eq. (3.6)]; we then use the JL corollary to represent the spectral energy of the original input signal. Thus, we first transform each N-dimensional processed vector $\mathbf{y}_\mathbf{ij}$ to an M-dimensional space (*i.e.,* to $\mathbf{\Phi y}_\mathbf{ij}$) through a *regularization matrix* $\mathbf{\Phi}$. We then solve for a mapping between the

regularized vector $\mathbf{\Phi y_{ij}}$ and a corresponding M-dimensional vector $\hat{\mathbf{y}}_{\mathbf{ij}}$ in the compressed domain.

In this section, we explore an alternative to the above regularization approach, where we attempt to solve Eq. (3.6) without using $\mathbf{\Phi}$. We, thus, seek a direct mapping between an N-dimensional vector $\mathbf{y_{ij}}$ and an M-dimensional vector $\hat{\mathbf{y}}_{\mathbf{ij}}$. We will see that this dimensionality mismatch in *the unregularized approach* will result in a non-unique mapping. Thus, the solution will implicitly involve reconstruction of the compressively-sensed signal. Further, since the regularized detector introduces error due to the least-squares solution [Eq. (3.25)] and the unregularized detector introduces error due to signal reconstruction, we will study the characteristics of the error in both approaches in Sec. 3.5.1. We will see that the error sources lead to new (effective) regularization matrices. Since the statistics of the regularization matrix is critical to using the JL corollary, we will study the statistics of the resulting effective regularization matrices in Sec. 3.5.2. We will show that the regularized detector provides adequate statistics while the unregularized detector does not. We first derive the formulation for the unregularized detector below.

Suppose we can find a CD-BPF $\widetilde{\mathbf{H}}_{\mathbf{i}}$ that yields $\widetilde{\mathbf{y}}_{\mathbf{ij}}$ from $\hat{\mathbf{x}}_{\mathbf{j}}$. Similar to the regularized approach, we have

$$\widetilde{\mathbf{y}}_{\mathbf{ij}} = \widetilde{\mathbf{H}}_{\mathbf{i}}\hat{\mathbf{x}}_{\mathbf{j}} = \widetilde{\mathbf{H}}_{\mathbf{i}}\mathbf{\Phi x_j}. \tag{3.21}$$

However, in the unregularized approach, we seek $\widetilde{\mathbf{y}}_{\mathbf{ij}}$ that attempts to estimate $\mathbf{y_{ij}}$ (rather than its random projection):

$$\widetilde{\mathbf{y}}_{\mathbf{ij}} = \mathbf{y_{ij}}. \tag{3.22}$$

This leads to the following relationship [from Eqs. (3.2) and (3.21)]:

$$\widetilde{\mathbf{H}}_{\mathbf{i}}\mathbf{\Phi x_j} = \mathbf{H_i x_j}, \tag{3.23}$$

which allows us to derive the unregularized CD-BPFs $\widetilde{\mathbf{H}}_\mathbf{i}$. To determine $\widetilde{\mathbf{H}}_\mathbf{i}$, we need to solve the following new set of linear equations:

$$\widetilde{\mathbf{H}}_\mathbf{i}\mathbf{\Phi} = \mathbf{H_i} \Leftrightarrow \mathbf{H}_\mathbf{i}^\mathbf{T} = \mathbf{\Phi}^\mathbf{T}\widetilde{\mathbf{H}}_\mathbf{i}^\mathbf{T}. \tag{3.24}$$

With $M \ll N$, in Eq. (3.24), matrix $\widetilde{\mathbf{H}}_\mathbf{i}^\mathbf{T}$ corresponds to $N \times M$ unknowns constrained by $N \times N$ equations. Eq. (3.24) thus corresponds to another set of overdetermined equations, requiring us to once again solve it in the least-square sense using the right pseudo-inverse as follows:

$$\begin{aligned}
\text{From Eq. (3.24):} \quad \widetilde{\mathbf{H}}_\mathbf{i}\mathbf{\Phi} &= \mathbf{H_i} \\
\text{Thus,} \quad \widetilde{\mathbf{H}}_\mathbf{i}\mathbf{\Phi}\mathbf{\Phi}^\mathbf{T} &= \mathbf{H_i}\mathbf{\Phi}^\mathbf{T} \\
\implies \widetilde{\mathbf{H}}_\mathbf{i} &= \mathbf{H_i}\underbrace{\mathbf{\Phi}^\mathbf{T}(\mathbf{\Phi}\mathbf{\Phi}^\mathbf{T})^{-1}} \\
&= \mathbf{H_i}\mathbf{\Phi}_\mathbf{R}^\dagger.
\end{aligned} \tag{3.25}$$

3.5.1 Projection Matrices with Fitting Error

In this section, we quantify the error in the least-square solution of Eq. (3.25) and compare it to the regularized case in Eq. (3.13). We will find that the two approaches entail the implicit use of projection matrices $\widetilde{\mathbf{\Omega}}$ and $\hat{\mathbf{\Omega}}$, respectively.

In the regularized case, after we apply the i^{th} CD-BPF from Eq. (3.13) to compressively-sensed data from the j^{th} EEG channel, we have the following relationship:

$$\hat{\mathbf{H}}_\mathbf{i}\mathbf{\Phi}\mathbf{x_j} = \left[\mathbf{\Phi}\mathbf{H_i}\mathbf{\Phi}_\mathbf{R}^\dagger\right]\mathbf{\Phi}\mathbf{x_j} = \mathbf{\Phi}\mathbf{H_i}\hat{\mathbf{R}}\mathbf{x_j}, \tag{3.26}$$

where $\hat{\mathbf{R}} = \mathbf{\Phi}_\mathbf{R}^\dagger\mathbf{\Phi}$ represents the fitting error of the least-squares solution. After we incorporate the fitting error from Eq. (3.26), we can re-write Eq. (3.7) as follows:

$$\underbrace{\hat{\mathbf{H}}_\mathbf{i}\mathbf{\Phi}}\,\mathbf{x_j} = \mathbf{\Phi}\mathbf{H_i}\mathbf{x_j} \Leftrightarrow \underbrace{\mathbf{\Phi}\mathbf{H_i}\hat{\mathbf{R}}}\,\mathbf{x_j} = \hat{\mathbf{\Omega}}\mathbf{H_i}\mathbf{x_j}. \tag{3.27}$$

where $\hat{\mathbf{\Omega}}$ is the *actual* projection matrix implicitly applied to $\mathbf{y_{ij}}$ and thus should be considered when discussing the JL corollary. If there is no fitting error (*i.e.*, if $\hat{\mathbf{R}} = \mathbf{I}$), Eq. (3.26) is identical to Eq. (3.7) and the true projection matrix $\hat{\mathbf{\Omega}}$ becomes equal to $\mathbf{\Phi}$.

We can similarly incorporate the fitting error into the unregularized formulation of Eq. (3.24), which leads us to the following relationship [from Eqs. (3.21) and (3.25)]:

$$\widetilde{\mathbf{H}}_\mathbf{i}\mathbf{\Phi}\mathbf{x_j} = \left[\mathbf{H_i}\mathbf{\Phi}_\mathbf{R}^\dagger\right]\mathbf{\Phi}\mathbf{x_j} = \mathbf{H_i}\widetilde{\mathbf{R}}\mathbf{x_j}, \tag{3.28}$$

where $\widetilde{\mathbf{R}} = \mathbf{\Phi}_\mathbf{R}^\dagger\mathbf{\Phi}$ represents the fitting error of the least-squares solution of the unregularized equations. After we incorporate the fitting error from Eq. (3.28), we can re-write Eq. (3.23) as follows:

$$\underbrace{\widetilde{\mathbf{H}}_\mathbf{i}\mathbf{\Phi}}\ \mathbf{x_j} = \mathbf{H_i}\mathbf{x_j} \Leftrightarrow \underbrace{\mathbf{H_i}\widetilde{\mathbf{R}}}\ \mathbf{x_j} = \widetilde{\mathbf{\Omega}}\mathbf{H_i}\mathbf{x_j}. \tag{3.29}$$

From Eqs. (3.27) and (3.29), we thus have

$$\hat{\mathbf{\Omega}} = \mathbf{\Phi}\mathbf{H_i}\hat{\mathbf{R}}\mathbf{H_i^{-1}} \ \text{ and } \ \widetilde{\mathbf{\Omega}} = \mathbf{H_i}\widetilde{\mathbf{R}}\mathbf{H_i^{-1}}. \tag{3.30}$$

If there is no fitting error (*i.e.*, if $\hat{\mathbf{R}} = \widetilde{\mathbf{R}} = \mathbf{I}$), $\hat{\mathbf{\Omega}}$ and $\widetilde{\mathbf{\Omega}}$ become equal to $\mathbf{\Phi}$ and $\mathbf{I}$, respectively. This, however, does not occur, except when $\mathbf{\Phi}$ is of full rank, which explains the identical performance at $\xi = 1\times$ shown ahead in Figs. 3.12 and 3.13.

3.5.2 Statistics of the Projection Matrices

Given that we implicitly use projection matrices $\hat{\mathbf{\Omega}}$ and $\widetilde{\mathbf{\Omega}}$ in the two compressed-domain detectors, we next study their statistics to see whether $\hat{\mathbf{\Omega}}$ and $\widetilde{\mathbf{\Omega}}$ actually meet the conditions required for the projection matrix $\mathbf{\Omega}$ in the JL corollary. We will observe that unlike $\widetilde{\mathbf{\Omega}}$, the statistics of the regularized projection matrix $\hat{\mathbf{\Omega}}$ closely satisfy the conditions of the JL corollary.

The JL corollary states that the pairwise inner products among a set of vectors B are preserved under projection $\mathbf{\Omega}$. The corollary also states that for such an $\mathbf{\Omega}$, each entry is sampled *i.i.d.* from the normal distribution $N(0,1)$ or from the uniform distribution $U(+1,-1)$. From Eq. (3.30), it is clear that $\hat{\mathbf{\Omega}}$ and $\widetilde{\mathbf{\Omega}}$ contain entries other than ± 1. Thus, $\hat{\mathbf{\Omega}}$ and $\widetilde{\mathbf{\Omega}}$ cannot be derived from $U(+1,-1)$. We hence proceed to see if the entries are derived from an underlying normal distribution $N(0,1)$. For this, we employ two common measures of Gaussianity, *viz.* the sample kurtosis excess κ and skewness ς.

Given Z samples, κ measures the peakedness in the data and can be defined as follows:

$$\kappa = \frac{m_4}{m_2^2} - 3 = \frac{\frac{1}{Z}\sum_{i=1}^{Z}\left(d_z - \overline{d}_z\right)^4}{\left[\frac{1}{Z}\sum_{i=1}^{Z}\left(d_z - \overline{d}_z\right)^2\right]^2} - 3, \tag{3.31}$$

where m_i represents the i^{th} moment and $\overline{d}_z$ represents the sample mean, respectively. The excess kurtosis is zero for $N(0,1)$. A negative or positive value of κ implies a flatness or peakedness in the distribution [and thus deviation from $N(0,1)$].

For the i^{th} CD-BPF $\mathbf{H_i}$, we see from Eq. (3.30) that $\hat{\mathbf{\Omega}}$ and $\widetilde{\mathbf{\Omega}}$ contain $M \times M$ and $M \times N$ samples, respectively. We evaluate the sample excess kurtosis of $\hat{\mathbf{\Omega}}$ and $\widetilde{\mathbf{\Omega}}$ for two choices of the measurement matrix $\mathbf{\Phi}$. First, we construct 100 instances of a random projection matrix $\mathbf{\Phi_U}$ with the elements derived from $U(+1,-1)$ (as typically chosen for low-energy compression on sensor nodes). Then, we also construct 100 instances of a random projection matrix $\mathbf{\Phi_N}$ with the elements derived from $N(0,1)$. We compute the kurtosis for each instance of $\mathbf{\Phi_U}$ and obtain κ_{avg} as the average of the hundred kurtosis values. Similarly, we also compute κ_{avg} using the 100 instances of $\mathbf{\Phi_N}$. The corresponding κ_{avg} plots for $\hat{\mathbf{\Omega}}$ and $\widetilde{\mathbf{\Omega}}$, averaged over the eight CD-BPFs and using the 100 instances of $\mathbf{\Phi_U}$ and $\mathbf{\Phi_N}$ are shown in Fig. 3.10(a). We observe that for both $\mathbf{\Phi_U}$ and $\mathbf{\Phi_N}$, the statistics of $\hat{\mathbf{\Omega}}$ are close to a Gaussian, while those of $\widetilde{\mathbf{\Omega}}$ deviate substantially.

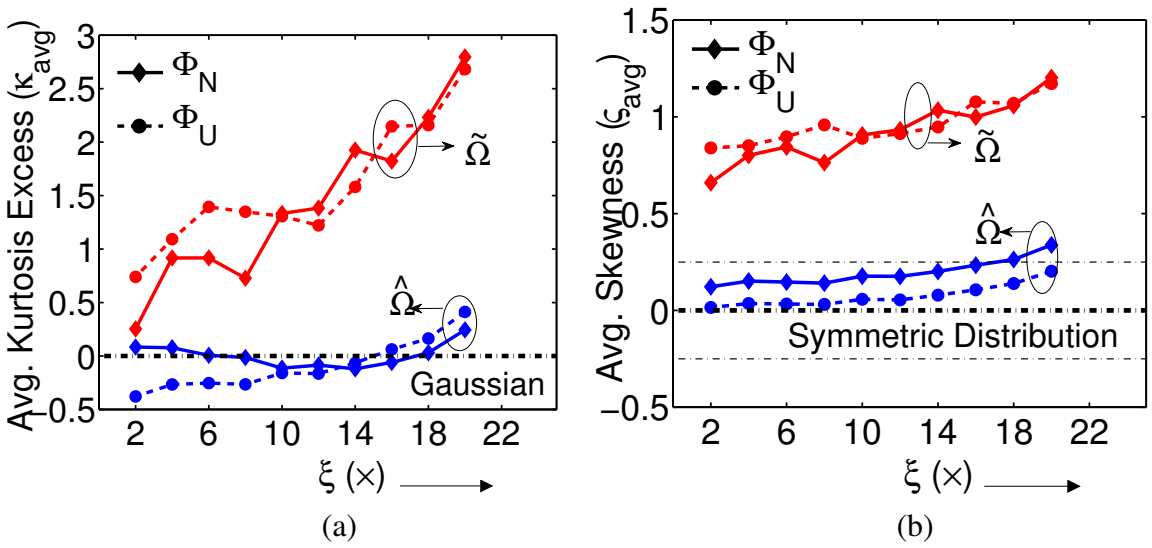


Figure 3.10: For both $\mathbf{\Phi_B}$ and $\mathbf{\Phi_G}$, the statistics of $\hat{\mathbf{\Omega}}$ are closer to $N(0,1)$, while $\widetilde{\mathbf{\Omega}}$ exhibits (a) a strong peakedness, and (b) a positive skew in its underlying density function.

Similarly, the skewness ς of the data measures the symmetry in a distribution. For Z samples, ς is defined as:

$$\varsigma = \frac{m_3}{m_2^{3/2}} = \frac{\frac{1}{Z}\sum_{i=1}^{Z}\left(d_z - \overline{d}_z\right)^3}{\left[\frac{1}{Z}\sum_{i=1}^{Z}\left(d_z - \overline{d}_z\right)^2\right]^{3/2}}. \tag{3.32}$$

A value of ς close to zero indicates symmetry in the underlying density (such as that exhibited by Gaussian, uniform, Laplace distributions, *etc.*). Fig. 3.10(b) shows ς_{avg} for $\hat{\mathbf{\Omega}}$ and $\widetilde{\mathbf{\Omega}}$, using 100 instances of $\mathbf{\Phi_U}$ and $\mathbf{\Phi_N}$. We compute ς_{avg} in a manner similar to κ_{avg}. We observe that $\widetilde{\mathbf{\Omega}}$ exhibits a strong positive skew. $\hat{\mathbf{\Omega}}$, however, has a skewness close to zero [expected of $N(0,1)$].

From the analysis of κ_{avg} and ς_{avg}, we conclude that unlike $\widetilde{\mathbf{\Omega}}$, each element of $\hat{\mathbf{\Omega}}$ is derived from an underlying distribution that has statistics close to $N(0,1)$. According to the JL corollary, the inner products are thus approximately preserved in the case of the regularized compressed-domain detector, but not necessarily for the unregularized compressed-domain detector.

3.6 Experimental Results

We quantify the performance of the proposed approach by first analyzing how the CD-BPF approach affects the mutual information in the FVs, and then by simulating the actual end-to-end performance of the detector.

3.6.1 Information Analysis of Compressed-domain FVs

With reference to Fig. 3.6, we remarked that although the error in the inner products increases linearly with ξ, it does not correspond to the performance of the end-to-end detector. To analyze this, we quantify the information content of the FVs based on the metrics of Shannon entropy and mutual information. In the regularized approach, we observe that the mutual information required for analysis is indeed maintained for large compression factors by the CD-BPFs. Next, we describe the methodology for computing mutual information.

At time $2E$ sec., let $F = \{f_k\}$ denote the set of D-dimensional FVs with $k \in [1, E]$. Using E epochs (observed up to time $2E$ sec.), we can specify n_b uniformly quantized bins between $min(f_{ij1}, f_{ij2}, \ldots, f_{ijE})$ and $max(f_{ij1}, f_{ij2}, \ldots, f_{ijE})$ for each FV dimension f_{ij}, where $i \in [1, 8]$ and $j \in [1, 18]$ for the seizure detector. Thus, we partition the D-dimensional FV space into n_b^D bins. Using these bins, the discrete probability density function (PDF) of the FV set F is given by $p(F) = [\#f_k \text{ in bin e}]\,/E$ where e ranges from 1 to n_b^D. Using this PDF, the Shannon entropy in the FVs can thus be computed as follows:

$$S(F) = -\sum_{e=1}^{n_b^D} p(F)\; log_2\left[p(F)\right]. \tag{3.33}$$

Given that the FVs correspond to instances from one of two classes (*i.e.*, seizure class or non-seizure class), we can define a set $C = \{c_l\}$ (with $l = 1$ for seizure FVs and $l = 0$ for non-seizures FVs). We can thus derive the conditional entropy of the full FV set as follows:

$$S(F|C) = \sum_{l=0,1} p(c_l)\, S(F|c_l), \tag{3.34}$$

where $p(c_l)$ is the probability of the class label c_l and $S(F|c_l)$ is the conditional entropy of the FV set F exhibited within class c_l, which is defined as

$$S(F|c_l) = -\sum_{e=1}^{n_b^D} p(F|c_l)\, log_2\left[p(F|c_l)\right],$$

The difference between the initial entropy of the FVs and the conditional entropy within the classes is defined as the mutual information $I(F;C)$ between the FV set F and the class values C. We thus have the following relationship:

$$I(F;C) = S(F) - S(F|C). \tag{3.35}$$

$I(F;C)$ is a commonly used metric for evaluating the ability of a classifier to successfully discriminate between data instances. Intuitively, it is the amount by which the knowledge provided by the set of class values C decreases the uncertainty in the FV set F [169, 170]. $I(F;C)$ thus provides an indication of the end-to-end performance achievable by the compressed-domain seizure detector.

Now, similar to the Nyquist case, we can define mutual information in the compressed domain as follows:

$$I(\hat{F};C) = S(\hat{F}) - S(\hat{F}|C). \tag{3.36}$$

where $S(\hat{F})$ and $S(\hat{F}|C)$ denote the entropy and conditional entropy of the FVs derived from compressed-domain processing.

$I(\hat{F};C)$ is thus desired to be as close as possible to $I(F;C)$ for correspondence between the two approaches. For the regularized approach, Fig. 3.11 shows numerical results for mutual information averaged over 21 patients in the CHB-MIT database. To keep the calculation of entropies tractable, we first reduced the FV dimensionality to 8 using PCA and

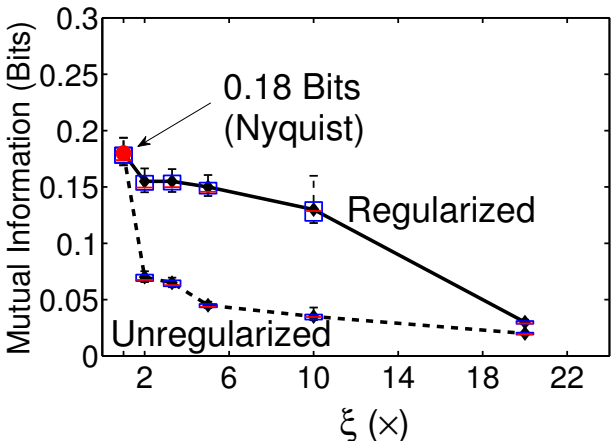


Figure 3.11: Mutual information is retained up to large ξ only for the regularized detector.

then quantized the probability distributions using $n_b = 8$ bins for each dimension of the FV set. Note that the y-axis in Fig. 3.11 is in bits, which means that the computation of individual entropies involves logarithm to base 2. $I(F;C) = 0.18$ bits is the information content of the Nyquist-domain FVs. Using the compressed-domain analysis, the mutual information in the FVs, $I(\hat{F};C)$, is over 0.15 bits for ξ up to 5×, and drops to 0.13 bits for $\xi = 10\times$. Compressing the EEG signals beyond this point reduces $I(\hat{F};C)$ to 0.03 bits at $\xi = 20\times$. Thus, the information in compressed-domain FVs is retained up to substantial compression factors, suggesting high end-to-end performance when using a flexible classifier. Fig. 3.11 also shows that the degradation in the mutual information for the unregularized compressed-domain FVs, denoted as $I(\widetilde{F};C)$, is much larger than that observed for the regularized compressed-domain FVs [*i.e.*, $I(\hat{F};C)$].

To validate the information trends observed, we next study the performance of the end-to-end detector. We expect to see some tolerance in a practical classifier, such as an SVM, to the degraded information in the signal features. Studying the detector performance will also allow us to determine the tolerance limits. These limits will then enable us to estimate the maximum ξ achievable for the seizure-detection application without significantly degrading the system performance.

Table 3.2: Comparison of Nyquist-domain performance with the baseline detector.

Performance Metric	Shoeb *et al.* [147]	This work ($\xi = 1\times$)
Sensitivity (%)	96.03	96.02
Latency (sec.)	4.59	4.59
Specificity (false alarms/hr.)	0.1471	0.1453
No. of SVs	242	253

3.6.2 Performance Analysis of the Compressed-domain Detector

In this section, we present the end-to-end performance, based on MATLAB simulations, of the compressed-domain seizure detector using an SVM classifier. We show a trade-off between the compression factor and system performance. Table 3.2 summarizes the performance of a baseline detection algorithm that uses the Nyquist-domain EEG [147]. Our Nyquist-domain implementation achieves a performance consistent with that shown in [147].

To evaluate the performance of the compressed-domain detector, we derive FVs at the rate of 0.5 Hz from the CHB-MIT database. We use these FVs to train and test the SVM classifier in a patient-specific manner. We employ a leave-one-out cross-validation scheme for measuring the performance of the detector. Accordingly, for each patient, we use all seizure records except one, along with all non-seizure records for training the classifier. We then apply the resulting SVM model only to the record that was left out in the training phase. We repeat this validation process for each record of a patient. Fig. 3.12 shows the scaling in performance (over 21 patients). The performance of the compressed-domain detector is very close to the Nyquist case at $\xi = 1\times$ (the performance at $\xi = 1\times$ is also shown in Table 3.2). Thus, for the regularized compressed-domain detector, at a compression of $1\times$, the sensitivity is 96.02%, latency is 4.59 sec. and the number of false alarms is 0.1453 per hour. These performance numbers begin to degrade with the compression factor. The corresponding numbers at $\xi = 10\times$ are 94.70%, 5.83 sec., and 0.1989/hr., respectively. Thus, at higher compression factors, the degradation in sensitivity is less than 1.33% up

to $\xi = 10\times$, beyond which it begins to drop more significantly. The scaling in the number of false alarms per hour and the latency also follow a similar trend. The mean latency of detection increases by 1.21 sec., while the specificity of the algorithm degrades by only 0.03 false alarms per hour at $\xi = 10\times$. We see that this performance profile exhibits a close correlation with the FV information loss investigated in the previous section. We suggest that very limited degradation is seen up to large compression factors, which enables the system model.

Performance of the unregularized detector. Fig. 3.13 shows the performance (over 21 patients) of the compressed-domain detector using the unregularized CD-BPFs $\widetilde{\mathbf{H}}_\mathbf{i}$. The detector sensitivity and specificity are much poorer when compared to the regularized detector, degrading substantially even at $\xi = 2\times$. Even though both approaches, at best, approximate the sought output signal due to the least-square fitting involved, the regularized approach seeks only to find *a suitable random projection* of $\hat{\mathbf{y}}_\mathbf{ij}$ rather than $\mathbf{y}_\mathbf{ij}$ directly. This approach enables much greater tolerance to approximation errors, and allows us to exploit the JL corollary in order to recover the desired features. Also, the degradation in the information content of the FVs (observed in the previous section) is in close agreement with the performance trends observed in Fig. 3.13.

Scaling in SV complexity. Despite an error of $\epsilon = 0.31$ in the inner-product computation at $\xi = 10\times$ (see Fig. 3.6), the performance of the regularized compressed-domain detector is maintained due to the mutual information when a flexible classifier, such as an SVM, is used. When trained using compressed-domain FVs, the SVM decision boundary is suitably adjusted. However, the complexity of the decision boundary, which is represented by the number of SVs required to represent it, depends on how the data are redistributed in the feature space. Fig. 3.14 shows that the number of SVs increases modestly as ξ is increased.

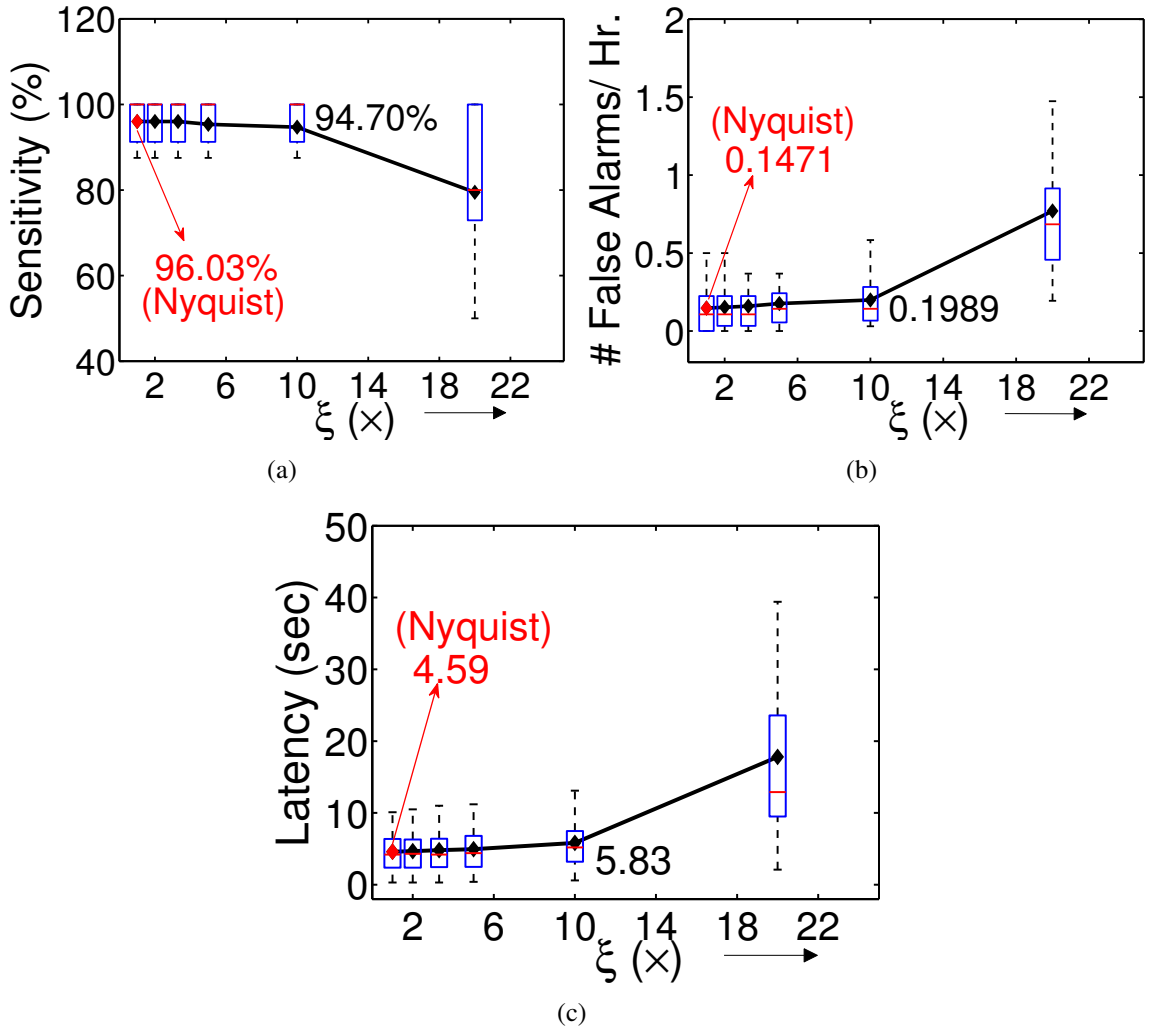


Figure 3.12: End-to-end performance [measured by (a) sensitivity, (b) false alarms/ hr., and (c) latency of detection] nearly equal to that in the Nyquist domain (shown in grey) is achieved at $\xi = 1\times$ in the compressed domain. Also, the performance is maintained in the compressed domain up to large values of ξ.

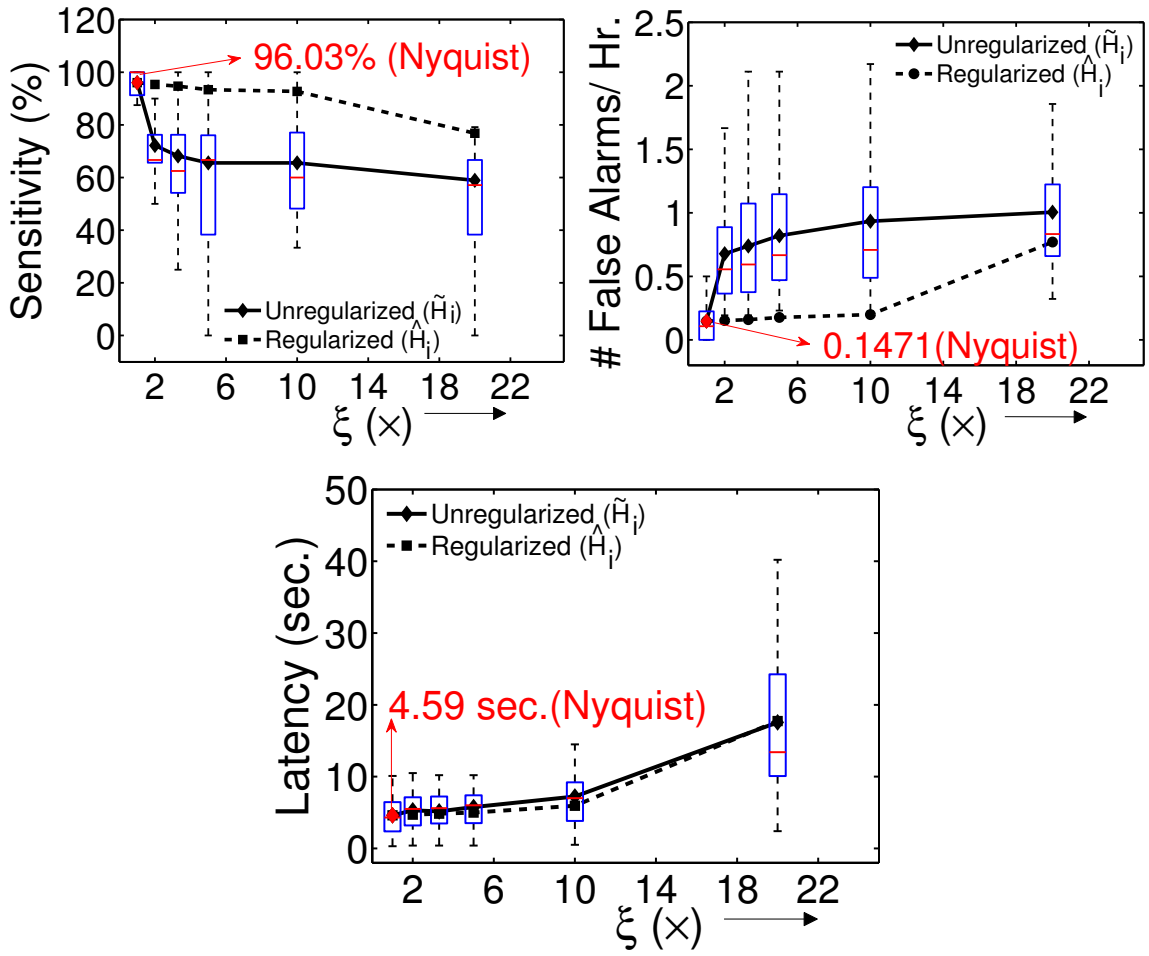


Figure 3.13: Unlike the regularized compressed-domain seizure detector, performance of the unregularized detector is poor at $\xi > 1\times$.

3.7 Reconstruction Error Analysis

As mentioned in Sec. 3.1, besides performing on-sensor analysis, we also want to acquire signal waveforms in the system shown in Fig. 3.1. Thus, in this section, we investigate the feasibility of signal reconstruction. We compute the reconstruction error and qualitatively observe a correlation with the information-content and detector-performance profiles. The correlation is expected and suggests that, generally, the performance of the compressed-

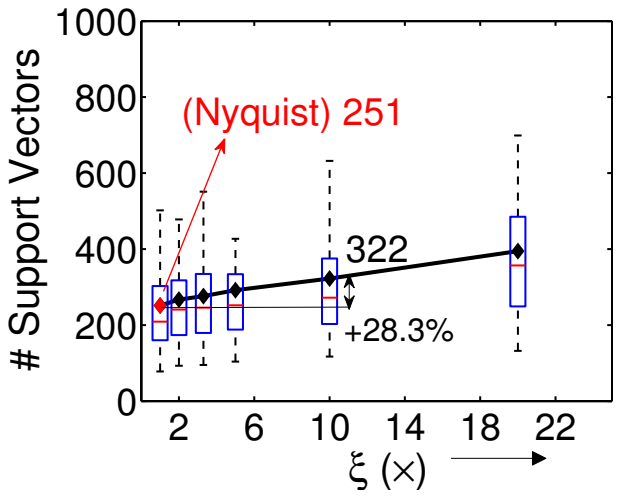


Figure 3.14: The complexity of the SVM decision model increases with ξ, indicating an adaptation to the errors in the compressed-domain FVs. The number of SVs in the Nyquist domain is also shown.

domain detector follows the accuracy achievable for signal reconstruction in compressive sensing.

Sparsity basis. EEG data have been shown to have a sparse representation in several bases, such as the Wavelet [171], Gabor [11,172], Mexican Hat [171], and Spline [148,173] bases. In our analysis, we use a Gabor dictionary $\mathbf{\Psi} \in \mathbb{R}^{N \times N}$ as the sparsity basis [11]. Functions in this dictionary are defined by cosines under a Gaussian envelope. The mn^{th} element of matrix $\mathbf{\Psi}$ is given by

$$\Psi_{mn}(\omega, \sigma) = \frac{1}{\sqrt{2\pi}\sigma}\left[e^{-(m-n)^2/\sigma^2}\right] cos(\omega m + \gamma), \tag{3.37}$$

where the center of the envelope is at column number n, $\omega \geq 0$ is the angular frequency of the cosine function, $\sigma \leq 1$ is the spread of the envelope, and $\gamma \in [0, \pi/2]$ is the phase angle. We use the values $\omega = 16$, $\gamma = 0$, and $\sigma = 0.1$ in our experiments, which are empirically determined to achieve the best reconstruction accuracy of the EEG data. Further, each column of $\mathbf{\Psi}$ is independently normalized such that the corresponding $l2$ norm is equal to one.

For the measurement matrix $\mathbf{\Phi}$, we use entries that are ± 1 with a uniform probability; this not only leads to a matrix that is incoherent with $\mathbf{\Psi}$, thus satisfying the RIP, but one that also enables low-energy compression by avoiding the need for actual multiplications (such a $\mathbf{\Phi}$ is commonly chosen for low-power sensing applications). Fig. 3.15(a) shows a 2 sec. epoch of Nyquist-sampled EEG from Patient #12 in the CHB-MIT database. The EEG is sampled at 256 Hz and clearly not sparse in the time domain. However, as shown in Fig. 3.15(b), the EEG is sparse in the Gabor basis with only a small number of non-zero coefficients. We can thus compressively-sense the EEG using $\mathbf{\Phi}$ [an instance of compression by $\xi = 5\times$ is shown in Fig. 3.15(c)].

Signal reconstruction. There are several signal reconstruction methods for compressive sensing [12, 69–71]. We use a gradient projection algorithm, which balances reconstruction accuracy with computational efficiency [12]. For each compressively-sensed EEG epoch $\hat{\mathbf{x}}_\mathbf{j}$ of M samples derived from the Nyquist-domain epoch $\mathbf{x_j}$, the algorithm solves the following non-smooth convex optimization problem:

$$\underset{\mathbf{w_j}}{\text{argmin}} \ \|\mathbf{\Phi\Psi w_j} - \hat{\mathbf{x}}_\mathbf{j}\|_2^2 + \lambda\|\mathbf{w_j}\|_1, \tag{3.38}$$

where $\mathbf{\Psi w_j}$ is the corresponding reconstructed EEG epoch and $\lambda \leq 1$ is a scaling constant, which controls the sparsity (determined by the $l1$-norm) of $\mathbf{w_j}$. Fig. 3.15(d) shows the reconstructed EEG epoch at $\xi = 5\times$, $10\times$, and $20\times$, respectively. We observe from the figure that EEG data can be accurately reconstructed up to large values of ξ. Reconstruction errors, absent at $\xi = 5\times$, begin to show up at $\xi = 10\times$ and increase substantially at $\xi = 20\times$.

To quantify the accuracy of signal reconstruction, we employ the metric of SNR, which is calculated for each epoch based on the expected ($\mathbf{x_j}$) and reconstructed ($\mathbf{\Psi w_j}$) EEG samples. Average SNR, over E epochs, in decibels (dB) is defined as

$$\text{SNR} = \frac{10}{E} \sum_{j=1}^{E} log\left[\frac{\|\mathbf{x_j}\|_2^2}{\|\mathbf{\Psi w_j} - \mathbf{x_j}\|_2^2}\right]. \tag{3.39}$$

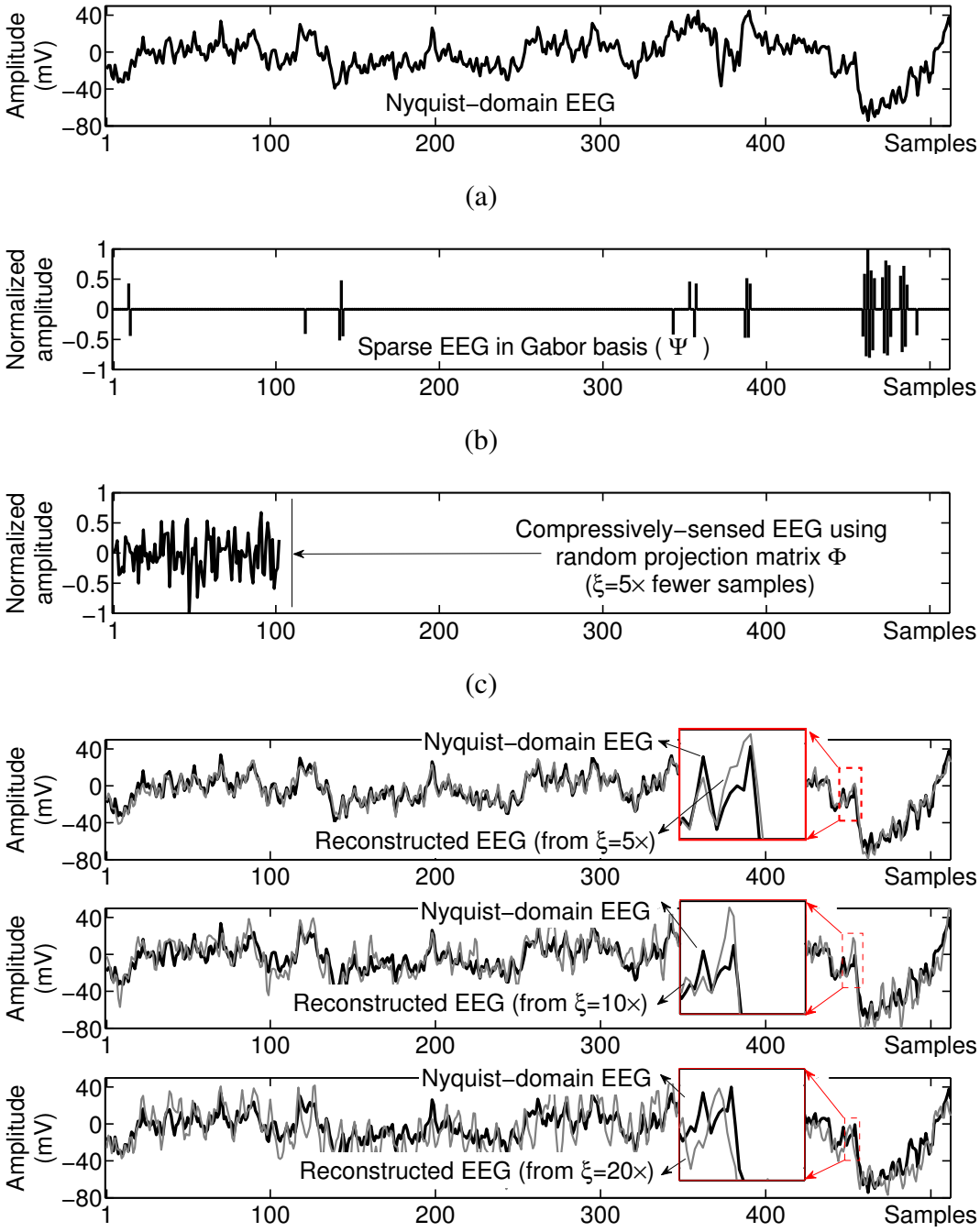


Figure 3.15: (a) An epoch has 512 samples in the Nyquist domain. (b) EEG is sparse in the Gabor basis. (c) Sparsity enables substantial compression (case of $\xi = 5\times$ is shown). (d) Reconstruction accuracy degrades at larger values of ξ.

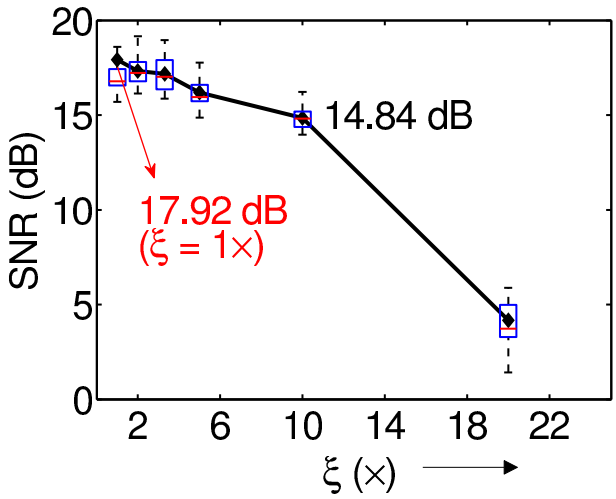


Figure 3.16: Accuracy of signal reconstruction follows a trend similar to detector performance.

Fig. 3.16 shows the SNR as a function of ξ, averaged over all epochs of 21 patients in the CHB-MIT database. With increasing ξ, the SNR of the reconstructed EEG drops minimally ($\leq 18\%$) up to $\xi = 10\times$, and more rapidly beyond that. This behavior correlates with the degradation in performance observed for the compressed-domain detector.

3.8 Discussion of Reduced Dimension Bounds

As remarked in Sec. 3.4.2, the expected value of the error ϵ introduced by the JL corollary for a basis vector set with cardinality B_c depends on the dimensionality of the random projection M. However, interestingly, it does not depend on the dimensionality of the original vector N. This implies that for signals with a very large number of samples N, high compression factors $\xi = N/M$ are achievable with low error. For the seizure detection application, an epoch size of 2 sec. (corresponding to $N = 512$ samples) is used in order to limit the latency of detection. Combined with the error, this limits the compression factor. However, in this section, we investigate the general performance of the proposed approach as the number of input samples in an application scales. We show that, in general, low-error

inner-product estimates, and thus high performance, can potentially be achieved for very high compression factors.

A lower bound on the dimensionality of the random projection M, when $\hat{\mathbf{\Omega}}$ comes from $N(0, 1)$ (denoted as $\hat{\mathbf{\Omega}}_{\mathbf{N}}$), is presented in [174]. It relates M, B_c, and ϵ as follows:

$$M \geq logB_c / \left[\epsilon^2 \log(1/\epsilon)\right]. \tag{3.40}$$

Fig. 3.17 shows the corresponding upper bound on ξ for a given error limit ϵ as well as the observed value of ϵ from Fig. 3.6 for the seizure detector. The bound on ξ, when $\hat{\mathbf{\Omega}}$ comes from $U(+1, -1)$ (denoted as $\hat{\mathbf{\Omega}}_{\mathbf{U}}$), is also shown [108]. For a given ξ, the empirical ϵ values observed for the seizure-detection application breach the bound for $\hat{\mathbf{\Omega}}_{\mathbf{U}}$, confirming the conclusion derived from the analysis in Sec. 3.5.2 that the elements of $\hat{\mathbf{\Omega}}$ are not derived from $U(+1, -1)$. In fact, the actual error is within the expected value for $\hat{\mathbf{\Omega}}_{\mathbf{N}}$. For large values of ξ, however, the inner-product error ϵ is high, ultimately limiting the compression that can be handled by the detector. Next, we consider increasing the epoch size beyond 2 sec. to illustrate the potentially large values of ξ that can be achieved.

For a given inner-product error, Fig. 3.18(a) shows the achievable compression factor with increasing number of samples N at $B_c = 100$. The case of $N = 512$ is shown by the dotted line. At an expected inner-product error of 0.1, we can achieve a maximum compression factor of 3× (note that the actual achievable compression for this error value is only 2× for the seizure-detection application). With increasing number of data dimensions (longer epoch sizes), we can achieve higher compression factors at the same expected error ϵ. For example, with $N = 10^3$, 10^4, and 10^6 at $\epsilon = 0.1$, we can achieve $\xi = 6\times$, 60×, and 600×, respectively. At a sampling rate of 256 Hz, these values of N correspond to epoch sizes of 3.9, 39, and 390 secs., respectively. Some applications may also have a larger number of underlying basis vectors. Fig. 3.18(b) shows the achievable compression factor at $B_c = 1$M. We observe that in such cases, the maximum ξ is just 1× at an inner-product error of 0.1 at $N = 512$. With increasing values of N, however, we can achieve $\xi = 2\times$,

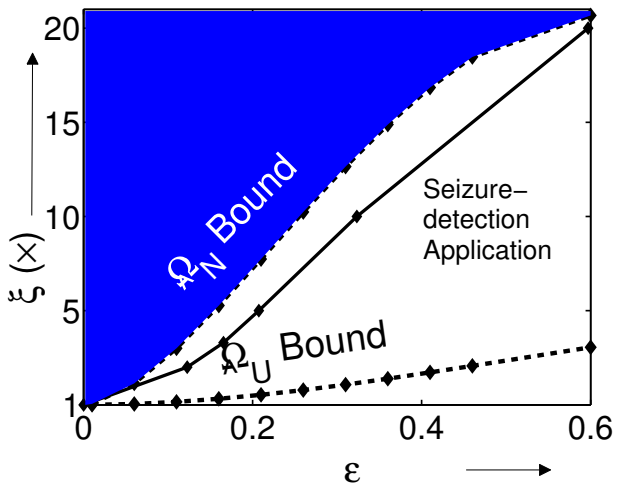


Figure 3.17: The inner-product error (ϵ) in the seizure-detection application is within the predicted upper bound for ξ when $\hat{\mathbf{\Omega}} \sim N(0, 1)$.

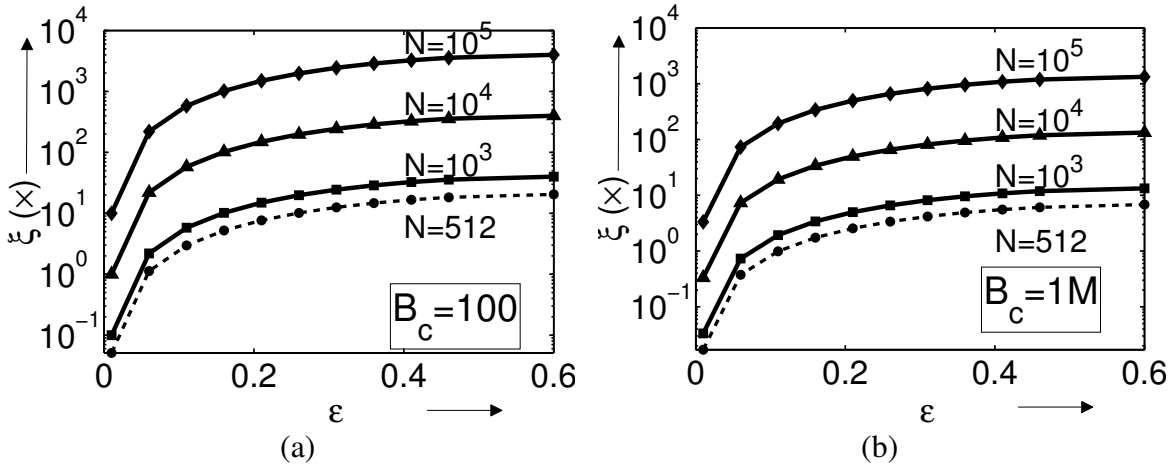


Figure 3.18: Scaling in the upper bound for ξ with respect to N and B_c shows the potential of the proposed approach at higher data dimensions.

20×, and 200× for $N = 10^3$, 10^4, and 10^6, respectively, at the same value of ϵ. Thus, in general, for applications that must handle large amounts of sensor data, the proposed compressed-domain processing approach can potentially yield substantial savings while retaining performance.

3.9 Hardware Implementation Analysis

In this section, we present an analysis of the hardware implementation cost for the compressed-domain detection algorithm. The proposed approach requires formulating feature computation as a matrix multiplication with non-zero elements. Moreover, the regularity of Nyquist-domain processing matrices (*e.g.*, convolution matrices used for FIR filtering) is disrupted in CA. Thus, CA precludes the use of filter optimizations, such as multiply-accumulate (MAC)-stage folding. On the other hand, the compressed-domain approach has the benefit of having to process fewer input samples. Thus, we present a trade-off related to the compression factors.

Filter optimizations. Fig. 3.19 shows the structure of the compressed- and Nyquist-domain detectors. MAC operations are the dominant computation in the system and are used for feature extraction in two stages: (i) the application of the spectral-analysis filters of Eq. (3.2) (identified as MAC0 in the figure), and (ii) the energy accumulation process of Eq. (3.3) (identified as MAC1). The number of operations per epoch are shown in dark boxes below the MAC units ($N = 512$ is the number of Nyquist samples, $k = 64$ is the filter order used, and $F = 8$ is the number of spectral-analysis filters). For the BPF stage, kN operations are performed per epoch per channel in the cascade-form FIR implementation. They can be reduced to $kN/2$ by exploiting the symmetry of filter coefficients $\mathbf{H_i}$ (necessary for a detector implementation with linear phase).

The optimizations for Nyquist-domain processing are possible because filtering actually corresponds to convolution, allowing matrix $\mathbf{H_i}$ to have a regular structure (with several zeroed entries), as shown in Fig. 3.19. With compressed-domain processing, however, the entries of the $\hat{\mathbf{H}}_\mathbf{i}$ matrix are determined by random projection matrix $\mathbf{\Phi}$, disrupting the regularity, and precluding the optimizations above. $\hat{\mathbf{H}}_\mathbf{i}$ thus potentially consists of $(N/\xi)^2$ non-zero coefficients. Therefore, in the most generic case, the compressed-domain implementation involves a matrix multiplication resulting in $(N/\xi)^2$ operations in MAC0. Fig. 3.20

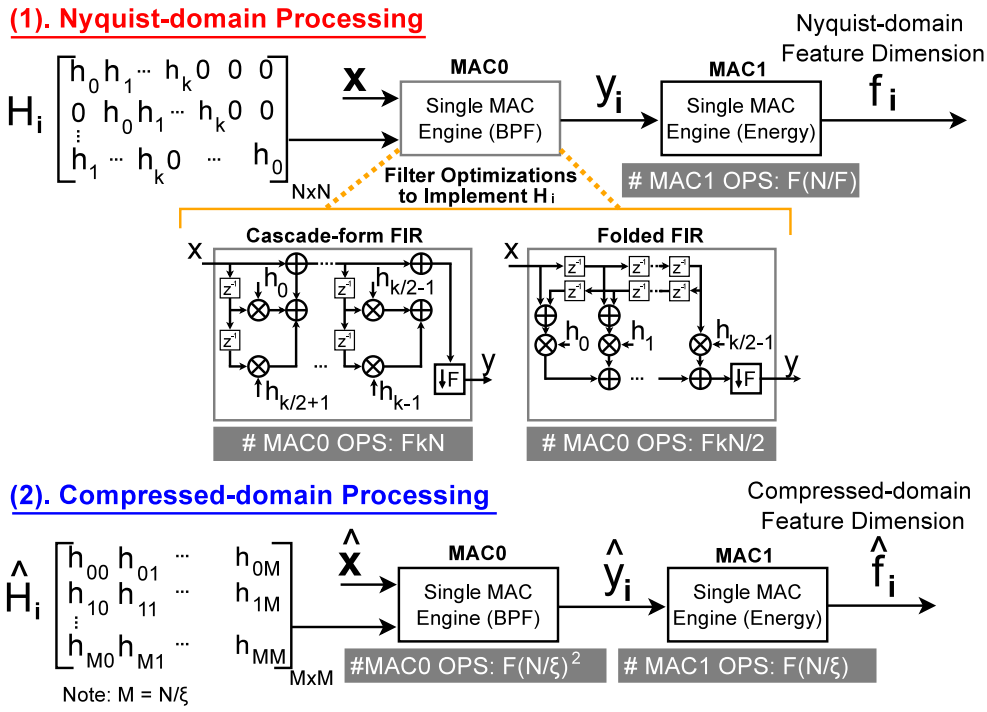


Figure 3.19: The structure of the convolution matrix $\mathbf{H_i}$ in the Nyquist domain enables filter optimizations to reduce the number of MAC operations.

shows a practical implementation of the detector using a single MAC engine for each of the operations, MAC0 and MAC1, described earlier. Fig. 3.21(a) shows the scaling in MAC operations with ξ (the number of MAC operations required for an optimized Nyquist implementation is also shown). We observe that at sufficient compression factors (around $\xi > 4\times$), compressed-domain processing can actually enable fewer hardware operations despite the optimizations possible in the Nyquist implementation. At $\xi = 10$, the number of MAC operations in the compressed domain are 6.15×, and 12.3× fewer as compared to the folded and cascade-form FIR architectures, respectively. Note that random projections in the compressed domain can be performed without any MAC operations and are excluded from this analysis.

Another key consideration in the implementation is the amount of memory required by the detectors [shown in Fig. 3.21(b)]. We need to store kF coefficients for the Nyquist implementation versus $k(N/\xi)^2$ for the compressed-domain detector. There is thus a subtle

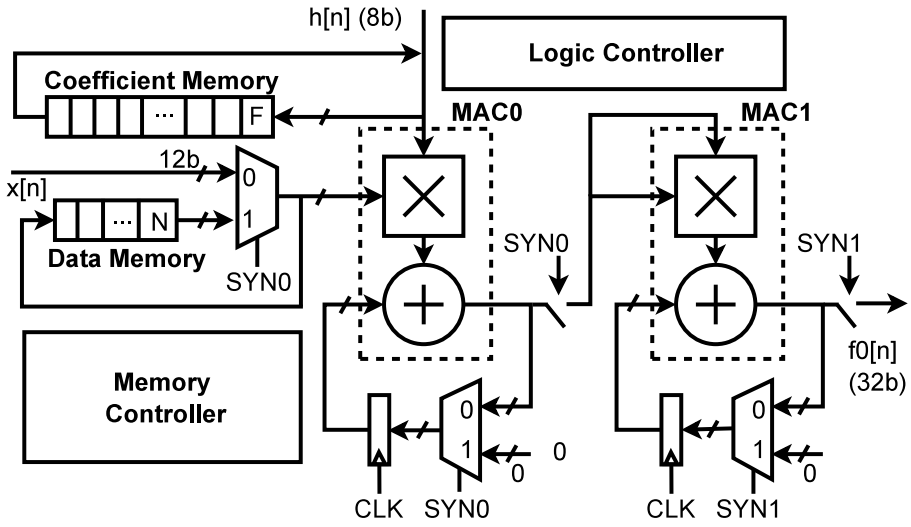


Figure 3.20: A practical implementation of the detector involves a single MAC engine for each filtering and spectral-energy estimation.

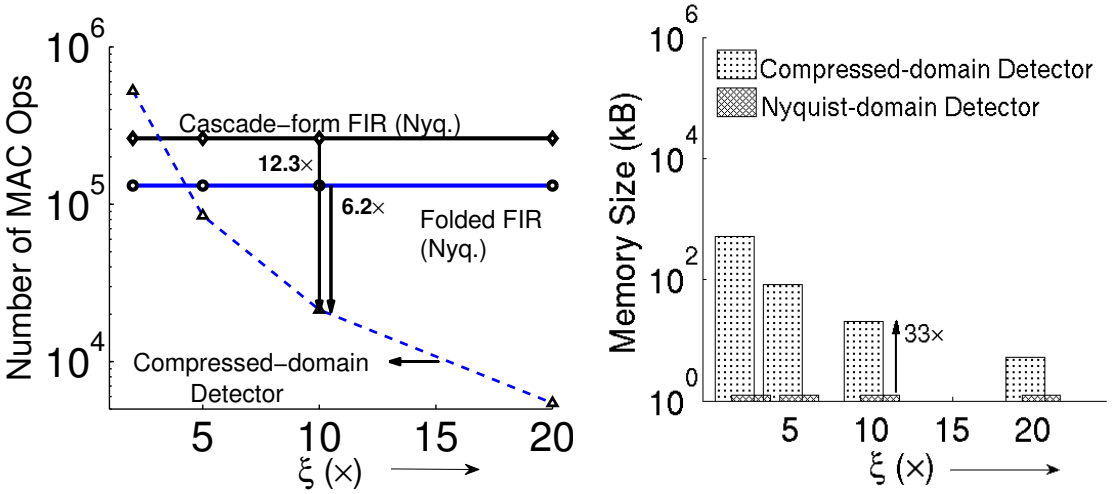


Figure 3.21: Hardware implementation analysis in the compressed domain shows (a) that processing in the compressed domain can potentially outperform processing in the Nyquist domain by allowing fewer computations for feature extraction, and (b) that there is a subtle trade-off between an efficient implementation and data storage.

computation-memory trade-off in the implementation of the compressed-domain detector. Further, the increased memory needs require new approaches for computational power management, which we will explore in Chapter 5. In the same chapter, we will also see that the system-level gains from reduced communication and computation costs, however, can be significant, typically far exceeding the memory overheads in a low-power sensor platform.

3.10 Chapter Summary

It is desirable in many sensing applications to analyze the sensed signal on energy-constrained sensor nodes. Though compressive sensing offers a compelling low-energy means for signal compression, it presents new challenges for signal analysis since the resulting representations are substantially altered due to the random projections involved. In this chapter, we presented a methodology for transforming linear signal-processing operations to the compressed domain using a least-squares approximation. We then extracted signal features that are based on inner-product computations. Using this approach, we estimated spectral features from EEG signals, which are shown to maintain the mutual information required for detecting epileptic seizures. Using an SVM classifier, we thus demonstrated a compressed-domain seizure detector that yields accurate end-to-end performance at high compression factors ($\xi \approx 10\times$). Since spectral features are biomarkers for a wide range of neurological applications, the proposed approach could have broad applicability. Further, we analyzed how the proposed methodology performs as the number of samples in the original Nyquist-domain signal increases. Generally, a large number of samples can be handled while keeping the expected error low, making the approach promising for applications targeting large data vectors. Our results from this chapter suggest that although regularization with a fixed random projection matrix $\mathbf{\Phi}$ helps us achieve acceptable performance in seizure detection up to large compression factors, the

error in the feature estimates can still be substantial. In the next chapter, we provide a methodology to derive a more accurate solution for the compressed-domain equations [Eq. (3.8)] through a designer-controllable auxiliary matrix $\boldsymbol{\Theta}$ instead of $\boldsymbol{\Phi}$. Since the projection matrix $\boldsymbol{\Theta}$ can be tailored, this solution can potentially help reduce the error in the signal features, enabling accurate detection up to much higher compression factors.

Chapter 4

Transformation to Compressed Domain with an Auxiliary Matrix

In this chapter, we use an auxiliary matrix $\mathbf{\Theta}$ [in place of $\mathbf{\Phi}$ in the right-hand side of Eq. (3.8)] to transform linear signal processing operations to the compressed domain. We show that the flexibility in choosing $\mathbf{\Theta}$ allows us to derive (1) an exact solution for $\hat{\mathbf{H}}$, which provides the highest accuracy, and (2) an approximate solution for $\hat{\mathbf{H}}$, which saves us computational energy. We also show that the elements of $\mathbf{\Theta}$ can be chosen freely by a designer based on the required end-to-end accuracy of the system. We focus on data-driven inference frameworks for analyzing sensor signals. In this case, our approach enables the use of compressively-sensed signals while completely avoiding signal reconstruction. Our transformations also reduce computational energy by enabling processing over fewer input samples. We demonstrate the approach through two case studies. First, we consider a system for neural prosthesis that extracts wavelet features directly from compressively-sensed spikes. Second, we consider a variant of the system presented in the previous chapter, where we use multi-rate FIR filters for band-pass filtering followed by spectral-energy feature extraction. We show that all of these computations can be performed directly with compressively-sensed EEG.

4.1 Introduction

As mentioned previously, sparse representations capture most or all information in a signal *via* a small number of samples. Such representations can significantly benefit several functions, such as communication, storage, and potentially computation. Compressive sensing is one specific technique that exploits sparsity in a transform basis to efficiently represent signals using simple random projections. However, compressive sensing significantly alters the Nyquist-domain samples. Consequently, before signal processing can be performed using conventional frameworks, it becomes necessary to reconstruct the original Nyquist-domain signal. The challenge is that reconstruction from random projections, can be extremely costly. In this chapter, we again consider *transforming linear signal processing operations so that they can be applied directly to the compressively-sensed signals*. Since we employ an auxiliary matrix, our transformations improve significantly upon the methodology presented in Chapter 3. Also, our approach can provide more reductions in computational energy since it enables processing over much fewer input samples. Our specific contributions in this chapter are as follows:

- We present a mathematical framework (improved over the one presented in Chapter 3) to derive compressed-domain equivalents of linear signal-processing functions. We consider both rate-preserving and multi-rate systems (*e.g.,* FIR filtering and wavelet transforms) as well as rate-transforming systems (*e.g.,* down-sampling).

- Since our approach solves for random projections of a processed signal using an auxiliary matrix $\mathbf{\Theta}$, it introduces important designer-controllable knobs for system-level trade-offs. First, the projections can be used to derive an exact solution for the compressed-domain equivalents. This limits the error in the resulting output signals close to the theoretical lower bound. Second, the projections can be used to derive approximate solutions, wherein fewer signal-processing operations are required, thereby enabling a new knob for computational power management. Unlike the ξ

knob presented in the previous section for data compression, this new knob allows us to scale computational energy at the cost of accuracy while keeping the amount of data compression fixed. Thus, the new knob provides additional energy savings (over that provided by ξ) while ensuring the required end-to-end performance for the system.

- To illustrate our approach for multi-rate systems, we derive compressed-domain equivalents of wavelet computations for neural prosthesis. We use the compressed-domain features to sort spikes and infer statistical parameters, which can be used to synthesize control function for prosthetics. We show that we can achieve system performance similar to an approach where features are extracted from signals that are first reconstructed.

- To illustrate our approach for rate-transforming systems, we derive compressed-domain equivalents for downsampling and FIR filtering in a seizure-detection system. We provide an exact solution for the compressed-domain operations and demonstrate a significant improvement in performance compared with a least-squares approximate solution presented in Chap. 3, which limited the performance that was previously achievable.

The rest of this chapter is organized as follows. In Sec. 4.2, we describe our approach for deriving compressed-domain processing functions. In Sec. 4.3, we describe the specific metrics used to evaluate system-level trade-offs. We then provide experimental results from two case studies: (1) a neural-prosthesis system, described in Sec. 4.4, and (2) a seizure-detection system (involving multi-rate processing) described in Sec. 4.5. Finally, we conclude in Sec. 4.6.

4.2 Signal Processing in the Compressed Domain

In this section, we show the feasibility of compressed-domain equivalents $\hat{\mathbf{H}}$ for any signal-processing function, which can be represented as a matrix operation $\mathbf{H}$. Our approach involves minimizing the error in the inner product between FVs, since this is a key computation in kernel functions for inference stages. We show that $\hat{\mathbf{H}}$ permits very low distortion errors with respect to the inner-products between FVs.

Relating back to the systems in Fig. 3.3, recall that our aim is to find a matrix transformations $\hat{\mathbf{H}}$ that leads to representations of a signal with the intended signal processing, but derived *by directly using* $\hat{\mathbf{x}}$.

Need for regularization. Suppose we can process each vector $\hat{\mathbf{x}}$ in CA by a matrix operator $\hat{\mathbf{H}}$ to derive the compressed-domain FV $\hat{\mathbf{y}}$. A naive approach might be to find $\hat{\mathbf{H}}$ such that the output vector $\hat{\mathbf{y}}$ equals $\mathbf{y}$ from NA. This gives the following formulation:

$$\mathbf{y} = \hat{\mathbf{y}} \;\Rightarrow\; \mathbf{H}\mathbf{x} = \hat{\mathbf{H}}\hat{\mathbf{x}} \;\Rightarrow\; \mathbf{H}\mathbf{x} = \hat{\mathbf{H}}\mathbf{\Phi}\mathbf{x}$$

$$\Rightarrow \mathbf{H} = \hat{\mathbf{H}}\mathbf{\Phi}. \qquad (4.1)$$

$$N \times N \quad N \times M \quad M \times N$$

However, with $M \ll N$, matrix $\hat{\mathbf{H}}$ above corresponds to $N \times M$ variables constrained by $N \times N$ equations. Such a system with fewer variables than equations is overdetermined and has no exact solution. In Chapter 3, we proposed an approach to regularize the left hand side of Eq. (4.1) through $\mathbf{\Phi}$. The resulting solution for $\hat{\mathbf{H}}$ was accurate only in the least-squares sense. We show next how an auxiliary matrix $\mathbf{\Theta}$ can be used instead of $\mathbf{\Phi}$ to introduce additional degrees of freedom in order to solve for $\hat{\mathbf{H}}$ exactly. Instead of solving for $\mathbf{y} = \hat{\mathbf{y}}$ [as in Eq. (4.1)], we solve for some K-dimensional projection $\mathbf{\Theta}\mathbf{y}$ of $\mathbf{y}$. This is similar to the approach presented in Chapter 3. However, in this case, the elements of the $K \times N$ auxiliary matrix $\mathbf{\Theta}$ are now design variables along with $\hat{\mathbf{H}}$. Thus, we need to solve

for $\boldsymbol{\Theta}$ and $\hat{\mathbf{H}}$ simultaneously in the following equation:

$$\boldsymbol{\Theta}\mathbf{y} = \hat{\mathbf{y}} \;\Rightarrow\; \boldsymbol{\Theta}\mathbf{H}\mathbf{x} = \hat{\mathbf{H}}\boldsymbol{\Phi}\mathbf{x}$$

$$\Rightarrow \underset{K\times N}{\boldsymbol{\Theta}}\underset{N\times N}{\mathbf{H}} = \underset{K\times M}{\hat{\mathbf{H}}}\underset{M\times N}{\boldsymbol{\Phi}}. \tag{4.2}$$

With $M \ll N$, $\boldsymbol{\Theta}$ and $\hat{\mathbf{H}}$ together correspond to $K \times (N + M)$ variables constrained by $K \times N$ equations. Thus, with more variables than constraints, Eq. (4.2) will have an infinite number of solutions. This lets us set constraints for finding unique solutions that make several useful design options available:

1. It enables us to *solve exactly* for the compressed-domain processing matrix $\hat{\mathbf{H}}$, avoiding additional error sources in the processing.

2. By using a smaller value of K, it also permits us to solve for an *approximate* $\hat{\mathbf{H}}$ *of smaller size*. This solution provides us with a knob to scale the number of computations performed in CA based on the required accuracy for solving Eq. (4.2).

Additionally, by introducing $\boldsymbol{\Theta}$, Eq. (4.2) allows us to extend our methodology from signal-processing operations where $\mathbf{H}$ is a square matrix to those where $\mathbf{H}$ is a non-square matrix (*e.g.,* multi-rate system). We consider the above cases in the sub-sections ahead.

Before proceeding, we parameterize the dimensionality of $\boldsymbol{\Theta}$ and relate it to the dimensionality of $\hat{\mathbf{H}}$; this will ease our consideration of the scaling trade-offs related to accuracy and energy. The size of compressed-domain processing matrix $\hat{\mathbf{H}}$ is governed by the size of $\boldsymbol{\Theta}$ and $\boldsymbol{\Phi}$ [see Eq. (4.2)]. Thus, in addition to compression factor $\xi = N/M$, we define a parameter called projection factor ν for $\boldsymbol{\Theta}$ as follows:

$$\nu = N/K. \tag{4.3}$$

Note that $\nu > 1$ (< 1) denotes a compressive (expansive) projection $\mathbf{\Theta}$. Similarly, $\xi > 1$ (< 1) denotes a compressive (expansive) projection $\mathbf{\Phi}$. These, in turn, imply fewer (more) computations associated with $\hat{\mathbf{H}}$.

4.2.1 Exact Solution for $\hat{\mathbf{H}}$ when H is Square (for Highest Accuracy)

The intuition behind solving for a projection of $\mathbf{y}$ instead of $\mathbf{y}$ itself in Eq. (4.2) is that many machine-learning stages (*e.g.,* SVMs) that act after feature extraction do not use the exact value of $\mathbf{y}$ but only its distance from other vectors. Thus, the Euclidean distance between FVs is the metric we should aim to preserve. The distance between any two FVs, $\mathbf{y_1}$ and $\mathbf{y_2}$, is given by the inner product: $\mathbf{y_1^T y_2}$. The corresponding distance in the compressed domain is given by:

$$\hat{\mathbf{y_1}}^{\mathbf{T}}\hat{\mathbf{y_2}} \Rightarrow (\mathbf{\Theta y_1})^{\mathbf{T}}(\mathbf{\Theta y_2}) \Rightarrow \mathbf{y_1^T}(\mathbf{\Theta^T\Theta})\mathbf{y_2}. \tag{4.4}$$

The right hand side will be equal to the inner product $\mathbf{y_1^T y_2}$ of NA if $\mathbf{\Theta^T\Theta}$ is equal to the $N \times N$ identity matrix $\mathbf{I}$. Thus, to solve for $\mathbf{\Theta}$ and $\hat{\mathbf{H}}$ exactly in Eq. (4.2), we have to solve the following constrained optimization problem:

$$\underset{\mathbf{\Theta}}{\text{argmin}} \; \| \mathbf{\Theta^T\Theta} - \mathbf{I} \|_2^2 \;\; \text{such that} \;\; \mathbf{\Theta H} = \hat{\mathbf{H}}\mathbf{\Phi}. \tag{4.5}$$

Assuming $\mathbf{H}$ is a square matrix (*e.g.*, DWT in NA), we can obtain the SVD of $\mathbf{\Phi H^{-1}}$ as $\mathbf{VSU^T}$, where $\mathbf{V}$ and $\mathbf{U}$ are orthogonal matrices (*i.e.*, $\mathbf{U^TU} = \mathbf{V^TV} = \mathbf{I}$) and $\mathbf{S}$ is an $M \times M$ diagonal matrix formed by the singular values of $\mathbf{\Phi H^{-1}}$. We thus have the following relationship for $\mathbf{\Theta^T\Theta}$:

$$\mathbf{\Theta^T\Theta} = (\hat{\mathbf{H}}\mathbf{\Phi H^{-1}})^{\mathbf{T}}\hat{\mathbf{H}}\mathbf{\Phi H^{-1}} = \mathbf{U}(\mathbf{SV^T}\hat{\mathbf{H}}^{\mathbf{T}}\hat{\mathbf{H}}\mathbf{VS})\mathbf{U^T}. \tag{4.6}$$

The distance from the above matrix to the identity will be at least the rank deficiency of $\mathbf{U}$. The lower bound in Eq. (4.5) will thus be achieved if we set $K = M$ (or $\nu = \xi$),

$$\hat{\mathbf{H}} = \mathbf{S}^{-1}\mathbf{V}^{\mathbf{T}}, \quad \text{and}$$
$$\boldsymbol{\Theta} = \hat{\mathbf{H}}\boldsymbol{\Phi}\mathbf{H}^{-1}. \tag{4.7}$$

4.2.2 Approximate Solution for $\hat{\mathbf{H}}$ when H is Square (for Designer-controllable Energy Savings)

In this section, we show how to solve for $\boldsymbol{\Theta}$ and an approximate $\hat{\mathbf{H}}$ to save computational energy in CA. The final solution for the compressed-domain processing matrix will have the dimensionality $K \times N$ and $K \times M$ ($K < M$ or $\nu > \xi$), respectively. Such an approach (with a smaller $\hat{\mathbf{H}}$ matrix) would reduce the number of computations by trading overall system accuracy. This can be done with the help of the JL lemma [110], which states that the inner product of vectors are preserved under random projections. According to this lemma, $\hat{\mathbf{y}_1}^{\mathbf{T}}\hat{\mathbf{y}_2}$ in Eq. (4.4) will be approximately equal to $\mathbf{y_1^T y_2}$, if the entries of the auxiliary matrix $\boldsymbol{\Theta}$ are drawn from normal distribution $N(0, 1)$ [109]. Thus, we can solve the following modified problem.

$$\text{Find } \boldsymbol{\Theta} \text{ and } \hat{\mathbf{H}} \text{ such that } \boldsymbol{\Theta}\mathbf{H} = \hat{\mathbf{H}}\boldsymbol{\Phi} \text{ and } \boldsymbol{\Theta} \sim N(0, 1).$$

Suppose $\boldsymbol{\Theta}$ and $\hat{\mathbf{H}}$ comprise row vectors $\theta_{\mathbf{i}}^{\mathbf{T}}$ and $\hat{\mathbf{h}}_{\mathbf{i}}^{\mathbf{T}}$, $i \in [1, K]$, where $\theta_{\mathbf{i}}^{\mathbf{T}} \in \mathbb{R}^{\mathrm{N}}$ and $\hat{\mathbf{h}}_{\mathbf{i}}^{\mathbf{T}} \in \mathbb{R}^{\mathrm{M}}$. We have the following representation:

$$\boldsymbol{\Theta} = \begin{bmatrix} — & \theta_{\mathbf{1}}^{\mathbf{T}} & — \\ & \vdots & \\ — & \theta_{\mathbf{K}}^{\mathbf{T}} & — \end{bmatrix}_{(K \times N)} \text{ and } \hat{\mathbf{H}} = \begin{bmatrix} — & \hat{\mathbf{h}}_{\mathbf{1}}^{\mathbf{T}} & — \\ & \vdots & \\ — & \hat{\mathbf{h}}_{\mathbf{K}}^{\mathbf{T}} & — \end{bmatrix}_{(K \times M)}$$

Given the above formulation, we can simplify and represent the i^{th} row of Eq. (4.2) as follows:

$$\theta_{\mathbf{i}}^{\mathbf{T}}\mathbf{H} = \hat{\mathbf{h}}_{\mathbf{i}}^{\mathbf{T}}\mathbf{\Phi} \;\Rightarrow\; \theta_{\mathbf{i}} = \mathbf{D}\hat{\mathbf{h}}_{\mathbf{i}} \tag{4.8}$$

where $\mathbf{D^T} = \mathbf{\Phi H^{-1}}$. Note that $\mathbf{D}$ in the above equation is of dimensionality $N \times M$. Suppose the SVD of $\mathbf{D}$ is $\mathbf{USV^T}$, where orthogonal matrices $\mathbf{U}$ and $\mathbf{V}$ are of dimensionality $N \times M$ and $M \times M$, respectively, and the diagonal matrix $\mathbf{S}$, comprising the singular values of $\mathbf{D}$, is of dimensionality $M \times M$. Then we can simplify Eq. (4.8) as follows:

$$\theta_{\mathbf{i}} = \mathbf{D}\hat{\mathbf{h}}_{\mathbf{i}} = \mathbf{USV^T}\hat{\mathbf{h}}_{\mathbf{i}}. \tag{4.9}$$

Since we seek $\theta_{\mathbf{i}} \sim N(0, I_{\mathbf{N}})$, to preserve the inner products according to the JL lemma, we draw $\hat{\mathbf{h}}_i$ from $N(0, \mathbf{\Sigma})$, where $\mathbf{\Sigma} = \mathbf{VS^{-2}V^T}$. Then we derive each row of $\mathbf{\Theta}$ based on Eq. (4.9). This choice of $\hat{\mathbf{h}}_i$, in fact, gives the exact JL solution for $\hat{\mathbf{H}}$ according to the following corollary:

Corollary 2. ***(JL solution for $\hat{\mathbf{H}}$)*** *Given orthogonal matrices* $\mathbf{U}$, $\mathbf{V}$ *of dimension* $N \times M$ *and* $M \times M$*, respectively, and an* $M \times M$ *diagonal matrix of singular values* $\mathbf{S}$*. Then,* $\hat{\mathbf{h}}_{\mathbf{i}} \sim N(0, \mathbf{\Sigma})$*, where* $\mathbf{\Sigma} = \mathbf{VS^{-2}V^T}$ *and* $\hat{\mathbf{h}}_{\mathbf{i}} \in \mathbb{R}^M$*, gives the solution for* $\theta_{\mathbf{i}} = \mathbf{USV^T}\hat{\mathbf{h}}_{\mathbf{i}}$ *such that the entries of the row vector* $\theta_{\mathbf{i}}$ *are drawn i.i.d from the multivariate normal* $N(0, \mathbf{I_N})$*.*

Proof. We complete the proof by deriving the mean and variance of $\hat{\mathbf{h}}_{\mathbf{i}}$ under the assumption of $\theta_{\mathbf{i}} \sim N(0, I_M)$. Consider the following equation:

$$\theta_{\mathbf{i}} = \mathbf{USV^T}\hat{\mathbf{h}}_{\mathbf{i}} = \mathbf{Uz_i} \tag{4.10}$$

where $\mathbf{z_i} = \mathbf{SV^T}\hat{\mathbf{h}}_{\mathbf{i}}$ is an M-dimensional vector of random variables. Since $\theta_{\mathbf{i}} \sim N(0, I_M)$ and $\mathbf{U}$ is a constant matrix, $z_i \sim N(0, \mathbf{I_N})$. Further, since $\hat{\mathbf{h}}_{\mathbf{i}} = \mathbf{VS^{-1}z_i}$, we can compute the

mean of $\hat{\mathbf{h}}_\mathbf{i}$ as $\mathbb{E}\left[\hat{\mathbf{h}}_\mathbf{i}\right] = \mathbb{E}\left[\hat{\mathbf{z}}_\mathbf{i}\right] = 0$, and the variance of $\hat{\mathbf{h}}_\mathbf{i}$ as follows:

$$\begin{aligned}\mathbb{E}\left[\hat{\mathbf{h}}_\mathbf{i}\hat{\mathbf{h}}_\mathbf{i}^\mathbf{T}\right] &= \mathbb{E}\left[\mathbf{V}\mathbf{S}^{-1}\mathbf{z}_\mathbf{i}\mathbf{z}_\mathbf{i}^\mathbf{T}\mathbf{S}^{-1}\mathbf{V}^\mathbf{T}\right] \\ &= \mathbf{V}\mathbf{S}^{-1}\,\mathbb{E}\left[\mathbf{z}_\mathbf{i}\mathbf{z}_\mathbf{i}^\mathbf{T}\right]\mathbf{S}^{-1}\mathbf{V}^\mathbf{T} \\ &= \mathbf{V}\mathbf{S}^{-2}\mathbf{V}^\mathbf{T}.\end{aligned}$$

■

Thus, the approximate solution for matrix $\hat{\mathbf{H}}$ is of dimension $K \times M$, where $K < M$ (or $\nu > \xi$). Such an $\hat{\mathbf{H}}$ saves computational energy in CA. This energy saving comes at the cost of accuracy. However, we will present a case study ahead that suggests that this cost can be small and, in fact, we can reliably employ $K \ll M$ ($\nu \gg \xi$). Next, we show that the above approach is also applicable to multi-rate signal-processing systems, and we solve Eq. (4.5) when $\mathbf{H}$ is a non-square matrix.

Note that the approximate solution is also applicable to the case when $\mathbf{\Theta}$ has more rows than columns *i.e.*, when $K > M$ ($\nu < \xi$). As compared to the exact solution where $K = M$ (or $\nu = \xi$), this case results in a $\hat{\mathbf{H}}$, which is bigger in size than the $\hat{\mathbf{H}}$ obtained from the exact solution. Thus, the design space of $K > M$ ($\nu < \xi$) results in a higher computational complexity and reduced accuracy as compared to the exact solution. It is thus not a very useful design space for low-energy operation. We highlight this case with a cross in all subsequent figures that show the performance/energy of the approximate solution.

4.2.3 Solution for $\hat{\mathbf{H}}$ when H is Non-square

For this case, we can also exploit the JL lemma to derive a near-orthogonal matrix $\mathbf{\Theta}$ and then solve for $\hat{\mathbf{H}}$ using the SVDs of $\mathbf{H}$ and $\mathbf{\Phi}$. To solve Eq. (4.5) for $\mathbf{\Theta}$ and $\hat{\mathbf{H}}$, we take the transpose of Eq. (4.2) and multiply with itself, obtaining the following relationship:

$$\begin{aligned}(\mathbf{\Theta}\mathbf{H})^\mathbf{T}(\mathbf{\Theta}\mathbf{H}) &= \left(\hat{\mathbf{H}}\mathbf{\Phi}\right)^\mathbf{T}\left(\hat{\mathbf{H}}\mathbf{\Phi}\right) \\ \mathbf{H}^\mathbf{T}\mathbf{\Theta}^\mathbf{T}\mathbf{\Theta}\mathbf{H} &= \mathbf{\Phi}^\mathbf{T}\hat{\mathbf{H}}^\mathbf{T}\hat{\mathbf{H}}\mathbf{\Phi} \\ \mathbf{R}\mathbf{Q}\mathbf{P}^\mathbf{T}\mathbf{\Theta}^\mathbf{T}\mathbf{\Theta}\mathbf{P}\mathbf{Q}\mathbf{R}^\mathbf{T} &= \mathbf{U}\mathbf{S}\mathbf{V}^\mathbf{T}\hat{\mathbf{H}}^\mathbf{T}\hat{\mathbf{H}}\mathbf{V}\mathbf{S}\mathbf{U}^\mathbf{T},\end{aligned} \tag{4.11}$$

where $\mathbf{H} = \mathbf{PQR^T}$ and $\mathbf{\Phi} = \mathbf{VSU^T}$ are the SVDs of $\mathbf{H}$ and $\mathbf{\Phi}$, respectively. Since $\mathbf{H}$ is of dimensionality $L \times N$ ($L < N$), $\mathbf{P}$, $\mathbf{Q}$, and $\mathbf{R}$ are of dimensionality $L \times L$, $L \times L$, and $N \times L$, respectively. Similarly, since $\mathbf{\Phi}$ is of dimensionality $M \times N$ ($M < N$), $\mathbf{U}$, $\mathbf{S}$, and $\mathbf{V}$ are of dimensionality $N \times M$, $M \times M$, and $M \times M$, respectively. If we let $\mathbf{\Theta} = \mathbf{BQ^{-1}P^T}$ and $\hat{\mathbf{H}} = \mathbf{AS^{-1}V^T}$ in Eq. (4.11), we have the following relationship:

$$\mathbf{RB^TBR^T} = \mathbf{UA^TAU^T}$$

$$\Rightarrow \mathbf{U^TRB^TBR^TU} = \mathbf{A^TA}$$

where $\mathbf{A}$ and $\mathbf{B}$ are unknown matrices that need to be determined. We can invoke the JL lemma and draw the $K \times L$ elements of $\mathbf{\Theta}$ from $N(0, 1)$. We can thus solve for the $K \times L$ matrix $\mathbf{B} = \mathbf{\Theta PQ}$ and use the above equation to derive the $K \times M$ matrix $\mathbf{A} = \mathbf{BR^TU}$. Finally, we obtain the $K \times M$ matrix $\hat{\mathbf{H}} = \mathbf{AS^{-1}V^T}$.

Algorithm 1 shows the pseudocode (with the correct scaling constants) that summarizes our approach of simultaneously solving for $\mathbf{\Theta}$ and $\hat{\mathbf{H}}$ under the three conditions described in this section. For the case of a non-square $L \times N$ ($L > N$) processing matrix $\mathbf{H}$, the algorithm also shows (on line 15) an optional step of orthogonalization (*e.g.,* by the Gram-Schmidt process) before deriving $\mathbf{B}$, $\mathbf{A}$, and $\hat{\mathbf{H}}$. This ensures a perfectly orthonormal $\mathbf{\Theta}$ when its row rank is greater than the column rank. Next, we describe system-level metrics that will be used to evaluate our approach in CA.

4.3 Metrics Used to Evaluate the Proposed Approach

The approach of the previous section opens up many system-design options. To understand the associated accuracy trade-offs, in this section, we discuss precise metrics that are relevant in inference applications. In addition to comparing the proposed CA with NA as a baseline approach, we also compare it with RA in which the sensor node transmits compressed data to an external platform to reduce the amount of data transmitted (hence, saving

Algorithm 1 Find compressed-domain processing matrix $\hat{\mathbf{H}}$

Require: projection dimension K and matrices $\mathbf{\Phi}$ and $\mathbf{H}$
Ensure: $\mathbf{\Theta}$ and $\hat{\mathbf{H}}$ with $\mathbf{\Theta H} = \hat{\mathbf{H}}\mathbf{\Phi}$

Init: $N \leftarrow$ # cols($\mathbf{\Phi}$); $M \leftarrow$ # rows($\mathbf{\Phi}$); $L \leftarrow$ # rows($\mathbf{H}$)
if $L = N$ **then**
  $\mathbf{D^T} := \mathbf{\Phi H^{-1}}$; $\mathbf{USV^T} \leftarrow$ SVD($\mathbf{D}$); {for $\theta_\mathbf{i} = \mathbf{D}\hat{\mathbf{h}}_\mathbf{i}$}
  if $K = M$ **then**
    $\hat{\mathbf{H}} = \sqrt{(N/M)}\left(\mathbf{S^{-1}V^T}\right)$; $\mathbf{\Theta} = \sqrt{(N/M)}\left(\hat{\mathbf{H}}\mathbf{\Phi H^{-1}}\right)$;
  else
    for $i = 1$ **to** K **do**
      $\mathbf{x_i} \sim N(0, \mathbf{I_M})/\sqrt{(K)}$; {for $\hat{\mathbf{h}}_\mathbf{i} \sim N\left(0, \mathbf{VS^{-2}V^T}\right)$}
      $\hat{\mathbf{h}}_\mathbf{i} = \mathbf{VS^{-1}x_i}$; $\theta_\mathbf{i} = \mathbf{Ux_i}$;
    end for
    $\mathbf{\Theta} = \sqrt{(N/M)}\left(\theta_1^T; \ldots; \theta_K^T\right)$; $\hat{\mathbf{H}} = \sqrt{(N/M)}\left(\hat{h}_1^T; \ldots; \hat{h}_K^T\right)$;
  end if
else
  $\mathbf{PQR^T} \leftarrow$ SVD($\mathbf{H}$); $\mathbf{VSU^T} \leftarrow$ SVD($\mathbf{\Phi}$);
  $\mathbf{\Theta} \sim N(0, 1) / \sqrt{(NK/M)}$; {ortho($\mathbf{\Theta}$) if $K > L$}
  $\mathbf{B} = \mathbf{\Theta PQ}$; $\mathbf{A} = \mathbf{BR^T U}$; $\hat{\mathbf{H}} = \sqrt{N/M}\left(\mathbf{AS^{-1}V^T}\right)$;
end if

communication energy and/or alleviating bandwidth constraints); the data are reconstructed on the external platform before performing signal processing. Fig. 4.1 shows the metrics we use. Since, in CA, we solve for a random projection $\mathbf{\Theta}$ of the FV [see Eq. (4.2)], we expect to be able to reconstruct the signal features accurately. Thus, we reconstruct the FVs in CA and compare them with the features extracted from reconstructed signals in RA. We also compare the variation in the inner-product error (IPE) and the accuracy of the inference stage with respect to both ξ and ν.

Reconstruction SNR with respect to ξ: Since CA solves for a projection of the processed signal ($\mathbf{\Theta y}$) in NA, the accuracy of processing in CA is expected to be correlated with our ability to recover the $\mathbf{y}$ features from $\mathbf{\Theta y}$. If we denote the reconstructed features as $\mathbf{y^*_{CA}}$, we can define the SNR in CA as follows:

$$\text{SNR}_{CA} = 10 \cdot log\left[\|\mathbf{y}\|_2^2 / \left(\|\mathbf{y^*_{CA}} - \mathbf{y}\|_2^2\right)\right] \text{ dB}. \tag{4.12}$$

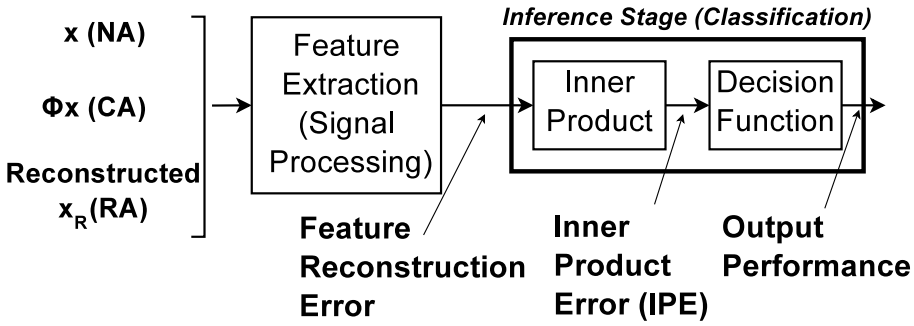


Figure 4.1: Metrics used to evaluate the performance of NA, RA, and CA.

Similarly, the performance in RA is governed by our ability to recover the $\mathbf{y}^*_{\mathbf{RA}}$ features. However, since reconstruction occurs before processing in RA, the reconstructed features $\mathbf{y}^*_{\mathbf{RA}}$ are related to the reconstructed signal $\mathbf{x}^*_{\mathbf{RA}}$ as $\mathbf{y}^*_{\mathbf{RA}} = \mathbf{H}\mathbf{x}^*_{\mathbf{RA}}$. Thus, the SNR in RA can be defined as follows:

$$\text{SNR}_{RA} = 10 \cdot log\left[\|\mathbf{y}\|_2^2 / \left(\|\mathbf{H}\mathbf{x}^*_{\mathbf{RA}} - \mathbf{y}\|_2^2\right)\right] \text{ dB}. \tag{4.13}$$

We will investigate how close the SNR in CA is with respect to the SNR in RA for the two case studies in Secs. 4.4.3 and 4.5.2, respectively.

IPE in feature extraction with respect to ξ: For feature extraction and classification, a primary concern is how the IPE of FVs scales with ξ. The total inner product error (IPE-T) between $\hat{\mathbf{y}}^{\mathbf{T}}\hat{\mathbf{y}}$ and $\mathbf{y}^{\mathbf{T}}\mathbf{y}$ comprises errors arising from the use of two projection matrices: (1) $\mathbf{\Phi}$, which gives rise to an error we define as IPE-1, and (2) $\mathbf{\Theta}$, which gives rise to an error we define as IPE-2. Also, for any two FVs $\mathbf{y_i}$ and $\mathbf{y_j}$, IPE-T between the inner product in CA (*i.e.,* $\hat{\mathbf{y}}_\mathbf{i}^{\mathbf{T}}\hat{\mathbf{y}}_\mathbf{j}$) and the inner product in NA (*i.e.,* $\mathbf{y}_\mathbf{i}^{\mathbf{T}}\mathbf{y}_\mathbf{j}$) is given by the following equation:

$$\text{IPE-T} = |\hat{\mathbf{y}}_\mathbf{i}^{\mathbf{T}}\hat{\mathbf{y}}_\mathbf{j} - \mathbf{y}_\mathbf{i}^{\mathbf{T}}\mathbf{y}_\mathbf{j}|/(\mathbf{y}_\mathbf{i}^{\mathbf{T}}\mathbf{y}_\mathbf{j}) \tag{4.14}$$

Earlier, we showed that the distance from $\mathbf{\Theta^T\Theta}$ to the identity matrix was the rank deficiency of $\mathbf{U}$ (*i.e.*, $\mathbf{UU^T}$). This distance actually represents a projection onto the subspace of $\mathbf{U}$, since a projection is defined as $\mathbf{U}\left(\mathbf{U^TU}\right)^{-1}\mathbf{U^T}$, which simplifies to $\mathbf{UU^T}$ owing to the orthogonality of $\mathbf{U}$. Also, since $\mathbf{U}$ is derived from $\mathbf{\Phi}$, the above distance actually represents the error introduced by projection $\mathbf{\Phi}$ (*i.e.*, IPE-1). This error is intrinsic to the system. IPE-1 is thus the best-achievable performance with any subsequent feature extraction. This lower bound can be obtained by solving Eq. (4.7). Using this CA solution, we can determine the error sub-components IPE-1 and IPE-2 as follows:

$$\begin{aligned} \text{IPE-1} &= |\mathbf{y_i^TUU^Ty_j} - \mathbf{y_i^Ty_j}|/(\mathbf{y_i^Ty_j}) \\ \text{IPE-2} &= |\hat{\mathbf{y}}_\mathbf{i}^\mathbf{T}\hat{\mathbf{y}}_\mathbf{j} - \mathbf{y_i^TUU^Ty_j}|/(\mathbf{y_i^TUU^Ty_j}). \end{aligned} \tag{4.15}$$

The exact solution in CA makes IPE-2 as close to 0 as possible. We study the scaling characteristics of IPE-T with respect to the dimensionality of $\mathbf{\Theta}$. We explore this trade-off for the spike-sorting application in Sec. 4.4.3.

IPE-T with respect to ν: In Sec. 4.4.3, we also investigate how scaling of the first dimension K (or ν) of $\hat{\mathbf{H}}$ and $\mathbf{\Theta}$ degrades IPE-T. If it degrades at a slow rate, it enables us to use a smaller $\hat{\mathbf{H}}$ and, hence, reduce the amount of computation significantly. The rate of degradation can be quantified by invoking the distance-preservation guarantees from [109]. For an input vector $\mathbf{x}$, we have the following relationship (from the near-orthogonality of $\mathbf{\Theta}$, as discussed in Sec. 4.2.2):

$$\|\mathbf{\Theta x}\| \approx \|\mathbf{UU^Tx}\| \tag{4.16}$$

However, since $\mathbf{\Phi}$ is a random projection, from [109], we have $\|\mathbf{UU^Tx}\| \approx \|\mathbf{x}\|$. With the scaling of dimensionality K of $\mathbf{\Theta}$, the rate of degradation in the above approximation (and thus in IPE-T) is indirectly governed by the minimum $[\sigma_{min}(\mathbf{\Theta})]$ and maximum $[\sigma_{max}(\mathbf{\Theta})]$

singular values of $\mathbf{\Theta}$ as follows:

$$1 - \sqrt{N/(MK)} < \sigma_{min}(\mathbf{\Theta}) < \sigma_{max}(\mathbf{\Theta}) < 1 + \sqrt{N/(MK)} \tag{4.17}$$

In Sec. 4.4.3, we determine if the measured bounds for the singular values of $\mathbf{\Theta}$ are indeed close to the above theoretical bounds and then investigate the distribution of IPE-T with respect to ν.

Inference performance with respect to ξ: Recall that $\xi = N/M$ quantifies the amount of compression achieved by compressive sensing. As ξ becomes larger, we expect the performance of RA and CA to deteriorate with respect to NA. The first question that arises is: till what value of ξ do RA and CA remain competitive with NA for an application of interest? The second question is: as we increase the value of ξ, does CA remain competitive with RA? If it does, then computations can viably be performed on the sensor node, with the additional benefit of computational energy reduction (due to the fewer operations required in CA). This suggests a new design approach to energy-constrained sensor nodes, wherein devices can be made more computationally powerful, thanks to energy savings enabled by the explicit use of efficient representations for the embedded signals; this approach to energy reduction can be exploited alongside algorithmic and architectural optimizations. We explore these questions for two case studies in Secs. 4.4.3 and 4.5.2, respectively.

Inference performance with respect to ν: Recall that $\nu = N/K$ provides us with a knob to obtain additional computational energy savings in our CA approach since the approximate solution permits a smaller $\hat{\mathbf{H}}$ matrix. These savings come at the cost of accuracy. The first question is what would the impact on performance and computational energy be if we simultaneously turn the ν and ξ knobs? The second question is how the accuracy and energy savings compare to the case where an exact solution is used for $\hat{\mathbf{H}}$? For the two case studies, we explore these scaling trends in Secs. 4.4.3 and 4.5.2, respectively.

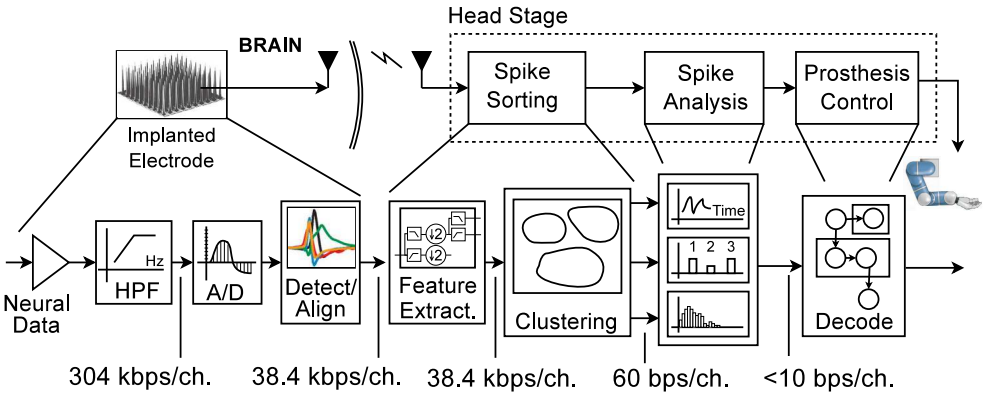


Figure 4.2: Block diagram of a typical system for neural prosthesis: we assume a sampling rate of 31.2 kilo Samples per second per channel (kSps/ch) and 10 bits per Sample (bpS). Also, the average firing rate is assumed to be 60 spikes/s/ch at 64 Samples/spike.

4.4 Case Study I: Neural Prosthesis with Compressively-sensed Spikes

In this section, we use the case study of a neural-prosthesis system to validate the system-level trade-offs arising from the proposed approach. We derive the compressed-domain equivalent of a square matrix $\mathbf{H}$, which computes the DWT of spike signals. We study the impact of the exact and approximate solutions for $\hat{\mathbf{H}}$ on system performance. We also analyze the IPE-T and SNR trends in NA and CA. Before we proceed, we present some background highlighting the platform-level constraints in neural prosthesis.

4.4.1 Neural Prosthesis System

Fig. 4.2 shows the block diagram of a typical neural-prosthesis system. After analog processing and digitization, it comprises four major computational blocks: spike detection/alignment, sorting, analysis, and prosthesis control [175]. Information in the spikes is contained in frequencies up to 8 kHz, necessitating sampling rates of at least 16 kHz [20]. Such high sampling rates over a large number of sensing channels can lead to large amounts of data. From an implant, transmission to an external head-stage is typically achieved serially, thus requiring buffering over all channels, amounting to data rates up to 1Mbps [2].

Many systems detect and align spikes locally before transmission [24, 176, 177]; as shown in Fig. 4.2, this can reduce the data rate by nearly 8×. Since an implanted electrode records the simultaneous firing of multiple neurons, spikes are sorted on the external head-stage before analysis [175]. This involves feature extraction and clustering [178–181]. DWT and K-means are two widely employed algorithms for feature extraction and clustering, respectively [175, 182]. After spike sorting, data rates can be significantly lowered, since the sorted spikes (corresponding to individual neurons) can be represented by threshold comparison using a Heaviside function [180]. Spike trains from each sorted cluster can then be analyzed to extract statistical parameters, such as spike count (SC), neuron firing rate (FR), inter-spike interval (ISI), coefficient of variation (CV), *etc.* [179]. These parameters eventually steer an algorithm for prosthesis control [183–185].

Energy constraints: Next, we discuss the energy constraints in the system and show how CA can help alleviate them. We also discuss trade-offs in implementing spike sorting on the implant versus on the head-stage. Detecting spikes on the implant can significantly reduce the amount of data transmitted to the external head-stage (from 304 kbps/channel to 38.4 kbps/channel). Processing spikes further on the implant (*i.e.*, sorting) may thus sound promising to reduce the amount of data even further (from 38.4 kbps/channel to 60 bps/channel). This can alleviate communication constraints on the implant significantly. However, power density constraints limit the number and type of computations, which can be supported *in vivo* [186]. A further limitation is imposed by the mode of power delivery to the implant – wireless powering of the implant (*e.g.,* through inductive coupling) is often preferred, and thus minimizing computations is critical to adhere to the implant power envelop. Nevertheless, some systems in the literature attempt to implement sorting on the implant [187, 188]. These approaches, however, do not scale easily with increasing data channels and are only designed to support specific algorithms. Thus, full-scale NA for spike sorting on the implant is not feasible. Table 4.1 quantifies the severity of the power constraints involved; analog processing alone can incur more than 200 $\mu W/mm^2$

Table 4.1: Estimated system energy requirements assuming 60 spikes/s/ch. and 60 S/spike. The number of OPS were obtained using Lightspeed [77].

Functions on implanted electrode				
	0.5 μm CMOS		0.13 μm CMOS	
AFE+ADC+HPF	487 $\mu W/mm^2$ [24]		262 $\mu W/mm^2$ [20]	
Spike Detect./ Align			6 $\mu W/mm^2$ [192]	
Functions on external computation platform				
Estimated power using 29.0 pW/OP from [187] and [181]				
Computational Block		μW/Ch. (MOPS)		μW/Ch. (MOPS)
Feature Extraction	DWT	14.25 (0.49)	PCA	5.77 (0.20)
Clustering	K-means	66.86 (2.31)		
Spike Train Analysis	FR	335.54 (11.57)	CV	71.59 (2.47)

of power on the implant (excluding any communication costs). Adding computations like DWT and K-means would push the power density very close to its maximum allowable value of 220 $\mu W/mm^2$. The middle ground of performing part of the sorting computations (*viz.* feature extraction) on the sensor, as suggested in [24, 181], may be viable. However, this adds to the computational energy burden on the implant without alleviating its communication burden (since still 38.4 kbps/channel must be transmitted). Thus, this is not desirable. Another approach could be to compress the detected spikes on the implant before transmission. Compressing spikes using DWT has been presented in [189] but the complexity of DWT limits scalability beyond 32 channels [190, 191].

Since compressive sensing permits large compression ($>$ 10$\times$) at a low cost in the energy required for compression, representing spikes using compressive sensing after detecting them on the implant seems like a viable approach. This could reduce the amount of data transmission from 38.4 kbps/ch to 3.84 kbps/ch. We would now need to reconstruct the spikes on the head-stage before sorting. This amounts to the RA architecture. However, this approach is not feasible since reconstruction can be very costly in energy and time. Recall

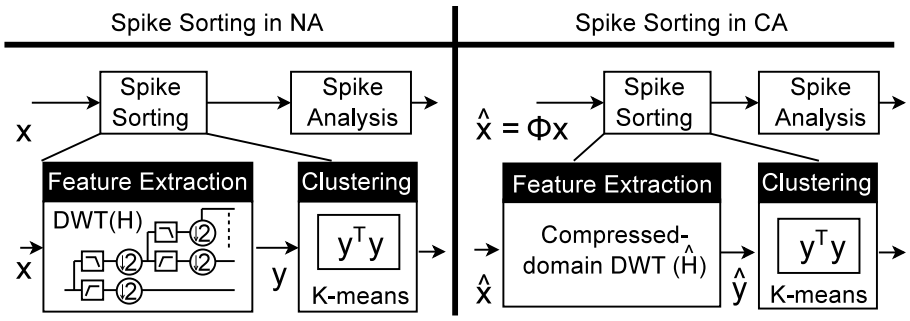


Figure 4.3: DWT feature extraction in the neural-prosthesis system; the DWT operation is formulated as a matrix $\mathbf{H}$ in NA, and a corresponding matrix $\hat{\mathbf{H}}$ is derived for CA.

from Sec. 2.2 that even at a compression of just 3×, it takes four orders of magnitude more operations to reconstruct than compress the signal. This is not just true for the neural prosthesis system, but for other applications (such as EEG) as well, as shown. Reconstruction also precludes real-time operation on the battery-powered external head-stage. The final choice is CA, which we described in the previous section. In CA, we perform spike sorting directly on compressively-sensed data. This can be done either on the external head-stage or on the implant itself. If done on the head-stage, it permits real-time operation by avoiding reconstruction, while potentially reducing the computational energy of spike sorting. Our results below suggest that the computational energy can be reduced by more than an order of magnitude. CA can reduce the communication constraints of the implant drastically since now only 60 bps/ch needs to be transmitted. This implies that low-energy or zero-energy communication links (*e.g.*, based on passive impedance modulation [2]) may be viable. The cost, however, is a small increase in computational energy (for the random projection of data) on the implant. In Sec. 4.4.3, we evaluate the benefits in computational energy delivered by CA in this context. Next, we formulate feature extraction as a matrix operation to enable a transformation to CA.

4.4.2 Formulating DWT as a Matrix Operation

Fig. 4.3 shows the computations we focus on for spike sorting and analysis. Each detected spike on the implanted sensor results in a vector of N samples. Suppose we denote each of these vectors as $\mathbf{x}$. In NA (shown on the left side of Fig. 4.3), the sensor directly transmits $\mathbf{x}$ to the external head-stage. $\mathbf{x}$ is then processed by a DWT block to derive the N-sample FV $\mathbf{y}$. The DWT function shown in Fig. 4.3 can be implemented as a filter bank [193]. To enable a transformation to CA, however, we require DWT to be formulated as a matrix operation $\mathbf{H}$.

In the filter bank implementation, the DWT of a signal is derived by passing it through a series of filters. First, vector $\mathbf{x}$ is passed through a low pass filter (LPF) through convolution. The signal is also decomposed simultaneously using a high-pass filter (HPF). However, with half the frequency band removed, the outputs can be down-sampled by 2× without risk of aliasing. This comprises one level of wavelet decomposition. The process is repeated with the LPF outputs to achieve higher levels of decomposition [193]. To formulate the entire process as a matrix operation in NA, we note that the processing between a vector of filter coefficients $\mathbf{g}$ and the N-sample spike vector $\mathbf{x}$ can be represented as a convolution operation:

$$\mathbf{z} = \mathbf{g} * \mathbf{x} = \sum_{k=-\infty}^{\infty} g[n-k]x[k] = \mathbf{G_N x} \tag{4.18}$$

where $\mathbf{z}$ is the filtered signal of N samples and $\mathbf{G_N}$ is the $N \times N$ convolution matrix whose rows are shifted versions of the coefficient vector $\mathbf{g}$. For the DWT algorithm, $\mathbf{G_N^L}$ and $\mathbf{G_N^H}$ can be used to represent the LPF and HPF operations, respectively. After the filtering process, we can then implement down-sampling by 2× at each level of decomposition through an $N/2 \times N$ matrix $\mathbf{D_{2,N}}$:

$$\mathbf{D_{2,N}} = \begin{bmatrix} 1 & 0 & 0 & 0 & \cdots & 0 \\ 0 & 0 & 1 & 0 & \cdots & 0 \\ \vdots & & & & \ddots & \\ 0 & 0 & 0 & 0 & \cdots & 1 \end{bmatrix}_{(N/2 \times N)}.$$

Using a cascade of **D**-**G** operators, we can thus represent the full DWT operation in NA as the following linear transformation:

$$\mathbf{y} = \mathbf{H}\mathbf{x} = \begin{bmatrix} \boxed{\mathbf{H_1}} \\ \boxed{\mathbf{H_2}} \\ \vdots \\ \boxed{\mathbf{H_{L+1}}} \end{bmatrix}_{N\times N} \begin{bmatrix} \\ \mathbf{x} \\ \\ \end{bmatrix}_{N\times 1} \tag{4.19}$$

where $\mathbf{y}$ is the N-sample DWT of spike samples $\mathbf{x}$. For L levels of decomposition, sub-matrices $\mathbf{H_n}$ $(1 \leq n \leq L+1)$ are given by:

$$\mathbf{H_n} = \begin{cases} \mathbf{D_{2,N}G^H_N} & \text{if } n = 1 \\ \prod_{\mathbf{k=0}}^{\mathbf{n-2}} \left(\mathbf{D_{2,N/2^k}G^L_{N/2^k}}\right) \left(\mathbf{D_{2,N/2^{n-1}}G^H_{N/2^{n-1}}}\right) & \text{if } 2 \leq n \leq L \\ \prod_{\mathbf{k=0}}^{\mathbf{n-1}} \left(\mathbf{D_{2,N/2^k}G^L_{N/2^k}}\right) & \text{if } n = L+1. \end{cases}$$

Each pair of matrices, $\mathbf{G^L_{N/2^j}}$ and $\mathbf{G^H_{N/2^j}}$, in the above equation is designed to be a quadrature mirror filter based on standard mother wavelets, *e.g.*, Haar, Daubechies, Coiflet, biorthogonal wavelet, *etc.* [193]. Given the DWT formulation in NA, we can derive the corresponding DWT transformation $\hat{\mathbf{H}}$ in the compressed domain based on the approach presented in Sec. 4.2.

4.4.3 Experimental Results

In this section, we compare the experimental results for NA, RA, and CA. We begin by describing our spike sorting framework and discussing how the representative parameters (SC, CV, and FR) for prosthetic control are computed.

The spike sorting and analysis system of Fig. 4.3 is implemented in MATLAB. For our experiments, we use four records (named as E1, E2, D1, and D2) from the dataset in [182].

Each record contains 60 sec. of simulated spike data from three electrodes sampled at 24 kHz, with each spike annotated with its true cluster class. The database closely mimics signals from the neocortex and basal ganglia. We first process each record to detect and align spikes using the thresholding algorithm described in [182]. This process results in a window of 64 samples per spike (denoted by vector $\mathbf{x}$). In NA, we then process the detected spikes by a matrix $\mathbf{H}$ to extract specific signal features. $\mathbf{H}$ corresponds to the DWT matrix, which is derived from four levels of decomposition of a Haar mother wavelet. In CA and RA, however, we first project the detected spikes using a matrix $\mathbf{\Phi}$ to obtain the compressively-sensed signal $\hat{\mathbf{x}} = \mathbf{\Phi x}$. We choose each entry of $\mathbf{\Phi}$ from a uniform distribution $U(-1, +1)$ to facilitate an efficient implementation. In RA, before performing computations, we reconstruct signal $\mathbf{x_R}$ from $\hat{\mathbf{x}}$ and then apply $\mathbf{H}$. In CA, however, we directly apply matrix $\hat{\mathbf{H}}$ to compressed signal $\hat{\mathbf{x}}$. We then sort the extracted wavelet features (in NA, RA, and CA) into three clusters using the K-means algorithm. Finally, for each spike cluster, we derive SC, CV, and FR.

Baseline performance of the spike sorting algorithm. Next, we describe how SC, CV, and FR can be computed and what their values are using the various analyses (NA, CA, and RA). SC is simply determined by counting the number of spikes in each cluster after K-means. The first step in computing CV is to determine the ISI histogram. We then model the envelope of the histogram as a Poisson distribution. We directly use this model to determine CV, which is defined as the ratio of the standard deviation to the mean of the distribution function of the ISI histogram. To compute FR for each class, we first determine the number of spikes, which occur in non-overlapping windows – each of width 300 ms. Fig. 4.4 shows this binned FR estimate for the second spike cluster in record E2. We then use a Gaussian filter with a length (L) of 30 and variance (σ) of 3 to smooth the binned FR estimates. Fig. 4.4 (at the bottom) shows the final continuous FR curve for the same spike cluster. The bin-width and smoothing filter parameters are chosen empirically to avoid discontinuities in the FR curve. The mean FR is then computed from the smoothed

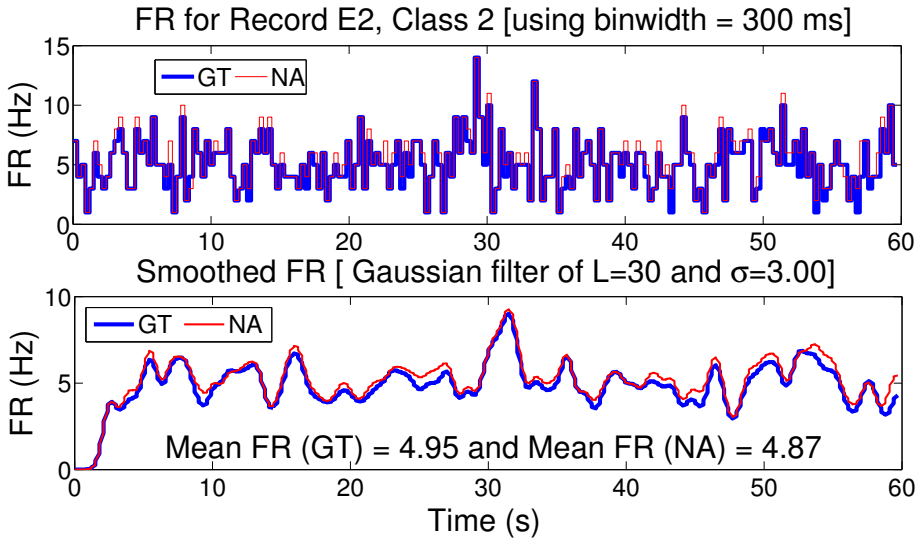


Figure 4.4: Mean FR: from the smoothed FR curve.

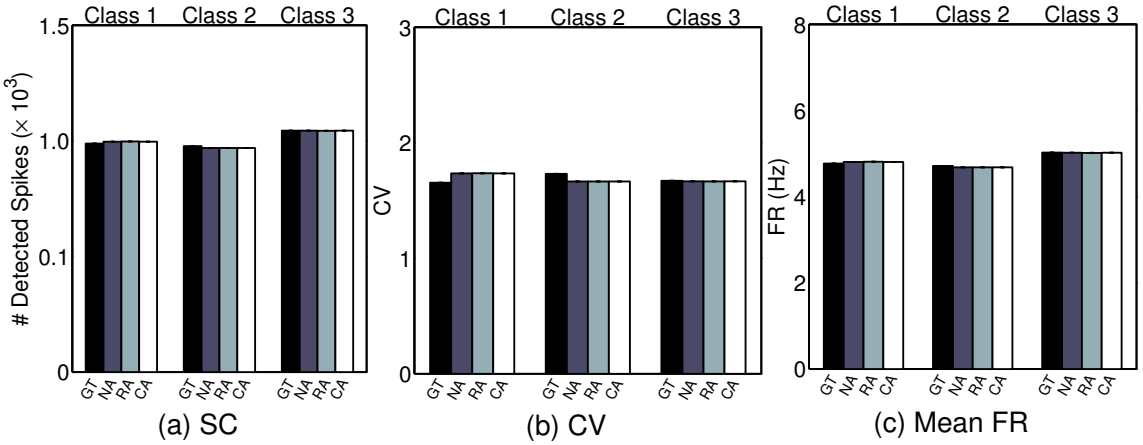


Figure 4.5: The accuracy of the spike sorting algorithm in NA is close to the ground truth (GT). Also shown are performance values for CA and RA at $\xi = 1\times$ (these are close to NA).

curve. Fig. 4.5 shows the performance of the spike sorting approach in comparison with the ground truth (GT) values. The GT values are obtained using annotations, which identify the true cluster association for each spike in the database. The end-to-end performance values for CA and RA (with no compression) are also shown. We observe that the performance of all four approaches is close to one another. Next, we study the performance metrics of Sec. 4.3 (namely SNR, IPE, and performance of CA, RA, and NA) when we scale ξ and ν.

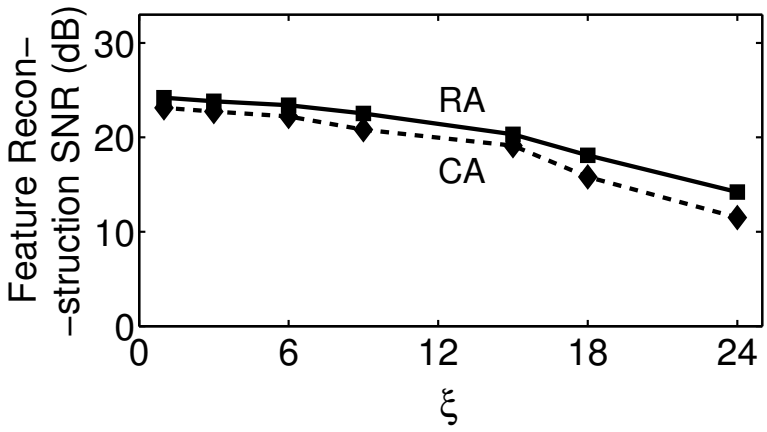


Figure 4.6: The SNR in CA is close to the SNR in RA, thus validating the similarity in performance trends.

Scaling in SNR with respect to ξ. Since the performance in CA and RA is related to our ability to reconstruct the FVs, we analyze the error introduced in each approach. Fig. 4.6 shows the mean SNR computed over spikes in all the four records. We observe that the SNR in RA is close to the SNR in CA. Although our ability to reconstruct features governs the performance trends in CA and RA, the inner-product between the features is the key parameter used in the K-means algorithm. Next, we study the IPE scaling of FVs with increasing ξ. Fig. 4.7 shows the reconstructed wavelet features in CA ($\nu = 1$) and RA at $\xi = 6\times$, $12\times$, and $24\times$. We see that the morphology of the reconstructed wavelet features in CA is similar to that obtained from RA at all compression factors (both approaches begin to show discernible degradation at $\xi = 24\times$).

IPE with respect to ξ. As mentioned earlier, IPE-T consists of IPE-1 (due to compressive-sensing error) and IPE-2 (due to $\hat{\mathbf{y}}$-approximation error). Fig. 4.8 illustrates the error components introduced before clustering in CA. We next show that IPE-2 is much smaller than IPE-1, and thus the option to approximate $\hat{\mathbf{y}}$ is beneficial for the energy savings it affords (*i.e.*, given that IPE-1 is the best-achievable lower-bound error on any subsequent feature-extraction processing). Fig. 4.9 shows histograms of IPE-1, IPE-2

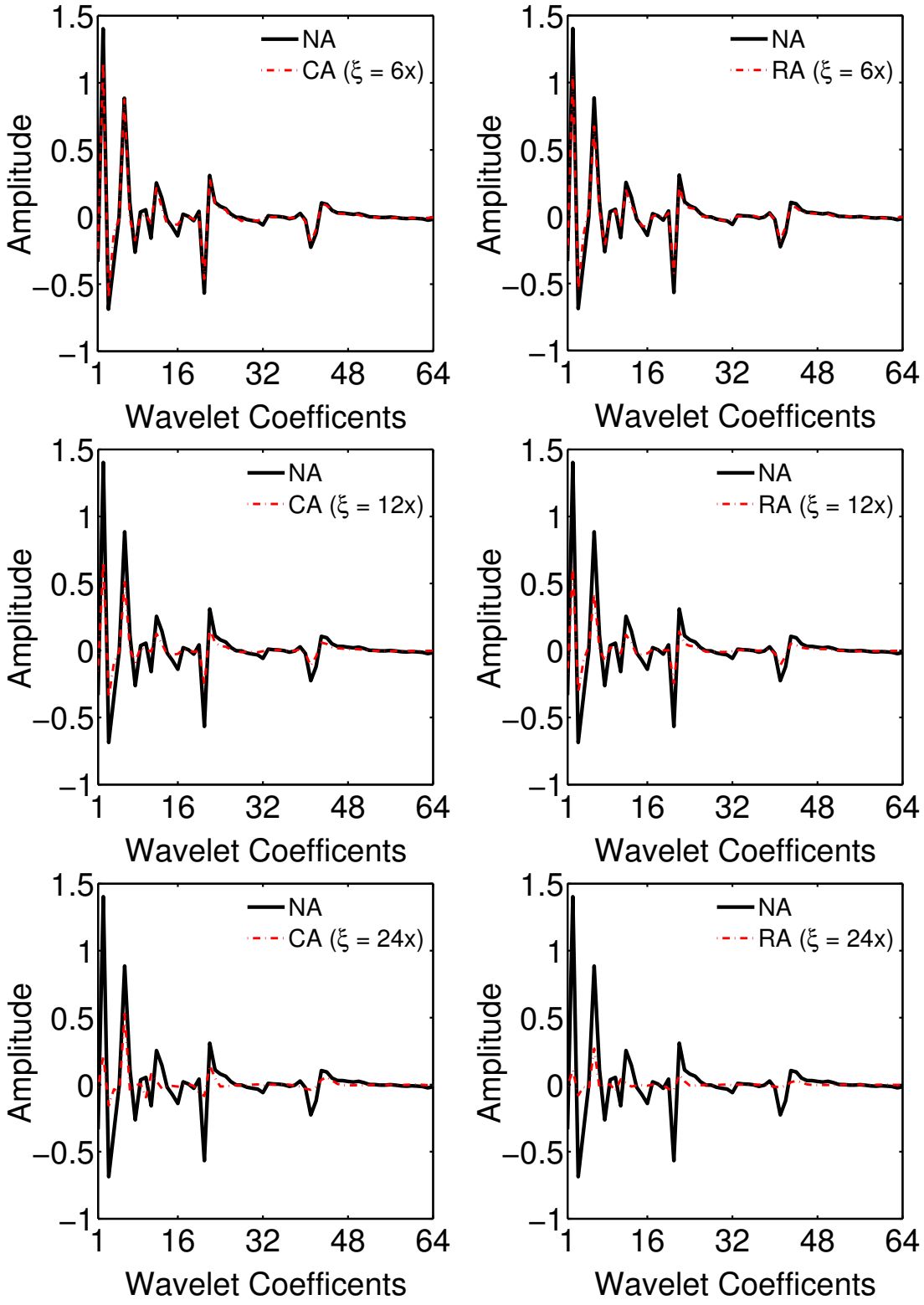


Figure 4.7: Reconstructed signal from CA (with the exact solution, $\nu = 1$) and RA (both dotted); CA is as good as RA, even at $\xi = 24\times$.

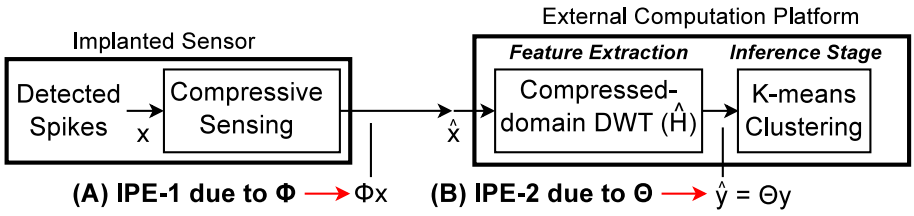


Figure 4.8: IPE-1 (due to $\mathbf{\Phi}$) and IPE-2 (due to $\mathbf{\Theta}$) contribute to IPE-T.

($\nu = 1\times$), and IPE-T at $\xi = 15\times$ when using 100 spike vectors (corresponding to $\mathbf{x}$). $\mathbf{x}$ has the dimension $N = 64$ (corresponding to one spike window) and $N = 512$ (corresponding to eight cascaded windows) in the two overlapping histograms. The IPEs are shown at $\xi = 15\times$. We can see from the figure that the significant error component is IPE-1. Further, Fig. 4.10 shows the IPE-T evaluated from the entire spike database. We see that the IPE-T in CA is only 19% even at $\xi = 24\times$ (RA has a similar error). At $\xi = 24\times$, only three compressively-sensed samples per spike are used for CA processing (compared to 64 samples for NA). In RA, we used gradient projection to reconstruct a sparse representation of spikes [12]. Also, we obtained IPE-T using 10-fold cross-validation on the total spike data. In each iteration, we learnt a new sparse dictionary $\mathbf{\Psi}$ from K-SVD using 90% of the total spike data [30].

IPE-T with respect to ν. For $\mathbf{\Theta}$ to be an orthonormal projection, it is desirable to have singular values close to 1. Fig. 4.11 shows that the measured singular values (for the entire spike database) are within the theoretical limits of Eq. (4.17). The theoretical bounds predict a much larger deviation than what we observe for the spike dataset; the singular values, however, diverge slightly more at higher values of ξ. From the figure, we also observe that there is only a small change in the singular values when we increase ν. The points of exact solution are shown as dark rectangles in the figure. The range for the singular values of $\mathbf{\Theta}$ can at best serve as an indirect proxy for the end-to-end performance of the system; IPE-T is a more direct parameter. Fig. 4.12 shows the IPE-T in the DWT spike features with respect to ν and ξ. For the exact solution ($\nu = \xi$) shown with dark

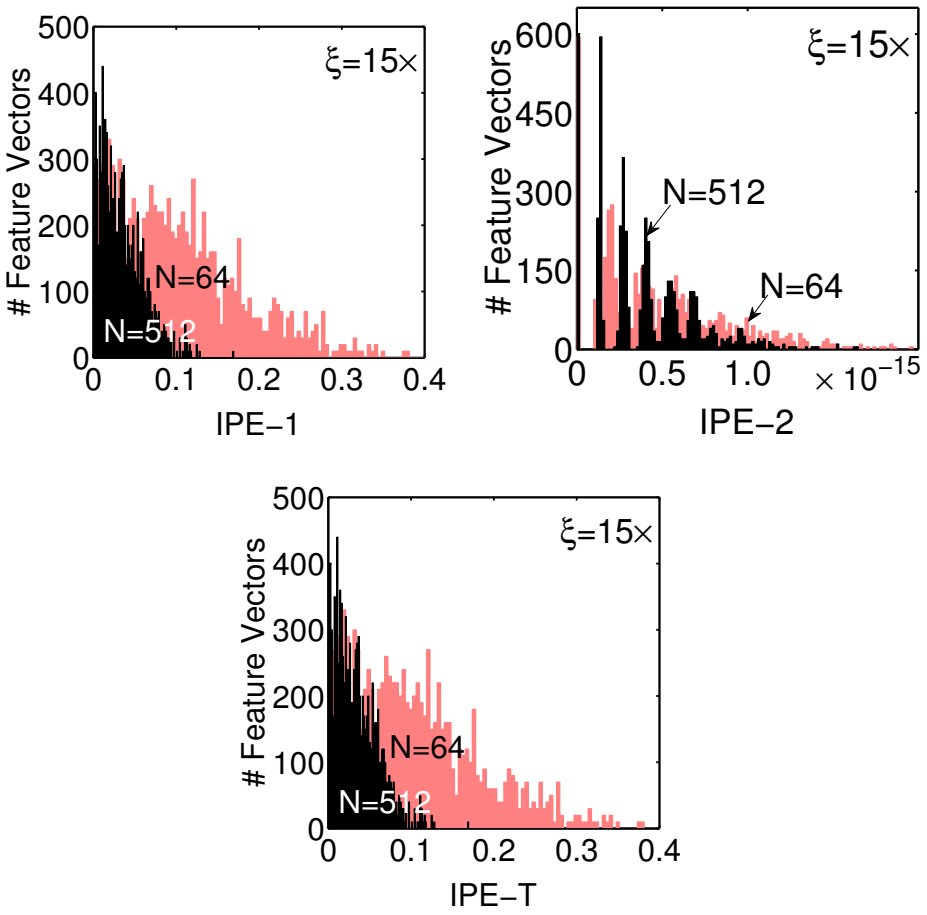


Figure 4.9: IPE-T is nearly equal to IPE-1 and IPE-2 ≈ 0 ($\nu = 1$).

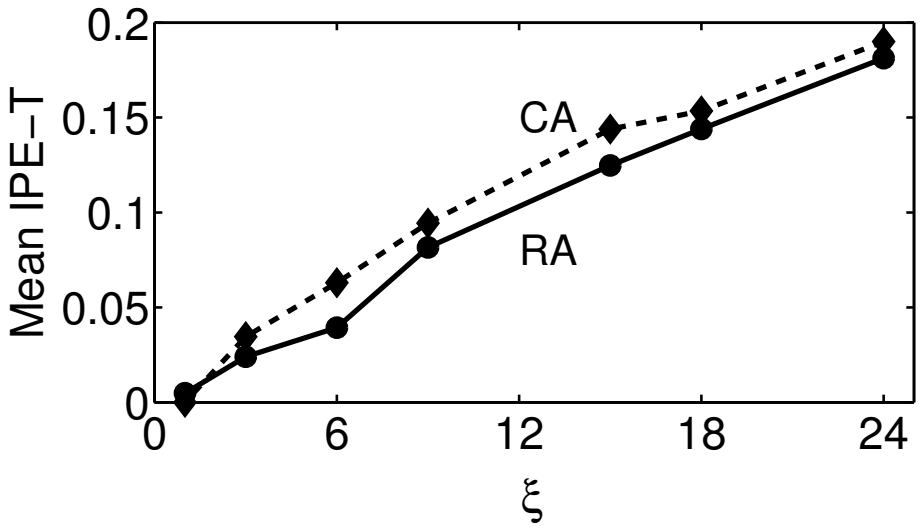


Figure 4.10: Since IPE-2 ≈ 0, mean IPE-T in CA is close to the mean IPE-T in RA.

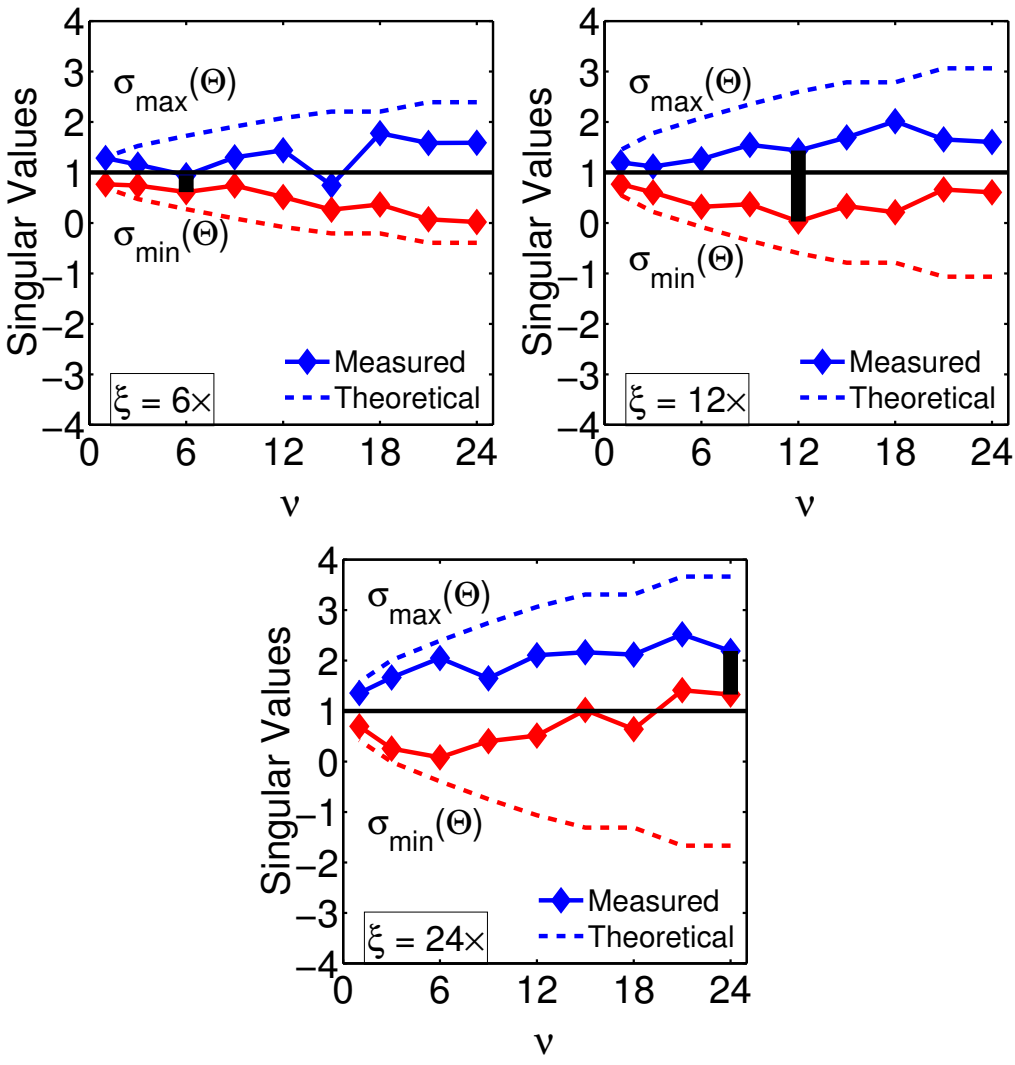


Figure 4.11: The singular values of $\mathbf{\Theta}$ are within the theoretical limits. They degrade minimally with increasing values of ν. The exact solution (*i.e.*, $\nu = \xi$) in each case is shown with a dark bar.

boxes in Fig. 4.12, IPE-T is below 19% even at $\xi = 24\times$. The figure also indicates that for $\xi < 12\times$, we can scale ν up to 15× while retaining the same level of IPE-T. This saves substantial computational energy in CA with 15× fewer operations associated with $\hat{\mathbf{H}}$.

Inference performance with ξ. With fewer compressively-sensed samples (*i.e.*, larger ξ), we expect the accuracy of SC, CV, and FR estimates to deteriorate in RA and CA. Since $\mathbf{H}$ is a square processing matrix in the neural prosthesis application, we use the exact solution for $\hat{\mathbf{H}}$ (from Sec. 4.2.1). Fig. 4.13 shows the estimation error for CA and RA.

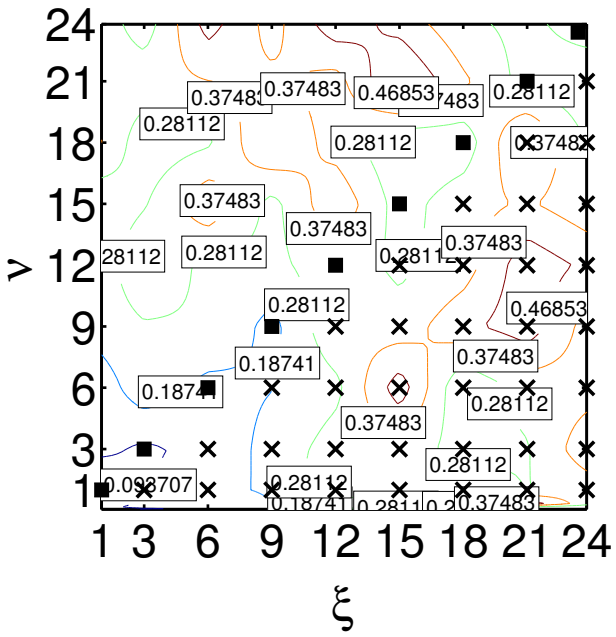


Figure 4.12: IPE-T in CA. The exact solution of Fig. 4.10 is shown with dark boxes. The cases of $\nu < \xi$, which are not favorable for low-energy operation are shown with a cross.

We use three methods for reconstructing the spikes from $\hat{\mathbf{x}}$, namely basis pursuit [12], l_1-Magic [194], and SPG-Lasso [195]. However, the results for only basis pursuit, which performs better than the other two algorithms, are shown in Fig. 4.13. Each performance metric is obtained for the exact solution (*i.e.,* $\xi = \nu$ case). The estimation errors are with respect to GT and averaged over the four records and three clusters. We observe that the performance trends for CA are similar to those of RA. Performance degrades gracefully, *e.g.*, even at $\xi = 24\times$, the estimation errors with respect to GT for SC, CV, and FR are below 8.65%, 5.06%, and 9.96% in CA and 6.66%, 4.91%, and 7.54% in RA, respectively. Thus, the exact solution enables CA to perform nearly as well as RA. Since compression does not introduce significant errors, we can significantly compress the spikes before transmitting them to the external head-stage. We next study the performance trends in CA under an approximate solution for $\hat{\mathbf{H}}$ (from Sec. 4.2.2).

Inference performance with ν: Since the approximate solution permits a smaller $\hat{\mathbf{H}}$ matrix, it enables additional savings in computational energy. However, as described in

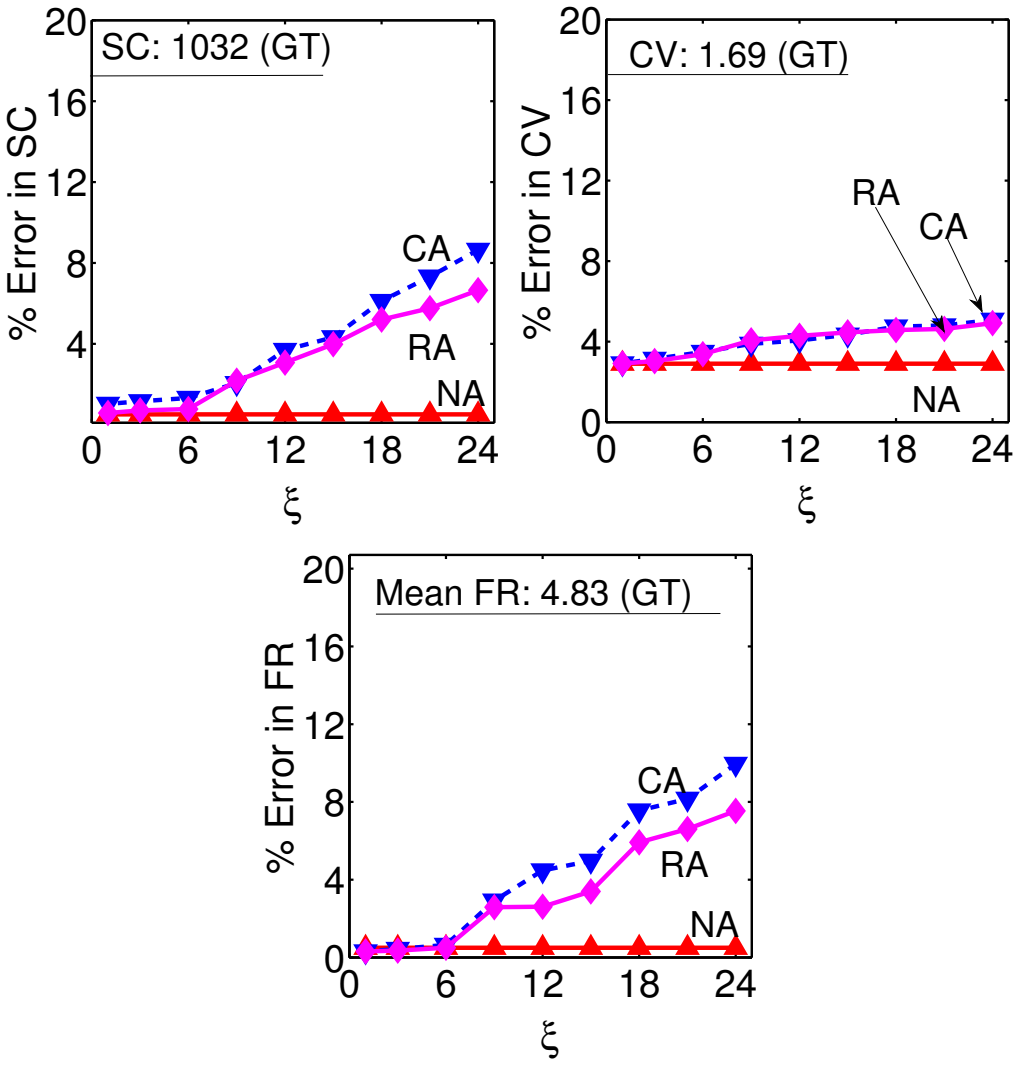


Figure 4.13: Scaling with ξ: trends for CA and RA are similar – they slowly degrade with increasing ξ, enabling us to compress the spikes significantly.

Sec. 4.2.2, due to the approximation required in $\hat{\mathbf{y}}$, this can impose a performance cost. Fig. 4.14 shows trends in performance as we simultaneously scale projection factor ν and compression factor ξ. The points corresponding to the exact solution for $\hat{\mathbf{x}}$ are shown as black squares; we can see that this performance corresponds to the IPE-T trends observed in Fig. 4.12. Fig. 4.14 also indicates that the degradation in the three parameters (*i.e.,* SC, CV, and FR) is small with increasing values of ν. For example, for both SC and FR (left and right plots), $\nu = 6\times$ (at $\xi = 9\times$) incurs very little error, yet enables >54× reduction in the size of the transformation matrix for CA processing; CV incurs somewhat higher error, but still quite small. All these error values are $\leq 3.5\%$

4.5 Case Study II: Epileptic Seizure Detection using Compressively-sensed EEG

In this section, we present a second case study that is a variant of the system presented in Chapter 3. The Nyquist-domain processing matrix $\mathbf{H}$ we consider is non-square. We thus derive the compressed-domain equivalent matrix $\hat{\mathbf{H}}$ using the solution in Sec. 4.2.3. First, we describe the modified Nyquist-domain algorithm for seizure detection, which employs patient-specific classifier training [147]. The Nyquist-domain algorithm is similar to that presented in Chap. 3 but with additional stages of downsampling incorporated to obtain a non-square processing matrix $\mathbf{H}$.

Fig. 4.15 shows the baseline algorithm for seizure detection. As before, a two-second epoch from each EEG channel is processed using eight BPFs with passbands of 0-3 Hz, 3-6 Hz, . . ., 21-24 Hz. Note that unlike in Chapter 3, however, the EEG signals are first downsampled before filtering. This process makes the BPFs rectangular, requiring the compressed-domain transformation for non-square $\mathbf{H}$ presented in Sec. 4.2.3. The spectral energy from each filter is then represented by summing the squared value of the output samples to form an FV, which is then used for classification by an SVM classifier. The

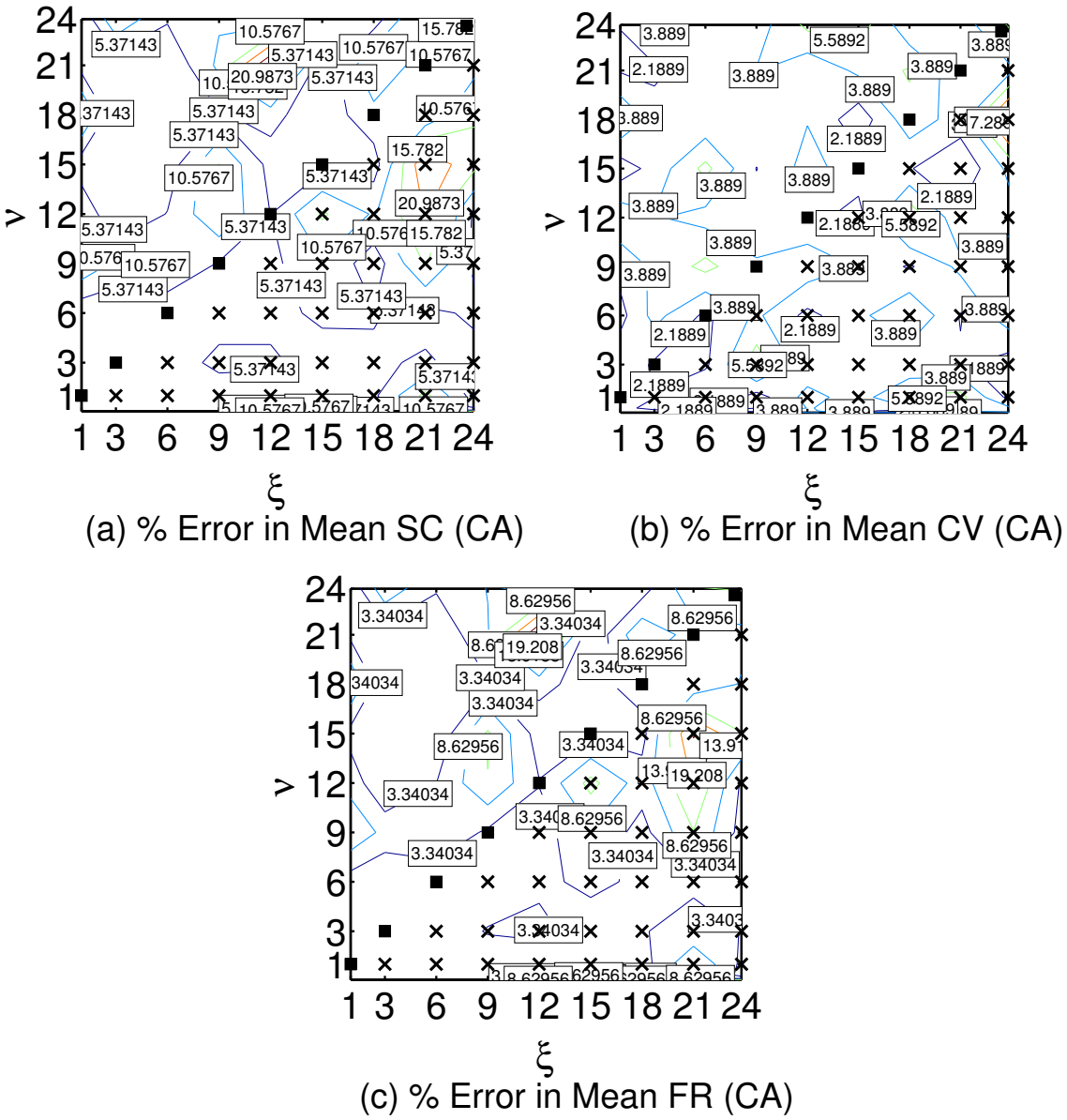


Figure 4.14: Scaling with ν: performance in CA is retained up to $\nu = 6\times$. Points of exact solution are also shown in the plots as dark squares. The cases of $\nu < \xi$, which are not favorable for low-energy operation are shown with a cross.

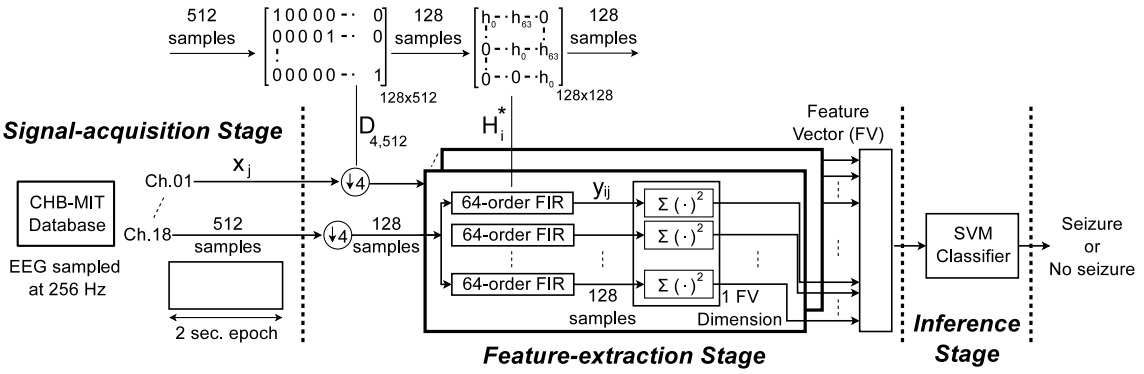


Figure 4.15: Spectral-energy feature extraction in NA can be represented as a product of the decimation matrix $\mathbf{D_{4,512}}$ and an FIR BPF $\mathbf{H}^*$.

feature-extraction process represents a special case of non-linear processing, (*i.e.*, we handle this by deriving a random projection and use the JL lemma to represent the signal energy). Further, note that since the feature-extraction process for this particular application involves spectral-energy extraction after filtering, the energy in the filtered EEG signal from each filter corresponds to one dimension of the FV. This operation can be represented as an inner-product computation: $\mathbf{f_{ij}} = \mathbf{y_{ij}^T y_{ij}}$. Relating the entire feature-extraction process with the stages in Fig. 4.1, we observe that there is an additional inner-product computation involved before classification (as also shown in Fig. 3.3). Thus, for this case study, the IPE metric defined in Sec. 4.3 directly represents the error in the signal features.

The baseline detector in NA is again validated on 558 hours of EEG data from 21 patients (corresponding to 148 seizures) in the CHB-MIT database [165]. For every patient, up to 18 channels of continuous EEG is processed using eight BPFs, leading to an FV dimensionality of 144.

4.5.1 Formulating Feature Extraction as a Matrix Operation

To enable a transformation to the compressed domain, we focus on computations in the feature-extraction stage of Fig. 4.15. Recall that to enable efficient processing with a low-order FIR filter, we down-sample the EEG signals before filtering. Since the BPFs in the

filter bank have a maximum cut-off frequency of 24 Hz and EEG signals in the CHB-MIT database are sampled at 256 Hz, we down-sample the data from each channel by a factor of 4. For each data channel, one EEG epoch (corresponding to 512 Nyquist samples) thus results in 128 decimated samples. These samples are then processed with eight BPFs of order 64. To represent the BPF computations as a matrix operation, we generalize the formulation given in Eq. (4.19) [Sec. 4.4.2 handled decimation of an N-sample signal by $2\times$]. We define a new decimation matrix $\mathbf{D_{4,512}}$, which acts upon a 512-sample EEG epoch to give 128 decimated samples. Suppose we represent one EEG epoch from the j^{th} channel as $\mathbf{x_j}$. Then $\mathbf{D_{4,512}}$, which has the structure shown in Fig. 4.15, acts upon $\mathbf{x_j}$ to give us 128 samples. Further, suppose we represent each 64-order BPF before energy accumulation as a convolution matrix $\mathbf{H_i^*}$, $0 \leq i \leq 7$, of dimensionality 128×128 – observe that $\mathbf{H_i^*}$ has a structure shown in Fig. 4.15, where each row is a shifted version of the previous row. We can then represent the decimation + filtering operation in the feature-extraction stage as the following cascaded operation:

$$\mathbf{y_{ij}} = \mathbf{H_i^*}\ \mathbf{D_{4,512}}\ \mathbf{x_j} \tag{4.20}$$

where $\mathbf{y_{ij}}$ is the filtered EEG data derived from the i^{th} filter acting upon the j^{th} EEG channel. The Nyquist-domain processing matrix for each BPF can thus be defined as $\mathbf{H_i} = \mathbf{H_i^* D_{4,512}}$. This matrix is rectangular and has a dimensionality of 128×512.

Next, we investigate the variation of IPE with ξ as well as its correlation with the performance of the end-to-end algorithm.

4.5.2 Experimental Results

In this section, we study the error in the FVs (represented by IPE) and the performance of the end-to-end system. We will see that the performance does not correlate directly with the IPE because the information content of the features is what controls the performance of the system (see Sec. 3.6.1). This behavior, which is unlike the previous case study, is due to the presence of the spectral-energy operation in the feature-extraction stage. We thus also

study the variation in mutual information with respect to ξ in CA and compare it with that in RA.

In CA, we derive compressed-domain processing matrices $\hat{\mathbf{H}}_\mathbf{i}$ from the corresponding rectangular NA matrices $\mathbf{H_i}$ using the solution in Sec. 4.2.3. Note that $\hat{\mathbf{H}}_\mathbf{i}$ has $K \times M$ [or $N(1/\nu + 1/\xi)$] entries. As in NA, we then obtain the processed signal from each filter as: $\hat{\mathbf{y}}_\mathbf{ij} = \hat{\mathbf{H}}_\mathbf{i}\ \mathbf{\Phi}\ \mathbf{x_j}$, where the processing matrix $\hat{\mathbf{H}}_\mathbf{i}$ acts directly on the compressively-sensed signal $\mathbf{\Phi x_j}$. We then derive a CA-estimate of the spectral energy as: $\hat{\mathbf{f}}_\mathbf{ij} = \mathbf{y}_\mathbf{ij}^\mathbf{T}\mathbf{y_{ij}}$.

Note that varying ν is not very informative in this case, since unlike square matrices, there is no possibility of an exact solution when $\xi = \nu$. Thus, we study the IPE characteristics and algorithmic performance by fixing ν at $1\times$ and only vary ξ. We present the simulation results below.

IPE with respect to ξ. The error in the FVs (IPE) is defined as: IPE $= \|\hat{\mathbf{f}}_\mathbf{ij} - \mathbf{f_{ij}}\|/\mathbf{f_{ij}}$. We expect the error to increase with increasing compression ($\xi > 1\times$). For our experiments, we keep $\nu = 1\times$ and scale ξ. The computational savings in CA thus increase with ξ [$\hat{\mathbf{H}}_\mathbf{i}$ has $N(1/\nu + 1/\xi)$ entries]. Fig. 4.16 shows the trend in IPE. The plot also shows the variation across all patients in the database. We observe that the IPE is less than 19.5% upto $\xi = 51\times$, at which point we only transmit and process 10 EEG samples per epoch. The figure also shows the IPE in RA, where we reconstruct each epoch using gradient projection [12]. For each patient, we learn a new sparse dictionary $\mathbf{\Psi}$ from K-SVD using 10-fold cross-validation [30]. We observe that IPE in CA is close to the IPE in RA, thus validating the solution for $\hat{\mathbf{H}}_\mathbf{i}$ in Sec. 4.2.3. We next proceed to study the impact of the error in the FVs on overall system accuracy.

Inference performance with respect to ξ. To evaluate the performance of the compressed-domain detector, we derive FVs from the CHB-MIT database. We use these FVs to train and test the SVM classifier in a patient-specific manner. We employ a leave-one-out cross-validation scheme for measuring the performance of the detector. Accordingly, for each patient, we use all seizure records except one, along with all non-

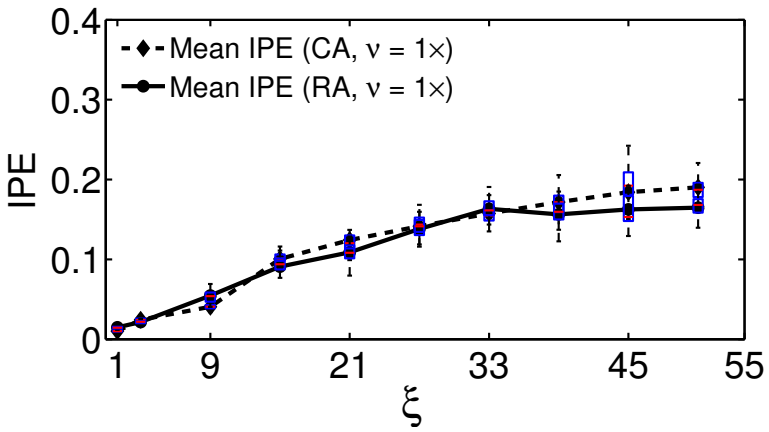


Figure 4.16: Variation in IPE-T across 23 patients in the CHB-MIT database. Mean IPE-T in CA is close to the mean IPE-T in RA.

seizure records for training the classifier. We then apply the resulting SVM model only to the record that was left out in the training phase. We repeat this validation process for each record of a patient. Fig. 4.17 shows the scaling in performance (over 21 patients) in CA. The performance of the compressed-domain detector is very close to the Nyquist case at $\xi = 1\times$. For CA, at a compression of $1\times$, the sensitivity is 95.53%, latency is 4.59 sec., and the number of false alarms is 0.1538/hr. These performance numbers begin to degrade with the compression factor. The corresponding numbers at $\xi = 21\times$ are 94.43%, 4.70 sec., and 0.1543/hr., respectively. Thus, at higher values of ξ (which give corresponding energy savings), the degradation in sensitivity is 1.1% at $\xi = 21\times$, beyond which it begins to drop more significantly. The scaling in the number of false alarms per hour and the latency also follows a similar trend. The mean latency of detection increases by 2.41% while the specificity of the algorithm degrades by only 0.33% at $\xi = 21\times$. Fig. 4.18 shows the performance of RA. We observe that it is close to the performance of CA. These trends, however, do not correlate with the IPE in Fig. 4.16. For example, at $\xi > 33\times$, IPE in RA is constant around 16% but the difference in performance for values of $\xi > 33\times$ is significant. Next, we study the information content in the FVs, which we showed to be a metric that directly indicates the end-to-end performance of the detector in Chapter 3.

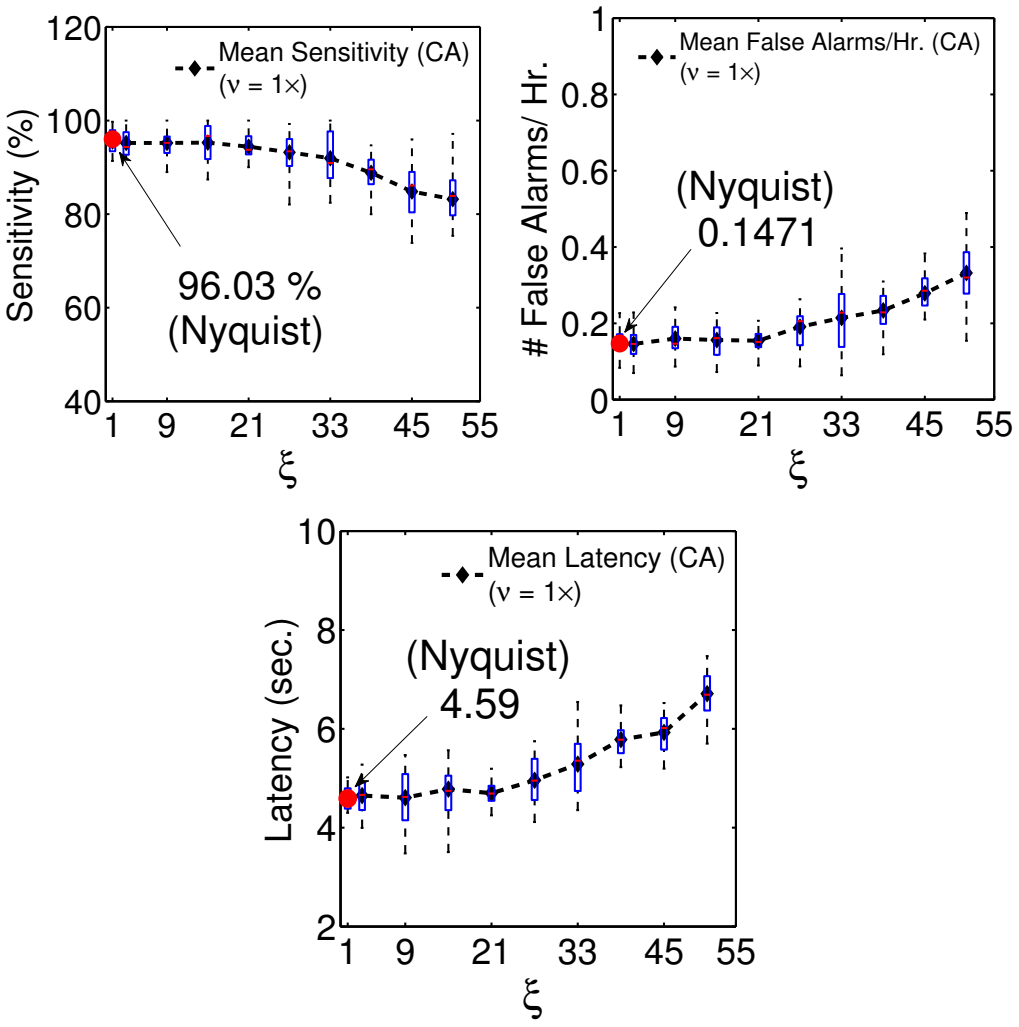


Figure 4.17: Performance of the seizure detection algorithm in CA by transforming a rectangular processing matrix $\mathbf{H}$: Performance is maintained upto $\xi = 24\times$.

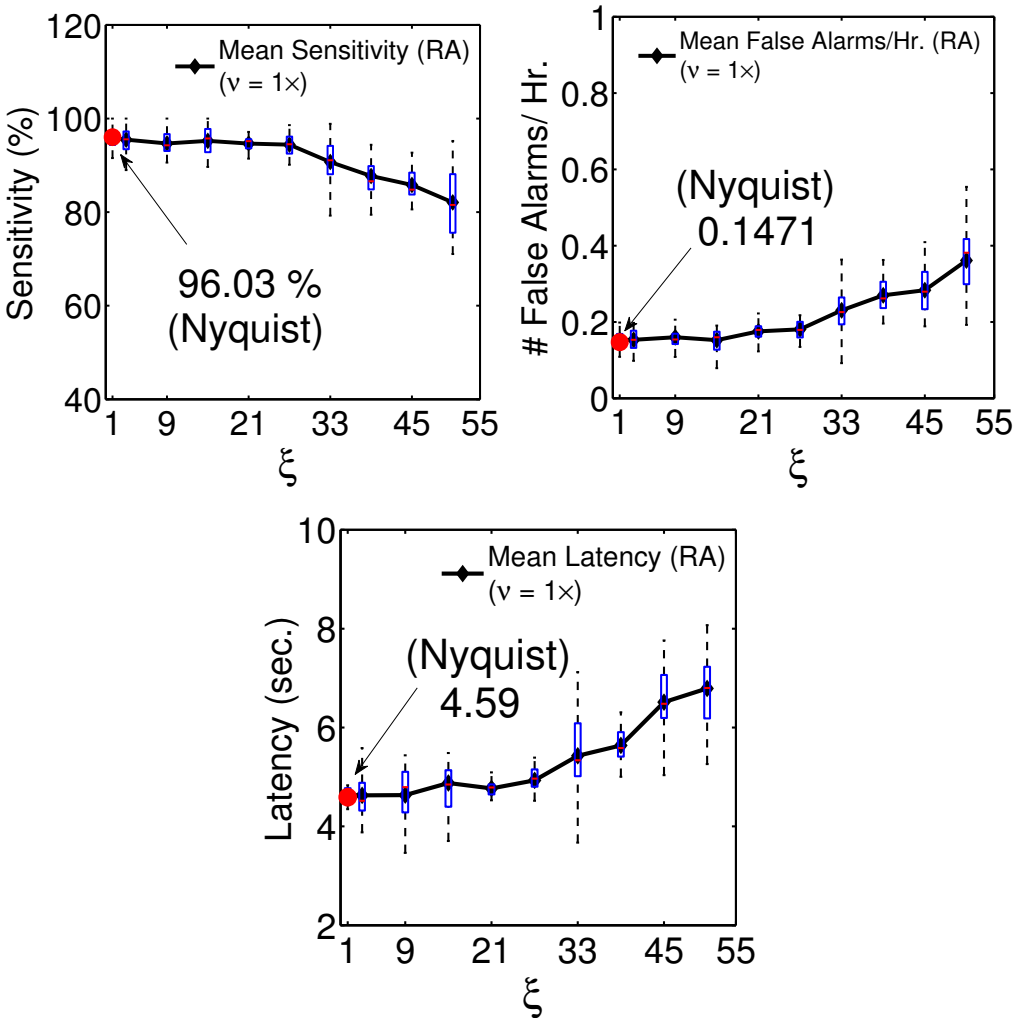


Figure 4.18: Performance of the seizure detection algorithm in RA by transforming a rectangular processing matrix $\mathbf{H}$: Performance is maintained upto $\xi = 24\times$.

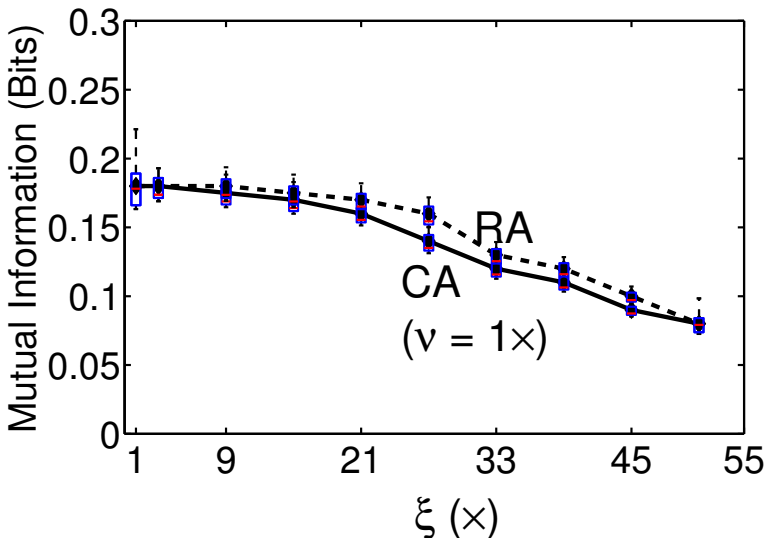


Figure 4.19: Information in CA and RA by transforming a rectangular processing matrix **H**: Mutual information follows the performance trends.

Mutual information with respect to ξ. Fig. 4.19 shows the variation in mutual information of the FVs in RA and CA vs. ξ. We see that the inference performance corresponds to the FV mutual information.

Based on the results for the multi-rate system presented in this section, very limited degradation is seen up to large compression factors. This supports the CA system model.

4.6 Chapter Summary

Sparsity of signals provides an opportunity to efficiently represent sensor data. Compressive sensing is one technique that exploits signal sparsity in a secondary basis to achieve very low-energy compression at the cost of high complexity in signal reconstruction. The energy for reconstruction can present a significant barrier to signal analysis, which is becoming increasingly important in emerging sensor applications. In this chapter, we improved upon the approach presented in Chapter 3 to derive computations that can be performed directly on compressively-sensed signals. Our methodology employed an auxiliary

matrix $\mathbf{\Theta}$ instead of $\mathbf{\Phi}$ used in the compressed-domain equation [Eq. (3.8)] in Chapter 3. Through analytical validations, we observed that this approach can significantly improve upon the error values in feature estimates obtained in Chapter 3. Thus, our approach can potentially permit higher amounts of data compression for the same amount of degradation in detector performance. We also showed that the new error values that we obtain are very close to the expected lower limit. We validated our approach with two case studies, namely spike sorting for neural prosthesis and EEG classification for seizure detection. For the neural-prosthesis application, our experimental results suggest that we can achieve computational energy savings of up to 54× while restricting detection errors to under 3.5%. Using our approach, the reductions in communication energy can also be significant. For instance, in the seizure-detection application, the detection error was under 2.41% when we used $\sim$ 21× fewer transmitted EEG samples. Our approach thus provides an approach for signal-processing systems that addresses system-resource constraints, such as energy and communication bandwidth, through efficient signal representation. This is in contrast to efforts that focus on efficient architectures and algorithms alone. An interesting offshoot of the auxiliary-matrix based approach is that it permits flexibility in the compressed-domain equations allowing us two kinds of solutions: (1) one that provides the highest accuracy, and (2) another that saves computational energy. This raises a new energy-accuracy trade-off in CA-based systems. In the next chapter, we explore this trade-off along with other practical implications of CA through a prototype IC implementation.

Chapter 5

IC Implementation: Seizure Detection in the Compressed Domain

Compressed-domain analysis can potentially help overcome the limitations of compressive sensing. In this chapter, we explore the implications of CA on a practical system. For a quantitative study, we use the seizure-detection system of Chapter 3, where we employ square BPF matrices followed by spectral-energy extraction and SVM classification. We provide measurements from a prototype IC that directly analyzes compressively-sensed EEG for embedded signal analysis. This has two advantages. First, with compressive sensing, reconstruction costs are typically severe, precluding embedded analysis; directly analyzing the compressed signals circumvents reconstruction costs, enabling embedded analysis within applications. Second, the use of compressed signals reduces the computational energy of signal analysis due to the reduced number of signal samples. We rely on the auxiliary matrix based algorithmic formulation for square matrices (from Chapter 4) to transform computations to CA. Thus, we describe a hardware architecture that exploits the two strong power-management knobs presented in the previous chapter. These knobs allow application-level performance to scale with computational energy. The two knobs are parameterized as follows: (1) ξ, which quantifies the amount of data compression, and

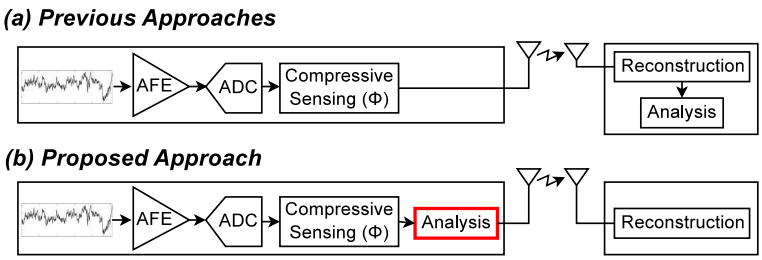


Figure 5.1: Comparison of our IC implementation with previous work: we propose to directly analyze compressively-sensed data.

(2) ν, which determines the approximation error within the proposed compressed-domain processing algorithm.

5.1 Introduction

Compressive sensing has been used to overcome communication-energy and -bandwidth constraints in low-power sensors. Owing to the high cost of reconstructing the signal, however, as shown in Fig. 5.1, previous implementations that exploit compressive sensing have resulted in practical implementations that are limited to primarily serving as nodes for transmitting raw data to a base station [10, 26]. As mentioned previously, in advanced medical devices, however, there is a need to also analyze signals on the node [91]. Such devices typically work by extracting signal features based on physiological biomarkers and then feeding these features to high-performance classifiers to detect targeted physiological states [91, 93]. However, since compressive sensing involves multiplication by random projection matrix $\mathbf{\Phi}$, the biomarkers get obscured and thus present a challenge for signal analysis. The CA methodology we presented in Chapter 4 allowed us to perform detection directly in the compressed domain. The IC implementation we describe next exploits this methodology using embedded power-management knobs in an EEG-based seizure-detector (that employs a *square* BPF matrix) enabled by operation in the compressed domain.

The rest of this chapter is organized as follows. In Sec. 5.2, we present simulation results showing the accuracy of seizure detection using the exact and approximate solutions. We also provide information analysis of the compressed-domain FVs. In Sec. 5.3, we describe the low-power compressed-domain processor for seizure detection. Within this section, we discuss circuit- and architecture-level opportunities for power management. We present measurement results from an IC prototype in Sec. 5.4. Finally, we conclude in Sec. 5.5.

5.2 Performance of the Compressed-domain Seizure Detector

We apply the NA to CA transformation methodology of Chapter 4 to the seizure detection algorithm presented in Chapter 3. Recall that to enable a transformation to the compressed domain, the i^{th} BPF for the j^{th} EEG channel can be formulated as matrix multiplication, namely of an input signal $\mathbf{x_j}$ by a matrix $\mathbf{H_i}$ to compute the filtered signal $\mathbf{y_{ij}}$. This formulation leads to the compressed-domain system shown in Fig. 3.5(b), which is based on the CD-BPFs. Since $\mathbf{H_i}$ is square, in this system, we can derive the corresponding matrix $\mathbf{\hat{H}_i}$ using (1) an exact solution, and (2) an approximate solution, as described in the previous chapter. Figs. 5.2 and 5.3 show the simulated performance of the two approaches, respectively. Note that these results are different from Fig. 4.17 since, in this case, we use an $\mathbf{\hat{H}}$ derived from a square matrix $\mathbf{H}$ in the Nyquist domain. In Fig. 4.17, we used a rectangular matrix to incorporate the additional downsampling process. Fig. 5.2 shows that for the exact solution, performance very close to the Nyquist-domain seizure detector is retained up to large values of compression factor ξ. For a given amount of data compression (*i.e,* at fixed ξ), Fig. 5.3 shows that using an approximate solution, the performance begins to degrade gradually with increasing values of ν. Further, these performance trends correlate

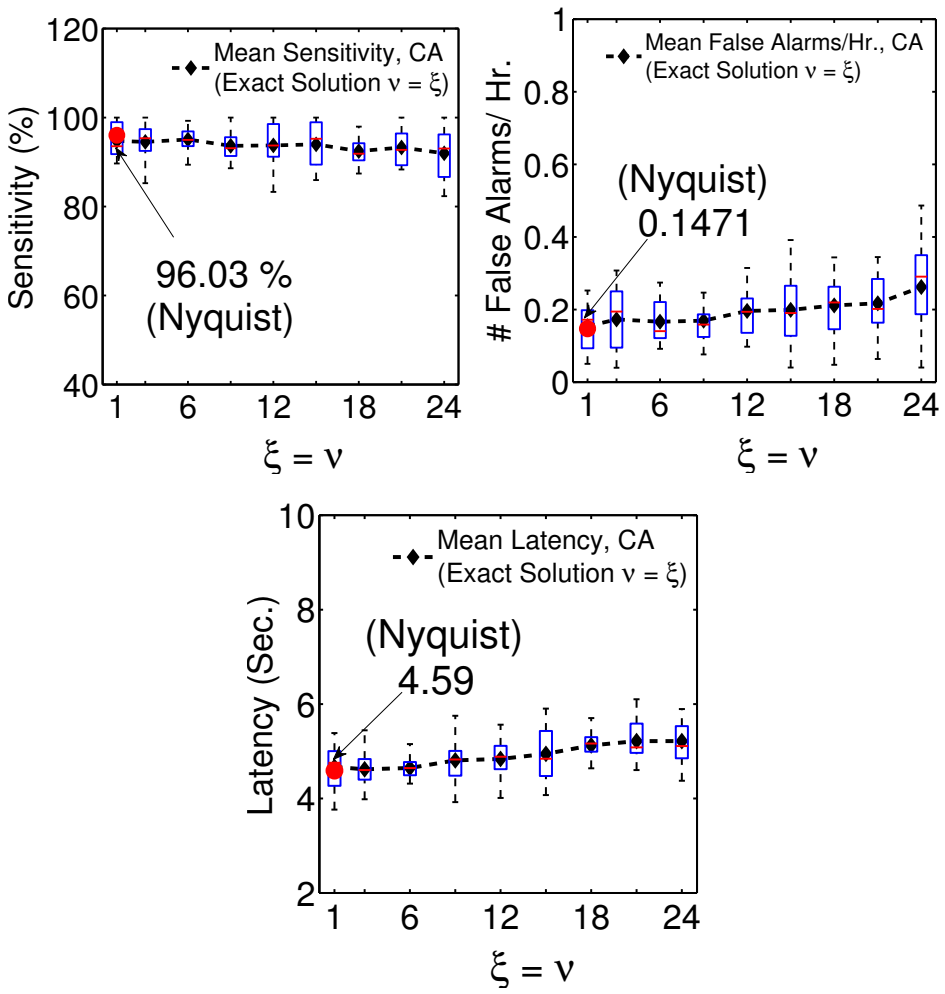
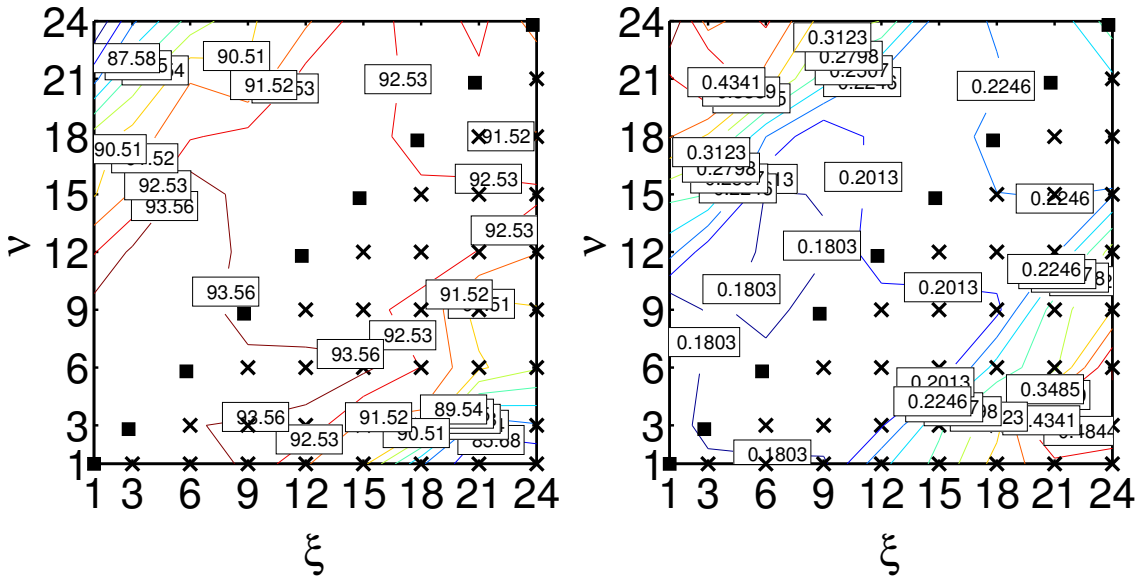


Figure 5.2: Performance in CA by transforming a square processing matrix **H**: Performance of the compressed-domain seizure detection algorithm using the exact solution (shown over 558 Hrs. of EEG data from 21 patients) is maintained up to large ξ.

well with the information content [13, 14], based on the mutual information of the FVs, which is shown for the exact and approximate solutions in Fig. 5.4.

Based on the performance analysis presented in this section, we can conclude that ξ and ν provide us with two knobs to control the end-to-end performance of the compressed-domain detector. Next, we study the impact of these two knobs on the size of the compressed-domain processing matrices $\hat{\mathbf{H}}_\mathbf{i}$ and thus on the processor implementation and energy.

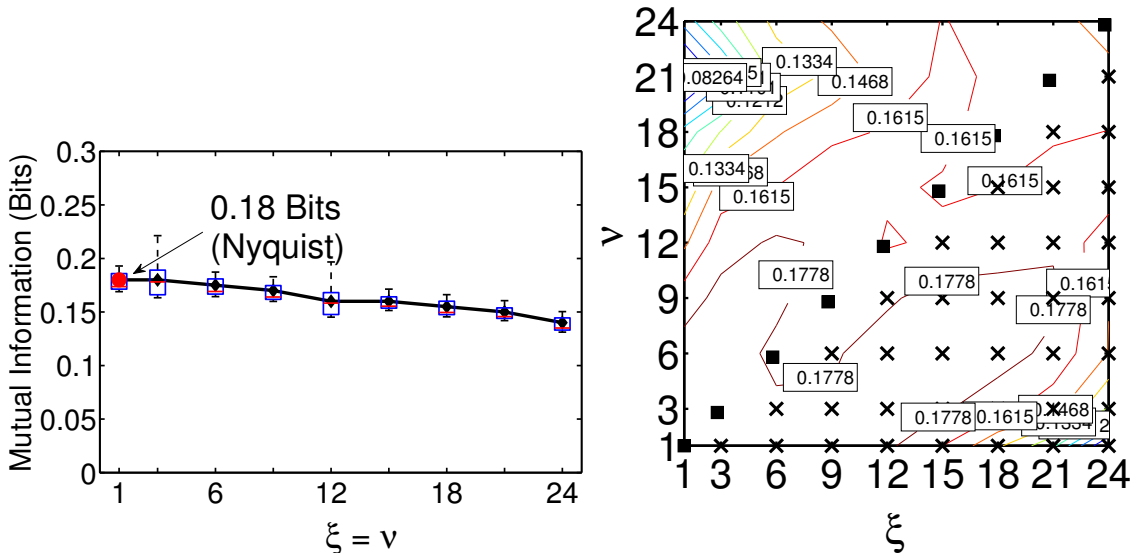


(a) Sensitivity (%)

(b) False alarms per hour

(c) Latency (seconds)

Figure 5.3: Performance in CA by transforming a square processing matrix **H**: When we use the approximation solution, the performance begins to degrade gradually due to the JL-approximation at higher values of ν. The exact solution of Fig. 5.2 is shown with dark boxes. The cases of $\nu < \xi$, which are not favorable for low-energy operation, are shown with a cross.



(a) Exact solution (b) Approx. solution: Mutual info. in bits

Figure 5.4: Information in CA by transforming a square processing matrix **H**: Mutual information in (a) the exact solution, and (b) the approximate solution follows the performance trends shown in Figs. 5.2 and 5.3, respectively.

5.2.1 Processor Architecture with Power Management

We exploit the scalability of ξ and ν as knobs for system power management. An important consequence of the algorithmic construction proposed is that the CD-BPF matrices $\hat{\mathbf{H}}_\mathbf{i}$ (which are of dimensionality $\frac{N}{\xi} \times \frac{N}{\xi}$ for the exact solution and $\frac{N}{\nu} \times \frac{N}{\xi}$ for the approximate solution) do not retain the regularity of $\mathbf{H}_\mathbf{i}$. Even though $\mathbf{H}_\mathbf{i}$ are of dimensionality $N \times N$, as shown in Fig. 5.5, the rows of $\mathbf{H}_\mathbf{i}$ are simply selected to implement convolution, and thus are shifted versions of the same FIR-filter impulse response. As a result, very few unique filter coefficients are required, and many of the coefficients are zero, as determined by the FIR-filter order k. However, in deriving $\hat{\mathbf{H}}_\mathbf{i}$, the shifted impulse responses and zero entries are disrupted. As shown in Fig. 5.5, the number of multiplications required thus no longer depends on the filter order, but rather (1) *quadratically* on the compression factor ξ for the exact solution, and (2) *linearly* on both ξ and ν for the approximate solution. This scaling can potentially reduce the number of multiplications required.

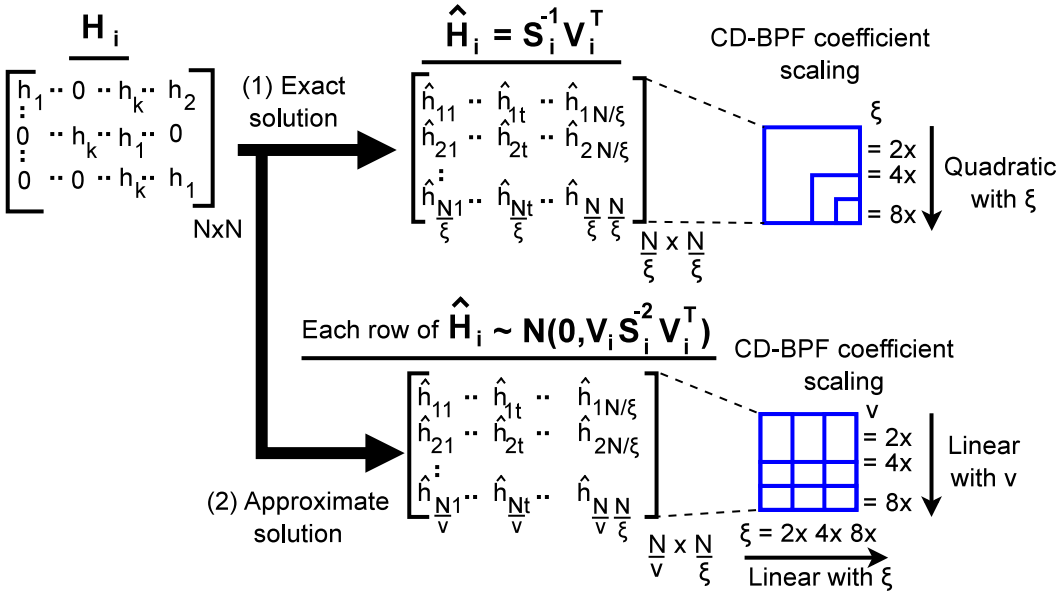


Figure 5.5: CD-BPF matrices $\hat{\mathbf{H}}_i$, derived using $\mathbf{H}_i$ and $\mathbf{\Phi}$, disrupt the regularity and zeros in $\mathbf{H}_i$. The complexity of the CD-BPFs thus scales (a) quadratically with ξ for the exact solution, and (b) linearly with ξ and ν for the approximate solution.

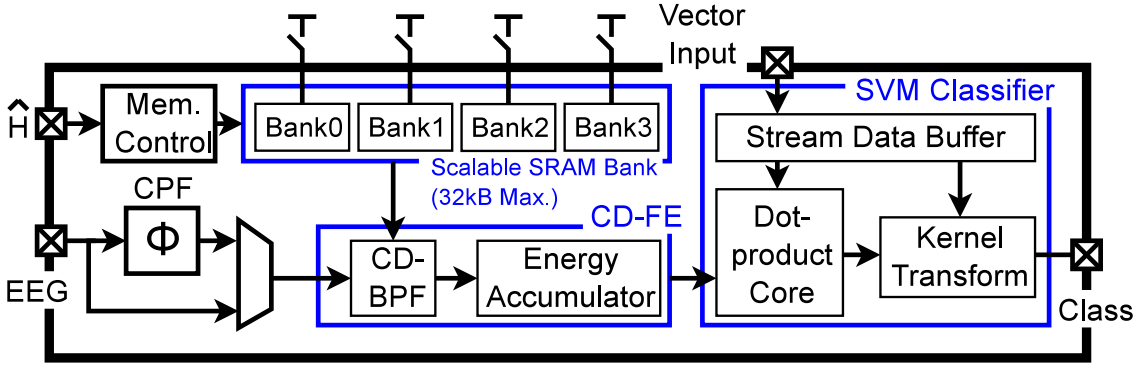


Figure 5.6: Architecture block diagram of energy-scalable, compressed-domain seizure detector.

To exploit this scaling, we propose the energy-scalable processor architecture whose block diagram is shown in Fig. 5.6. The processor consists of two computational stages: compressed-domain feature extraction followed by SVM classification. The compressed-domain feature extractor (CD-FE) includes a CD-BPF and energy-accumulator block. The coefficients for the CD-BPF are pulled from a scalable SRAM bank. Due to the disruption in regularity, the $\hat{\mathbf{H}}_\mathbf{i}$ matrices need a larger number of distinct coefficients to be stored, potentially increasing the memory requirements. Scalability in the SRAM bank is thus an important aspect of power management. We achieve this through the use of multiple subarrays, which enable fine-grained power-gating as well as reduced bit-line and word-line access energy. The total bank size in our implementation is 32kB, which is partitioned into four subarrays. An SVM classifier (which comprises an inner-product core followed by a kernel transform) is also integrated to perform real-time seizure detection using the derived FVs. Compressively-sensed EEG signals are input directly to the processor for seizure detection. However, for the case of Nyquist inputs, a compressive-projection frontend (CPF) is also included to explicitly multiply inputs by a random projection matrix $\mathbf{\Phi}$; thus the energy savings derived from a reduced number of samples can be exploited even if the original input signal is not compressively sensed.

5.3 Low-energy Compressed-domain Processor

In this section, we describe the circuits used in the compressed-domain processor. We also present an analysis of SRAM energy, which will help us understand the impact of energy scalability in the processor with respect to ξ and ν.

5.3.1 Circuits

Fig. 5.7 shows the circuits used in the compressed-domain processor. The CPF is selectable for front-end signal compression. It uses a 16b linear feedback shift register (LFSR) to im-

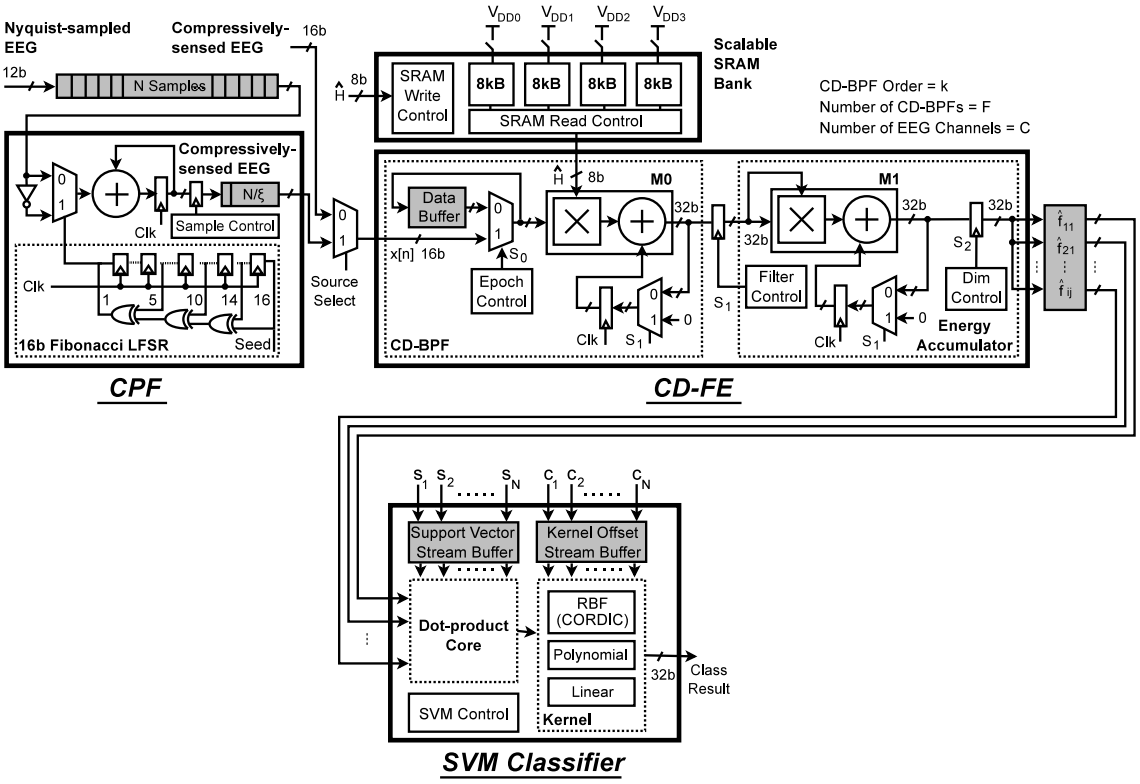


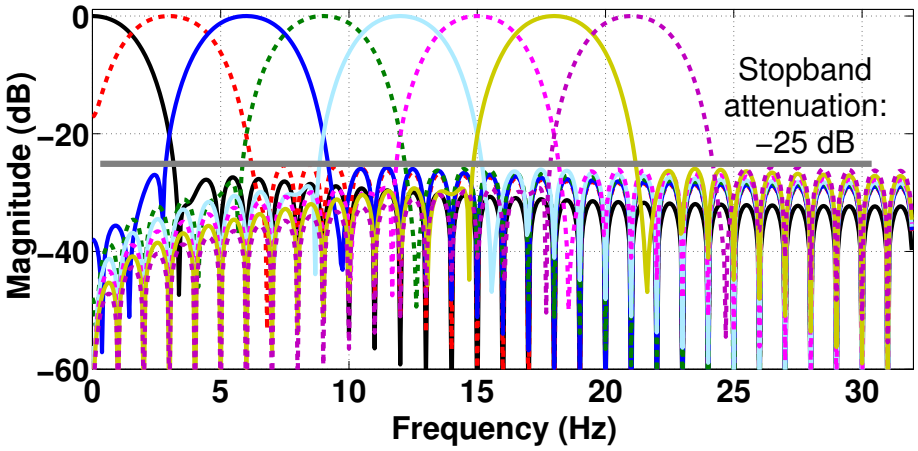
Figure 5.7: Circuits used in the compressed-domain processor for seizure detection.

plement multiplication with a random projection matrix $\mathbf{\Phi}$, as shown. Since the processor operates on an EEG epoch of 2 seconds, FVs are derived at the rate of 0.5 Hz. At this low rate, the CD-FE can compute each feature dimension sequentially and store the intermediate results in a data buffer. The CD-FE can be configured to compute up to eight spectral features ($i = 0, \ldots, 7$) for each EEG channel (j) over as many as 18 channels, yielding a maximum FV dimensionality of 144. Within the CD-FE, the control pulse S_0 initiates CD-BPF computations. A multiply-accumulate (MAC) unit (M0) is used to perform the matrix multiplications required for compressed-domain band-pass filtering using $\hat{\mathbf{H}}_\mathbf{i}$. Each filtered EEG epoch is then registered by control pulse S_1. Energy accumulation over the output vector is then performed by a second MAC unit (M1). After the feature-extraction process [which requires $(N/\xi)(N/\xi + 1)$ MAC operations], each FV dimension ($\hat{f}_{ij}$) is stored in an intermediate FV buffer based on control pulse S_2.

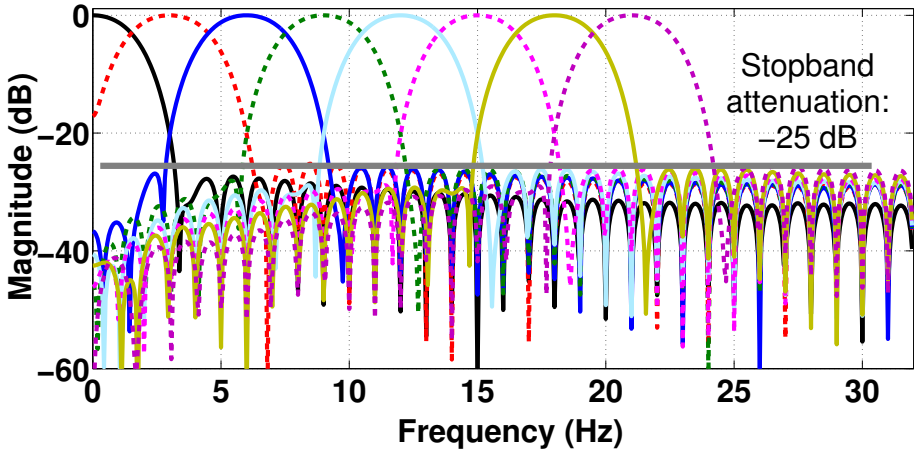
Fig. 5.8 shows the frequency response of the Nyquist-domain BPFs at different coefficient resolutions. We observe that even at a resolution of 8 bits, the stopband attenuation is below -25 dB. To investigate further, all EEG epochs in the CHB-MIT database that contain seizures are processed using floating-point BPF coefficients. These filtered epochs constitute the baseline signals. The same epochs are then processed using BPF coefficients at resolutions of 4-16 bits. The difference between these filtered signals and the baseline represents the noise. The SNR in the filtered EEG epochs is thus computed using the noise and the baseline EEG epochs. Fig. 5.9 shows the SNR of the filtered EEG epochs processed by the BPFs quantized at different resolutions. The figure shows that the SNR remains above 40 dB even at 6 bits of coefficient precision. In our implementation, we choose to represent filter coefficients using 8 bits of precision. To support CD-FE computations, the processor thus requires a maximum of 32kB accesses per second from the memory bank.

Fig. 5.10 shows that the SRAM energy per access (E_{acc}^{sram}) is reduced by choosing smaller-sized subarrays [196]. Since the ξ and ν knobs scale the memory required, designing a single 32 kB array would be sub-optimal for many of the parameter points. Instead, we use four subarrays (each of size 8 kB) to balance savings in energy per access with the overhead of further partitioning. With this partitioning, leakage energy saving can be achieved by independently power-gating each sub-array (from off-chip).

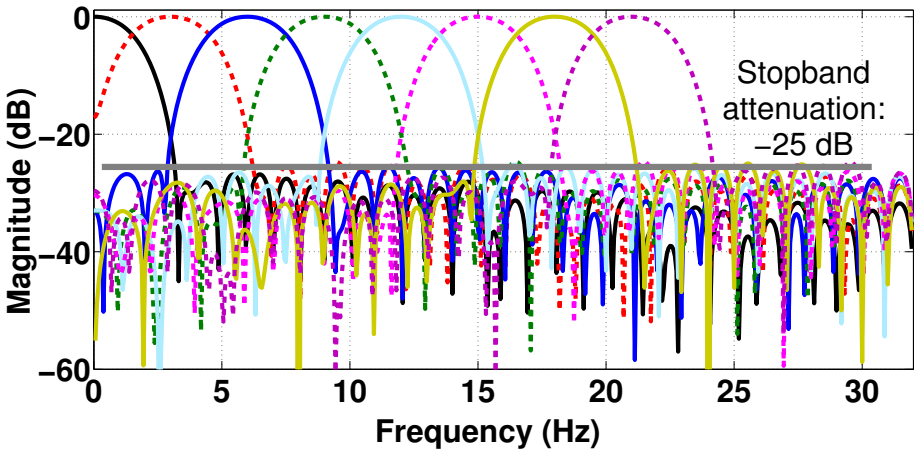
After the CD-FE computations, each FV is processed by the SVM block within the epoch duration of two seconds. The SVM can apply linear, polynomial, or RBF kernel transformations (*via* an embedded CORDIC engine). The SVs are derived from offline training of the classifier and are provided through a dedicated interface. The classification result is encoded in the MSB of the SVM output (MSB = 1 for seizure detected, MSB = 0 for no seizure detected).



(a) Floating point



(b) 12 bits



(c) 8 bits

Figure 5.8: Frequency response of the Nyquist-domain BPF matrices at different coefficient resolutions: (a) floating point, (b) 12 bits, and (c) 8 bits. The stopband attenuation is below -25 dB even at 8 bits of resolution.

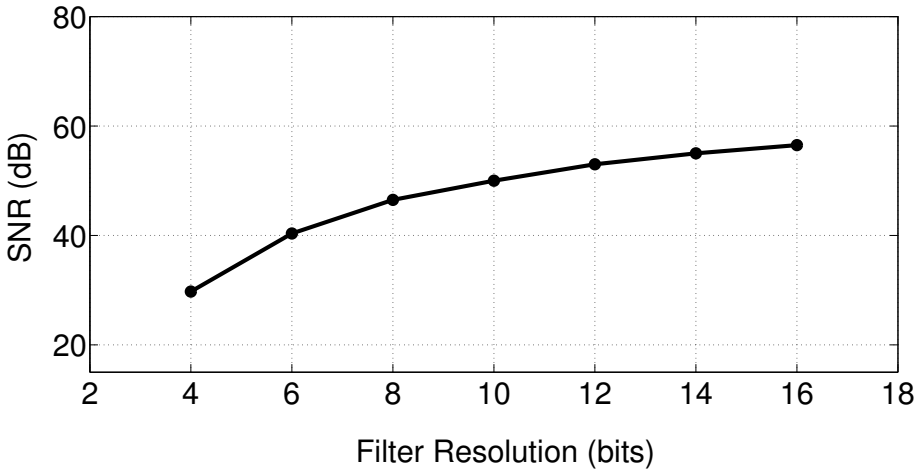


Figure 5.9: SNR of filtered seizure epochs at different coefficient resolutions.

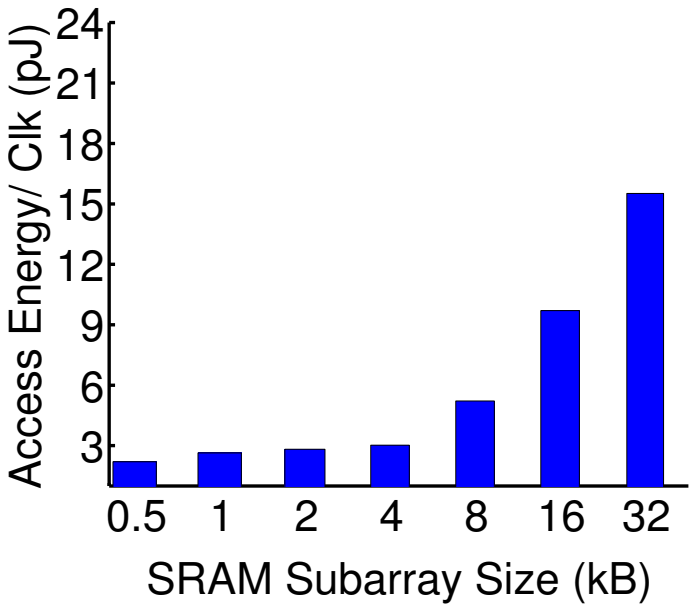


Figure 5.10: SRAM access energy (from NanoSim [197] at 0.7 V) is lower for smaller sized arrays.

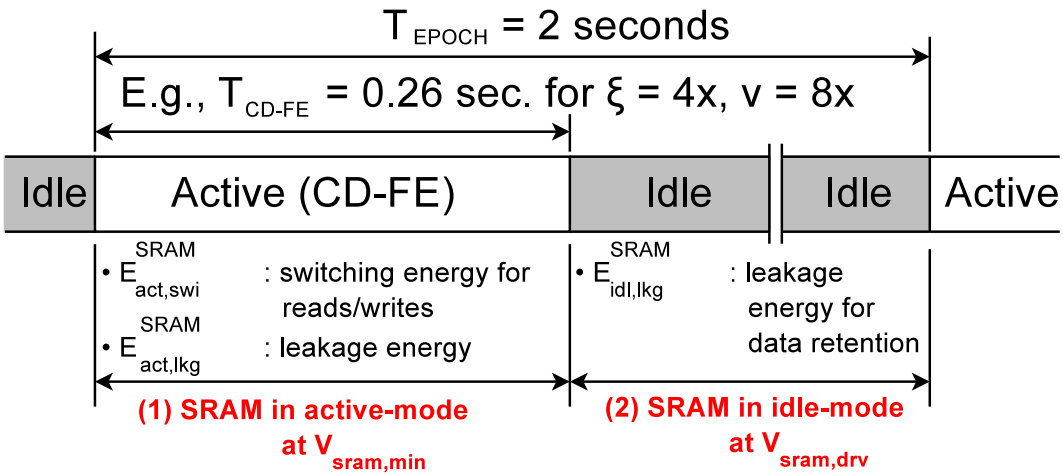


Figure 5.11: Summary of energy components contributing to total SRAM energy (the $\xi = 4\times$, $\nu = 8\times$ case is shown for illustration).

5.3.2 SRAM Energy Analysis

The CD-FE energy comprises the logic and SRAM energy subcomponents. The SRAM consumes a substantial portion of the total CD-FE energy. Its optimization to exploit scalability with respect to ξ and ν is thus a key factor. The detector processes an EEG epoch every $T_{EPOCH} = 2$ sec. However, the optimal operating frequency (and supply voltage $V_{dd,opt}$) for the CD-FE logic is determined by minimizing the overall CD-FE energy, while ensuring a minimum throughput that allows the active CD-FE computations to be completed in T_{CD-FE} (< 2) seconds for each value of ξ and ν. For the remainder of the epoch (*i.e.*, $T_{EPOCH} - T_{CD-FE}$), the logic and SRAMs can be placed in low-energy idle modes.

Fig. 5.11 summarizes the SRAM operating modes and energies [196]. The total SRAM energy is the sum of the active-mode (E_{act}^{SRAM}) and idle-mode (E_{idl}^{SRAM}) energies for each subarray that is enabled (numbering N_{sub}); under the assumption that the SRAMs cannot by fully power-gated in order to ensure data retention, E_{idl}^{SRAM} is not zero. During the active mode, the SRAM operates at the minimum operational supply voltage ($V_{sram,min}$), which is the lowest possible voltage at which data can be read from and written to the SRAM. The SRAM thus fails to operate normally below $V_{sram,min}$. For the process and technology node

considered in this chapter, we empirically determine $V_{sram,min}$ to be 0.7 V. At this voltage, it operates at 920 kHz, thus providing sufficient performance for CD-FE. During the idle mode, the SRAM operates at its minimum data-retention voltage ($V_{sram,drv}$), which is the lowest possible supply voltage at which the data can be retained inside the SRAM. For the process and technology node considered in this chapter, we empirically determine $V_{sram,drv}$ to be 0.42 V.

In the active mode, while set to a supply voltage of $V_{sram,min}$, E_{act}^{SRAM} comprises active-switching ($E_{act,swi}^{SRAM}$) and leakage ($E_{act,lkg}^{SRAM}$) energies for a period of T_{CD-FE}. In the idle mode, while set to a supply voltage of $V_{sram,drv}$, E_{idl}^{SRAM} comprises only the leakage energy ($E_{idl,lkg}^{SRAM}$) for the duration ($T_{EPOCH} - T_{CD-FE}$). Thus, we can represent the SRAM energy components as follows:

$$\begin{aligned} E_{lkg}^{SRAM} &= E_{idl,lkg}^{SRAM} + E_{act,lkg}^{SRAM} \\ &= N_{sub} T_{CD-FE} \{I_{V_{sram,min}} V_{sram,min}\} \\ &\quad + N_{sub}(T_{EPOCH} - T_{CD-FE})\{I_{V_{sram,drv}} V_{sram,drv}\} \end{aligned} \tag{5.1}$$

$$E_{act,swi}^{SRAM} = E_{acc}^{sram} \times \#\text{accesses} \tag{5.2}$$

The duration of the active mode (T_{CD-FE}) in Eq. (5.1) depends on ξ, ν, and the optimum logic voltage $V_{dd,opt}$. For smaller (larger) values of ξ and ν, there are more (fewer) coefficients in $\hat{\mathbf{H}}_\mathbf{i}$ and T_{CD-FE} (the active CD-FE time) is higher (lower). For instance, T_{CD-FE} is 0.26 sec. for $\xi = 4\times$ and $\nu = 8\times$, as shown in Fig. 5.11. It increases to 0.52 sec. at $\xi = \nu = 4\times$ and reduces to 0.13 sec. at $\xi = 4\times$ and $\nu = 16\times$. Further, the number of active subarrays (N_{sub}) is also a function of ξ and ν; Fig.5.12 shows this dependence. Eqs. (5.1) and (5.2) also show that although $E_{act,swi}^{SRAM}$ remains invariant with changing values of V_{dd}, it is impacted by ξ and ν (since # accesses changes with ξ and ν). Similarly, the SRAM leakage energy E_{lkg}^{SRAM} scales substantially with ξ and ν. Consequently, the optimal logic voltage

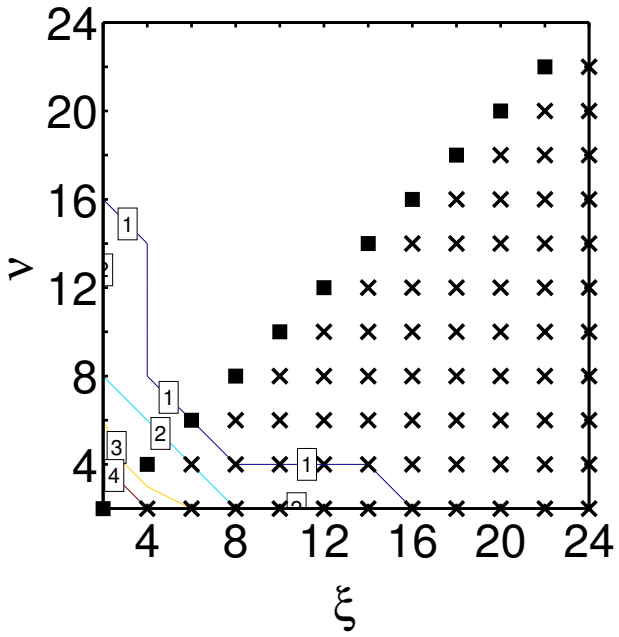


Figure 5.12: Number of active sub-arrays (N_{sub}) scales substantially with ξ and ν affecting the SRAM leakage energy.

$V_{dd,opt}$, which minimizes the SRAM and the logic CD-FE energy, also changes with respect to ξ and ν. We study the variation of $V_{dd,opt}$ in the next section.

5.4 Measurement Results

The IC was prototyped in a 0.13μm CMOS process from IBM. The die photograph and performance summary are shown in Fig. 5.13 and Table 5.1, respectively. 18 channels of Nyquist EEG signals are sampled at a rate of 256 Hz, and eight CD-BPFs are derived corresponding to eight Nyquist-domain BPFs, each of order $k = 64$ (based on the filter specifications required for seizure detection [93]). This leads to a total FV dimensionality of 144. Table 5.1 shows that the CPF permits EEG compression by a factor of ξ = 2-24×, consuming 85-7.3 pJ of energy. Also, the total processor energy is in the 2.2-0.3 μJ range (for linear SVMs), 10.5-38.5 μJ range [for non-linear SVMs using a fourth-order polynomial kernel (poly4)], and 16.0-53.2 μJ range (for SVMs with an RBF kernel). Since the feature processing rate is 0.5 Hz, the total processor power lies between 0.6-106 μW

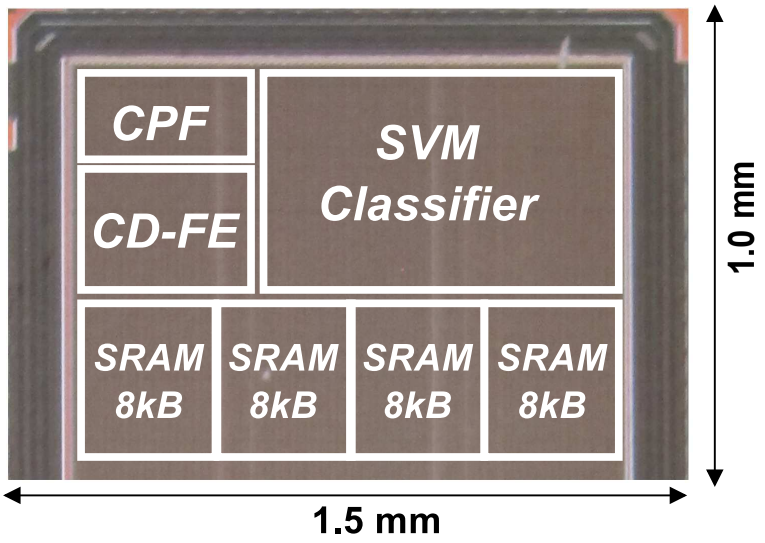


Figure 5.13: Die photo of IC.

for all SVM kernels. The results presented next consider the impact of ξ and ν scaling on the feature-extractor, classifier, and overall processor energies.

5.4.1 Determining the Optimal Voltage for CD-FE Logic

As described in the previous section, the SRAM leakage energy changes with both ξ and ν. Thus, the optimal voltage ($V_{dd,opt}$) for the CD-FE logic changes with both ξ and ν. In order to determine $V_{dd,opt}$, we minimize the total CD-FE energy comprising the logic and SRAM energies.

Fig. 5.14 shows the measured subcomponents of the CD-FE energy with respect to V_{dd} when N_{sub} ranges from 1 to 4 (corresponding to four different values of ξ and ν). For all values of N_{sub}, the active energy (E_{swi}^{logic}) of the CD-FE logic increases and the leakage energy (E_{lkg}^{logic}) decreases with increasing values of V_{dd}, leading to the minimum-energy point of 0.46 V [198]. However, this is not $V_{dd,opt}$ since we need to also consider the SRAM energy. The SRAM operates at 0.7 V in the active mode. We see from Fig. 5.14 that the SRAM active-mode access energy $E_{act,swi}^{SRAM}$ does not change with V_{dd} [consistent with Eq. (5.2)]. Further, the leakage energies in the active ($E_{act,lkg}^{SRAM}$) and idle modes ($E_{idl,lkg}^{SRAM}$)

Table 5.1: Performance summary: energy-scalable, compressed-domain processor IC.

Technology		IBM 130nm LP CMOS	
Supply voltage		CD-FE: 1.2-0.44 V SRAM: 0.7/0.42 V CPF/ SVM: 0.48 V	
EEG sampling rate		256 Hz	
Clock frequency		10.2−0.3 MHz	
CPF compression factor ξ		2-24×	
Projection factor ν		2-24×	
Feature computation rate		0.5 Hz	
CD-BPF memory size		0.44-32 kB	
SUBBLOCK ENERGY MEASUREMENTS			
		per FV	per Clock
CPF (at 0.48 V)		85.0-7.3 pJ	10.6 fJ
CD-FE logic (at $V_{dd,opt}$)		70.8-1.3 nJ	1.3 pJ
SRAM subarray (at 0.7 V)		2.1-0.1 μJ	5.0 pJ
Total Feature Extraction		**2.1 μJ-93.2 nJ**	**6.3 pJ**
SVM	RBF	16.0-53.2 μJ	6.0 pJ
	Poly4	10.5-38.4 μJ	4.8 pJ
	Linear	62.9-209.0 nJ	2.0 pJ
Total Processor (linear SVM)		**2.2-0.3 μJ**	**8.3 pJ**

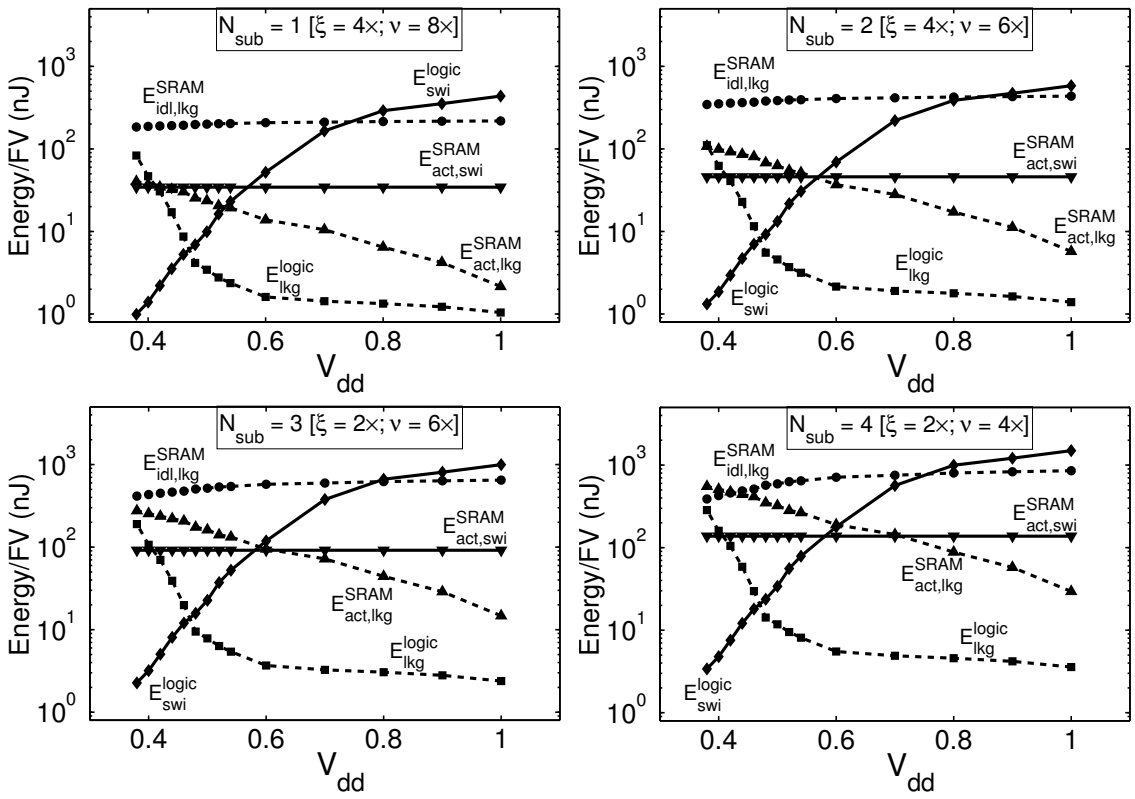


Figure 5.14: The CD-FE energy subcomponents introduce non-linear dependence on ξ and ν. Primarily, the SRAM leakage energy in the active ($E^{SRAM}_{act,lkg}$) and idle mode ($E^{SRAM}_{idl,lkg}$) is substantially impacted by N_{sub} and T_{CD-FE}. The active-mode SRAM access energy ($E^{SRAM}_{act,swi}$), however, does not change with ξ or ν.

increase as N_{sub} increases. This is also expected since from Eq. (5.1), E^{SRAM}_{lkg} depends on N_{sub}. However, since both $E^{SRAM}_{act,lkg}$ and $E^{SRAM}_{idl,lkg}$ also depend on $V_{dd,opt}$, the increase in the leakage energies is not proportional to the increase in N_{sub} [Eq. (5.1)]. The total CD-FE energy is thus a non-linear function of ξ and ν, which necessitates $V_{dd,opt}$ to be determined numerically.

Fig. 5.15 shows the measured CD-FE energy at different voltage values for the cases considered in Fig. 5.14. For these four instances, we see from the figure that $V_{dd,opt}$ for the CD-FE logic is either 0.48 V or 0.5 V. The corresponding frequencies are determined to be 380 or 420 kHz, respectively, from the frequency vs. V_{dd} plot in Fig. 5.16. With more measurements, we determine $V_{dd,opt}$, frequency, and active time (T_{CD-FE}) for the CD-FE logic when ξ and ν vary in the 2-24× range. The results are shown in Figs 5.17(a), (b), and

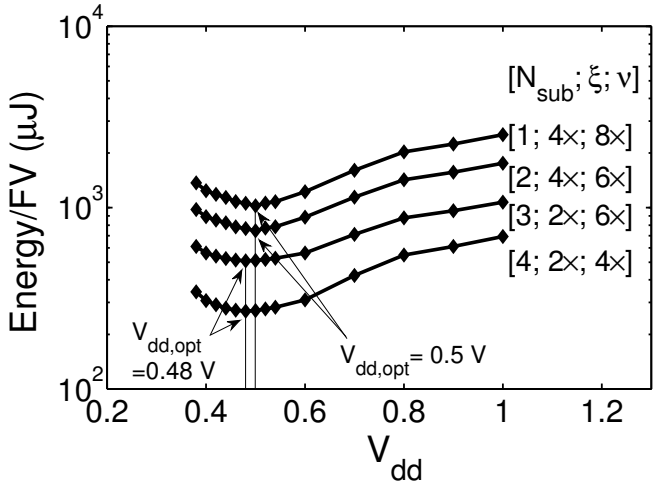


Figure 5.15: For the four cases considered in Fig. 5.14, the optimal voltage for the CD-FE logic ($V_{dd,opt}$) is either 0.48 V or 0.5 V.

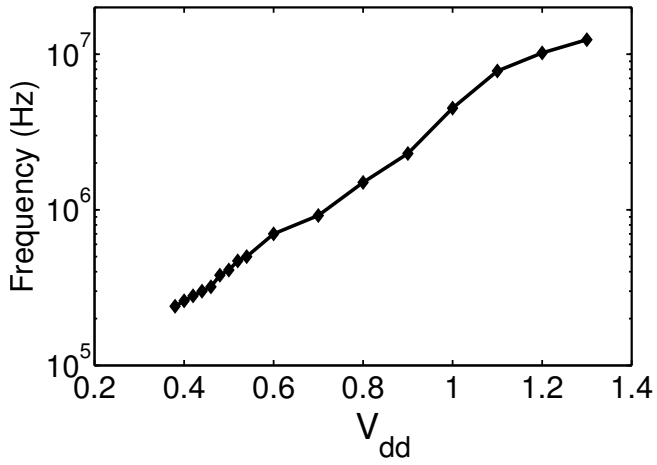
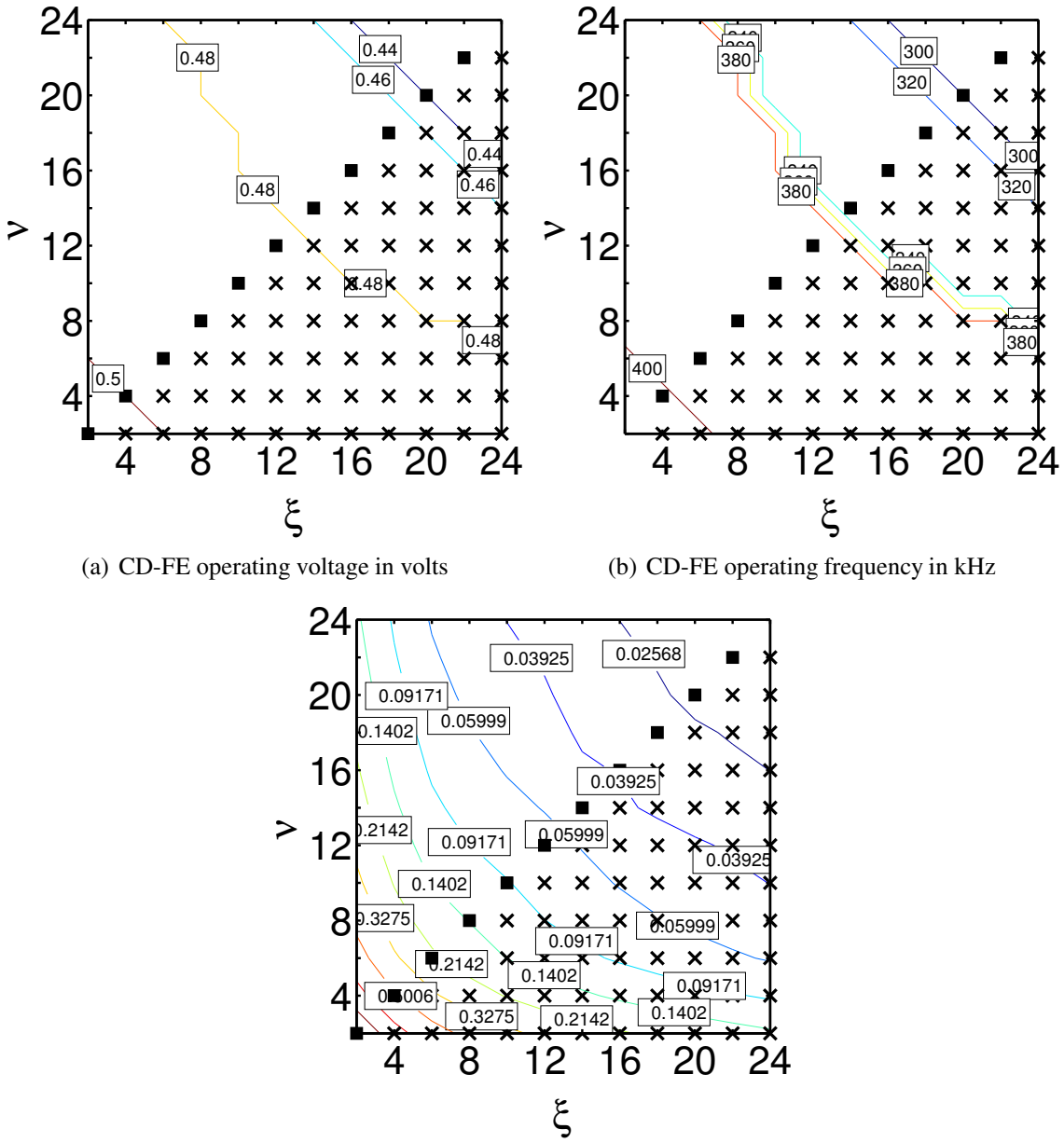


Figure 5.16: The operating frequency vs. V_{dd} for the CD-FE logic.

(c), respectively. For all values of ξ and ν, the active time T_{CD-FE} varies in the 0.9-0.02 sec. range and is below the epoch time of 2 sec., which allows sufficient time ($T_{EPOCH} - T_{CD-FE}$) for the SVM classifier to finish computing.

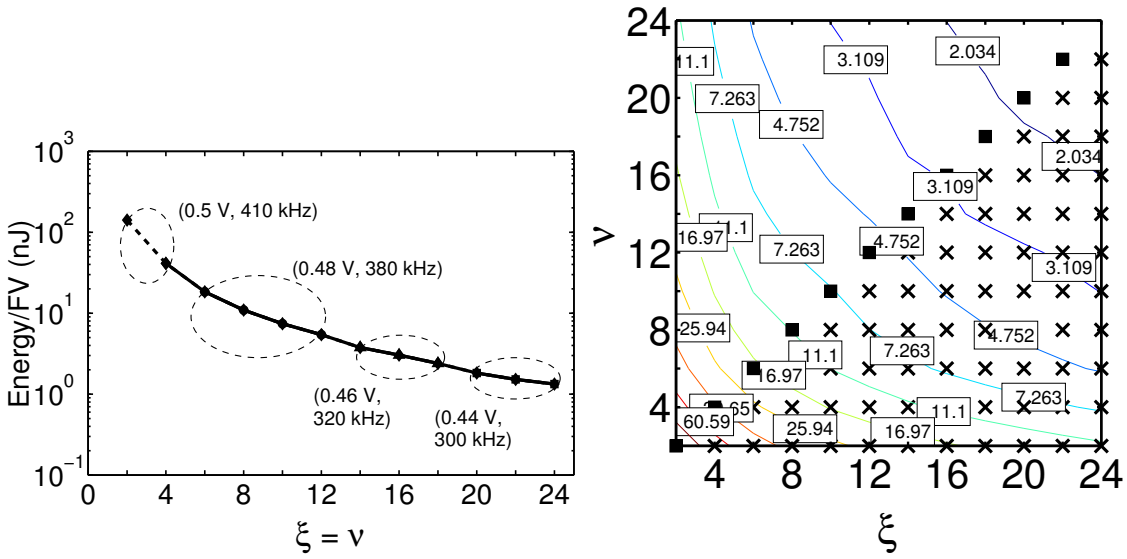


(a) CD-FE operating voltage in volts

(b) CD-FE operating frequency in kHz

(c) CD-FE active time in seconds

Figure 5.17: As ξ and ν scale in the 2-24× range, (a) the optimal voltage for the CD-FE logic ($V_{dd,opt}$) varies in the 0.5-0.44 V range, (b) the corresponding operating frequency varies in the 420-300 kHz range, and (c) the CD-FE active time (T_{CD-FE}) varies in the 0.9-0.02 sec. range.



(a) Exact solution (b) Approx. solution: Logic energy in *n*J

Figure 5.18: The CD-FE logic energy for (a) the exact solution, and (b) the approximate solution measured at $V_{dd,opt}$. The energy scales substantially with ξ and ν.

5.4.2 Feature-extractor Energy

As mentioned previously, the CD-FE energy comprises the logic and SRAM energies. In this section, we provide measurement results for these energy subcomponents using both the exact and approximate solutions for $\hat{\mathbf{H}}_\mathbf{i}$.

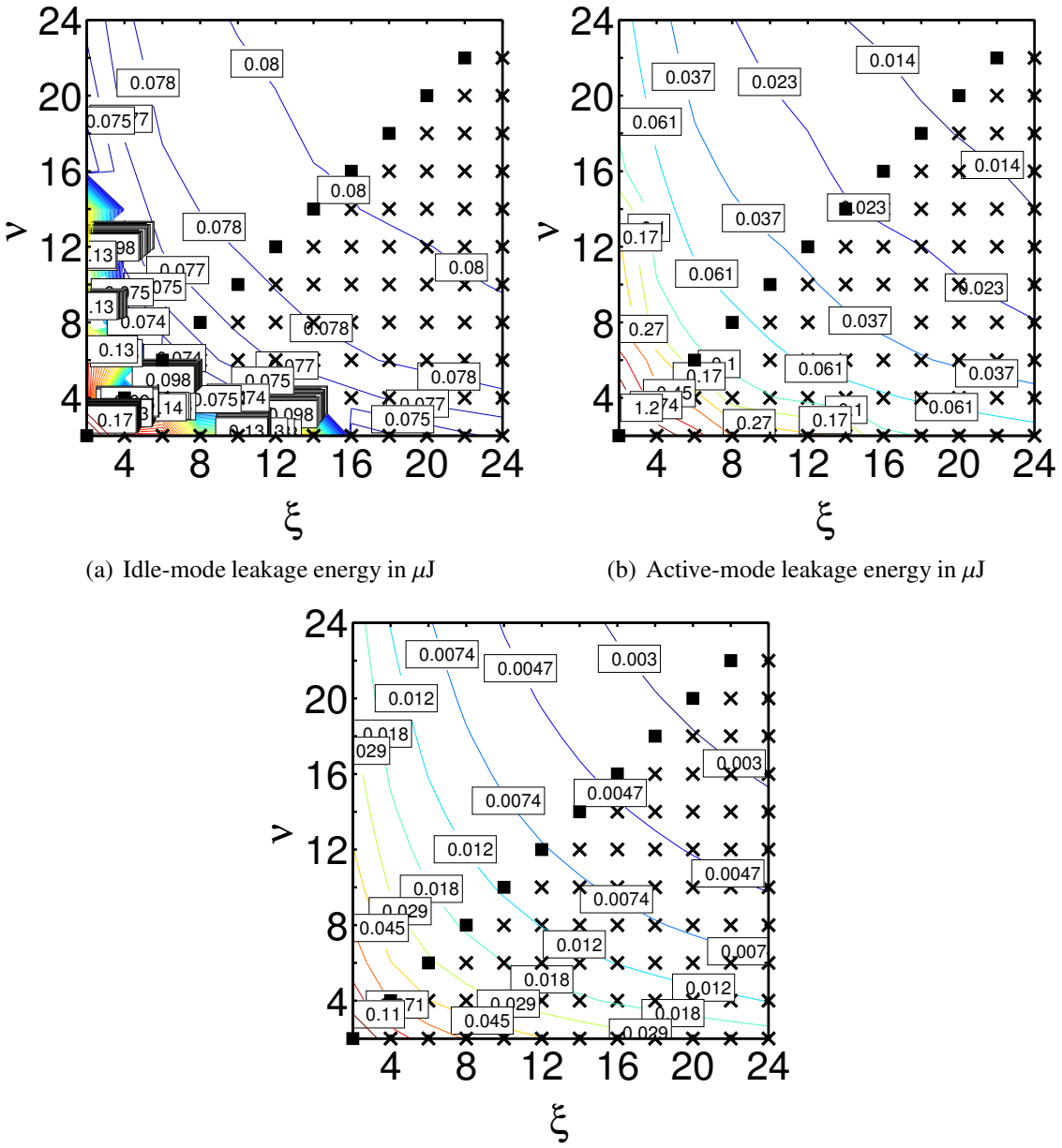
CD-FE logic energy. Recall from Sec. 5.2.1 that for the exact solution (*i.e.,* when $\xi = \nu$), the CD-FE complexity scales quadratically with ξ; for the approximate solution, it scales linearly with both ξ and ν. Figs. 5.18(a) and (b) show the CD-FE energy for the exact and approximate solutions, respectively. For each value of ξ and ν, the energy is reported for $V_{dd,opt}$, which minimizes CD-FE's active-switching and leakage energies as well as the SRAM energy; the $V_{dd,opt}$ values are also annotated in Fig. 5.18(a) and can be correlated to Fig. 5.17(a).

SRAM energy. Figs. 5.19(a) and (b) show the SRAM leakage energies in the idle and active modes and Fig. 5.19(c) shows the SRAM access energy in the active mode, versus ξ and ν. We can see from the figures that for smaller values of ξ and ν, since the size

of $\hat{\mathbf{H}}_\mathbf{i}$ is larger, T_{CD-FE} is higher and the SRAM active energy dominates the idle-mode energy. This is also consistent with a higher value of $V_{dd,opt}$ at these values of ξ and ν, which helps the CD-FE computations to finish sooner. In contrast, at larger values of ξ and ν, however, there are fewer coefficients in $\hat{\mathbf{H}}_\mathbf{i}$ and the SRAM spends most of the time in the idle mode. This behavior is clear from Figs. 5.20(a) and (b), which show the total SRAM energy for the exact and approximate solutions. The figures show that the total SRAM energy is nearly equal to the SRAM idle-mode energy at higher values of ξ and ν. Further, the figures also show substantial scaling in the total SRAM energy and in its constituents with respect to ξ and ν. This scaling occurs due to the variation in N_{sub} and T_{CD-FE} versus ξ and ν [see Eqs. (5.1) and (5.2) and Figs. 5.12 and 5.17(c)]. The SRAM energy thus eventually begins to saturate due to the granularity limit of the four subarrays; a finer granularity would enhance scaling at the cost of additional hardware overhead.

Total feature-extraction energy. From the above results, we see that the SRAM energy can significantly dominate the CD-FE logic energy at all values of ξ and ν. This behavior validates the focus on optimizing the SRAM energy in Sec. 5.3.2; for instance, at $\xi = 4\times$ and $\nu = 2\times$, the total SRAM energy is 2.1 μJ and the CD-FE logic energy is 70.8 nJ. The contribution of the energy subcomponents is also apparent in the total CD-FE energy plots shown for the exact and the approximate solutions in Figs. 5.21(a) and (b), respectively (results are for 18 EEG channels with eight CD-BPFs). These plots show that the CD-FE energy profile is similar to the SRAM energy profile presented in the previous section.

Comparison with Nyquist-domain processing. Since $\mathbf{H}_\mathbf{i}$ are convolution matrices, the filter order determines the number of non-zero coefficients in $\mathbf{H}_\mathbf{i}$ (see Fig. 5.5), which, in turn, determine the feature-extraction energy in the Nyquist domain. However, in the compressed domain, due to the loss of regularity in $\hat{\mathbf{H}}_\mathbf{i}$, the feature-extraction energy does not depend on the filter order in the same way. Thus, in the compressed domain, the energy can initially increase due to loss of regularity in $\hat{\mathbf{H}}_\mathbf{i}$, but then it can eventually decrease owing to scaling in the size of $\hat{\mathbf{H}}_\mathbf{i}$ due to both ξ and ν. Further, at a given value of ξ and ν, we can

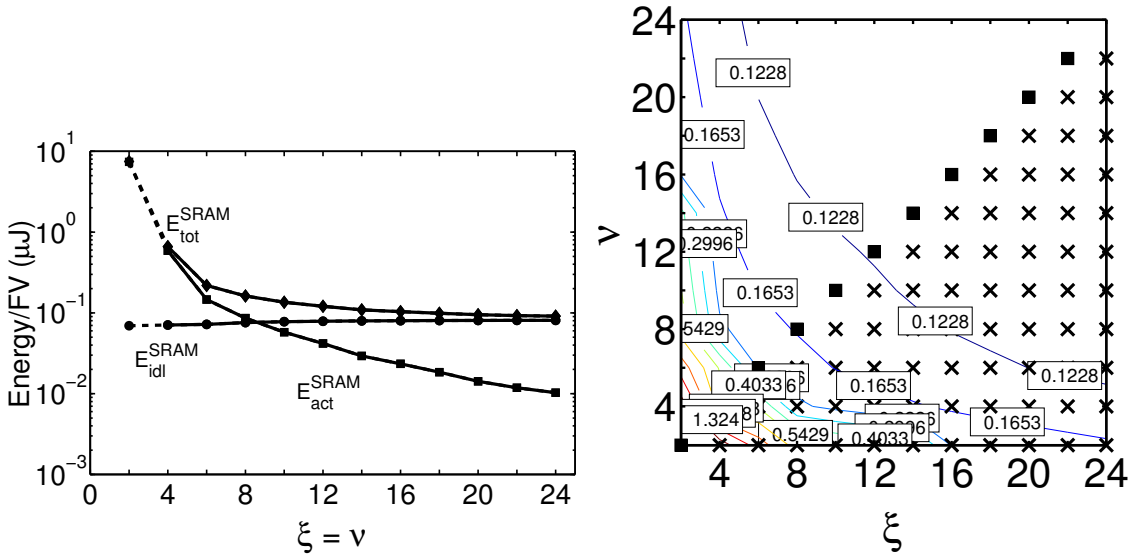


(a) Idle-mode leakage energy in μJ

(b) Active-mode leakage energy in μJ

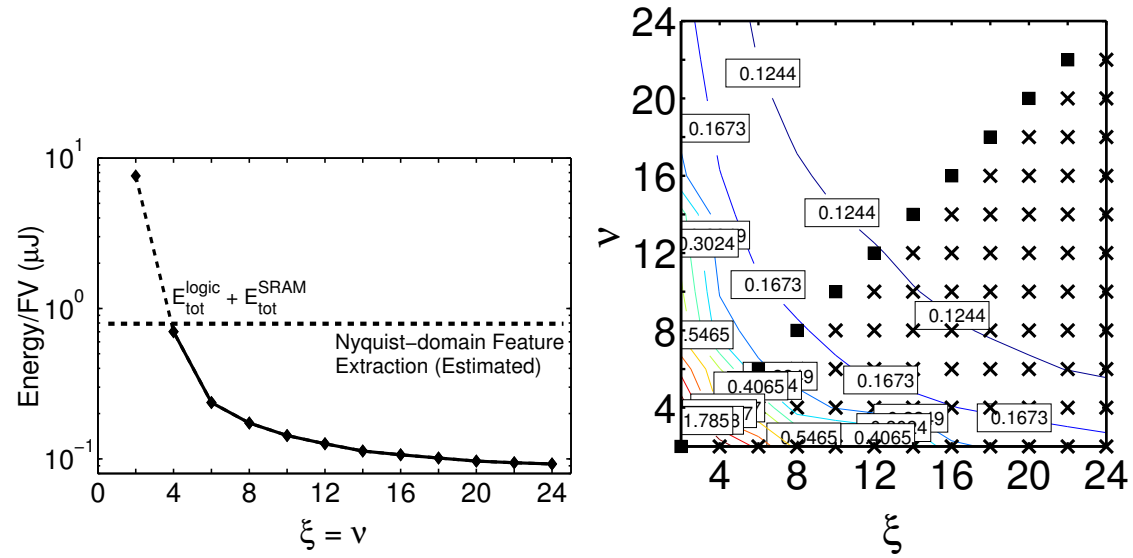
(c) Active-mode switching energy in μJ

Figure 5.19: Each of the SRAM energy subcomponents, *i.e.,* (a) idle-mode leakage ($E^{SRAM}_{idl,lkg}$), (b) active-mode leakage ($E^{SRAM}_{act,lkg}$), and (c) active-mode access ($E^{SRAM}_{act,swi}$) scale with both ξ and ν. $E^{SRAM}_{act,lkg}$ tends to dominate at smaller values of ξ and ν.



(a) Exact solution (b) Approx. solution: SRAM energy in μJ

Figure 5.20: The total SRAM energy for (a) the exact solution, and (b) the approximate solution scales substantially at smaller values of ξ and ν. At higher values of ξ and ν, CD-BPF matrices $\hat{\mathbf{H}}_\mathbf{i}$ are smaller, which makes $E_{idl,lkg}^{SRAM}$ dominate the total SRAM energy.



(a) Exact solution (b) Approx. solution: CD-FE energy in μJ

Figure 5.21: The total CD-FE energy (logic + SRAM) for (a) the exact solution, and (b) the approximate solution. The SRAM energy tends to dominate and thus provides scalability with both ξ and ν.

scale the total CD-FE energy by the ratio of the number of non-zero coefficients in $\mathbf{H_i}$ and $\hat{\mathbf{H}}_\mathbf{i}$ to derive an estimate for the Nyquist-domain feature-extraction energy. Fig. 5.21 shows that for the exact solution, at $\xi > 4\times$, the total energy of compressed-domain processing is less than that projected for Nyquist-domain processing (for a 64-order FIR filter).

5.4.3 Classifier Energy

One downside of directly processing compressively-sensed EEG is that the SVM model for classification can become somewhat more complex at higher amounts of data compression. For instance, Fig. 5.22 shows that N_{SV} can increase up to approximately 28% at $\xi = 10\times$. Intuitively, this happens due to the additional error introduced in the FVs when we solve the compressed-domain equations [Eq. (4.5)], which necessitates complex decision boundaries in the classifier. Fig. 5.23 shows the SVM energy for the exact solution using three kernels: RBF, 4^{th}-order polynomial (poly4), and linear. Figs. 5.24(a), (b), and (c) show the classifier energy for the approximate solution using the same three kernels, respectively. In each of these cases, the SVM operates at its minimum-energy point of 0.48 V. From Fig. 5.24, we can see that the increase in classifier energy opposes the reduction in CD-FE energy. We can also see that the SVM energy increase becomes worse when ν is significantly higher than ξ, which reflects the extra error introduced at the algorithmic level due to a degradation in the JL-approximation.

5.4.4 Total Processor Energy

Fig. 5.25 shows the effect of ξ scaling on the total processor energy for the exact solution. Figs. 5.26(a), (b), and (c) show the effect of ξ and ν scaling on the total processor energy for the approximate solution using the RBF, poly4, and linear classification kernels, respectively. The SVM operates at 0.48 V, the CD-FE operates at $V_{dd,opt}$ [specified in Fig. 5.17(a)], and the SRAMs operate at 0.7/0.42 V during the active/idle modes. The figures show that non-linear SVM kernels (*i.e.,* RBF and poly4) consume significant en-

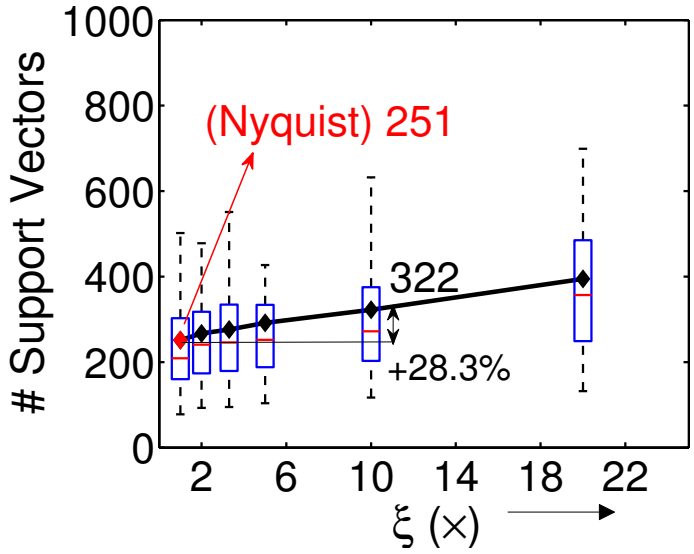


Figure 5.22: The SVM classification model increases with the amount of compression.

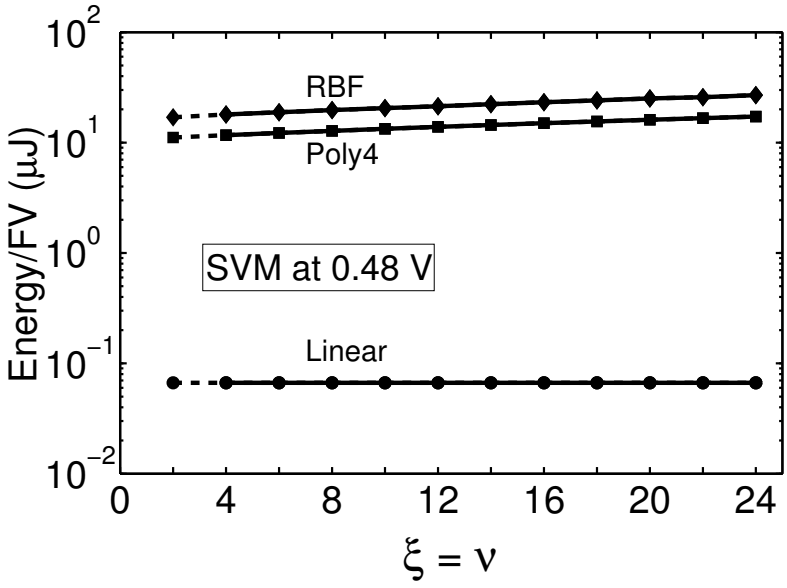
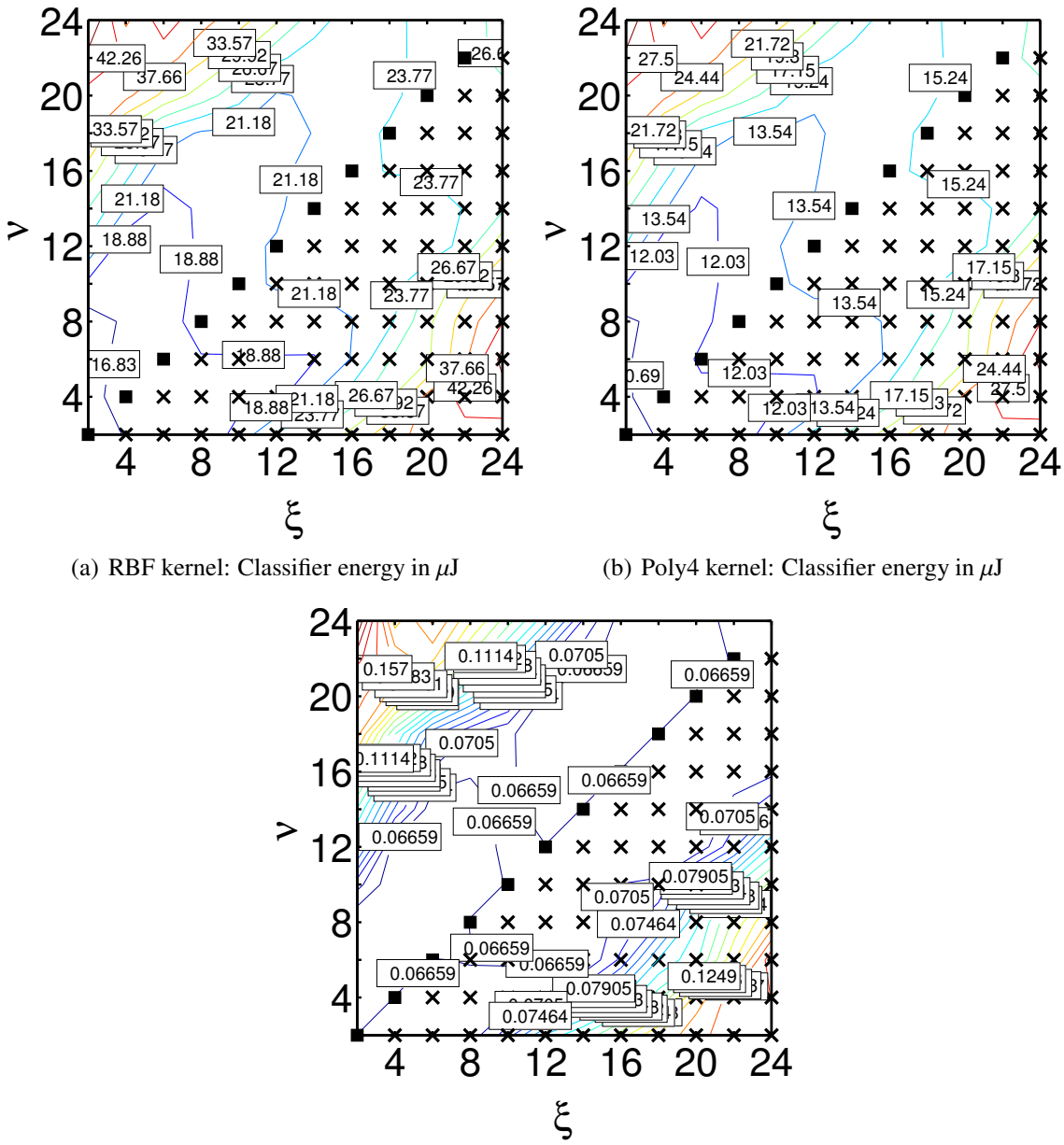


Figure 5.23: SVM classifier energy for the exact solution.



(a) RBF kernel: Classifier energy in μJ

(b) Poly4 kernel: Classifier energy in μJ

(c) Linear kernel: Classifier energy in μJ

Figure 5.24: The SVM classifier energy measured at the minimum-energy point of 0.48 V for the approximate solution using (a) RBF, (b) poly4, and (c) linear kernel.

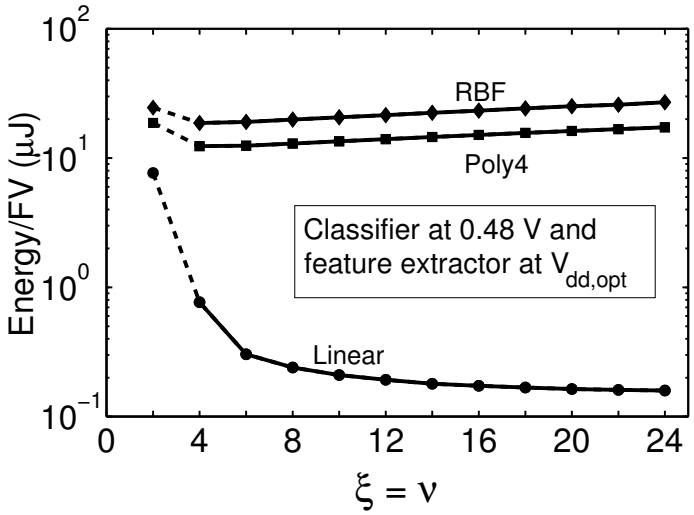
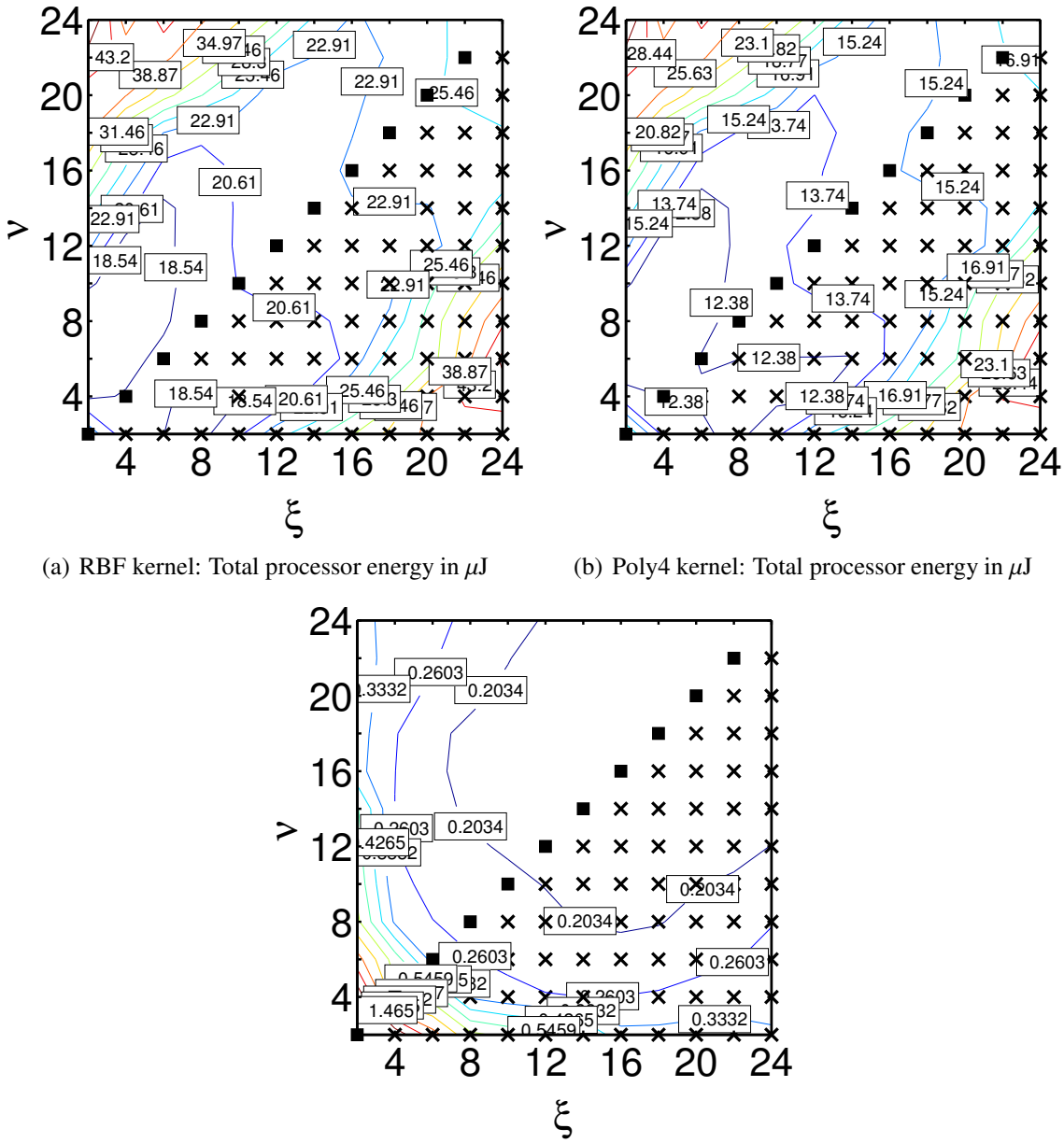


Figure 5.25: Total processor energy for the exact solution.

ergy, while SVMs with a linear kernel incur minimal energy, causing the energy-scaling characteristics to be dominated by CD-FE at all values of ξ and ν. For the non-linear cases, the SVM energy actually leads to optimal ξ and ν values (*e.g.,* for the exact solution, from Fig. 5.25, an optimal ξ of approximately 5× minimizes the total processor energy).

5.5 Chapter Summary

Compressive sensing exploits signal sparsity in a secondary basis to achieve very low-energy compression on the sensing node. In this chapter, we presented the design of a processor that applies the auxiliary matrix based algorithmic formulation described in Chapter 4 to the seizure detection application (with square BPFs) described in Chapter 3. Our processor thus enables on-node signal analysis to detect epileptic seizures by directly using compressively-sensed EEG. Using the methods described in Chapter 4, we first derived an exact solution for the CD-BPFs. Through simulations, we showed that the performance of the compressed-domain detector is retained up to high compression factors. Additionally, by using an approximate solution, we also derived smaller-sized CD-BPFs, saving us more energy in the compressed domain. By taking advantage of a subarray of SRAMs in a pro-



(a) RBF kernel: Total processor energy in μJ

(b) Poly4 kernel: Total processor energy in μJ

(c) Linear kernel: Total processor energy in μJ

Figure 5.26: Total processor energy for the approximate solution using (a) RBF, (b) poly4, and (c) linear kernel. For non-linear SVMs, classifier energy dominates due to the extra modeling complexity in the compressed domain, while for linear kernels, CD-FE energy is higher and permits substantial scalability with ξ and ν.

totype IC implementation, we showed how to exploit the two knobs, ξ and ν, to control the energy consumption of the processor. Thus, due to the ξ and ν knobs, in addition to communication energy savings, through end-to-end data reduction in a system, our processor provided a mode of power management where the computational energy scaled due to both a reduction in the number of input samples that needed to be processed and approximations introduced at the algorithmic level.

Our results also suggest a caveat. Although compressed-domain processing can help significantly lower the energy for feature extraction by employing higher amounts of data compression, the classification energy can increase modestly at higher values of ξ. In fact, many applications require non-linear classifiers; for instance, RBF or polynomial transformation functions are required for accurate arrhythmia detection [199]. In such applications, the energy for classification can significantly dominate the feature-extraction energy in NA itself. In these cases, processing signals using CA can add to the high initial complexity of the classifier. In the next chapter, we thus explore VLSI optimizations to reduce the overall processor energy by focusing on the optimization of the classifier computations. For this purpose, we employ hardware specialization, while simultaneously retaining selective flexibility to accommodate applications involving a range of classifier complexities.

Chapter 6

Hardware Study: Energy-efficient Classification

In the previous three chapters, we saw that the need for high-order classification models in CA arises due to the additional error introduced by the compressed-domain equations. For instance, Fig. 3.14 showed that in a seizure-detection application, although the model complexity of processing EEG signals in NA was low (only 251 SVs), the complexity in CA can be significantly higher (up to 28.3% higher at $\xi = 10\times$). This complexity increase was also reflected in the energy measurements presented in the previous chapter *e.g.*, see Figs. 5.23 and 5.24 in Sec. 5.4.3. Further, in several embedded applications, especially in those that employ non-linear kernels, the SVM complexity can be higher to begin with, *i.e.*, it can be higher in NA itself. For instance, when we have to discriminate between very similar ECG signals to detect abnormal heartbeats. Both of these situations call for a reduction in the classifier energy. In fact, reducing the classifier energy for the latter set of applications will also help us handle the additional complexity increase arising due to the CA transformation. In this chapter, we thus focus on an application where the classifier model complexity is extremely high in NA. In particular, we study two algorithms for detecting cardiac arrhythmias that employ morphological and wavelet feature extrac-

tion followed by SVM classification. These algorithms require a large number of SVs for accurate classification. A large number of SVs in linear SVM kernels do not pose a computational challenge [199]. However, for arrhythmia detection, linear kernels exhibit poor classification performance, necessitating the use of non-linear transformation functions such as the polynomial and the RBF [199]. These limitations imply high energy costs for classification in NA. We thus investigate the energy bottlenecks in these algorithms and perform an architectural design-space exploration for energy reduction. Based on our analysis, we propose a coprocessor-based platform for signal analysis. In the arrhythmia application, since the feature-extraction energy is very small compared to classification, we offload feature-extraction computations, along with all preprocessing steps like segmentation and beat alignment, to an embedded processor, while performing classification on a dedicated coprocessor. Our aim in building this hybrid architecture is to provide hardware specialization to save computational energy, while simultaneously allowing application-level flexibility. This balancing act makes our platform adaptable to a range of algorithms for signal analysis. We also quantify the achievable energy savings in the proposed platform through post-layout simulations of a prototype IC implementation.

6.1 Introduction

The central need when embedded devices aim to provide actionable outputs through signal analysis is the ability to detect specific states of interest from signals that are available through low-power sensors [91]. However, the targeted and background variations are often expressed through subtle manifestations in signal waveforms. These manifestations raise the need for extremely high-order classification models. As an illustration, the challenges associated with arrhythmia detection are shown in Fig. 6.1. Abnormal beat morphologies in an ECG, annotated as premature ventricular contraction (V) and ventricular trigeminy (T), exhibit subtle intra- and inter-patient variability. Their discrimination from normal

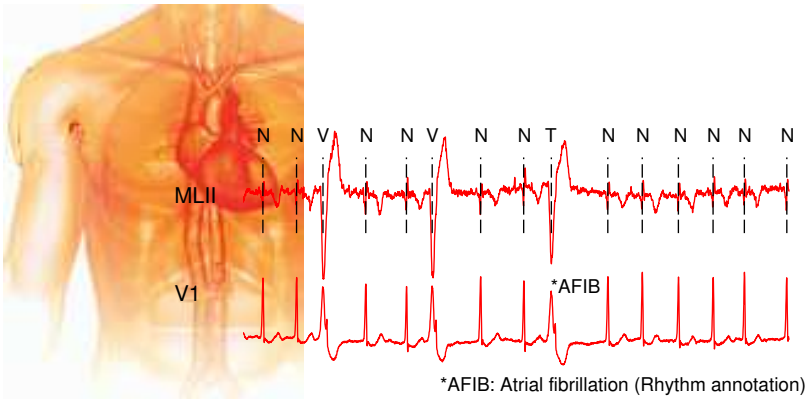


Figure 6.1: Signal correlations to clinically relevant states are complex to model based on physiology and precise correlations are hard to distinguish from normal physiologic activity.

(N) beats thus poses a challenge for signal detection and inference. Thus, since both the targeted and background variations are often expressed through very specific manifestations in physiological waveforms, their detection necessitates high-order models for signal analysis.

In fact, for arrhythmia detection, the detection models can be so complex that the classification energy can substantially dominate the feature-extraction energy. In this chapter, we thus present a specialized computational platform for data-driven signal analysis that can help reduce the energy costs associated with high-order classification models. We investigate the principles behind a hardware-specialized architecture by quantitatively evaluating various platform options, from general-purpose CPUs to custom instructions to hardware coprocessors. For the coprocessor approaches, we also evaluate the potential of microarchitecture- and circuit-level opportunities, such as dynamic voltage and precision scaling. We thus propose an algorithm-driven methodology that takes advantage of the computational structure and the characteristics of data-driven signal-analysis algorithms. Our specific contributions are as follows:

- We present an energy analysis of representative signal-analysis algorithms (algorithms for cardiac arrhythmia detection are considered in detail). Our results are based on patient data from the MIT-BIH database [200] and show that classification,

the complexity of which depends on the characteristics of the data, can often pose a major energy limitation. We also show that the computational structure of classification limits the energy savings attainable through the use of custom instructions.

- Based on the energy analysis and the computational requirements of various parts of the algorithms, we propose a generalized architecture for an embedded signal-analysis platform. This attempts to employ programmability where computational flexibility is required, while leveraging hardware specialization for classification, where set computations are required at very high energy efficiency.

- We propose a transistor-level design of a classification coprocessor that leverages a low-power technology (*i.e.*, low-leakage FD-SOI). Specific requirements for computational flexibility are identified and incorporated through hardware scalability in a parallelized subthreshold implementation that operates at the minimum-energy supply voltage.

The rest of the chapter is organized as follows. In Sec. 6.2, we present an analysis of the arrhythmia algorithms and identify the computational bottlenecks involved. In Sec. 6.3, we explore the architecture of a low-energy data-driven computational platform through custom instructions and a coprocessor. In Sec. 6.4, we describe specialized circuits at the transistor level, including a variable-precision MAC unit for the coprocessor based implementation. In Sec. 6.5, we present post-layout simulation results for the coprocessor. Finally, we conclude in Sec. 6.6.

6.2 Application-domain Algorithmic Study: Arrhythmia Detection

In this section, we describe the computational structure of algorithms used for analyzing ECG signals. Since arrhythmia detection involves high-order detection models, we focus

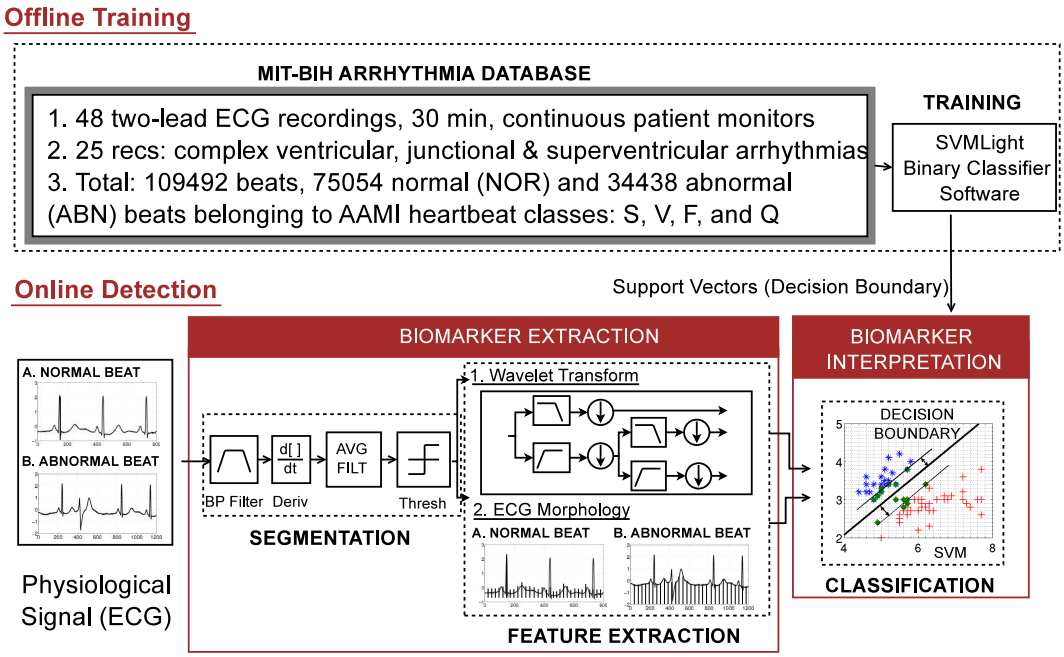


Figure 6.2: The structure of arrhythmia detection algorithms includes offline training and online detection (employing biomarker extraction and interpretation).

on it as an example in our algorithmic study. The algorithms we use employ morphological and wavelet features and perform detection computations based on an SVM classifier.

6.2.1 General Structure of Algorithms

Cardiac arrhythmias refer to abnormal heart beats that are indicative of a range of cardiovascular conditions.

Fig. 6.2 illustrates the structure of the arrhythmia-detection algorithms we consider. ECG data are preprocessed for noise removal (band-pass filtering), QRS detection [201], and beat segmentation [202]. Subsequently, the detection process involves the following two primary steps: (1) biomarker extraction, and (2) biomarker interpretation (through a classifier) [203]. Recall that biomarkers refer to specific signal parameters that are indicative of the physiological state of interest [204, 205]. For arrhythmia detection, a range of biomarkers has been used (including ECG morphology, beat intervals, spectral features, *etc*. [206–209]). The diversity in the choice of biomarkers is due to the various clinical

trade-offs introduced by each, which can also be variable across patients [210, 211]. In this study, we use two prominent biomarkers: waveform morphology [206] and spectral wavelets [212]. The associated processing steps, including segmentation (to isolate individual beats), are then implemented in software (enabling the energy analysis presented next). The outputs from these stages form the FVs that are used for classification.

Following the feature-extraction process, an SVM is used for modeling and classification. Recall from Sec. 3.2 that the actual classification computation in an SVM [for RBF and polynomial transformation kernels] is as follows:

$$\text{Data Class} = sgn\left[\sum_{i=1}^{N_{SV}} K\left(\vec{\mathbf{x}} \cdot \vec{\mathbf{sv_i}}\right) \alpha_i y_i - b\right] \tag{6.1}$$

$$\text{where } K\left(\vec{\mathbf{x}} \cdot \vec{\mathbf{sv_i}}\right) = \begin{cases} exp(-\gamma \|\vec{\mathbf{x}} - \vec{\mathbf{sv_i}}\|^2) & \text{RBF kernel} \\ F(\vec{\mathbf{x}} \cdot \vec{\mathbf{sv_i}} + \beta)^d & \text{Poly. kernel} \end{cases}$$

where $sgn[]$ is the *signum function*, $\vec{\mathbf{x}}$ is the FV to be classified, and $\vec{\mathbf{sv_i}}$ is the i^{th} SV (b, d, α_i, β, γ, and y_i are training parameters). N_{SV} is the total number of SVs used in the computation. Note that as the number of SVs (N_{SV}) and the FV dimensionality (D_{SV}) scale, the classification computations are dominated by the dot-product between $\vec{\mathbf{x}}$ and $\vec{\mathbf{sv_i}}$. Further, K represents a kernel function, whose choice, along with N_{SV} and D_{SV}, can have a major impact on classifier complexity.

6.2.2 Need for Advanced Classification Models

In this section, we present the limitations of using simple classification models (which would address the complexity challenge described above). Experimental results show that as the model complexity is reduced, the classifier performance degrades, necessitating high-order models. The detection algorithms illustrated in Fig. 6.2 are implemented using SVM-Light [213], an open-source implementation of an SVM, for the classifier. To

correctly analyze the computational complexities and trade-offs imposed by the model, we use patient data from the MIT-BIH database [200][1].

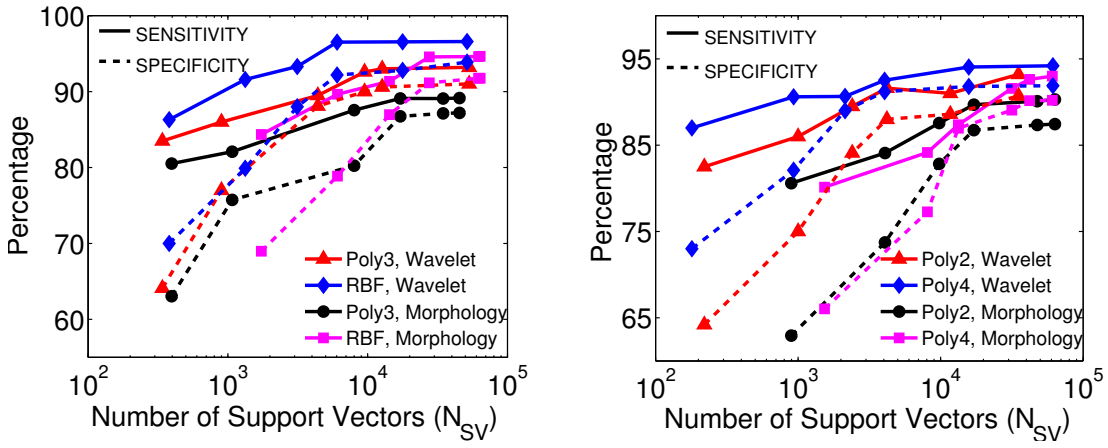
If the FVs were linearly separable, a linear kernel function could be used for classification [214]. The test vector could then be pulled out of the summation in Eq. (3.1), enabling precomputation of a single decision vector. As a result, even when N_{SV} scales, the classification energy can remain constant. However, several applications that require embedded signal analysis, for instance in the biomedical domain, have shown to perform poorly when linear decision functions are used [101]. Non-linear functions, such as high-order polynomials, RBFs or sigmoidal kernels, are thus needed for acceptable classifier accuracies [215]. Several signal-analysis algorithms in the literature employ non-linear kernels for classification. For instance, using polynomial and RBF kernels, 15.8% and 47.4% improvements in detection accuracy are reported in [216] and [217], respectively.

The performance of the classifier depends on the SV model and the characteristics of the application data. Figs. 6.3(a) and (b) show how the sensitivity and specificity for arrhythmia detection degrade as N_{SV} is reduced. In order to reduce N_{SV}, the training parameters b, d, α_i, β, γ, and y_i are adjusted along with the choice of the data subset used for training. Thus, the model complexity depends on the characteristics of the application data and introduces an unavoidable trade-off with respect to accuracy performance. We next present an energy analysis of the end-to-end arrhythmia detection algorithm, which employs high-order detection models for accurate signal classification. This analysis will enable an architectural study towards a low-energy application-domain processor.

6.3 Application-domain Architectural Study

We adopt three approaches in our architectural study. First, we implement the entire arrhythmia detection algorithm on an embedded low-power base processor. We find that

[1]Based on the ANSI/AAMI-recommended practice, four of the 48 MIT-BIH records (102, 104, 107, and 217) included paced beats and were not used in our evaluations.



(a) (b)

Figure 6.3: Reducing model complexity (by reducing N_{SV}) degrades the accuracy of classification (a) for RBF and polynomial kernels of order 3, and (b) for polynomial kernels of order 2 and 4.

classification poses an energy bottleneck due to the complexity of the models required. We then explore opportunities for hardware specialization through custom instructions and finally through a coprocessor.

For the base processor, we use the Xtensa processor from Tensilica, which is a customizable and extensible platform [218]. A family of processors can be built around the base instruction set architecture (ISA) of the synthesizable Xtensa processor core [219, 220]. As a result, custom processor configurations can be obtained with optimized performance, power dissipation, code size, and die size. Design automation algorithms and tools used in extensible and configurable processors, such as Tensilica, are discussed in [221, 222]. A processor generator provides *configurability* through selectable additional instructions, accelerators, memory/cache architectures, exception/interrupt configurations, and debugging support. It provides *extensibility* through single- or multi-cycle custom instructions and accelerators that can be defined via the Tensilica Instruction Extension (TIE) hardware description language. Design-space exploration of the Tensilica processor parameters allows us to customize the base processor to achieve minimum-energy consumption. The customizable features of the Xtensa architecture are illustrated in Fig. 6.4.

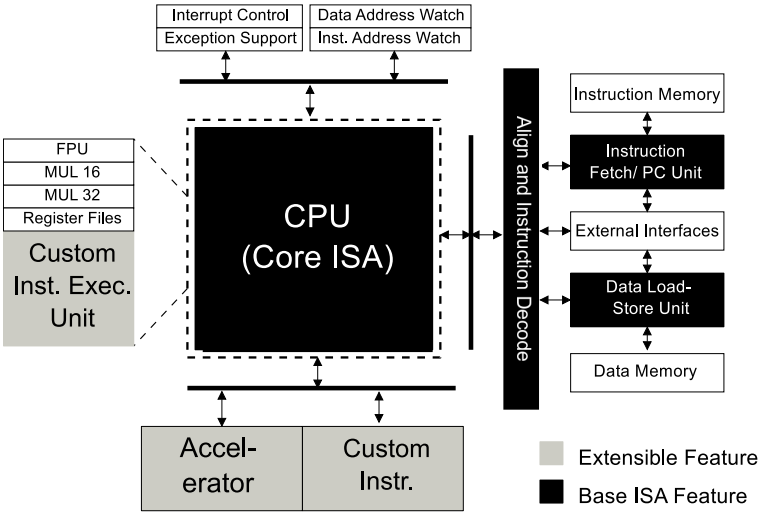


Figure 6.4: The Xtensa architecture provides a base processor ISA along with optional instructions and features (*e.g.*, MUL-16/32, register files, *etc*.). Custom instructions and accelerators can be defined using the TIE hardware description language and added to the processor for application extensibility. A software tool suite (*e.g.*, retargetable compiler, instruction set simulator, *etc*.) is automatically generated for the custom processor.

The major parameters include the choice of multipliers (MUL32/MUL16), instruction/data cache sizes and associativities (range of 0-16 kB and 2-16), and datapath/instruction path widths (range of 8-32 bits). We pick the design parameters that lead to minimum-energy consumption based on an initial parameter-space exploration. The configuration obtained for the base processor is shown in Table 6.1. We present energy profiling results of the arrhythmia detector on the configured base processor.

6.3.1 Implementation on the Base Processor

In this section, we present energy analysis of a software implementation of the arrhythmia detector on the Tensilica base processor; initial design profiling leads to the configuration parameters shown in Table 6.1. Note that even though classification involves dot-product computations, including multipliers (MUL16 and MUL32) in the base processor leads to a higher energy consumption. We will see ahead in Sec. 6.3.2 that this energy increase is due to the overhead of fetching high-dimensional data for the multipliers.

Table 6.1: Xtensa custom processor configuration

Parameter	Configuration
Instruction width	24 bits
Pipeline length	5 stages
Pipeline type	Uniscalar
General-purpose registers	16
ALUs	1
Branch units	1
Core frequency	10 MHz
Instruction RAM	2 kB
Data RAM	4 kB
Datapath width	32 bits
Disabled Options	
Multipliers (MUL32/MUL16), Viterbi unit, single-cycle MAC, zero-overhead loop, normalized shift, min/max unit, ICache/DCache associativity	

Table 6.2: Software energy (per test vector) for preprocessing and feature extraction on the base Xtensa processor core.

Computational step	**Energy/test vector**
Pre-processing segmentation	84.02 μJ
Morphology feature extraction	2.61 μJ
Wavelet feature extraction	29.28 μJ

The energy profiling results for the preprocessing and feature-extraction steps are shown in Table 6.2 (results are shown at the operating frequency of 10 MHz and supply voltage of 1.2 V). An FV is derived every heart beat and consumes 84.02 μJ for segmentation, which includes the process of isolating individual heartbeats along with the filtering of noise and other interference sources. Subsequently, 2.61 μJ and 29.28 μJ of energy is consumed for morphological and wavelet feature extraction, respectively.

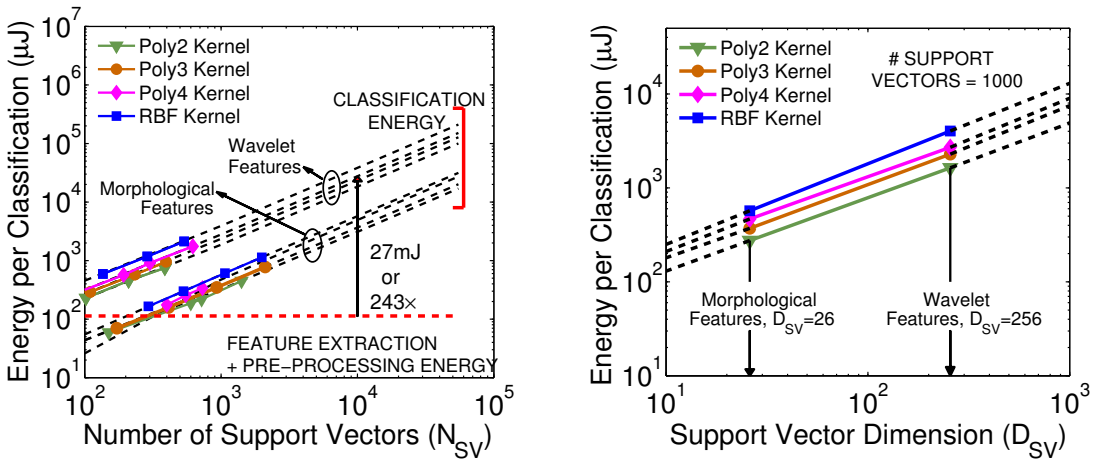


Figure 6.5: Arrhythmia detector: (a) classification energy scales with N_{SV} and thus dominates that of feature extraction, and (b) classification energy scales with D_{SV} (N_{SV} and D_{SV} represent classification complexity.).

Figs. 6.5(a) and 6.5(b) show the energy of classification versus N_{SV} and D_{SV}, respectively. Both the number and dimensionality of the SVs are representative of the model complexity. It can be seen that due to energy scaling, classification energy rapidly dominates that of feature extraction. $N_{SV} = 10,000$, using wavelet features and a fourth-order polynomial kernel, for instance, leads to an energy consumption ratio of 941:3:1 for classification, preprocessing, and feature extraction, respectively. At $N_{SV} = 100,000$, the ratio increases to 5172:3:1. As described in Sec. 6.2.2, to avoid compromising accuracy, the arrhythmia application generally requires complex models, causing the classification energy to be dominant.

The output of the Xtensa profiling tool for classification computations is shown in Table 6.3. The dot-product computations required in Eq. (3.1) are shown in bold (sprod_ns and kernel). These functions correspond to the N_{SV} and D_{SV} analysis presented in Sec. 6.2.2, and constitute over 70% of the computations. Thus, classification is the energy-intensive computational step and is targeted next for hardware specialization through custom instructions and a coprocessor.

Table 6.3: Tensilica code profiler output for the SVM classifier.

MORPHOLOGICAL FEATURES, D_{SV}=26, N_{SV}=10,000, FREQUENCY = 10 MHz				
Function name	Percentage time (%)	Self secs.	Cumulative secs.	Number of cycles
sprod_ns	**59.12**	**0.09**	**0.09**	**925748**
kernel	**11.67**	**0.02**	**0.11**	**182738**
smult_s	7.95	0.01	0.12	124487
add_vector_ns	6.54	0.01	0.13	102408
WAVELET FEATURES, D_{SV}=256, N_{SV}=10,000, FREQUENCY = 10 MHz				
Function name	Percentage time (%)	Self secs.	Cumulative secs.	Number of cycles
sprod_ns	**63.30**	**0.57**	**0.57**	**5688984**
kernel	**12.89**	**0.12**	**0.69**	**1158467**
smult_s	8.34	0.08	0.77	753113
add_vector_ns	6.60	0.06	0.83	593164

[2] *Self secs.* is the number of seconds accounted for by a particular function alone. *Cumulative secs.* is a running sum of the number of seconds accounted for by a function and those listed above it.

6.3.2 Custom-instruction Based Platform

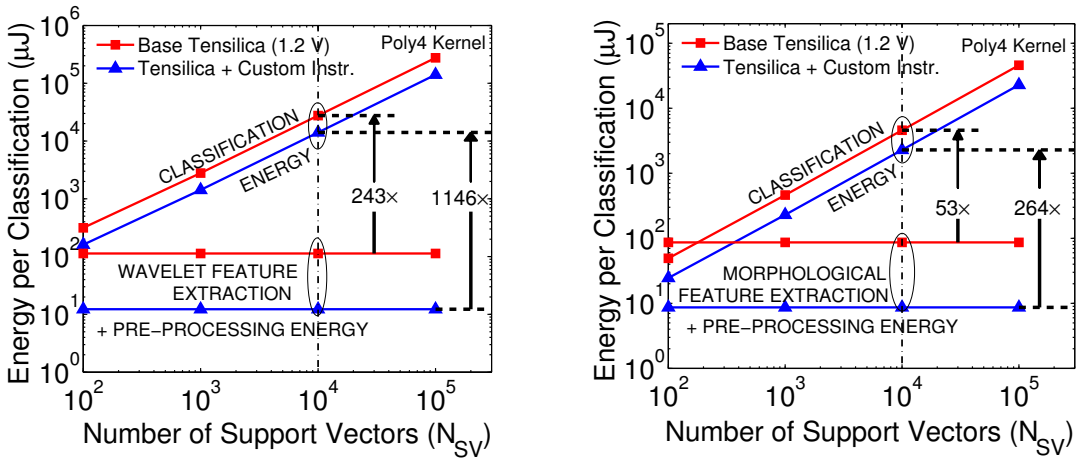
In this section, we explore the use of custom instructions as a hardware-specialization option. We primarily focus on the classifier due to its importance in determining the total energy. We find that the custom instructions are insufficient for achieving significant classifier energy savings. This is due to the overheads that remain for fetching high-dimensional data from memory. Thus, the energy reductions achievable through the use of custom instructions for classification are limited due to the large number of operands involved in the dot-product computation.

The Tensilica Xpress compiler [218] is used to optimize the software implementation. This involves an automatic design-space exploration of the potential custom instructions. SVM classification, wavelet transform with Daubechies wavelets of order four, and mor-

Table 6.4: Custom instructions from the Xpress compiler at N_{SV}=10,000.

Custom instruction	% of total instr.
Preproc. + Morphology	
fusion.nop.loopgt.extui	32.6
fusion.abs.add8x8.extui	11.4
fusion.nop.neg8.extui	9.0
Preproc. + Wavelet	
fusion.ssl.mul8x16_0.extui	21.9
fusion.l8rzl.extai	17.2
fusion.movt.z.extui	8.4
Classification	
fusion.movi8x16.extui	13.1
fusion.add.sdd8x16.simcw.extui	5.4
fusion.sll.sub16x16_0.extui	5.3

phological feature-extraction functions (including threshold selection and QRS isolation) are chosen for implementation as custom instructions. We perform Xpress synthesis using FLIX, Fusion, and SIMD instructions provided by Tensilica [218]. These design options provide optimization techniques, including automatic vectorization in the custom processor. Fusion instructions enable the lowest-energy implementation. Table 6.4 shows the top three custom instructions obtained for the feature-extraction and classification computations. The number of calls to each custom instruction is also shown as a percentage of the total instructions. We observe that there is no commonality in the custom instructions across feature-extraction and classification computations. Figs. 6.6(a) and 6.6(b) show the classification energy after a custom-instruction based optimization (corresponding to wavelet and morphological features, respectively). The energy consumption is still substantially dominated by classification. At $N_{SV} = 10,000$, for a fourth-order polynomial kernel, the ratios of energy consumption for classification, preprocessing, and feature-extraction computations are 4432:3:1 and 720:3:1, for the wavelet and morphological features, respectively.



(a) (b)

Figure 6.6: Custom-instruction based implementation results in only modest energy improvement: classification energy still dominates preprocessing + feature extraction (a) by 1146× for wavelet features, and (b) by 264× for morphological features.

Table 6.5 summarizes the resulting energy savings obtained through custom-instruction based optimization. This optimization leads to roughly 10× energy improvement for the preprocessing and feature-extraction operations. However, optimization of classification computations leads to roughly only 2× reduction in energy. The limited energy reduction for classification is due to the large amount of data, which need to be fetched from memory, involved in the dot-product computations (*i.e.*, large N_{SV} and D_{SV}) [219, 221]. This limitation is not adequately addressed by custom-instruction based optimization. To gain further intuition, consider the following equation [219], which is used to rank candidate templates for custom-instruction based implementation:

$$\text{Priority} = \frac{\text{OT}}{\max(In - \omega, 0) + \max(Out - \sigma, 0) + \rho}.$$

In the above equation, *OT* is the fraction of the total execution time of the original program spent in the template, *In* and *Out* are the number of inputs and outputs of the template, respectively, ω is the number of inputs that can be encoded in the instruction, σ is the number of outputs that can be encoded, and ρ is the number of cycles required by the template

Table 6.5: Speedup and energy saving using custom instructions with a base Xtensa processor running at V_{dd} = 1.2 V and 10 MHz, with $N_{SV} = 10,000$.

Processor Configuration		**Preproc. + Morphology**	**Preproc. + Wavelet**	**Classification**	
				$D_{SV} = 26$	$D_{SV} = 256$
Base Xtensa	No. Cycles	27.56k	36.04k	1.57M	8.99M
	Energy (μJ)	86.62	113.29	4935.2	27574
Base Xtensa + Custom Instr.	No. Cycles	2.75k	3.91k	0.78M	4.58M
	Energy (μJ)	8.65	12.28	2455.3	14068
Energy Improvement		**10.01**×	**9.23**×	**2.01**×	**1.96**×

when implemented as a custom instruction. For custom instruction candidate templates for classification computations, *OT* has a large value of 0.70 (according to Table 6.3). However, the high-dimensional input vectors (corresponding to D_{SV}) and the large number of cycles ρ required to fetch a high-order decision model (corresponding to N_{SV}) reduce the priority as a potential choice for custom-instruction based implementation. Choosing custom instructions for the dot-product computation based on an alternate priority function would still result in sub-optimal energy savings. This is because the processor architecture would limit the data width in the classifier custom instructions, necessitating additional cycles for load-store operations. Thus, system memory overheads offset the benefits of custom-instruction based speedup in classification.

The use of a vector processor core for handling high-dimensional input data can thus provide a potential solution for the classification computations. However, since the application data are not inherently vector in nature, such an architecture incurs unnecessary overheads for the general-purpose computations required, namely the feature-extraction computations required across clinical applications. Rather, the representation of data in a vector form is a specific transformation introduced by the classification framework. Thus, the complexity and associated energy overheads [223] incurred by a vector-processor based implementation of the preprocessing and feature-extraction steps are best avoided for a platform-level design. To exploit both the canonical computations and data structuring

required in the classification framework, we next turn our attention to coprocessor based hardware specialization.

6.3.3 Coprocessor Based Platform

In this section, we discuss a coprocessor based specialization that allows the data structures used by the classifier to be efficiently handled, yielding substantial energy savings. Further, this degree of specialization raises the opportunity for microarchitectural optimizations based on parallelism, which can be readily exploited in the computation.

The architecture proposed in Fig. 6.7 aims to take advantage of the structure of the signal-analysis algorithms where the classifier energy can be dominant. In such cases, a high degree of flexibility is primarily required in the preprocessing and feature computations. The need for flexibility arises due to the range of preprocessing and feature computations involved in various applications. For instance, morphological and wavelet features are employed in arrhythmia detection [206, 212], proteomic classification [224], and heart-rate estimation [225]; spectral features are employed in brain-machine interfaces [92], sleep disorder analysis [164], *etc.*

Thus, a general-purpose processor is employed for preprocessing and feature computation, while an optimized coprocessor is employed for kernel based SVM classification. The feature-extraction computations are optimized through custom instructions, providing significant energy savings, as shown in Table 6.5. These computations involve floating-point operations in the Tensilica CPU, incurring somewhat higher energy than a fixed-point implementation. However, as explained in the previous section, in the targeted applications, the contribution of the feature-extraction energy to the overall processor energy is very small. Thus, we focus on optimizing the coprocessor. Here, in addition to hardware specialization, circuit and microarchitectural enhancements aim to achieve minimum-energy operation [198] through voltage scaling and parallelism, whereby the throughput constraints for real-time detection can be met. In addition to energy efficiency, however, the need for

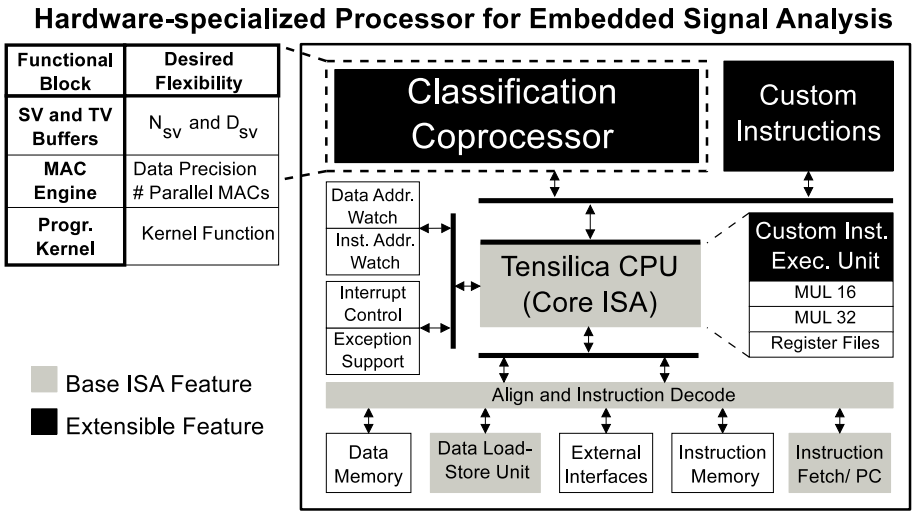


Figure 6.7: General architecture of a processor for embedded signal analysis.

selective flexibility is also recognized so that the classification needs across a wide range of signal-analysis algorithms can be supported. For example, the rate of processing mass spectrometry data [224] could be different from ECG signals [206, 212]. For these applications, N_{SV}, D_{SV}, data precision, as well as the kernel functions will also be different. The coprocessor thus introduces this flexibility through a precision-scalable multiplier. It also yields programmability in the classification model, computation precision, and the choice of kernel transformation function (these aspects are summarized in the block diagram of Fig. 6.7).

6.4 Low-energy Classification Coprocessor

In this section, we describe the architecture of the classification coprocessor in further detail.

6.4.1 Coprocessor Microarchitecture

Fig. 6.8 shows the architecture and layout of the classification coprocessor. It has three major functional blocks: (1) SV and test vector (TV) buffers, (2) MAC engine, and (3)

programmable polynomial kernel core. Following offline training, SVs are loaded into the SV preload buffers. The TVs, produced through feature extraction by the general-purpose processor, are loaded into the TV line buffer. TVs and SVs are then fed dimension-by-dimension to the MAC array in order to perform the dot-product operations in Eq. (3.1). Readout from these buffers is optimized using a multiplexer based array decoder. Hardware parallelism is employed through an array of MAC units: *MAC_1 to MAC_N*, each of which is associated with an SV preload buffer. The coprocessor operates on integer data. Once multiplication over all the dimensions is complete, the dot-products are multiplexed to the kernel transformation block, where a second-, third- or fourth-order polynomial transformation is computed. Furthermore, a CORDIC module in the kernel block would potentially accommodate additional transformation functions, such as RBF, sigmoid, *etc.* The results are scaled and summed by a final accumulator whose output sign determines the classification result.

6.4.2 Voltage Scaling and Parallelism

In this section, we describe the energy optimization pursued for the coprocessor through voltage scaling. Since the dot-product derivation (in the MAC array) dominates the computation, we focus on optimizing its energy.

The total energy is determined primarily by the sum of active-switching (E_{act}) and subthreshold leakage (E_{lk}). The reduction in E_{act} due to V_{dd} scaling is opposed by the increase in leakage energy (due to longer resulting leakage-current integration time T_{MAC}). The energy-optimal point thus typically occurs in the subthreshold region, since here the circuit speed begins to degrade rapidly [198]. Although this implies that energy optimization leads to low circuit performance, computational throughput constraints can be efficiently met if the required computations can be performed in parallel without imposing substantial overheads due to parallelization [226]. We can thus exploit the parallelism possible in the classifier dot-product computation (*i.e.*, MAC array) to achieve minimum-energy operation

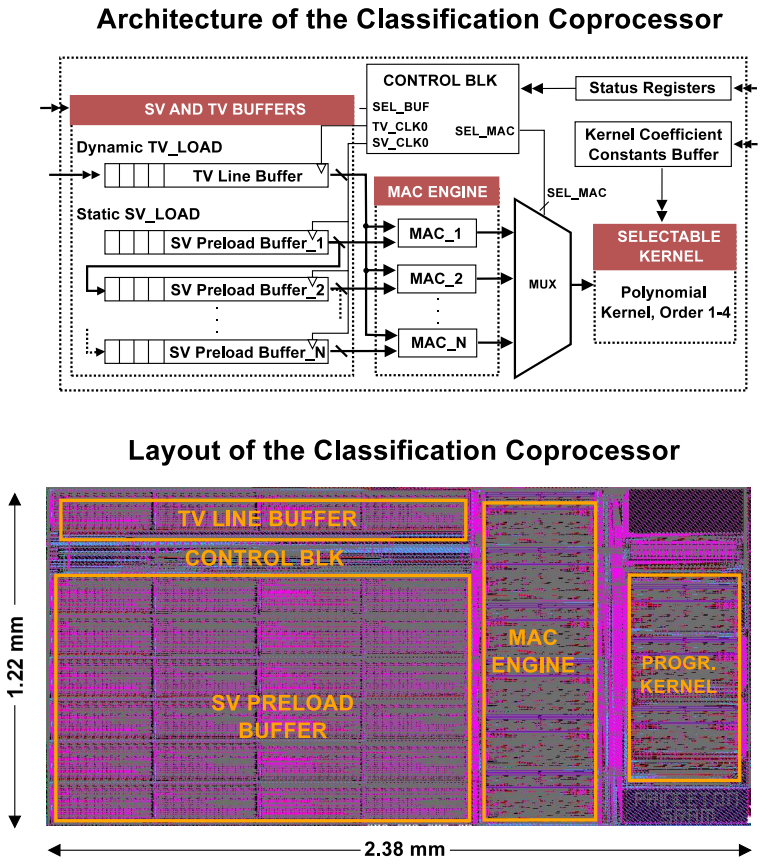


Figure 6.8: The architecture and layout of the classification coprocessor designed in an FD-SOI process. The coprocessor architecture comprises an array of MAC units to compute the dot-product of SVs and TVs. The output is then transformed by a kernel function in order to evaluate the classification result.

for real-time signal detection. To do this, we first determine the minimum-energy V_{dd} of a MAC unit. We then determine its performance at this V_{dd} (*i.e.*, seconds per MAC operation, T_{MAC}). The total rate of MAC operations ($R_{T.MAC}$) required in the classifier computation [of Eq. (3.1)] is given by

$$R_{T.MAC} = \lceil N_{SV} \times D_{SV} \times R_{CLASS} \rceil, \tag{6.1}$$

where R_{CLASS} is the classification rate. The required parallelism is then $R_{T.MAC} \times T_{MAC}$. For the application considered, the $R_{T.MAC}$ required ranges from 2.7M to 7.7M MACs/second [19, 25].

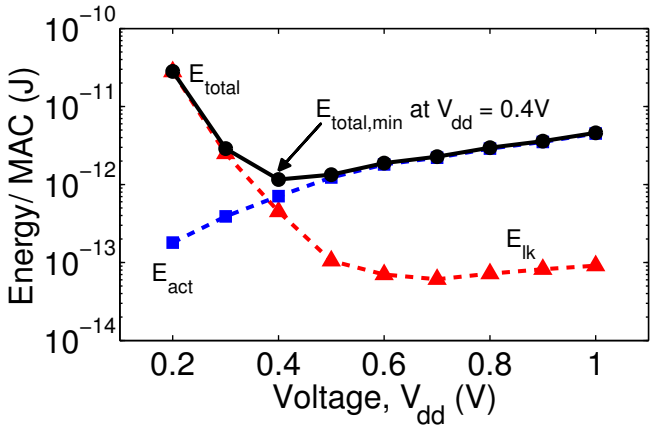


Figure 6.9: The E_{act} and E_{lk} profiles for a MAC unit with the minimum total energy occurring at $V_{dd} = 0.4V$.

Fig. 6.9 shows the E_{act} and E_{lk} of a MAC unit (based on a transistor-level simulation) implemented in the target 150nm FD-SOI CMOS process (described further in Sec. 6.4.4). The total energy, E_{total}, is minimized at a V_{dd} of 0.4V, which is in the subthreshold region for the technology. Fig. 6.10 shows the performance achieved by a MAC unit as V_{dd} is scaled. Under worst-case process and temperature conditions (*i.e.*, low temperature in subthreshold), the maximum frequency at the minimum-energy V_{dd} is 520 kHz (*i.e.*, $T_{MAC} = 1.92\,\mu$s). The level of parallelism required is thus 6 to 15 MAC units. Fig. 6.9, however, shows that the energy minimum is shallow, particularly if V_{dd} is increased slightly. For instance, to increase the MAC performance by a factor of three (in order to cover the target $R_{T.MAC}$ range), V_{dd} must be increased by less than 50 mV (based on Fig. 6.10), causing a negligible increase in total energy (based on Fig. 6.9). We thus optimize for the lower performance (by employing 6 MAC units) and use voltage scaling, with minimal impact on the optimization, to elevate the performance when required.

6.4.3 Circuit-level Optimization

In this section, we describe how the scalability desired in the classification coprocessor is achieved.

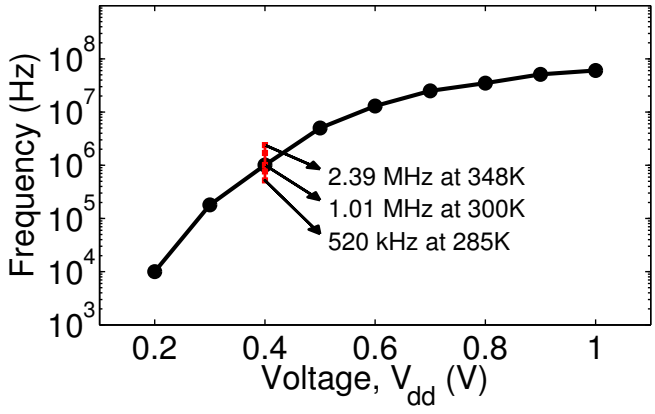


Figure 6.10: The operating frequency at $V_{dd} = 0.4V$ is 520 kHz at 285K (low temperature is slowest in subthreshold).

SV and TV buffers. The energy of the buffers is optimized for read operations since the SVs are loaded infrequently (*i.e.*, only when a new classification model is required). The coprocessor buffers support a $D_{SV} \times N_{SV}$ of 64. If additional storage is required to represent the classification model, the control block permits expansion by allowing up to 16,384 write sequences from the processor cache or from off-chip memory to the local buffers. As an example, $4,095$ SVs and 256 feature dimensions can be supported, along with any other combination that results in the same product.

Variable-precision MAC. Due to the wide range of SVs and feature dimensions across applications, the precision requirements of the classifier computation are variable. Several approaches for scalable-precision multipliers have been reported, *e.g.*, those in [198]. The approach used here exploits the efficiency of the Booth encoding algorithm [227].

Fig. 6.11 shows the architecture of the variable-precision MAC unit. In the MAC unit, the *BOOTH ENC* blocks compute the partial products based on the select bits of the multiplier (y). The shifted partial products are output as PP_i, $i \in [0, 5]$. This allows a maximum precision of 12 bits for the input operands (corresponding to six partial products). In Fig. 6.11, PP_0 and PP_1 are the partial products used when precisions of 12 and 10 bits, respectively, are required; otherwise the precision is 8 bits. The carry-bypass adders (CBAs) consist of M = 4-bit full adder chains, and N represents the total input bit-width of each

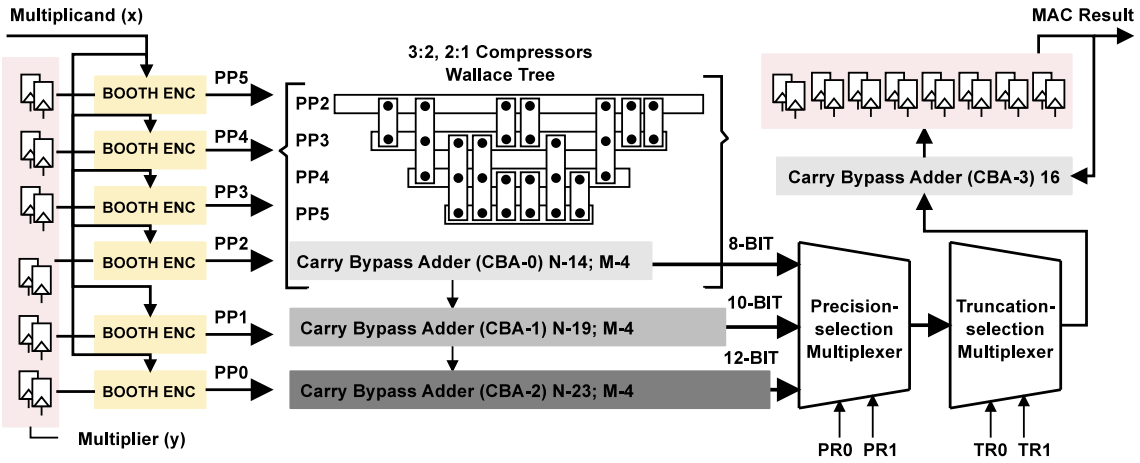


Figure 6.11: The variable-precision MAC unit. Partial product additions can be terminated at CBA-0/1/2 to scale the precision for 8/10/12-bit inputs.

adder. The common partial products required for the 8/10/12 precision bits are added using 3:2 and 2:1 compressors in a Wallace tree. The outputs of *CBA*-0/1/2 are read out via a precision-selection multiplexer (for 8/10/12-bit precision, respectively). The unused CBAs can be power-gated. Fig. 6.12 shows the energy reductions due to precision scaling. Although the minimum-energy V_{dd} remains the same, scaling the precision from 12 to 8 bits reduces the energy per multiplication by 17.6%. Following precision selection, the output of the multiplier has either a 24-, 20-, or 16-bit output. The truncation-selection multiplexer selects a level of truncation (to 12, 10, or 8 bits, programmable via the status register). The output of the truncation-selection multiplexer is accumulated into an output register using a 16-bit final CBA.

MAC delay estimation. Based on the proposed MAC architecture (Fig. 6.11), the critical path delay through a MAC unit can be estimated as follows:

$$\begin{aligned} T_{delay} &= T_{CQ} + T_{BOOTH} + T_{WAL} + T_{CBA,19\text{-}4} + T_{CBA,23\text{-}4} \\ &\quad + 2T_{AND} + 3T_{MUX} + T_{CBA,16\text{-}4} + T_{SU}, \end{aligned}$$

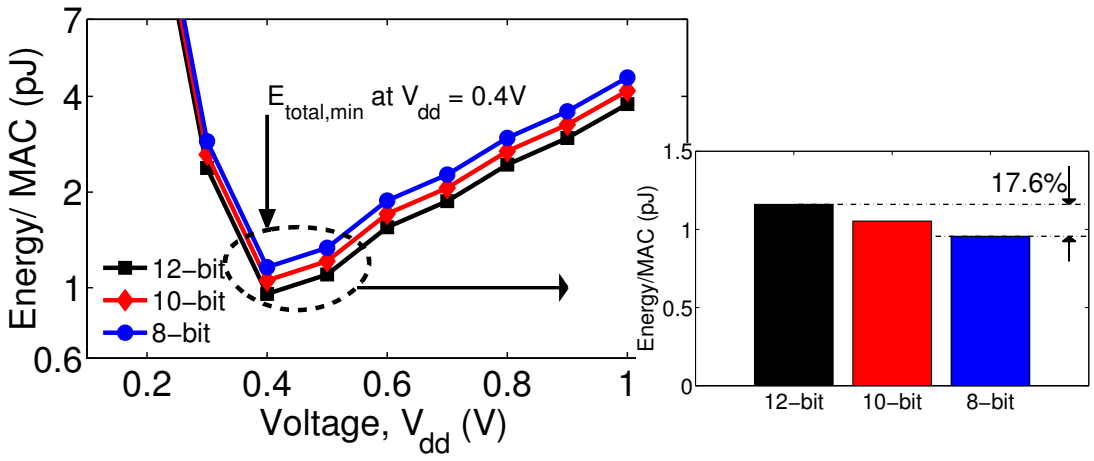


Figure 6.12: Scaling precision of input data enables a second level of energy optimization. The figure shows energy savings of 17.6% while scaling the data-representation precision from 12 to 8 bits.

where $T_{CBA,N\text{-}M}$ is the delay of a CBA, which has a segment length of M and an operator bit length of N, T_{BOOTH} is the delay through the Booth encoder unit, T_{WAL} is the delay through the Wallace-tree compressor chain, T_{CQ} and T_{SU} are the clock-to-output and clock setup delays, respectively, and T_{AND} and T_{MUX} are the AND gate and multiplexer delays, respectively. Further, these delays can be simplified using a sum of delays through basic sub-blocks as follows:

$$\begin{aligned} T_{CBA,N\text{-}M} &= T_{SU} + MT_C\left(\frac{N}{M} - 1\right)T_{MUX} + (M-1)T_C \\ &\quad + T_S \\ T_{BOOTH} &= T_{MUX} + 4T_{NOR} + T_{CBA,12\text{-}4} \\ T_{WAL} &= 3T_S + T_{CBA,14\text{-}4}, \end{aligned}$$

where T_C and T_S are the delays for the carry and sum paths in a full adder, respectively, and T_{NOR} is the NOR gate delay.

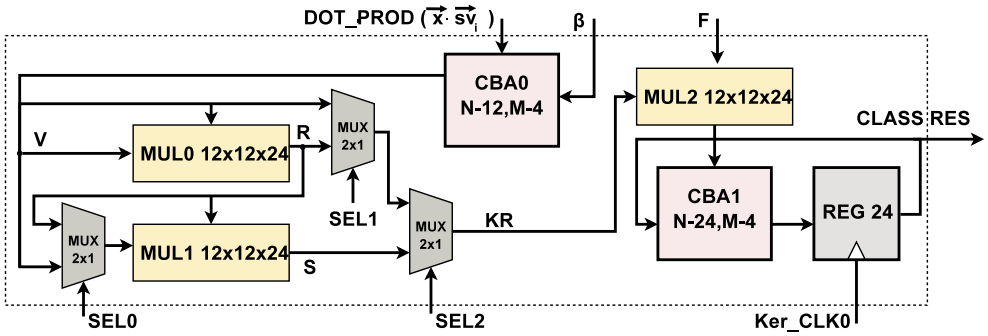


Figure 6.13: The programmable kernel enables choice among kernels of degree one through four.

The delay through the critical path of the MAC unit can thus be estimated systematically. Estimating the MAC delay based on the performance of the mentioned sub-blocks thus facilitates rapid estimation of the maximum operating frequency, level of parallelism, and hence the associated system parameters, thereby overcoming the need for extensive transistor-level simulations in the early phases of system design.

Flexible polynomial kernel. A polynomial kernel can be selected to transform the dot-product output from the MAC engine. The flexible kernel module comprises two 12×12×24 multipliers to support polynomial transformations of order two through four. Only one such multiplier is needed for a second-order polynomial function. Going from a second-order to a third-order polynomial kernel, however, incurs the cost of using an additional 12×12×24 multiplier. Further, the difference between a fourth-order and a third-order polynomial function is only an additional array of multiplexers. These aspects are summarized in Fig. 6.13.

6.4.4 Choice of Technology

Due to the modest performance requirements of typical sensing applications (owing to, for instance, the relatively low bandwidth of physiological signals), employing a technology that is aggressively optimized for low leakage is beneficial. As an example, for arrhythmia detection, a performance on the order of 5 million MACs/second is required. The tech-

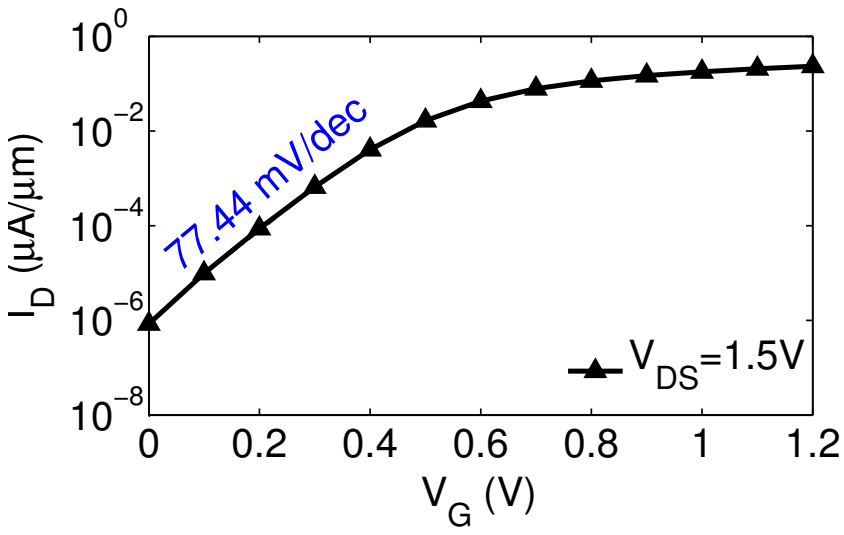


Figure 6.14: The FD-SOI device has a steep sub-threshold slope to minimize leakage and maintain high transistor on-to-off ratios in subthreshold CMOS gates.

nology we use for this study is thus a 1.5 V 150 nm ultra low-leakage FD-SOI CMOS process [228]. FD-SOI allows the technology to exhibit reduced process variations due to a reduction in random-dopant fluctuations and gives the transistors steep subthreshold slopes; the I_d-V_{gs} characteristic for the devices is shown in Fig. 6.14. Additionally, the devices are designed to have high threshold voltages to reduce the leakage current (*i.e.*, $V_{t,N} = 0.65V$, $|V_{t,P}| = 0.53V$) [228].

6.5 Results and Analysis

In this section, we present post-layout simulation results of the coprocessor.

Table 6.6 summarizes the impact of the architectural optimizations considered (which include feature extraction on the Tensilica processor and classification using custom instructions or a coprocessor). A 2× improvement in the energy consumption per SV dimension is obtained while going from an implementation on a base Tensilica processor (which consumes 11.85 *n*J) to a design with custom instructions (which consumes 5.89 *n*J). The limited energy reductions, as mentioned in Sec. 6.3.2, is due to the high-dimensional input

Table 6.6: Energy per SV dimension

Tensilica (at 1.2V)	**Tensilica + Custom Instr.**	**Tensilica + Custom Instr. + Coprocessor**		
		$V_{dd}^{COPROC} = 1.2V$	$V_{dd}^{COPROC} = 0.6V$	$V_{dd}^{COPROC} = 0.4V$
11.85 nJ - **1**×	5.89 nJ - ↓**2**×	22.84 pJ - ↓**519**×	5.59 pJ - ↓**2119**×	5.31 pJ - ↓**2231**×

data required for classification. More aggressive specialization, through the use of a coprocessor, leads to a 519× energy reduction over the base case. Subsequent voltage scaling, taking advantage of the parallelism possible in the coprocessor implementation, leads to energy reductions of 2119× and 2231× for V_{dd} corresponding to 0.6 V and 0.4 V, respectively.

6.5.1 Coprocessor Energy Measurements

In this section, we present an analysis of the energy consumption of the classification coprocessor versus N_{SV} and D_{SV}. Further, we also quantify the energy reductions achievable through voltage and precision scaling.

Energy versus N_{SV} and D_{SV}. We perform energy measurements on the post-layout extracted netlist at various values of N_{SV} for the wavelet and morphological features. Fig. 6.15 shows the scaling in the energy consumption versus the number of SVs for the classification computation. The energy for classification scales roughly linearly with N_{SV}. A similar behavior is observed with D_{SV} (as shown in Fig 6.16, where the energy numbers are provided at $V_{dd} = 1.2$ V). Fig. 6.15 shows results for a fourth-order polynomial kernel and spectral wavelet features ($D_{SV} = 256$) for arrhythmia detection. At 100,000 SVs, the optimization of a base Tensilica processor with custom instructions leads to an energy reduction by 1.96× (Sec. 6.3.2). For the FD-SOI coprocessor based design, energy reductions by 228× are achieved at a supply voltage of 1.2 V. Voltage and precision scaling applied to the coprocessor leads to further energy reductions. For instance, as shown in Fig. 6.15, classification computations after wavelet feature extraction ($D_{SV} = 256$) consume 12.00-

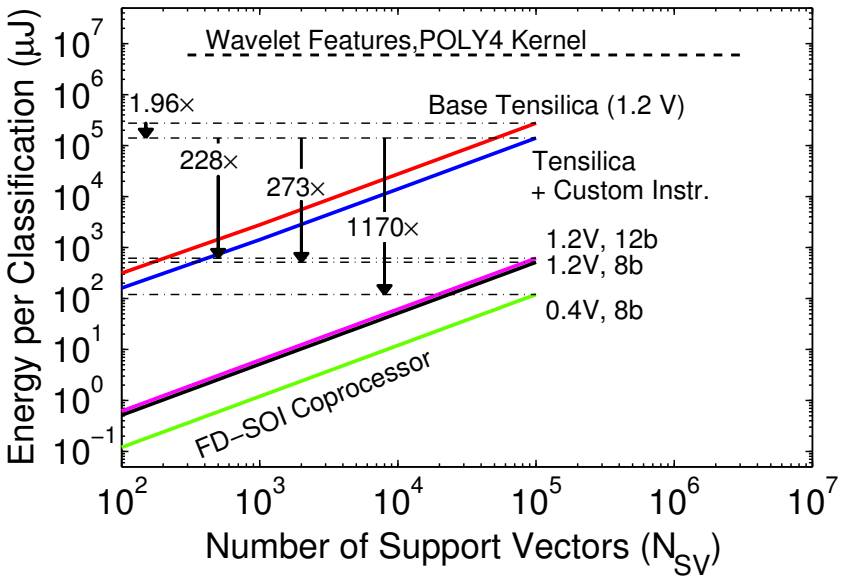


Figure 6.15: The classification coprocessor enables energy reductions of 228× at 1.2 V; energy reductions are increased to 1170× at V_{dd} = 0.4 V and 8 bits of precision.

120.05 μJ using 10,000-100,000 SVs at 8 bits of data precision and a supply voltage of 0.4 V. This represents 1170× lower energy than that of a Tensilica processor with custom instructions alone. Classification computations after morphological feature extraction, at the same precision and voltage levels, consume 10.24-24.51 μJ of energy, which is 1548× smaller than one based on a custom-instruction based design. Detailed analysis results for the voltage- and precision-scaling experiments are presented next.

Energy versus V_{dd}. E_{act} and E_{lk} measurements from the coprocessor are shown in Fig 6.17, demonstrating the benefit of voltage scaling. The results are shown at 12 bits of data precision. A second-order polynomial kernel is used for the simulations. On an average, for a data precision of 12 bits, E_{act} accounts for 98.1%, 95.1%, and 71.2% of the total energy at supply voltages of 1.2 V, 0.6 V, and 0.4 V, respectively. Voltage scaling thus enables energy reduction by up to 77%.

Energy versus precision. Table 6.7 shows the E_{act} and E_{lk} measurements from post-layout simulation of the coprocessor using 8 bits of data-representation precision. A second-order polynomial kernel is used for the results shown. On an average, for a

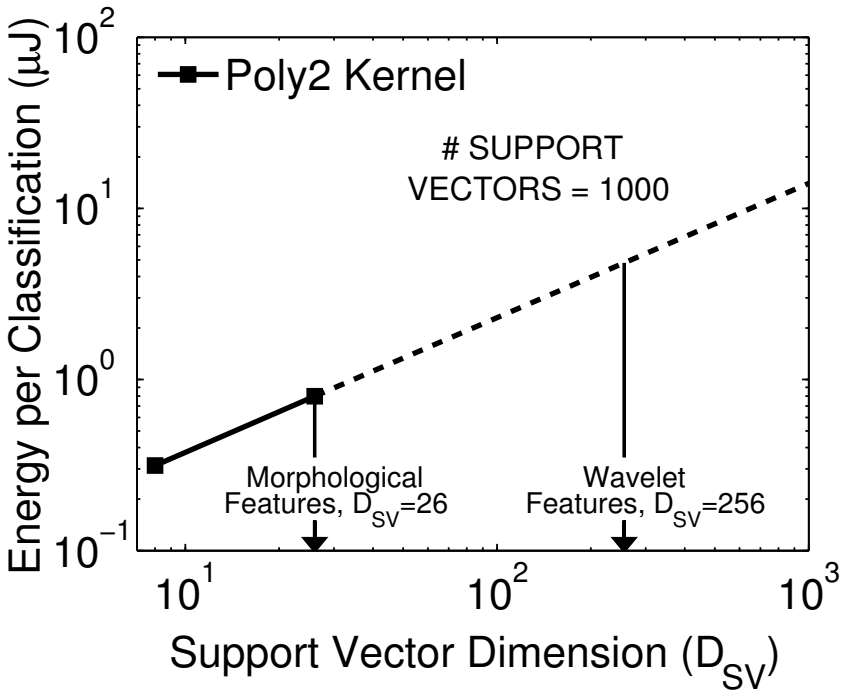


Figure 6.16: Coprocessor energy versus D_{SV} per TV (at 12-bit data precision).

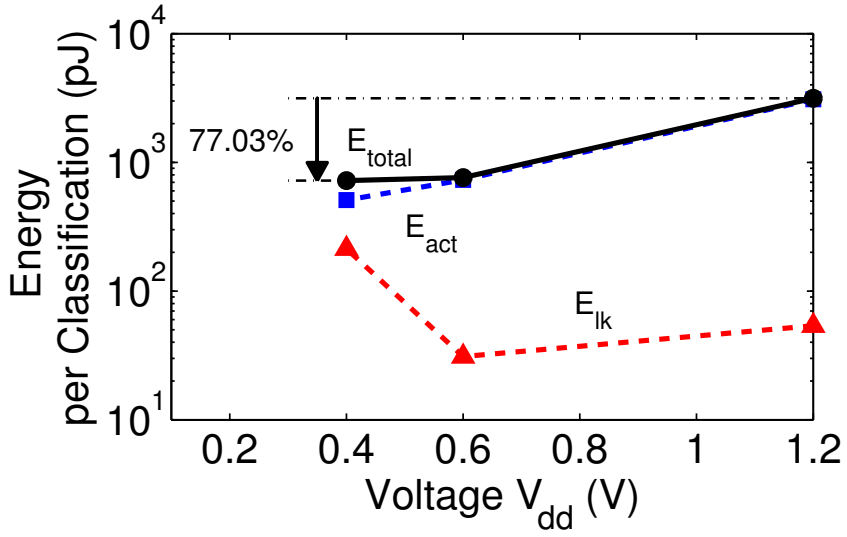


Figure 6.17: Coprocessor energy versus V_{dd} per TV (at N_{SV} = 10, D_{SV} = 8, and 12-bit data precision). V_{dd} scaling enables energy reduction by up to 77%.

Table 6.7: Precision scaling enables up to 9.25% reduction in the coprocessor energy.

V_{dd} (V)	D_{SV}	N_{SV}	Precision	E_{act} (pJ)	E_{lk} (pJ)	E_{total} (pJ)	f_{op}, T=287K
1.2	8	10	12 bit 8 bit	3089.1 2799.8	53.7 52.4	3142.8 2852.2 } **9.25%↓**	10 MHz
0.6	8	10	12 bit 8 bit	730.5 667.2	31.0 25.1	761.5 692.3 } **9.09%↓**	2 MHz
0.4	8	10	12 bit 8 bit	508.7 457.1	213.2 199.2	721.9 656.3 } **9.09%↓**	550 kHz

Table 6.8: Coprocessor energy scaling with respect to the kernel function.

Specification	E_{act} (pJ)	E_{lk} (pJ)	E_{total} (pJ)	Kernel Order
V_{dd}=1.2V, N_{SV}=10, D_{SV}=8, 12-bit precision	3078.7	64.1	3142.8 } **30.92%↓**	Poly2
	4036.2	78.2	4114.4 } **7.56%↓**	Poly3
	4344.4	81.1	4425.5	Poly4
V_{dd}=1.2V, N_{SV}=10, D_{SV}=8, 8-bit precision	2799.8	52.4	2852.2 } **33.57%↓**	Poly2
	3738.7	70.9	3809.6 } **4.12%↓**	Poly3
	3914.6	74.7	3989.3	Poly4

data precision of 12 bits, E_{act} accounts for 98.1%, 95.2%, and 71.3% of the total energy at supply voltages of 1.2 V, 0.6 V, and 0.4 V, respectively. Consequently, performing computations at a data-representation precision of 8 bits enables an overall 9.25%, 9.09%, and 9.09% reduction in total energy as compared to an implementation that relies on 12 bits of data precision [most savings come from a 17.6% reduction from the MAC unit alone (see Fig. 6.12)].

Energy versus kernel selection. Table 6.8 shows the energy consumption for various classifier kernels at a data precision of 12 and 8 bits. Since the kernel transform is not the dominant computation, the energy scaling is modest.

On an average, going from a second- to a third- to a fourth-order polynomial costs 30.92% and 7.56% extra energy for 12 bits and 33.57% and 4.12% for 8 bits of data-

Table 6.9: Energy (per TV) and area of the sub-blocks.

Meas. Condition			**Total**	**BUF**	**MAC**, pJ	**KER**, pJ	**CNTRL**
V_{dd}	D_{SV}	N_{SV}	pJ	pJ	(% Total)	(% Total)	pJ
8-BIT DATA PRECISION							
1.2 V	8	10	**2852.2**	38.8	1511.8 (**53.0**)	900.9 (**31.6**)	400.7
0.6 V			**692.3**	18.9	360.7 (**52.1**)	220.2 (**31.8**)	92.5
0.4 V			**656.3**	18.5	343.8 (**52.4**)	203.9 (**31.1**)	90.1
12-BIT DATA PRECISION							
1.2 V	8	10	**3142.8**	40.2	1640.5 (**52.2**)	990.0 (**31.5**)	472.1
0.6 V			**761.5**	19.3	402.8 (**52.9**)	242.2 (**31.8**)	97.2
0.4 V			**721.9**	20.1	383.3 (**53.1**)	223.8 (**31.0**)	94.7
Area (in mm^2)			**2.90**	**1.62**	**0.61**	**0.65**	**0.02**

representation precision, respectively. The incremental change in energy between a third- and a fourth-order polynomial kernel is due to the optimization enabled by the programmable kernel architecture [19, 25].

6.5.2 Sub-block Energy Measurements

Table 6.9 shows the computational energy contributions for the coprocessor sub-blocks during online classification at a data-representation precision of 8 and 12 bits. In the table, the energy consumed in the TV and SV buffers, the MAC array engine, kernel, and the control block are shown in the BUF, MAC, KER, and CNTRL columns, respectively. The MAC engine consists of six MAC units and the KER block consists of three MUL 12×12×24 multipliers, which are sub-blocks within a MAC unit. As shown, the MAC+KER energy dominates (~84%), validating the need to focus on its energy through the optimization discussed in Sec. 6.4.2. Although the buffers dominate the transistor count, their low energy contribution shown in the table is due to the low leakage enabled by the choice of technology. The buffers have a very weak influence on the minimum energy point of the coprocessor due to the low activity factor and low leakage energy.

6.5.3 Processor-level Energy Measurements

In this section, we present energy measurements for the entire processor. The architecture, which employs custom instructions for feature extraction and a coprocessor for classification, achieves over two orders of magnitude energy reductions as compared to an implementation, which employs only custom instructions for the entire computation.

Energy versus N_{SV}. In this section, we show energy scaling results versus N_{SV} for the full signal detection process (which, along with classification, includes the energy numbers for a custom-instruction based implementation of preprocessing and feature-extraction computations).

Fig. 6.18 shows simulation results for a fourth-order polynomial kernel and spectral wavelet features ($D_{SV} = 256$). Significant energy reductions are observed for the total detection process. For instance, using $N_{SV} = 10,000$, the total detection energy at 1.2 V and 12 bits of data precision is 73.98 μJ. It is reduced to 63.69 μJ using 8 bits of precision at 1.2 V. Voltage scaling applied to this optimized coprocessor configuration results in a total detection energy of 24.29 μJ at 0.4 V (this is about 580× lower than an implementation using a base Tensilica processor, which consumes 14.08 mJ at $V_{dd} = 1.2$ V).

Computational-energy contributions. Fig. 6.19 shows the proportions of energy consumption for the preprocessing, feature extraction, and classification computations following the various optimizations. The design space ranges from a full-software, base Tensilica implementation to a coprocessor based architecture. The percentages are shown for $N_{SV} = 20,000$. It is observed that even after a custom-instruction based optimization, the classification computations dominate feature extraction and preprocessing (less than 1% of the total energy). After an optimization through the use of the FD-SOI coprocessor, however, the classification energy is reduced substantially. The energy ratio is 1.8:1 for classification and preprocessing + feature-extraction computations. However, at $N_{SV} = 100,000$, the corresponding ratio is 9:1 (the total energy consumption for the associated computa-

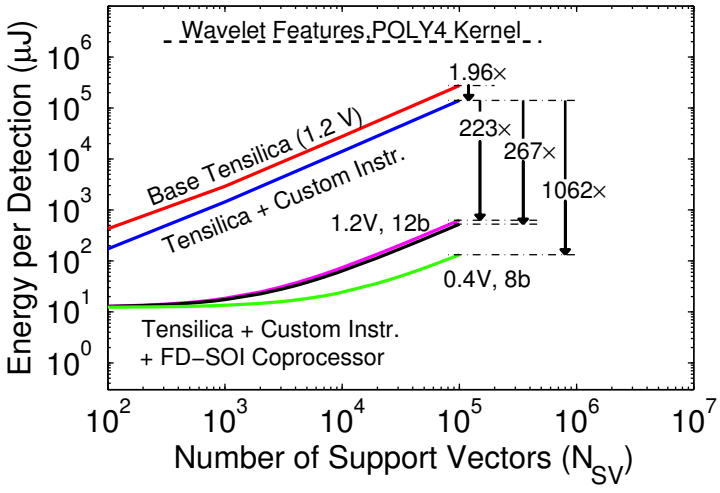


Figure 6.18: Total application energy (segmentation + feature extraction + classification) is reduced by up to 1062× through the use of a coprocessor for classification.

tions being 120.32 μJ at a supply voltage of 0.4 V and a data-representation precision of 8 bits).

Energy versus V_{dd}. We next illustrate the benefits of voltage scaling on the coprocessor. This leads to operation of the arrhythmia detector at a power level < 500 μW.

Figs. 6.20(a) and (b) show the energy optimizations achieved for the detection process using a polynomial kernel of order four for the wavelet and morphological features, respectively. Similarly, Figs. 6.21 (a) and (b) show the energy optimizations achieved for the detection process using a second-order polynomial kernel for the wavelet and morphological features, respectively. It is observed that for wavelet features, voltage and precision scaling applied to the coprocessor enable computations in signals-analysis algorithms within an energy consumption range of 24.29-132.33 μJ for N_{SV} = 10,000-100,000. Similar experiments with morphological features demonstrate the full detection process within 10.24-24.51 μJ for N_{SV} = 10,000-100,000 [see Fig. 6.20(b) and Fig. 6.21(b)]

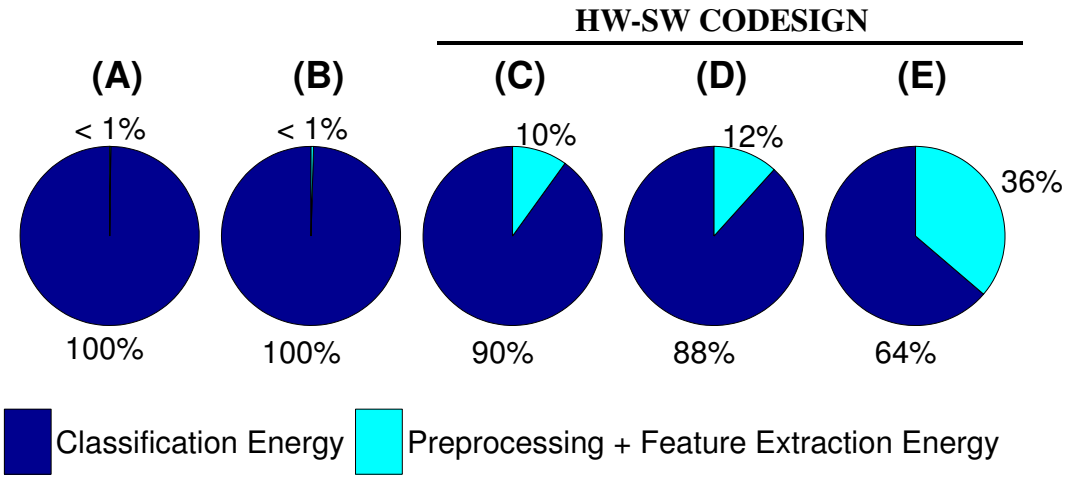
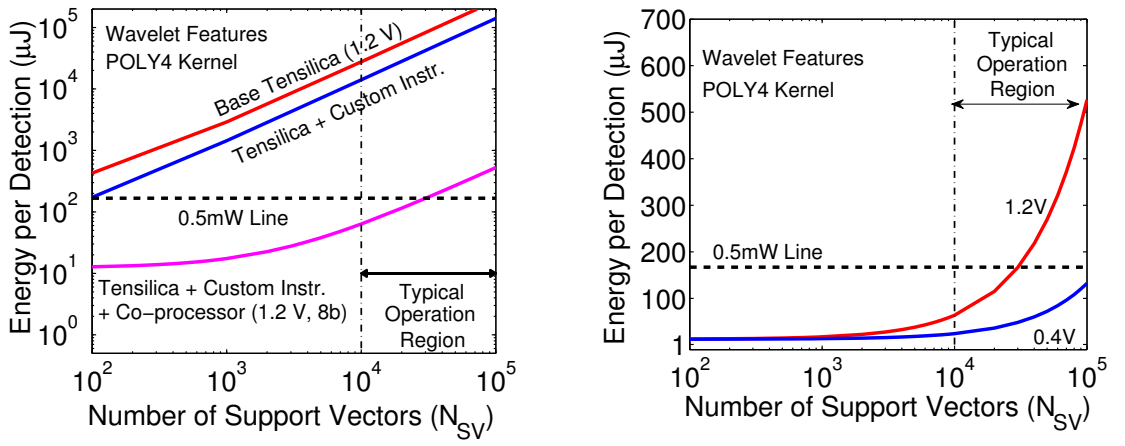


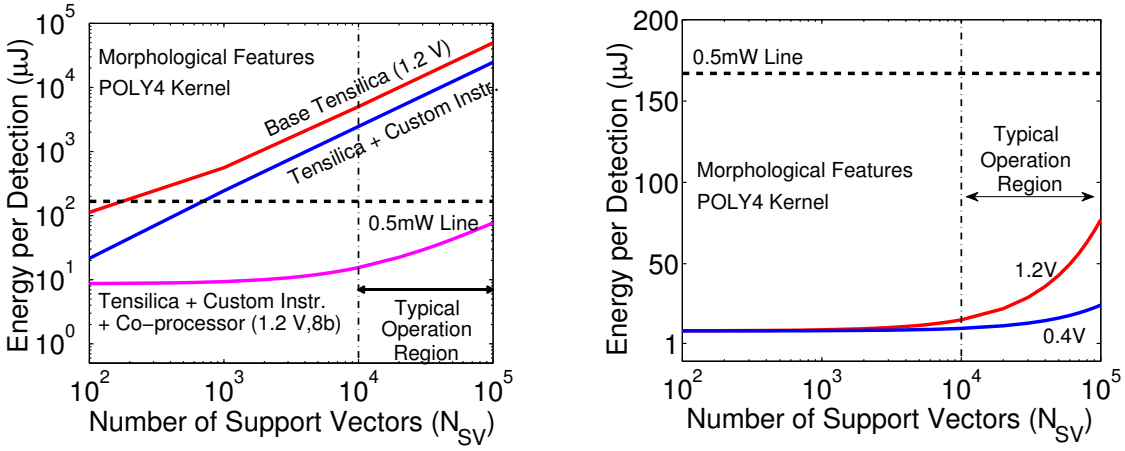
Figure 6.19: Energy proportions for preprocessing, feature extraction, and classification computations illustrate the benefits of voltage and precision scaling applied to the classification coprocessor for low-energy operation: (A) base Tensilica processor operating at V_{dd} = 1.2 V, (B) custom-instruction based implementation, (C) hardware-software codesign with custom instructions for preprocessing + feature extraction and the classification computations implemented on a coprocessor at V_{dd} = 1.2 V and 12 bits of data precision, (D) coprocessor at V_{dd} = 1.2 V and 8 bits, and (E) coprocessor at V_{dd} = 0.4 V and 8 bits.

6.6 Chapter Summary

Machine-learning based algorithms for signal analysis are emerging as a highly promising means for detecting specific states of interest in the sensed signals. In several such algorithms, although exploiting sparsity through CA can help reduce the network data, the complexity of classification can be extremely high, often dominating the preprocessing and feature-extraction computations. The structure in these algorithms can be exploited towards the design of a generalizable low-energy computation platform. In this chapter, we showed how to optimize kernel-based classification through the use of hardware specialization. We observed that although feature-extraction computations can be implemented efficiently as custom instructions on a low-power processor, the energy reductions achievable through the use of custom instructions for classification were limited due to the large number of operands involved in the dot-product computation. We thus explored opportunities for optimization through the use of a hardware coprocessor. This specialization

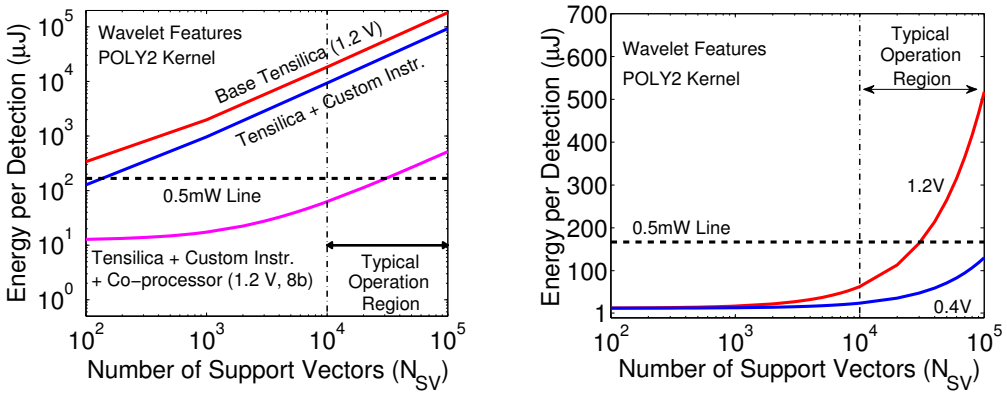


(a) Wavelet features (D_{SV}=256)

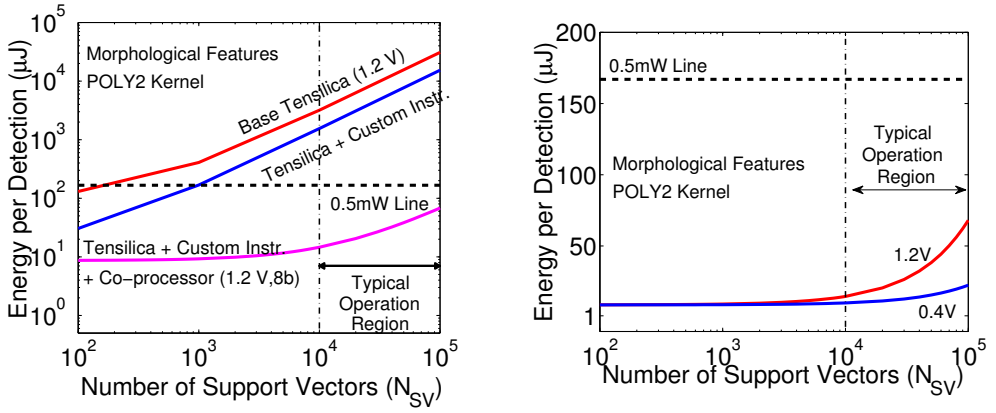


(b) Morphological features (D_{SV}=26)

Figure 6.20: Benefits of voltage scaling applied to the coprocessor (with the Tensilica processor at 1.2 V) using a fourth-order polynomial kernel and a beat classification rate (R_{CLASS}) of 3 beats/sec. Voltage scaling enables the full detector computations at less than 500 μW.



(a) Wavelet features (D_{SV}=256)



(b) Morphological features (D_{SV}=26)

Figure 6.21: Benefits of voltage scaling applied to the coprocessor (with the Tensilica processor at 1.2 V) using a second-order polynomial kernel and a beat classification rate (R_{CLASS}) of 3 beats/sec.

provided an approach for hardware-software codesign, expanding the scope of the embedded processor architecture to a broader range of applications. By employing specific hardware configurability in the classification coprocessor, we not only exploited the fixed kernel computations required for classification, but also incorporated selective flexibility required across a range of applications. For a case study of arrhythmia detection, the optimized coprocessor reduced the computational energy of the embedded platform by over three orders of magnitude compared to that of a low-power processor with custom instructions alone. We showed that implementing signal-analysis algorithms on a base Tensilica processor consumes about 100 mW for the entire computation. Thus, a wearable device

(which runs on a typical 3V, 560 mAh capacity coin-cell battery) employing a Tensilica-like processor would have an average recharge cycle of 16.8 hours. If the computational power for the entire processor can be reduced by two-three orders of magnitude, continuous signal analysis can become more viable (with battery lifetimes extended to 2-24 months). Our platform thus introduces great promise for applications employing real-time embedded signal analysis, such as healthcare networks, which are becoming increasingly important, as a wide range of real-time patient signal correlations is being discovered with new clinical states of interest.

Chapter 7

Conclusions and Future Work

Energy constraints have posed a major challenge to meeting the demand for ever-increasing functionality in embedded systems. The dominant approaches to overcoming this obstacle have focused on developing efficient signal-processing circuits and architectures. In this thesis, our primary focus instead has been on the way in which information is represented throughout the system. We attempted to answer the question of what the impact on energy would be if the embedded signals explicitly represent information more efficiently. The challenge we faced is that much of the signal-processing theory has been developed in the context of time- and frequency-domain methods. Though practical and substantially-general methods for representing information efficiently based on sparsity have emerged, they involve signal transformations that prevent the use of conventional signal-processing approaches. To address this issue, we developed methodologies that suitably transform all linear signal-processing operations so that they can be applied directly to representations that are based on sparsity. In addition to enabling generalized signal-processing systems based on efficient representations, we also quantitatively characterized the impact of such a methodology on system energy and algorithmic accuracy. To achieve this goal, we brought together mathematical approaches based on theoretical concepts and architectural innovations for practical signal-processing systems.

7.1 Thesis Summary

In this thesis, we attempted to unite a framework based on sparsity (*i.e.*, compressive sensing) with frameworks for linear signal processing. To demonstrate our results, we considered three biomedical applications as case studies: detecting epileptic seizures using EEG, sorting spikes for neural prosthesis, and detecting cardiac arrhythmias using ECG. The rest of this section summarizes our results and contributions in each chapter of the thesis.

In Chapter 3, we focused on transforming linear signal-processing computations in a seizure detection application. To achieve this goal, we first formulated feature-extraction computations using matrix operations. This enabled us to derive corresponding compressed-domain processing matrices by solving a set of overdetermined linear equations. In order to achieve this, we employed a least-squares approximation. We also showed that solving for a projection of the Nyquist-domain features (instead of the features themselves) helps achieve smaller error values in spectral-energy estimates, which were key to accurate seizure detection. The intuition was that the projection of the features preserves the inner-product between vectors. This projection was used to extract the spectral-energy values from the processed EEG signals before classification. To show that the inner-products are indeed preserved by our compressed-domain processing matrices, we showed that EEG signals can be represented through a small set of basis vectors. This allowed us to apply the JL lemma, validating the hypothesis of inner-products preservation. To further validate our methodology, we also used statistical metrics of kurtosis and skewness, as well as information metrics of KL divergence and mutual information in the FVs obtained from compressed-domain processing. Through these metrics and through simulation of the end-to-end algorithm, we showed that at a compression factor of 10×, despite $\approx$ 30% error in the spectral-energy features, the accuracy of seizure detection suffers minimally. This chapter led us to identify two key limitations of our methodology. First, the error introduced by the least-squares approximation negatively impacted the end-to-end performance of the detector. This meant that an accurate method of solving

the equations could potentially help us achieve higher compression factors for the same degradation in detection accuracy. Second, we learnt that the inner-product error was impacted by the length of the input vector used for compressed-domain processing; the length of the input EEG vector processed for seizure detection was 512 samples. By exploring the theoretical error bounds, we saw that the inner-product error could be alleviated by processing high-dimensional input vectors.

In Chapter 4, we addressed the first limitation identified in Chapter 3. We introduced a designer-controllable auxiliary matrix $\mathbf{\Theta}$ to solve the compressed-domain equations. The system of linear equations now became underdetermined, allowing us an infinite number of valid solutions. We showed that by constraining these equations appropriately, we can derive two types of solutions: (1) an exact solution with minimum error in the compressed-domain equations, and (2) an approximate solution with smaller-sized processing matrices in the compressed domain, potentially saving us additional computational energy. Further, introducing $\mathbf{\Theta}$ allowed us to also transform multi-rate signal-processing systems. We validated our methodology using a neural prosthesis application where we performed spike sorting followed by classification using a K-means algorithm. We showed that the inner-product error among FVs could be reduced to below 20% at a compression factor of 10×. This lower error helped us achieve much higher detection accuracies than the least-squares approach. Further, we also showed the applicability of our methodology to multi-rate systems by using a modified version of the seizure-detection algorithm, where we downsampled EEG signals before processing. From this chapter, we learnt that compressed-domain processing is limited by a theoretical lower bound on the inner-product errors. These errors are governed by the random projections used in compressive sensing. We also learnt that an auxiliary matrix based approach provides us with new knobs to explore the energy-accuracy tradeoff in compressed-domain processing systems.

In Chapter 5, we took advantage of the knobs provided by the methodology presented in Chapter 4 in a practical IC implementation. We presented a hardware architecture for

the seizure-detection algorithm used in Chapter 3. We showed that the compressed-domain processing matrices disrupted the regularity of FIR filters used in the Nyquist-domain algorithm. This necessitated the use of an SRAM to store the filter coefficients for processing. We observed that in compressed-domain processing, the SRAM energy dominates the logic energy. However, the size of the compressed-domain processing matrices could potentially be much smaller. The size of the matrices was controllable through two knobs: (1) ξ, which determined the amount of compression and (2) ν, which determined the error in the compressed-domain equations. By utilizing these knobs, we thus showed that a subarray of SRAMs could be extremely advantageous for power management. Through measurement results, we studied the energy scaling characteristics of the compressed-domain seizure detector with respect to ξ and ν. We observed that the compressed-domain feature-extraction energy scaled substantially with both parameters. Thus, we demonstrated that our methodology could provide an effective energy-accuracy knob for processing signals. Based on the prototype IC presented in this chapter, we learnt that by directly processing compressively-sensed signals, we can simultaneously reduce energy for communication and computation. The computational energy savings are obtained since we not only avoid the reconstruction process but also process fewer input samples compared to the Nyquist domain. However, a limitation we observed was that for algorithms, which rely on non-linear SVMs (*i.e.*, those that employ RBF or polynomial kernels), the energy for classification could significantly dominate the feature-extraction energy. This could potentially obviate the computational-energy benefits of compressed-domain processing.

In Chapter 6, we attempted to address the limitation observed in Chapter 5. We performed an application-driven algorithmic study of data-driven algorithms for signal analysis. Our aim was to investigate the reason for the high complexity of classification. We used two algorithms for arrhythmia detection as representative examples. We saw that in these algorithms, the classifier complexity could be extremely high (as much as 243× higher than preprocessing and feature extraction when implemented on a Tensilica processor) in

the Nyquist domain itself. The reason for this complexity was the high-order detection models used in the SVM classifier. High-order models were necessary to maintain accurate detection. We thus conducted an architectural study for optimizing such algorithms. We observed that using custom instructions was not very effective for reducing the energy consumption of classification computations since the large data vectors pulled from memory posed the main energy bottleneck. However, we found that a coprocessor based architecture could be more effective in minimizing classifier energy. The challenge, however, was to provide selective flexibility along with hardware specialization. This was necessary to be able to support classifier complexities across a range of applications. We thus proposed a coprocessor architecture for SVM classification, which had a variable-sized buffer, variable-precision MAC, and programmable polynomial kernel. We implemented the coprocessor using a low-power FD-SOI technology where we also employed circuit-level optimizations of voltage scaling and parallelism. Through post-layout simulations, we demonstrated reductions of up to three orders of magnitude for SVM classification as compared to implementations using custom instructions. The results from this chapter also suggest that our coprocessor based platform could bring down the energy consumption of the end-to-end signal-analysis algorithms to under 0.5 mW. This could potentially be promising for enabling continuous on-node signal analysis, which, in several applications, is often constrained by limited battery resources.

7.2 Future Directions

In this thesis, we attempted to explore the question of whether we can directly process signals when they are represented efficiently. In particular, we considered signal representations that exploit sparsity. There are several avenues along which the techniques we presented can be further explored.

Develop a suite of compressed-domain processing functions. Our approach focused on transforming computations so that they can be applied directly to compressively-sensed signals, which are derived by sub-Nyquist sampling. Specifically, we demonstrated methodologies to transform linear signal-processing functions, including multi-rate systems. For the considered applications, we derived compressed-domain processing matrices corresponding to wavelet transforms and FIR filters (including downsampling) in the Nyquist domain. In the future, it would be beneficial to derive compressed-domain matrices for other linear signal-processing functions. Eventually, this could help build a suite of compressed-domain functions, which can also be mapped to a hardware library. This process can help characterize the range of energy savings provided by different linear signal-processing operations. Building a suite of compressed-domain functions would also help designers quickly explore error bounds and compression factors for a much broader range of applications.

Study compression limits for algorithms that process high-dimensional data vectors. In Sec. 3.8, we explored theoretical bounds for inner-product errors achievable by our methodology. We observed that the error can potentially be lowered when we process high-dimensional vectors. Our input-vector dimensionality was restricted by the algorithms we used for evaluation. However, our methodology can potentially be of more benefit to those algorithms that process high-dimensional input vectors. In the future, it would thus be interesting to explore the compression limits for algorithms that must handle large amounts of sensor data. This can arise either due to the high sampling rates involved or processing of longer signal epochs. This research direction could be especially compelling for image-processing and computer-vision algorithms. With increasing emphasis on mobile computer vision, compressed-domain processing could be very beneficial for substantial energy reduction and increased battery lives.

Derive transformations for algorithms that process low-dimensional data vectors derived from a large number of measurement channels. Another related direction is to

extend our methodology to applications with very low-dimensional input vectors. The idea in this case is to derive new compressed-domain matrices so that they can be applied to a concatenated vector derived using signals from multiple data channels. Concatenating channels could help our methodology since it artificially generates high-dimensional vectors, thereby reducing the inner-product error. The challenge in this case, however, is to be able to subsequently isolate the processed results for individual channels without any loss of information. A potential method for achieving this may involve projecting data from each sensing channel onto an orthogonal subspace and then applying the compressed-domain processing matrix. Application domains, such as brain-machine interfaces that rely on ECoGs, typically require a large number of low-dimensional measurement channels [229]. Such applications can benefit substantially from this approach.

Extend the compressed-domain processing methodology to non-linear systems and systems with feedback. The processing algorithms we considered in this thesis assume feed-forward signal-processing functions in the Nyquist domain (*e.g.*, FIR filters). Extending our methodology to systems with feedback (*e.g.*, infinite impulse response filters) would be interesting. However, this research path may require overcoming a new set of challenges like buffering in the compressed domain. Another compelling exploration for future research is to extend our methodology to non-linear signal-processing functions. This could be a very challenging task. One possible way to tackle this problem could be through piecewise linear approximations of a non-linear processing function. We could model each linear component using a matrix and transform it to the compressed domain. The challenge, however, is in stitching together the processed pieces in the compressed domain so that it effectively mimics the result of transforming the non-linear function.

Bibliography

[1] J. B. Predd, S. R. Kulkarni, and H. V. Poor, "Distributed learning in wireless sensor networks," *IEEE Signal Processing Magazine*, vol. 23, no. 4, pp. 56–69, Jul. 2006.

[2] S. Mandal and R. Sarpeshkar, "Power-efficient impedance-modulation wireless data links for biomedical implants," *IEEE Trans. Biomedical Circuits and Systems*, vol. 2, no. 4, pp. 301–315, Dec. 2008.

[3] J. Pandey and B. P. Otis, "A sub-100 μW MICS/ISM band transmitter based on injection-locking and frequency multiplication," *IEEE J. Solid-State Circuits*, vol. 46, pp. 1049–1058, May 2011.

[4] N. Verma, A. H. Shoeb, J. Bohorquez, J. Dawson, J. Guttag, and A. P. Chandrakasan, "A micro-power 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.

[5] Texas Instruments Inc., "Low-cost low-power 2.4 GHz RF transmitter," [Online]. Available: http://focus.ti.com/docs/prod/folders/print/cc2550.html, Oct. 2007.

[6] ——, "2.4 GHz bluetooth low-energy SoC," [Online]. Available: http://www.ti.com/lit/ds/symlink/cc2540.pdf, Jul. 2011.

[7] E. J. Candès and T. Tao, "Near optimal signal recovery from random projections: Universal encoding strategies," *IEEE Trans. Information Theory*, vol. 52, no. 12, pp. 5406–5425, Dec. 2006.

[8] D. Donoho, "Compressed sensing," *IEEE Trans. Information Theory*, vol. 52, no. 4, pp. 1289–1306, Apr. 2006.

[9] J. N. Laska, S. Kirolos, M. F. Duarte, T. S. Ragheb, R. G. Baraniuk, and Y. Massoud, "Theory and implementation of an analog-to-information converter using random demodulation," in *Proc. Int. Symp. Circuits and Systems*, May 2007, pp. 1959–1962.

[10] F. Chen, A. P. Chandrakasan, and V. M. Stojanovic, "Design and analysis of a hardware-efficient compressed sensing architecture for data compression in wireless sensors," *IEEE J. Solid-State Circuits*, vol. 47, no. 3, pp. 744–756, Mar. 2012.

[11] S. Aviyente, "Compressed sensing framework for EEG compression," in *Proc. IEEE Int. Wkshp. Statistical Signal Processing*, Aug. 2007, pp. 181–184.

[12] M. A. T. Figueiredo, R. D. Nowak, and S. J. Wright, "Gradient projection for sparse reconstruction: Application to compressed sensing and other inverse problems," *IEEE J. Sel. Topics in Signal Processing*, vol. 1, no. 4, pp. 586–597, Dec. 2007.

[13] M. Shoaib, N. K. Jha, and N. Verma, "Enabling advanced inference on sensor nodes through the direct use of compressively-sensed signals," in *Proc. IEEE Design, Automation, and Test in Europe Conf.*, Mar. 2012, pp. 437–443.

[14] ——, "Compressed-domain signal analysis for low-energy medical sensors with application to a seizure-detection system," *IEEE Trans. Biomedical Circuits and Systems, under review*.

[15] ——, "Signal processing with direct computations on compressively-sensed data," *IEEE Trans. VLSI Systems, under review*.

[16] ——, "A compressed-domain processor for seizure detection to simultaneously reduce computation and communication energy," in *Proc. IEEE Conf. Custom Integrated Circuits*, Sep. 2012, pp. 1–4.

[17] M. Shoaib, K. H. Lee, N. K. Jha, and N. Verma, "A 0.6-106 μW energy scalable processor for seizure detection with compressively-sensed EEG," *IEEE Trans. Circuits and Systems - I, under review*.

[18] M. Shoaib, N. K. Jha, and N. Verma, "Sub-threshold computational circuits for high-order data-driven analysis of physiological signals," in *Proc. Conf. Subthreshold Microelectronics*, Sep. 2011, pp. 72–76.

[19] ——, "Algorithm-driven architectural design space exploration of domain-specific medical-sensor processors," *IEEE Trans. VLSI Systems*, pp. 1–14, Oct. 2012.

[20] H. Gao, R. M. Walker, P. Nuyujukian, K. A. A. Makinwa, K. V. Shenoy, B. Murmann, and T. H. Meng, "Hermese: A 96-channel full data rate direct neural interface in 0.13 μm CMOS," *IEEE J. Solid State Circuits*, vol. 47, no. 4, pp. 1043–1055, Apr. 2012.

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

[22] A. B. Schwartz, X. T. Cui, D. J. Weber, and D. W. Moran, "Brain-controlled interfaces: Movement restoration with neural prosthetics," *Neuron*, vol. 52, no. 1, pp. 205–220, Oct. 2006.

[23] W. Wattanapanitch and R. Sarpeshkar, "A low-power 32-channel digitally programmable neural recording integrated circuit," *IEEE Trans. Biomedical Circuits and Systems*, vol. 5, no. 6, pp. 592–602, Dec. 2011.

[24] R. R. Harrison, P. T. Watkins, R. J. Kier, R. O. Lovejoy, D. J. Black, B. Greger, and F. Solzbacher, "A low-power integrated circuit for a wireless 100-electrode neural recording system," *IEEE J. Solid State Circuits*, vol. 42, no. 1, pp. 123–133, Jan. 2007.

[25] M. Shoaib, N. K. Jha, and N. Verma, "A low-energy computation platform for data-driven biomedical monitoring algorithms," in *Proc. Design Automation Conf.*, Jun. 2011, pp. 591–596.

[26] H. Mamaghanian, N. Khaled, D. Atienza, and P. Vandergheynst, "Compressed sensing for real-time energy-efficient ECG compression on wireless body sensor nodes," *IEEE Trans. Biomedical Engineering*, vol. 58, no. 9, pp. 2456–2466, Sep. 2011.

[27] T. O. Ayodele, *New Advances in Machine Learning*. InTech, Feb. 2010, ch. Types of Machine Learning Algorithms.

[28] J. A. Tropp, "Topics in sparse approximation," Ph.D. Thesis, Computer Science, University of Texas at Austin, Austin, TX, USA, Aug. 2004.

[29] M. Aharon, "Overcomplete dictionaries for sparse representation of signals," Ph.D. Thesis, Computer Science, Technion - Israel Institute of Technology, Haifa, Israel, Nov. 2006.

[30] M. Aharon, M. Elad, and A. Bruckstein, "K-SVD: An algorithm for designing overcomplete dictionaries for sparse representation," *IEEE Trans. Signal Processing*, vol. 54, no. 11, pp. 4311–4322, Nov. 2006.

[31] K. Kreutz-Delgado, J. F. Murray, B. D. Rao, K. Engan, T.-W. Lee, and T. J. Sejnowski, "Dictionary learning algorithms for sparse representation," *J. Neural Computing*, vol. 15, no. 2, pp. 349–396, 2003.

[32] J. Haupt, W. U. Bajwa, M. Rabbat, and R. Nowak, "Compressed sensing for networked data," *IEEE Signal Processing Magazine*, vol. 25, no. 2, pp. 92–101, Mar. 2008.

[33] R. Robucci, L. K. Chiu, J. Gray, J. Romberg, P. Hasler, and D. Anderson, "Compressive sensing on a CMOS separable transform image sensor," in *Proc. IEEE Int. Conf. Acoustics, Speech and Signal Processing*, Apr. 2008, pp. 5125–5128.

[34] M. Wakin, J. Laska, M. Duarte, D. Baron, S. Sarvotham, D. Takhar, K. Kelly, and R. Baraniuk, "An architecture for compressive imaging," in *Proc. IEEE Int. Conf. Image Processing*, Oct. 2006, pp. 1273–1276.

[35] M. Lustig, D. Donoho, and J. M. Pauly, "Sparse MRI: The application of compressed sensing for rapid MR imaging," *Magnetic Resonance in Medicine*, vol. 58, no. 6, pp. 1182–1195, Dec. 2007.

[36] D. Salomon, *A Concise Introduction to Data Compression*. Berlin: Springer, 2008.

[37] C. Strydis and G. N. Gaydadjiev, "Profiling of lossless compression algorithms for a novel biomedical implant architecture," in *Proc. Int. Conf. Hardware/Software Codesign and System Synthesis*, Oct. 2008, pp. 109–114.

[38] A. E. Cetin and H. Köymen, *The Biomedical Engineering Handbook: Second Edition*. Boca Raton: CRC Press LLC, 2000, ch. Compression of Digital Biomedical Signals.

[39] S. M. S. Jalaleddine, C. G. Hutchens, R. D. Strattan, and W. Coberly, "ECG data compression techniques: A unified approach," *IEEE Trans. Biomedical Engineering*, vol. 37, no. 4, pp. 329–343, Apr. 1990.

[40] E. U. Ruttimann and H. V. Pipberger, "Compression of the ECG by prediction or interpolation and entropy encoding," *IEEE Trans. Biomedical Engineering*, vol. BME-26, no. 11, pp. 613–623, Nov. 1979.

[41] J. R. Cox, F. M. Nolle, H. A. Fozzard, and G. C. Oliver, "AZTEC: A preprocessing program for real-time ECG rhythm analysis," *IEEE Trans. Biomedical Engineering*, vol. BME-15, no. 2, pp. 128–129, Apr. 1968.

[42] D. R. Weber, "A synopsis on data compression," in *Proc. National Telemetering Conference*, Oct. 1965, pp. 9–16.

[43] Y. Zigel, A. Cohen, and A. Katz, "ECG signal compression using analysis by synthesis coding," *IEEE Tran. on Biomedical Engineering*, vol. 47, no. 10, pp. 1308–1316, Oct. 2000.

[44] R. Gryder and K. Hake, "Survey of data compression techniques," *ORNL/TM-11797: Technical Report, Oak Ridge National Laboratory, TN, USA*, Sep. 1991.

[45] G. Antoniol, "EEG data compression techniques," *IEEE Trans. Biomedical Engineering*, vol. 44, no. 2, pp. 105–114, Feb. 1997.

[46] A. S. Alvarado, C. Lakshminarayan, and J. C. Pr´ncipe, "Time-based compression and classification of heartbeats," *IEEE Trans. Biomedical Engineering*, vol. 6, no. 1, pp. 1–9, Jan. 2007.

[47] V. K. Goyal, "Theoretical foundations of transform coding," *IEEE Signal Processing Magazine*, vol. 18, no. 5, pp. 9–21, Sep. 2001.

[48] N. S. Jayant and P. Noll, *Digital Coding of Waveforms*. Englewood Cliffs, New Jersey: Prentice Hall, 1984.

[49] M. Gastpar, P. L. Dragotti, and M. Vetterli, "The distributed karhunenloeve transform," *IEEE Trans. Information Theory*, vol. 52, no. 12, pp. 5177–5196, Dec. 2006.

[50] M. Akay, *Time Frequency and Wavelets in Biomedical Signal Processing*. New York: IEEE press, 1998.

[51] M. L. Hilton, "Wavelet and wavelet packet compression of electrocardiograms," *IEEE Trans. Biomedical Engineering*, vol. 44, no. 5, pp. 394–402, May 1997.

[52] M. F. Duarte, G. Shen, A. Ortega, and R. G. Baraniuk, "Signal compression in wireless sensor networks," *Phil. Trans. Royal Society A: Mathematical, Physical, and Engineering Sciences*, vol. 370, no. 1958, pp. 118–135, Jan.

[53] S. Rein and M. Reisslein, "Low-memory wavelet transforms for wireless sensor networks: A Tutorial," *IEEE Communications Surveys and Tutorials*, vol. 13, no. 2, pp. 291–307, May. 2011.

[54] A.-Y. Wu and K. J. R. Liu, "Algorithm-based low-power transform coding architectures: The multirate approach," *IEEE Trans. Very Large Scale Integration (VLSI) Systems*, vol. 6, no. 4, pp. 707–718, Dec. 1998.

[55] M. E. Womble, J. S. Halliday, S. K. Mitter, M. C. Lancaster, and J. H. Triebwasser, "Data compression for storing and transmitting ECGs/VCGs," *Proc. IEEE*, vol. 65, no. 5, pp. 702–706, May 1997.

[56] J. Han, H. Chang, L. Loss, K. Zhang, F. L. Baehner, J. W. Gray, P. Spellman, and B. Parvin, "Comparison of sparse coding and kernel methods for histopathological classification of gliobastoma multiforme," in *Proc. Int. Symp. Biomedical Imaging: From Nano to Macro*, Apr. 2011, pp. 711–714.

[57] X. Lu, Y. Wang, and Y. Yuan, "Sparse coding from a bayesian perspective," *IEEE Trans. Neural Networks and Learning Systems*, vol. 24, no. 6, pp. 929–939, Jun.. 2013.

[58] M. Morup and M. N. Schmidt, "Transformation invariant sparse coding," in *Proc. Int. Wkshp. Machine Learning for Signal Processing*, Sep. 2011, pp. 1–6.

[59] I. S. Duff, "A survey of sparse matrix research," *Proc. IEEE*, vol. 65, no. 4, pp. 500–535, Apr.

[60] J. Yang, A. Bouzerdoum, and M. G. Amin, "Multi-view through-the-wall radar imaging using compressed sensing," in *Proc. European Signal Processing Conf.*, Aug., pp. 1429–1433.

[61] Z. Chaozhu and L. Jing, "Distributed video coding based on compressive sensing," in *Proc. Int. Conf. Multimedia Technology*, Jul. 2011, pp. 3046–3049.

[62] C. R. Berger, Z. Wang, J. Huang, and S. Zhou, "Application of compressive sensing to sparse channel estimation," *IEEE Communications Magazine*, vol. 48, no. 11, pp. 164–174, Nov. 2010.

[63] N. Shental, A. Amir, and O. Zuk, "Identification of rare alleles and their carriers using compressed se(que)nsing," *J. Nucleic Acids Research*, vol. 38, no. 19, pp. 179–191, Aug. 2010.

[64] Y. Lu and A. Finger, "Channel model-based sensing for indoor ultrasonic location systems," in *Proc. Wkshp. Positioning Navigation and Communication (WPNC)*, Apr. 2011, pp. 83–88.

[65] C. R. Berger, S. Zhou, and P. Willett, "Signal extraction using compressed sensing for passive radar with OFDM signals," in *Proc. Int. Conf. Information Fusion*, Jul. 2008, pp. 1–6.

[66] S. Shah, Y. Yu, and A. Petropulu, "Step-frequency radar with compressive sampling (SFR-CS)," in *Proc. IEEE Int. Conf. Acoustics Speech and Signal Processing*, Mar. 2010, pp. 1686–1689.

[67] J. Ma, "Compressed sensing for surface characterization and metrology," *IEEE Trans. Instrumentation and Measurement*, vol. 59, no. 6, pp. 1600–1615, Jun. 2010.

[68] M. F. Duarte, M. A. Davenport, M. B. Wakin, and R. G. Baraniuk, "Sparse signal detection from incoherent projections," in *Proc. Int. Conf. Acoustics Speech and Signal Processing*, May 2006, pp. 3–8.

[69] S. S. Chen, D. L. Donoho, and M. A. Saunders, "Atomic decomposition by basis pursuit," *SIAM Review*, vol. 43, no. 1, pp. 129–159, Feb. 2001.

[70] S. G. Mallat and Z. Zhang, "Matching pursuits with time-frequency dictionaries," *IEEE Trans. Signal Processing*, vol. 41, no. 12, pp. 3397–3415, Dec. 1993.

[71] Y. C. Pati, R. Rezaiifar, and P. S. Krishnaprasad, "Orthogonal matching pursuit: Recursive function approximation with applications to wavelet decomposition," in *Proc. Asilomar Conf. Signals, Systems, and Computers*, Nov. 1993, pp. 40–44.

[72] A. Borghi, J. Darbon, S. Peyronnet, T. F. Chan, and S. Osher, "A compressive sensing algorithm for many-core architectures," *Advances in Visual Computing*, vol. 6454, pp. 678–686, Dec. 2010.

[73] M. Andrecut, "Fast gpu implementation of sparse signal recovery from random projections," *Engineering Letters*, vol. 17, no. 3, pp. 151–158, Jan. 2009.

[74] J. A. Tropp and A. Gilbert, "Signal recovery from random measurements *via* orthogonal matching pursuit," *IEEE Trans. Information Theory*, vol. 53, no. 12, pp. 4655–4666, Dec. 2007.

[75] A. Septimus and R. Steinberg, "Compressive sampling hardware reconstruction," in *Proc. Int. Symp. Circuits and Systems*, Jun. 2010, pp. 3316–3319.

[76] J. L. Stanislaus and T. Mohsenin, "High performance compressive sensing reconstruction hardware with QRD process," in *Proc. Int. Symp. Circuits and Systems*, Jun. 2012, pp. 29–32.

[77] T. Minka, "The lightspeed MATLAB toolbox," [Online]. Available: http://www.research.microsoft.com/~minka/software/lightspeed, May. 2011.

[78] J. Yoo, S. Becker, M. Loh, M. Monge, E. J. Candès, and A. Emami-Neyestanak, "A 100 MHz-2 GHz 12.5x sub-Nyquist rate receiver in 90 nm CMOS," in *Proc. IEEE Symp. Radio Frequency Integrated Circuits*, Jun., pp. 31–34.

[79] H. Mamaghanian, N. Khaled, D. Atienza, and P. Vandergheynst, "Design and exploration of low-power analog to information conversion based on compressed sensing," *IEEE J. Emerging and Selected Topics in Circuits and Systems*, vol. 2, no. 3, pp. 493–501, Sep. 2012.

[80] T. Ragheb, J. N. Laska, H. Nejati, S. Kirolos, R. G. Baraniuk, and Y. Massoud, "A prototype hardware for random demodulation based compressive analog-to-digital conversion," in *Proc. Midwest Symp. Circuits and Systems*, Dec. 2008, pp. 37–40.

[81] S. Kirolos, J. Laska, M. Wakin, M. Duarte, D. Baron, T. Ragheb, Y. Massoud, and R. Baraniuk, "Analog-to-information conversion via random demodulation," in *Proc. IEEE Dallas/CAS Wkshp. Design, Applications, Integration, and Software*, Oct. 2006, pp. 71–74.

[82] J. A. Tropp, J. N. Laska, M. F. Duarte, J. K. Romberg, and R. G. Baraniuk, "Beyond Nyquist: Efficient sampling of sparse bandlimited signals," *IEEE Trans. Information Theory*, vol. 56, no. 1, pp. 520–544, Jan. 2010.

[83] S. R. Becker, "Practical compressed sensing: Modern data acquisition and signal processing," Ph.D. Thesis, Applied and Computational Mathematics, California Institute of Technology, Pasadena, CA, USA, Apr. 2011.

[84] Z. Zhang, T.-P. Jung, S. Makeig, and B. D. Rao, "Compressed sensing of EEG for wireless telemonitoring with low energy consumption and inexpensive hardware," *IEEE Trans. Biomedical Engineering*, vol. 60, no. 1(2), pp. 221–224, Jan. 2013.

[85] P. K. Baheti and H. Garudadri, "An ultra low-power pulse oximeter sensor based on compressed sensing," in *Proc. Int. Wkshp. Wearable and Implantable Body Sensor Networks*, Jun. 2009, pp. 144–148.

[86] H. Garudadri, P. K. Baheti, and S. Majumdar, "Low complexity sensors for body area networks," in *Proc. Int. Conf. Pervasive Technologies Related to Assistive Environments*, 2010, pp. 1–7.

[87] E. Dishman, "Inventing wellness systems for aging in place," *IEEE Computer*, vol. 37, no. 5, pp. 34–41, May. 2004.

[88] A. L. Benabid, "Deep brain stimulation for Parkinson's disease," *Current Opinion in Neurobiology*, vol. 13, no. 6, pp. 696–706, Dec. 2003.

[89] S. C. Schachter and C. B. Saper, "Vagus nerve stimulation," *Epilepsia*, vol. 39, no. 7, pp. 677–686, Jul. 1998.

[90] M. A. Lebedev and M. A. L. Nicolelis, "Brain-machine interfaces: Past, present and future," *Elsevier Trends in Neurosciences*, vol. 29, no. 9, pp. 536–546, Jul. 2006.

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

[92] A.-T. Avestruz, W. Santa, D. Carlson, R. Jensen, S. Stanslaski, A. Helfenstine, and T. Denison, "A 5 μW/channel spectral analysis IC for chronic bidirectional brain-machine interfaces," *IEEE J. Solid-State Circuits*, vol. 43, no. 12, pp. 3006–3024, Dec. 2008.

[93] N. Verma, A. H. Shoeb, J. Guttag, and A. Chandrakasan, "A micro-power EEG acquisition SoC with integrated seizure detection processor for continuous patient monitoring," in *Proc. Int. Symp. VLSI Circuits*, Jun. 2009, pp. 62–63.

[94] A. Chandrakasan, N. Verma, and D. Daly, "Ultralow-power electronics for biomedical applications," *Annual Review: Biomedical Engineering*, vol. 4, pp. 247–274, Aug. 2008.

[95] P. Bonato, "Wearable sensors/systems and their impact on biomedical engineering," *IEEE Engineering in Medicine and Biology Magazine*, vol. 22, no. 3, pp. 18–20, May-Jun. 2003.

[96] L. Schwiebert, S. K. S. Gupta, and J. Weinmann, "Research challenges in wireless networks of biomedical sensors," in *Proc. Int. Conf. Mobile Computing and Networking*, Jul. 2001, pp. 151–165.

[97] J. Kwong and A. P. Chandrakasan, "An energy-efficient biomedical signal processing platform," *IEEE J. Solid-State Circuits*, vol. 46, no. 7, pp. 1742–1753, Jul. 2011.

[98] H. Kim, R. F. Yazicioglu, T. Torfs, P. Merken, C. V. Hoof, and H.-J. Yoo, "An integrated circuit for wireless ambulatory arrhythmia monitoring systems," in *Proc. Int. Conf. Engineering in Medicine and Biology*, Sep. 2009, pp. 5409–5412.

[99] S. R. Sridhara, M. DiRenzo, S. Lingam, S.-J. Lee, R. Blazquez, J. Maxey, S. Ghanem, Y.-H. Lee, R. Abdallah, P. Singh, and M. Goel, "Microwatt embedded processor platform for medical system-on-chip applications," in *Proc. Int. Symp. VLSI Circuits*, Feb. 2010, pp. 15–16.

[100] D. Hau and E. Coiera, "Learning qualitative models from physiological signals," in *Proc. AAAI Symp. Artificial Intelligence in Medicine*, Mar. 1994, pp. 67–71.

[101] R. Somorjai, M. Alexander, R. Baumgartner, S. Booth, C. Bowman, A. Demko, B. Dolenko, M. Mandelzweig, A. Nikulin, and N. Pizzi, "A data-driven, flexible machine learning strategy for the classification of biomedical data," *Artificial Intelligence Methods And Tools For Systems Biology, Computational Biology*, vol. 5, pp. 67–85, Sep. 2004.

[102] N. Verma, K. H. Lee, and A. H. Shoeb, "Data-driven approaches for computation in intelligent biomedical devices: A case study of EEG monitoring for chronic seizure

detection," *J. Low Power Electronic Applications*, vol. 1, no. 1, pp. 150–174, Apr. 2011.

[103] G. Meyfroidt, F. Guiza, J. Ramon, and M. Bruynooghe, "Machine learning techniques to examine large patient databases," *Best Practice and Research Clinical Anaesthesiology*, vol. 23, no. 1, pp. 127–143, Mar. 2009.

[104] K. H. Lee, K. J. Jang, A. H. Shoeb, and N. Verma, "A data-driven modeling approach to stochastic computation for low-energy biomedical devices," in *Proc. IEEE Int. Conf. Acoustics Speech and Signal Processing*, Sep. 2011, pp. 826–829.

[105] V. K. Chippa, D. Mohapatra, A. Raghunathan, K. Roy, and S. T. Chakradhar, "Scalable effort hardware design: Exploiting algorithmic resilience for energy efciency," in *Proc. Design Automation Conference*, Jun. 2010, pp. 555–560.

[106] V. K. Chippa, A. Raghunathan, K. Roy, and S. T. Chakradhar, "Dynamic effort scaling: Managing the quality-efficiency tradeoff," in *Proc. Design Automation Conference*, Jun. 2011, pp. 603–608.

[107] A. H. Shoeb, D. Carlson, E. Panken, and T. Denison, "A micropower support vector machine based seizure detection architecture for embedded medical devices," in *Proc. Int. Conf. Engineering in Medicine and Biology*, Nov. 2005, pp. 4202–4205.

[108] D. Achlioptas, "Database-friendly random projections," in *Proc. Symp. Principles of Database Systems*, May 2001, pp. 274–281.

[109] M. Rudelson and R. Vershynin, "Non-asymptotic theory of random matrices: Extreme singular values," *[Online]. Available: arXiv preprint arXiv:1003.2990*, Apr. 2010.

[110] S. Dasgupta and A. Gupta, "An elementary proof of the Johnson-Lindenstrauss lemma," *Random Structures and Algorithms*, vol. 22, no. 1, pp. 60–65, Jun. 2002.

[111] S. Kaski, "Dimensionality reduction by random mapping: Fast similarity computation for clustering," in *Proc. IEEE Int. Joint Conf. Computational Intelligence and Neural Networks*, May. 1998, pp. 413–418.

[112] D. Fradkin and D. Madigan, "Experiments with random projections for machine learning," in *Proc. ACM Int. Conf. Knowledge Discovery and Data Mining*, Jul. 2003, pp. 517–522.

[113] R. Calderbank, S. Jafarpour, and R. Schapire, "Compressed learning: Universal sparse dimensionality reduction and learning in the measurement domain," *Technical Report, Princeton University*, Dec. 2009.

[114] V. Vapnik, *Estimation of Dependences Based on Empirical Data*. Springer-Verlag, Mar. 2006.

[115] ——, *The Nature of Statistical Learning Theory*. Springer, Nov. 1999.

[116] C. J. C. Burges, "A tutorial on support vector machines for pattern recognition," *Data Mining and Knowledge Discovery*, vol. 2, pp. 121–167, Jun. 1998.

[117] S. Zhou and J. Lafferty, "Compressed and privacy-sensitive sparse regression," *IEEE Trans. Information Theory*, vol. 55, no. 2, pp. 846–866, Feb. 2009.

[118] O. A. Maillard and R. Munos, "Compressed least-squares regression," *Neural Information Processing Systems*, pp. 1–9, Oct. 2009.

[119] M. M. Fard, Y. Grinberg, J. Pineau, and D. Precup, "Compressed least-squares regression on sparse spaces," in *Proc. AAAI Conf. Artificial Intelligence*, Jul. 2012, pp. 1054–1060.

[120] P. Indyk and R. Motwani, "Approximate nearest neighbors: Towards removing the curse of dimensionality," in *Proc. ACM Symp. Theory of Computing*, Jun. 1998, pp. 604–613.

[121] R. J. Durrant and A. Kaban, "Compressed Fisher linear discriminant analysis: Classification of randomly projected data," in *Proc. ACM Int. Conf. Knowledge Discovery and Data Mining*, Jul. 2010, pp. 1119–1128.

[122] S. Dasgupta, "Experiments with random projection," in *Proc. Conf. Uncertainty in Artificial Intelligence*, Jul. 2000, pp. 143–151.

[123] A. Bourrier, R. Gribonval, and P. Perez, "Compressive Gaussian mixture estimation," in *Proc. Int. Conf. Acoustics, Speech, and Signal Processing*, May 2013.

[124] S. Dasgupta, D. Hsu, and N. Verma, "A concentration theorem for projections," in *Proc. Conf. Uncertainty in Artificial Intelligence*, Jul. 2006, pp. 114–121.

[125] N. E. Karoui, "Concentration of measure and spectra of random matrices: Applications to correlation matrices, elliptical distributions and beyond," *The Annals of Applied Probability*, vol. 19, no. 6, pp. 2362–2405, Nov. 2009.

[126] J. E. Fowler, "Compressive-projection principal component analysis," *IEEE Trans. Image Processing*, vol. 18, no. 10, pp. 2230–2242, Oct. 2009.

[127] Y. Wu, Y. Chi, and R. Calderbank, "Compressive blind source separation," in *Proc. Int. Conf. Image Processing*, Sep. 2010, pp. 89–92.

[128] R. G. Baraniuk and M. B. Wakin, "Random projections of smooth manifolds," *J. Foundations of Computational Mathematics*, vol. 9, no. 1, pp. 51–77, Jan. 2009.

[129] J. B. Tenenbaum, V. de Silva, and J. C. Langford, "A global geometric framework for nonlinear dimensionality reduction," *Science*, vol. 290, no. 5500, pp. 2319–2323, Dec. 2000.

[130] S. T. Roweis and L. K. Saul, "Nonlinear dimensionality reduction by locally linear embedding," *Science*, vol. 290, no. 5500, pp. 2323–2326, Dec.

[131] M. F. Duarte, M. A. Davenport, M. B. Wakin, J. N. Laskar, D. Takhar, K. F. Kelly, and R. G. Baraniuk, "Multiscale random projections for compressive classification," in *Proc. Int. Conf. Image Processing*, Oct. 2007, pp. 161–164.

[132] D. L. Donoho and C. Grimes, "Hessian Eigenmaps: New locally linear embedding techniques for high-dimensional data," *Proc. National Academy of Sciences*, vol. 100, no. 10, pp. 5591–5596, May.

[133] C. Hegde, M. B. Wakin, and R. G. Baraniuk, "Random projections for manifold learning," in *Proc. Conf. Neural Information Processing Systems*, Dec. 2007, pp. 15–23.

[134] K. Huang and S. Aviyente, "Sparse representation for signal classification," *Advances in Neural Information Processing Systems*, vol. 19, no. 609, pp. 609–617, Dec. 2007.

[135] R. O. Duda, P. E. Hart, and D. G. Stork, *Pattern Classification*. New York: Wiley, Oct. 2000.

[136] M. A. Davenport, P. T. Boufounos, M. B. Wakin, and R. G. Baraniuk, "Signal processing with compressive measurements," *IEEE J. Selected Topics in Signal Processing*, vol. 4, no. 2, pp. 445–460, Apr. 2010.

[137] M. A. Davenport, M. F. Duarte, M. B. Wakin, J. N. Laska, D. Takhar, K. F. Kelly, and R. G. Baraniuk, "The smashed filter for compressive classification and target recognition," in *Proc. SPIE Computational Imaging V*, Jan. 2007, pp. 64 980H–64 980H.

[138] S. C. Saxena, V. Kumar, and S. T. Hamde, "Feature extraction from ECG signals using wavelet transforms for disease diagnostics," *J. Systems Science*, vol. 33, no. 13, pp. 1073–1085, Nov. 2002.

[139] J. Wagner, J. Kim, and E. Andre, "From physiological signals to emotions: Implementing and comparing selected methods for feature extraction and classification," in *Proc. IEEE Int. Conf. Multimedia and Expo*, Jul. 2005, pp. 940–943.

[140] W. Ting, Y. Guo-Zheng, Y. Bang-Hua, and S. Hong, "EEG feature extraction based on wavelet packet decomposition for brain computer interface," *Measurement*, vol. 41, no. 6, pp. 618–625, Jul. 2008.

[141] R. Nevatia and K. R. Babu, "Linear feature extraction and description," *Computer Graphics and Image Processing*, vol. 13, no. 3, pp. 257–269, Jul. 1980.

[142] O. D. Trier, A. K. Jain, and T. Taxt, "Feature extraction methods for character recognition: A survey," *Pattern Recognition*, vol. 29, no. 4, pp. 641–662, Apr. 1996.

[143] E. L. Hall, R. P. Kruger, S. J. Dwyer, D. L. Hall, R. W. Mclaren, and G. S. Lodwick, "A survey of preprocessing and feature extraction techniques for radiographic images," *IEEE Trans. Computers*, vol. C-20, no. 9, pp. 1032–1044, Sep. 1971.

[144] T. R. Reed and J. M. H. Dubuf, "A review of recent texture segmentation and feature extraction techniques," *Image Understanding*, vol. 57, no. 3, pp. 359–372, May 1993.

[145] M. D. Levine, "Feature extraction: A survey," *Proc. IEEE*, vol. 57, no. 8, pp. 1391–1407, Aug. 1969.

[146] E. Hjelmas and B. K. Low, "Face detection: A survey," *Computer Vision and Image Understanding*, vol. 83, no. 3, pp. 236–274, Sep. 2001.

[147] A. H. Shoeb and J. Guttag, "Application of machine learning to seizure detection," in *Proc. Int. Conf. Machine Learning*, Jun. 2010, pp. 975–982.

[148] A. M. Abdulghani, A. J. Casson, and E. Rodriguez-Villegas, "Quantifying the feasibility of compressive sensing in portable electroencephalography systems," *Neuroergonomics and Operational Neuroscience J. Foundations of Augmented Cognition*, pp. 319–328, Jul. 2009.

[149] L. Kuhlmann, A. N. Burkitt, M. J. Cook, K. Fuller, D. B. Grayden, L. Seiderer, and I. M. Y. Mareels, "Seizure detection using seizure probability estimation: Comparison of features used to detect seizures," *Annals of Biomed. Engineering*, vol. 37, no. 10, pp. 2129–2145, Oct. 2009.

[150] H. Qu, "Self-adapting algorithms for seizure detection during EEG monitoring," Ph.D. Thesis, Electrical Engineering, McGill University, Montreal, Quebec, Canada, Nov. 1994.

[151] A. Subasi and E. Erçelebi, "Neural network classification of EEG signals by using AR with MLE preprocessing for epileptic seizure detection," *J. Mathematical and Computational Appl.*, vol. 10, no. 1, pp. 57–70, Feb. 2005.

[152] K. M. Kiymik, A. Subasi, and R. H. Ozcalik, "Neural networks with periodogram and autoregressive spectral analysis methods in detection of epileptic seizure," *J. Medical Systems*, vol. 28, no. 6, pp. 511–522, Dec. 2004.

[153] M. E. Saab and J. Gotman, "A system to detect the onset of epileptic seizures in scalp EEG," *Clinical Neurophysiology*, vol. 116, no. 2, pp. 427–442, Feb. 2005.

[154] H. Qu and J. Gotman, "Patient-specific algorithm for the detection of seizure onset in long-term EEG monitoring: Possible use as a warning device," *IEEE Trans. Biomedical Engineering*, vol. 44, no. 2, pp. 115–122, Feb. 1997.

[155] R. Meier, H. Dittrich, A. Schulze-Bonhage, and A. Aertsen, "Detecting epileptic seizures in long-term human EEG: A new approach to automatic online and real-time detection and classification of polymorphic seizure patterns," *J. Clinical Neurophysiology*, vol. 25, no. 3, pp. 119–131, Jun. 2008.

[156] Y. Lecun, L. Bottou, Y. Bengio, and P. Haffner, "Gradient-based learning applied to document recognition," *Proc. IEEE*, vol. 86, no. 11, pp. 2278–2324, Nov. 1998.

[157] P. Y. Simard, D. Steinkraus, and J. C. Platt, "Best practices for convolutional neural networks," in *Proc. Int. Conf. Document Analysis and Recogntion*, 2003, pp. 958–962.

[158] B. Hammer and T. Villmann, "Generalized relevance learning vector quantization," *Neural Networks*, vol. 15, pp. 1059–1068, 2002.

[159] M. R. Abdullah, K.-A. Toh, and D. Srinivasan, "A framework for empirical classifiers comparison," in *Proc. IEEE Conf. Industrial Electronics and Applications*, 2006, pp. 1–6.

[160] J. C. Platt, *Advances in Kernel Methods - Support Vector Learning*. Cambridge, MA, USA: MIT Press, 1999, ch. Fast Training of Support Vector Machines Using Sequential Minimal Optimization.

[161] I. Durdanovic, E. Cosatto, and H. P. Graf, "Large scale parallel SVM implementation," *Large Scale Kernel Machines*, pp. 105–138, 2007.

[162] T. Hofmann, B. Scholkopf, and A. J. Smola, "Kernel methods in machine learning," *The Annals of Statistics*, vol. 36, no. 3, pp. 1171–1220, Jun. 2008.

[163] A. Rahimi and B. Recht, "Random features for large-scale kernel machines," *Advances in Neural Information Processing Systems*, vol. 20, pp. 1177–1184, Dec. 2007.

[164] W. B. Mendelson, D. A. Sack, S. P. James, J. V. Martin, R. Wagner, D. Garnett, J. Milton, and T. A. Wehr, "Frequency analysis of the sleep EEG in depression," *Psychiatry Research*, vol. 21, no. 2, pp. 89–94, Jun. 2007.

[165] Physionet, "CHB-MIT Physionet database," [Online]. Available: http://www. physionet.org/physiobank/database, Jun. 2000.

[166] G. H. Golub, "Numerical methods for solving linear least squares problems," *Numerische Mathematik*, vol. 7, no. 3, pp. 206–216, Oct. 1965.

[167] R. I. Arriaga and S. Vempala, "An algorithmic theory of learning: Robust concepts and random projection," in *Proc. Symp. Foundations of Computer Science*, Oct. 1999, pp. 616–623.

[168] E. Kushilevitz, R. Ostrovsky, and Y. Rabani, "Efficient search for approximate nearest neighbor in high dimensional spaces," in *Proc. ACM Symp. Theory of Computing*, May 1998, pp. 614–623.

[169] J. W. Fisher, M. Siracusa, and K. Tieu, "Estimate of signal information content for classification," in *Proc. IEEE DSP Wkshp. and IEEE Signal Processing Education Wkshp.*, Jan. 2009, pp. 353–358.

[170] R. Battiti, "Using mutual information for selecting features in supervised neural net learning," *IEEE Trans. Neural Networks*, vol. 5, no. 4, pp. 537–550, Jul. 1994.

[171] A. M. Abdulghani, A. J. Casson, and E. Rodriguez-Villegas, "Quantifying the performance of compressive sensing on scalp EEG signals," in *Proc. Symp. Applied Science in Biomedical and Communication Technologies*, Oct. 2010, pp. 1–5.

[172] C. Sieluzycki, R. Konig, A. Matysiak, R. Kus, D. Ircha, and P. J. Durka, "Single-trial evoked brain responses modeled by multivariate matching pursuit," *IEEE Trans. Biomedical Engineering*, vol. 56, no. 1, pp. 74–82, Jan. 2009.

[173] R. D. Larsen, E. F. Crawford, and P. W. Smith, "Reduced spline representations for EEG signals," in *Proc. IEEE*, May 1977, pp. 804–807.

[174] N. Alon, "Problems and results in extremal combinatorics- I ," *Discrete Mathematics*, vol. 273, no. 1, pp. 31–53, May 2003.

[175] J. C. Letelier and P. P. Weber, "Signal processing challenges for neural prostheses," *IEEE Signal Processing Magazine*, vol. 25, no. 1, pp. 18–28, Sep. 2008.

[176] R. H. Olsson and K. D. Wise, "A three-dimensional neural recording microsystem with implantable data compression circuitry," *IEEE J. Solid State Circuits*, vol. 40, no. 12, pp. 2796–2804, Dec. 2005.

[177] A. M. Sodagar, K. D. Wise, and K. Najafi, "A fully integrated mixed-signal neural processor for implantable multichannel cortical recording," *IEEE Trans. Biomedical Engineering*, vol. 54, no. 7, pp. 1075–1088, Jun. 2007.

[178] M. Abeles and M. H. Goldstein Jr., "Multispike train analysis," *Proc. IEEE*, vol. 65, no. 5, pp. 762–773, May 1977.

[179] E. N. Brown, R. E. Kass, and P. P. Mitra, "Multiple neural spike train data analysis: State-of-the-art and future challenges," *Nature Neuroscience*, vol. 7, no. 5, pp. 456–461, Apr. 2004.

[180] M. S. Lewicki, "A review of methods for spike sorting: The detection and classification of neural action potentials," *Computational Neural Systems*, vol. 9, no. 4, pp. 53–78, Jan. 1998.

[181] S. Gibson, J. W. Judy, and D. Marković, "Technology-aware algorithm design for neural spike detection, feature extraction, and dimensionality reduction," *IEEE Trans. Neural Systems and Rehabilitation Engineering*, vol. 18, no. 5, pp. 469–478, Oct. 2010.

[182] R. Q. Quiroga, Z. Nadasdy, and Y. Ben-Shaul, "Unsupervised spike detection and sorting with wavelets and superparamagnetic clustering," *Neural Comp.*, vol. 16, no. 8, pp. 1661–1687, Aug. 2004.

[183] K. V. Shenoy, D. Meeker, S. Cao, S. A. Kureshi, B. Pesaran, C. A. Buneo, A. P. Batista, P. P. Mitra, J. W. Burdick, and R. A. Andersen, "Neural prosthetic control signals from plan activity," *Neuroreport*, vol. 14, no. 4, pp. 591–596, Mar. 2003.

[184] S. Musallam, B. D. Corneil, B. Greger, H. Scherberger, and R. A. Andersen, "Cognitive control signals for neural prosthetics," *Science*, vol. 305, no. 5681, pp. 258–262, Jul. 2004.

[185] R. E. Kass, V. Ventura, and E. N. Brown, "Statistical issues in the analysis of neuronal data," *J. Neurophysiology*, vol. 94, no. 1, pp. 8–25, Feb. 2005.

[186] S. Kim, "Thermal impact of an active 3-D microelectrode array implanted in the brain," *IEEE Trans. Neural Systems and Rehabilitation Engineering*, vol. 15, no. 4, pp. 493–501, Dec. 2007.

[187] 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.

[188] 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. Int. Symp. Solid State Circuits*, Feb. 2008, pp. 146–147.

[189] K. G. Oweiss, "A systems approach for data compression and latency reduction in cortically controlled brain machine interfaces," *IEEE Trans. Biomedical Engg.*, vol. 53, no. 7, pp. 1364–1377, Jul. 2006.

[190] Z. S. Zumsteg, C. Kemere, S. O. Driscoll, G. Santhanam, R. E. Ahmed, K. V. Shenoy, and T. H. Meng, "Power feasibility of implanted digital spike sorting circuits for neural prosthetic systems," *IEEE Trans. Neural Systems and Rehabilitation Engineering*, vol. 1, no. 3, pp. 272–279, Sep. 2005.

[191] K. G. Oweiss, A. Mason, Y. Suhail, A. M. Kamboh, and K. E. Thomson, "A scalable wavelet transform VLSI architecture for real-time signal processing in high-density intra-cortical implants," *IEEE Trans. Circuits and Systems-I*, vol. 54, no. 6, pp. 1266–1278, Jun. 2007.

[192] J. Holleman, A. Mishra, C. Diorio, and B. Otis, "A micro-power neural spike detector and feature extractor in 0.13 μm CMOS," in *Proc. IEEE Conf. Custom Integrated Circuits*, Sep. 2008, pp. 333–336.

[193] T. K. Sarkar, C. Su, R. S. Adve, M. Salazar-Palma, L. E. García-Castillo, and R. R. Boix, "A tutorial on wavelets from an electrical engineering perspective I: Discrete wavelet techniques," *IEEE Mag. Antennas and Propagation*, vol. 40, no. 5, pp. 49–68, Oct. 1998.

[194] E. J. Candès and J. Romberg, "l_1-magic: Recovery of sparse signals *via* convex programming," [Online]. Available: www.acm.caltech.edu/l1magic/downloads/l1magic.pdf, Oct. 2005.

[195] E. V. D. Berg and M. P. Friedlander, "Probing the Pareto frontier for basis pursuit solutions," [Online]. Available: http://www.optimization-online.org/DB_FILE/

2008/01/1889.pdf, Dept. Computer Science, University of British Columbia, Tech. Rep. TR-2008-01, Jan. 2008.

[196] N. Verma, "Analysis towards minimization of total SRAM energy over active and idle operating modes," *IEEE Trans. VLSI Systems*, vol. 19, pp. 1695–1703, Sep. 2011.

[197] Synopsys Inc., "NanoSim: A next-generation solution for SoC integration verification," [Online]. Available: http://www.synopsys.com/Tools/Verification/Pages/socintegration.aspx, Apr. 2013.

[198] A. Wang and A. P. Chandrakasan, "A 180-mV subthreshold FFT processor using a minimum energy design methodology," *J. Solid-State Circuits*, vol. 40, no. 1, pp. 310–319, Jan. 2005.

[199] K. H. Lee, S.-Y. Kung, and N. Verma, "Low-energy formulations of support vector machine kernel functions for biomedical sensor applications," *Journal of Signal Processing Systems*, vol. 69, no. 3, pp. 339–349, Dec. 2012.

[200] Physionet, "MIT-BIH Physionet database," [Online]. Available: http://www. physionet.org/physiobank/database, Jun. 2000.

[201] J. Pan and W. J. Tompkins, "A real time QRS detection algorithm," *IEEE Trans. Biomedical Engineering*, vol. BME-32, no. 3, pp. 232–236, Mar. 1985.

[202] P. Laguna, N. V. Thakor, P. Caminal, R. Jané, H.-R. Yoon, A. B. de Luna, V. Marti, and J. Guindo, "New algorithm for QT interval analysis in 24-hour Holter ECG: Performance and applications," *Medical and Biological Engineering and Computing*, vol. 28, no. 1, pp. 67–73, Jan. 1990.

[203] P. Sajda, "Machine learning for detection and diagnosis of disease," *Annual Review: Biomedical Engineering*, vol. 8, pp. 537–565, Aug. 2006.

[204] U. Manne, R. G. Srivastava, and S. Srivastava, "Keynote review: Recent advances in biomarkers next term for cancer diagnosis and treatment," *Drug Discovery Today*, vol. 10, no. 14, pp. 965–976, Jul. 2005.

[205] T. Sunderland, R. E. Gur, and S. E. Arnold, "The use of biomarkers in the elderly: Current and future challenges," *Biological Psychiatry*, vol. 58, no. 4, pp. 272–276, Aug. 2005.

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

[207] T. H. Yeap, F. Johnson, and M. A. Rachniowski, "ECG beat classification by a neural network," in *Proc. Int. Conf. Engineering in Medicine and Biology*, Nov. 1990, pp. 1457–1458.

[208] O. Stanislaw and T. H. Linh, "ECG beat recognition using fuzzy hybrid neural network," *IEEE Trans. Biomedical Engineering*, vol. 48, pp. 1265–1271, Nov. 2001.

[209] L. Senhadji, G. Carrault, J.-J. Bellanger, and G. Passariello, "Comparing wavelet transforms for recognizing cardiac patterns," *IEEE Engineering in Medicine and Biology Magazine*, vol. 14, no. 2, pp. 167–173, Mar./Apr. 1995.

[210] A. S. Jaffe, L. Babuin, and F. S. Apple, "Biomarkers in acute cardiac disease: The present and the future," *J. American College of Cardiology*, vol. 48, pp. 1–11, Jun. 2006.

[211] P. de Chazal and R. B. Reilly, "A comparison of the ECG classification performance of different feature sets," in *Proc. IEEE Conf. Computer in Cardiology*, Sep. 2000, pp. 327–330.

[212] E. D. Ubeyli, "ECG beats classification using multiclass support vector machines with error correcting output codes," *Digital Signal Processing*, vol. 17, no. 3, pp. 675–684, May 2007.

[213] T. Jaochims, "SVM-Light, support vector machine," [Online]. Available: http:// www.svmlight/ jaochims.org, Aug. 2008.

[214] O. Chapelle and V. Vapnik, "Advances in neural information processing systems," Cambridge, MA, USA, Jan. 1999.

[215] 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. Acoustics Speech and Signal Processing*, May. 2011, pp. 1597–1600.

[216] C. H. Tang, P. M. Middleton, A. V. Savkin, G. S. H. Chan, S. Bishop, and N. H. Lovell, "Non-invasive classification of severe sepsis and systemic inflammatory response syndrome using a nonlinear support vector machine: A preliminary study," *Physiological Measurement*, vol. 31, no. 6, pp. 775–793, May 2010.

[217] A. H. Shoeb, "Application of machine learning to epileptic seizure onset detection and treatment," Ph.D. Thesis, Electrical and Medical Engineering, Massachusetts Institute of Technology, Boston, Massachusetts, Sep. 2009.

[218] Tensilica Inc., "The Xtensa processor," [Online]. Available: http://www.tensilica .com, Jun. 2010.

[219] F. Sun, S. Ravi, A. Raghunathan, and N. K. Jha, "Custom-instruction synthesis for extensible-processor platforms," *IEEE Trans. Computer-Aided Design*, vol. 23, no. 2, Feb. 2004.

[220] ——, "Application-specific heterogeneous multiprocessor synthesis using extensible processors," *IEEE Trans. Computer-Aided Design*, vol. 25, no. 9, Sep. 2006.

[221] ——, "A scalable synthesis methodology for application-specific processors," *IEEE Trans. VLSI Systems*, vol. 14, no. 11, Nov. 2006.

[222] ——, "A synthesis methodology for hybrid custom instruction and co-processor generation for extensible processors," *IEEE Trans. Computer-Aided Design*, vol. 26, no. 11, Nov. 2007.

[223] L. Seiler, D. Carmean, E. Sprangle, T. Forsyth, P. Dubey, S. Junkins, A. Lake, R. Cavin, R. Espasa, E. Grochowski, T. Juan, M. Abrash, J. Sugerman, and P. Hanrahan, "Larrabee: A many-core x86 architecture for visual computing," *IEEE Micro*, vol. 29, no. 1, pp. 10–21, Jan./Feb. 2009.

[224] F.-M. Schleif, M. Lindemann, M. Diaz, P. Maab, J. Decker, T. Elssner, M. Kuhn, and H. Thiele, "Support vector classification of proteomic profile spectra based on feature extraction with the bi-orthogonal discrete wavelet transform," *Computer Visualization Science*, vol. 4, no. 12, pp. 189–199, Dec. 2007.

[225] N. Krupa, M. M. A. Ali, E. Zahedi, S. Ahmed, and F. M. Hassan, "Antepartum fetal heart rate feature extraction and classification using emperical mode decomposition and support vector machine," *Biomedical Engineering Online*, vol. 10, no. 1, pp. 1–15, Jan. 2011.

[226] V. Sze and A. P. Chandrakasan, "A 0.4 V UWB baseband processor," in *Proc. IEEE Int. Symp. Low Power Electronics and Design*, Aug. 2007, pp. 262–267.

[227] I. S. Abu-Khater, A. Bellaouar, and M. I. Elmasry, "Circuit techniques for CMOS low-power high-performance multipliers," *IEEE J. Solid-State Circuits*, vol. 31, no. 10, pp. 1535–1546, Oct. 1996.

[228] S. A. Vitale, P. W. Wyatt, N. Checka, J. Kedzierski, and C. L. Keast, "FD-SOI process technology for subthreshold-operation ultralow-power electronics," *Proc. IEEE*, vol. 98, no. 2, pp. 333–342, Feb. 2010.

[229] S. Ortiz Jr., "Brain-computer interfaces: Where human and machine meet," *IEEE Computer*, vol. 40, no. 1, pp. 17–21, Jan. 2007.