

© 2017 Yingyan Lin

ENERGY-EFFICIENT SYSTEMS FOR INFORMATION TRANSFER AND PROCESSING

BY

YINGYAN LIN

DISSERTATION

Submitted in partial fulfillment of the requirements
for the degree of Doctor of Philosophy in Electrical and Computer Engineering
in the Graduate College of the
University of Illinois at Urbana-Champaign, 2017

Urbana, Illinois

Doctoral Committee:

Professor Naresh R. Shanbhag, Chair
Professor Andrew C. Singer
Professor Elyse Rosenbaum
Professor Pavan K. Hanumolu

ABSTRACT

Machine learning (ML) systems are finding excellent utility in tackling the data deluge of the big data era thanks to the exponential increase in computing power. Current ML systems adopt either centralized cloud computing or distributed edge computing. In both, the challenge of energy efficiency has been drawing increased attention. In cloud computing, data transfer due to inter-chip, inter-board, inter-shelf and inter-rack communications (I/O interface) within data centers is one of the dominant energy costs. This will intensify with the growing demand for increased I/O bandwidth of high-performance computing in data centers. On the other hand, in edge computing, energy efficiency is the primary design challenge, as mobile devices have limited energy, computation and storage resources. This challenge is being exacerbated by the need to embed ML algorithms such as convolutional neural networks (CNNs) for enabling local on-device inference capabilities. In this dissertation, we investigate techniques to address these challenges.

To address the energy efficiency challenge in data centers, this dissertation focuses on reducing the energy consumption of the I/O interface. Specifically, in the emerging analog-to-digital converter (ADC)-based multi-Gb/s serial link receivers, the power dissipation is dominated by the ADC. ADCs in serial links employ signal-to-noise-and-distortion-ratio (SNDR) and effective-number-of-bits (ENOB) as performance metrics because these are the standard for generic ADC design. This dissertation presents the use of information-based metrics such as bit-error-rate (BER) to design a BER-optimal ADC (BOA) for serial links. First, theoretical analysis is developed to show when the benefits of BOA over a conventional uniform ADC (CUA) in a serial link receiver are substantial. Second, a 4 GS/s, 4-bit on-chip ADC in a 90 nm CMOS process is designed and integrated into a 4 Gb/s serial link receiver to verify the aforementioned analysis. Specifically, measured results demonstrate that a 3-bit

BOA receiver outperforms a 4-bit CUA receiver at a BER $< 10^{-12}$ and provides 50% power savings in the ADC. In the process, it is demonstrated conclusively that BER as opposed to ENOB is a better metric when designing ADCs for serial links.

For the problem of resource-constrained computing at the edge, this dissertation tackles the issue of energy-efficient implementation of ML algorithms, particularly CNNs which have recently gained considerable interest due to their record-breaking performance in many recognition tasks. However, their implementation complexity hinders their deployment on power-constrained embedded platforms. This dissertation develops two techniques for energy-efficient CNN design.

The first technique is a *predictive* CNN (PredictiveNet), which makes use of high sparsity in well-trained CNNs to bypass a large fraction of power-dominant convolutions at run-time without modifying the CNN structure. Analysis supported by simulations is provided to justify PredictiveNet’s effectiveness. When applied to both the MNIST and CIFAR-10 datasets, simulation results show that PredictiveNet achieves $7.2\times$ and $4.4\times$ reduction in the computational and representational costs, respectively, compared with a conventional CNN. It is further shown that PredictiveNet enables computational and representational cost reductions of $2.5\times$ and $1.7\times$, respectively, compared to a state-of-the-art CNN, while incurring only 0.02 classification accuracy loss.

The second technique is a variation-tolerant architecture for CNN capable of operating in near threshold voltage (NTV) regime for aggressive energy efficiency. It is well-known that NTV computing can achieve up to $10\times$ energy savings but is sensitive to process, temperature, and voltage (PVT) variations which can lead to timing errors. To leverage the great potential of NTV for energy efficiency, this dissertation develops a new statistical error compensation (SEC) technique referred to as rank decomposed SEC (RD-SEC). RD-SEC makes use of inherent redundancy in CNNs to handle timing errors due to NTV computing. When evaluated in CNNs for both the MNIST and CIFAR-10 datasets, simulation results in 45 nm CMOS show that RD-SEC enables robust CNNs operating in the NTV regime. Specifically, the proposed RD-SEC can achieve up to $11\times$ improvement in variation tolerance and enable up to $113\times$ reduction in the standard deviation of classification accuracy while incurring marginal degradation in the median classification accuracy.

*To my family whose selfless support made this journey possible and inspires me every day,
and to everyone who has blessed me with kindness along the way.*

ACKNOWLEDGMENTS

Many great people have helped make my Ph.D. journey at UIUC more challenging, rewarding and memorable. I am grateful from the bottom of my heart for their contributions to both my research and my personal growth.

First, I would like to extend my sincerest thanks to my advisor, Prof. Naresh R. Shanbhag, for his valuable guidance and support throughout the years. It is he who encouraged me to start this journey and who has trained me along the path of becoming an independent researcher. Prof. Shanbhag has consistently emphasized the importance of big vision, analytical justification, effective presentation, and concise writing. He has held me to high standards, and my interactions with him have made me much stronger, more appreciative, and more confident. I believe he has thoroughly prepared me to face "the real world" that I will enter upon graduation, for which I will always be grateful.

Second, I would like to sincerely thank Prof. Andrew C. Singer, Prof. Pavan K. Hanumolu, and Prof. Elyse Rosenbaum for serving on my thesis committee and for providing their insightful comments. Their suggestions have helped me strengthen my research and improve this dissertation. In addition, I have heartfelt appreciation to all three of them for their valuable support on my faculty applications. I am also grateful for staff members at UIUC, including Jennifer Carlson, Laurie Fisher, Brooke Newell, Rachel Palmisano, Jennifer Summers, and James Hutchinson, for their help with all kinds of administrative and logistical tasks or editorial check of this dissertation.

Third, I truly appreciate my past and current research group colleagues, notably Dr. Eric Kim, Dr. Sai Zhang, Dr. Yongjune Kim, Dr. Mingu Kang, Sujan Gonugondla, Ameya Patil, Charbel Sakr, and Sungmin Lim. I could not have wished for better colleagues: they are very tough and honest, they have a keen sense for technical details, and their feedback

was always balanced with kindness. The numerous inspiring discussions and all of your generous encouragement will not be forgotten. Special thanks go to Dr. Yongjune Kim for his comprehensive feedback to improve this dissertation, to Dr. Sai Zhang for his useful contributions to the RD-SEC project, and to Charbel Sakr and Dr. Yongjune Kim for their helpful input on the PredictiveNet project.

Fourth, I would like to acknowledge my other friends for their priceless help and support, notably Dr. Yang Zhang, Katie Martin, Nicole Joy, Dr. Zhangyang Wang, Lili Su, Dr. Songbin Gong, Dr. Tejasvi Anand, Dr. Guanghua Shu, Dr. Woo-Seok Choi, Eric Ehmann, Yang Xiu, Dr. Peter Kairouz, and Dr. Saurabh Saxena. Special thanks go to Katie Martin for her valuable critiques to improve my English and for her encouragement during difficult times. I also sincerely thank Nicole Joy for her great Zumba instruction, which has not only made this form of exercise my true-love, but has also helped me find the delicate balance between research, personal life and health.

Finally, and most importantly, my deepest gratitude goes to my family who have supported me with more love and care than one could hope for. My parents and brothers have been models for my life: they showed me how to strive wholeheartedly towards your full potential; they demonstrated what it looks like to choose what you love to do, and then to do your very best; they taught by example how to help others in need, even when you are in need yourself. My parents and parents-in-law have taken turns to travel the long distance from China to the U.S. to help me take care of my son when I am at school. The work in this dissertation would not have been possible without their selfless support. I am very fortunate to have my husband Jing with me in the U.S., who has always encouraged me to pursue my dreams. Although this resulted in a long distance between us, he put in extra efforts to visit and to demonstrate his love and support, including driving back and forth between Texas and Illinois more than twenty times during my time at UIUC. I am extremely grateful to have my son, Chengzhong, who brightens my life like a joyful angel. He has been and will continue to be my greatest motivation for growth in all aspects of life.

CONTENTS

LIST OF TABLES	viii
LIST OF FIGURES	ix
Chapter 1 INTRODUCTION	1
1.1 Related Work	5
1.2 Dissertation Contributions and Organization	10
Chapter 2 BER-OPTIMAL ADC-BASED RECEIVER FOR SERIAL LINKS	12
2.1 Background	12
2.2 Achievable BER Improvement via BOA	17
2.3 Implementation of BOA Receiver	24
2.4 Measurement Results	31
2.5 Summary	37
2.6 Derivation of BERR	38
Chapter 3 PREDICTIVENET	41
3.1 Background	42
3.2 The Proposed PredictiveNet Technique	44
3.3 Simulation Results	49
3.4 Summary	56
3.5 Derivation of (3.10) and (3.11)	56
Chapter 4 RANK DECOMPOSED STATISTICAL ERROR COMPENSATION	58
4.1 Background	59
4.2 The Proposed RD-SEC Technique	61
4.3 Error Model Generation and Validation	67
4.4 Simulation Results	72
4.5 Summary	77
4.6 Derivation of α in (4.10)	78
Chapter 5 CONCLUSIONS AND FUTURE WORK	79
5.1 Dissertation Contributions	79
5.2 Future Work	81
REFERENCES	84

LIST OF TABLES

2.1	Performance Summary of the ADC-based Receivers	35
2.2	Performance Comparison with State-of-the-art ADC-based Receivers and Analog Receivers in CMOS	36
3.1	Errors of MSB-CNN and PredictiveNet with Respect to FP-CNN	46
3.2	Parameters Summary of the CNN for the MNIST Dataset	50
3.3	Computational and Representational Cost Comparison among CNNs for the MNIST Dataset	51
3.4	Parameters Summary of the CNN [78] for the CIFAR-10 Dataset	51
3.5	Computational and Representational Cost Comparison among CNNs for the CIFAR-10 Dataset	55
4.1	Parameters for the γ_C and γ_F in Fig. 4.4	67
4.2	Summary of CNN Parameters from [68]	74
4.3	Summary of CNN Parameters for CIFAR-10 Dataset [79]	76

LIST OF FIGURES

1.1	Illustration of: (a) centralized cloud computing, and (b) distributed edge computing.	2
1.2	Power breakdown in a state-of-the-art 48-core processor at both low and high power modes [5].	3
1.3	Comparison of available resource between a standard desktop and a Google glass [6, 7].	4
1.4	The scaling of supply voltage, a quadratic knob for energy efficiency, remains stagnant beyond 45 nm [11].	5
1.5	Role of an ADC in a serial link: (a) block diagram of a serial link, and (b) idealized model for the ADC in (a).	6
2.1	Channel response $\mathbf{h} = [0.0949, 0.2539, 0.1552, 0.0793, 0.0435, 0.0356, 0.0220]$ with $L = 7$ of a 20-inch backplane channel carrying 10 Gb/s data [57].	13
2.2	An illustrative example: (a) the block diagram of an ADC-ML receiver, (b) the conditional pdf of the channel output given $b[n] = -1$, (c) the conditional pdf of the channel output given $b[n] = +1$, (d) BOA's quantization thresholds (inverted triangles in yellow) and uniform quantization thresholds (dashed lines in red) for channel $\mathbf{h} = [0.08, 0.07, 0.1, 0.04]$ when $SNR = 36$ dB, and (e) the simulated $BERR(SNR, 4, 3)$, $p_{eo}(SNR, 3)$ and $p_{eu}(SNR, 4)$ versus SNR plot.	15
2.3	Examples of m -clustering and h_t : (a) m -clustering with $m = 3$, and (b) h_t for $\mathbf{t}_o = [\pm 0.3, \pm 0.2, \pm 0.1, 0]$ (case I where $h_t = 1$) and $\mathbf{t}_o = [\pm 0.3, \pm 0.11, \pm 0.09, 0]$ (case II where $h_t = 0.7564$) with $y_{max} = 0.3$	20
2.4	An illustrative example for $d_{o,k}^*$, μ_k^* and $d_{u,k}^*$	21
2.5	Comparison among p_{eo} from MC simulation ($p_{eo(sim)}$), p_{eo} estimated using (2.9) ($p_{eo(10)}$), p_{eu} from MC simulation ($p_{eu(sim)}$), and p_{eu} estimated using (2.12) ($p_{eu(13)}$), for channels (a) $\mathbf{h} = [0.09, 0.1, 0.08, -0.05]$ when $B_u = 6$ and $B_o = 3$, and (b) $\mathbf{h} = [0.08, 0.07, 0.1, 0.04]$ when $B_u = 4$ and $B_o = 3$, respectively.	22
2.6	Comparison between $BERR(SNR, B_u, B_o)$ from MC simulation ($BERR_{sim}$), and $BERR(SNR, B_u, B_o)$ estimated using (2.6) ($BERR_{(7)}$) for channels (a) $\mathbf{h} = [0.09, 0.1, 0.08, -0.05]$ when $B_u = 6$ and $B_o = 3$, and (b) $\mathbf{h} = [0.08, 0.07, 0.1, 0.04]$ when $B_u = 4$ and $B_o = 3$, respectively.	22

2.7	$BERR(SNR, B_u, B_o)$ vs. h_t , where $B_u = B_o = \log_2(m + 1)$, for channels $\mathbf{h} = [1, a_1, a_2, a_3]$ ($a_i = 0.1 : 0.1 : 0.9, i = 1, 2, 3$) using an ADC-ML receiver when $SNR = 38$ dB. The value of h_t and measured $BERR(SNR, 4, 3)$ for a FR4 channel using an ADC-LE receiver (described in section 2.3) are also shown.	24
2.8	$BERR(SNR, B_x, B_x)$ vs. B_x of (a) an ADC-ML receiver, when the channel impulse response is $\mathbf{h} = [0.09, 0.1, 0.08, 0.04]$ and $SNR = 36$ dB, and (b) an ADC-LE receiver, when the channel impulse response is equal to the FR-4 channel (see Fig. 2.1) and $SNR = 36$ dB.	25
2.9	Block diagram of the BOA receiver.	25
2.10	Block diagram of the BOA chip.	26
2.11	The 8-bit single-core multiple-output DAC: (a) circuit schematic, and (b) timing diagram.	27
2.12	Architectures of: (a) the QL-UD unit, and (b) the i^{th} RL-UD unit.	29
2.13	Schematics of the preamplifier and latches.	31
2.14	(a) Micrograph of the BOA chip, and (b) the test set-up.	32
2.15	Standalone ADC measurement results: (a) DNL and (b) INL characteristics before/after calibration, and (c) measured ENOB vs. input frequency.	32
2.16	Measured ADC output: (a) eye diagram, and (b) histogram for a 20-inch FR4 channel at 4 Gb/s when TX amplitude is 180 mV_{ppd} and ADC FSR is 100 mV_{ppd}	33
2.17	ENOB and BER measurements: (a) ENOB vs. input frequency, and (b) BER vs. TX amplitude at 4 Gb/s when the FSR of the CUA is 100 mV_{ppd}	34
2.18	Measured BER vs. sampling phase using a 20-inch channel when TX amplitude is 180 mV_{ppd} and the FSR of the CUA is 100 mV_{ppd}	35
3.1	Illustration of a state-of-the-art CNN [68] showing a convolutional layer (C-layer), a subsampling layer (S-layer), feature maps (FMs), and the squashing function $f(\cdot)$	43
3.2	An architecture implementing (3.4) in PredictiveNet.	45
3.3	Illustration of the empirical values of $MSE_{MSB-CNN}$ and $MSE_{PredictiveNet}$ and the upper bound on $MSE_{PredictiveNet}$ with respect to $B_{\mathbf{x},msb}$ for: the (a) first and (b) second C-layers over FP-CNN where $B_{\mathbf{w},msb} = 5$ bits, $B_{\mathbf{w}} = 8$ bits, $B_{\mathbf{x}} = B_{\delta} = 7$ bits, and $B_{\delta,msb} = B_{\mathbf{x},msb}$. Both curves are obtained by averaging over all pixels of the two C-layers' output FMs.	48
3.4	Simulation results for the MNIST dataset comparing FP-CNN (FP-CONV and FP-ZS), PredictiveNet, and MSB-CNN in terms of: (a) classification error rates, (b) normalized computational cost (# of full adders (FAs)), and (c) normalized representational cost (# of bits), where $B_{\mathbf{x},msb} = B_{\delta,msb} = 4$ bits and $B_{\mathbf{w},msb} = 5$ bits.	52
3.5	Comparison on the C-layer input sparsity of FP-CONV/FP-ZS, PredictiveNet and MSB-CNN for the MNIST dataset.	53

3.6	Simulation results for the CIFAR-10 dataset comparing FP-CNN (FP-CONV and FP-ZS), PredictiveNet, and MSB-CNN in terms of: (a) classification error rates, (b) normalized computational cost (# of full adders (FAs)), and (c) normalized representational cost (# of bits), where $B_{\mathbf{x},\text{msb}} = B_{\delta,\text{msb}} = 6$ bits and $B_{\mathbf{w},\text{msb}} = 5$ bits.	54
3.7	Comparison on the C-layer input sparsity of FP-CONV/FP-ZS, PredictiveNet and MSB-CNN for the CIFAR-10 dataset.	55
4.1	Algorithmic noise-tolerance (ANT): (a) architecture, and (b) the error statistics in the M -block and E -block [50].	60
4.2	Architecture of: (a) a (N, M) dot product ensemble (MVM), where $\mathbf{w}_{ml} = [w_{1ml}, \dots, w_{Nml}]$ and $\mathbf{W}_l = [\mathbf{w}_{1l}, \dots, \mathbf{w}_{Ml}]$, and (b) one stage MVM-based CNN consisting of a C-layer and an S-layer.	63
4.3	RD-SEC applied to an MVM.	64
4.4	Overhead of the RD-SEC-based MVM: computational overhead γ vs. N , where the corresponding parameters are summarized in Table 4.1.	66
4.5	Block diagram of: (a) model generation methodology, and (b) error modeling framework.	68
4.6	Illustration of the AND gate within-the-die (WID) delay histograms from HSPICE Monte Carlo (MC) simulations and the AND gate delay model at: (a) $V_{dd} = 1.2\text{V}$, (b) $V_{dd} = 0.4\text{V}$, with 1000 MC iterations.	69
4.7	Validating the error model generation methodology by comparing SNR from HDL simulations and the NTV methodology based on 30 MVM instances with 10^5 random input samples for each instance operating at gate level delay variation of 3%-39%.	71
4.8	Characterization of: (a) process variations in terms of $(\sigma/\mu)_d$ vs. V_{dd} , and (b) the impact of process variations on MVM error rate \bar{p}_η based on 30 MVM instances.	72
4.9	Simulation results for the MNIST dataset: (a) \bar{p}_{det} vs. $(\sigma/\mu)_d$, and (b) $\sigma_{p_{det}}$ vs. $(\sigma/\mu)_d$, based on 30 CNN instances in the presence of process.	73
4.10	An example of the C1 FMs and the output vector from: (a) the Conv CNN, and (b) the RD-SEC CNN, when the input digit is "5" and $(\sigma/\mu)_d = 27\%$	75
4.11	Simulation results for the CIFAR-10 dataset: (a) \bar{p}_{det} vs. $(\sigma/\mu)_d$, and (b) $\sigma_{p_{det}}$ vs. $(\sigma/\mu)_d$, based on 30 CNN instances in the presence of process variations.	77

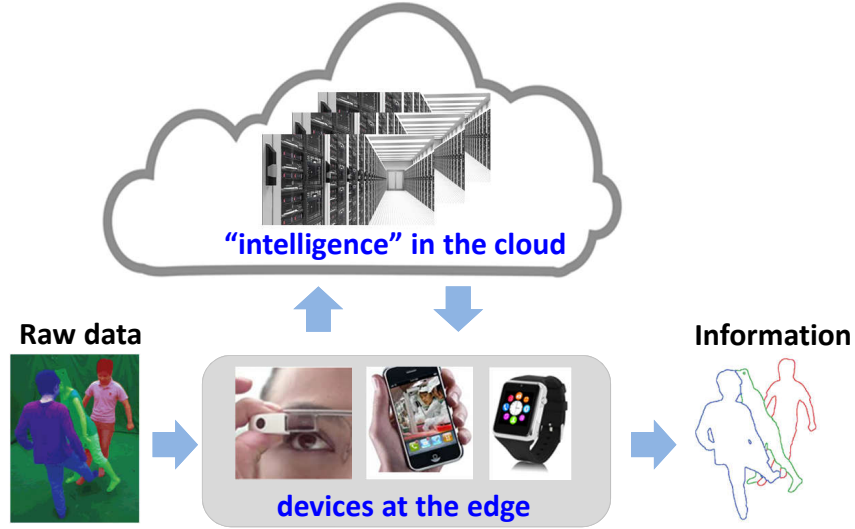
Chapter 1

INTRODUCTION

Machine learning (ML) systems have been dramatically transforming the way we live and work by enhancing our ability to recognize, analyze, and classify the world around us. In fact, many see this as the fourth industrial revolution [1]. Such unprecedented transformation is made possible by the explosion in computing power and the availability of vast amounts of data. Indeed, ML systems have transformed science fiction into everyday reality. Examples include self-driving cars or aircraft, household robots, virtual assistants, and many others. Recently, ML systems exceeded human performance in some applications such as million-scale object recognition [2]. However, this record-breaking performance comes at a large energy cost. For example, Google's AlphaGo, which amazed everyone by beating the human Go champion early in 2016, runs on 1202 CPUs and 176 GPUs [3] and consumes more than four-orders-of-magnitude higher power than the human brain. Therefore, there is an imperative need to design energy-efficient ML systems for enabling their pervasive deployment in our daily lives.

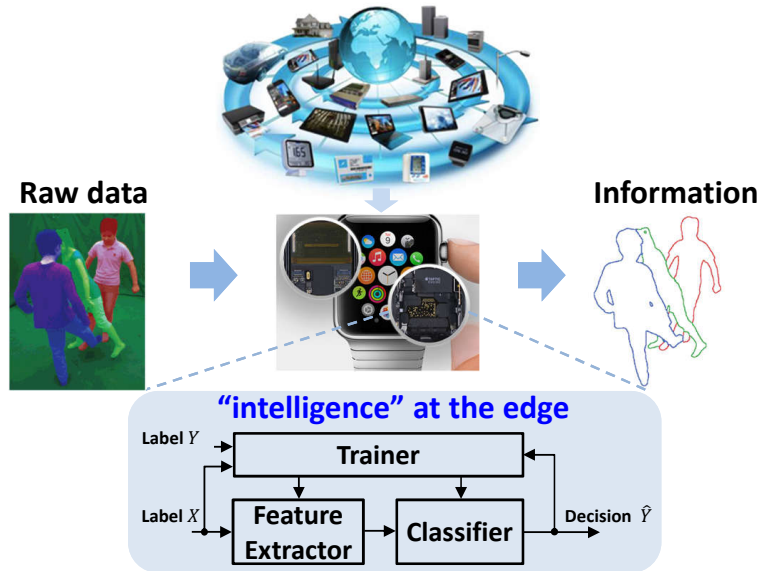
Current ML systems are either centralized in a cloud (see Fig. 1.1(a)) or distributed at the edge (see Fig. 1.1(b)). Specifically, in cloud platforms, data from the devices of end users, such as mobile phones, are transferred to the data centers which execute ML algorithms on CPU and GPU clusters. The extracted information is then transferred back to users' devices. While cloud computing is rapidly expanding, recent work [4] shows that the energy cost of transferring data between data centers and local devices can be a significant percentage of the total energy cost in cloud computing if the usage rate and data volume are large. Therefore, there has been an increasing interest in enabling local inference capability at the edge such as end users' devices. Local processing of raw data reduces energy and latency, and enhances privacy. In both centralized cloud computing and distributed edge computing,

Centralized Cloud Computing



(a)

Distributed Edge Computing



(b)

Figure 1.1: Illustration of: (a) centralized cloud computing, and (b) distributed edge computing.

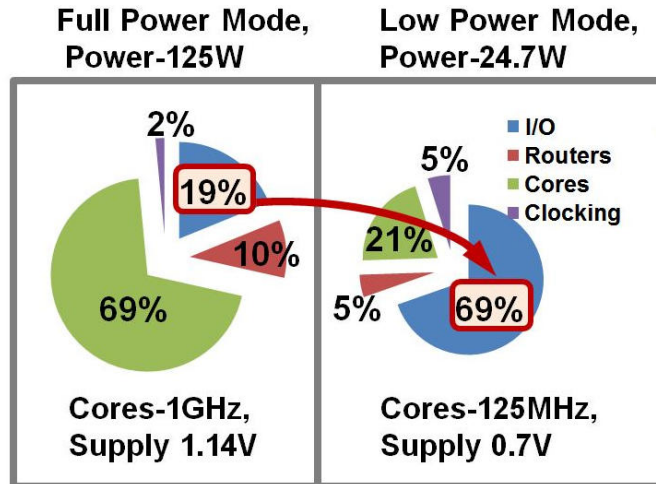
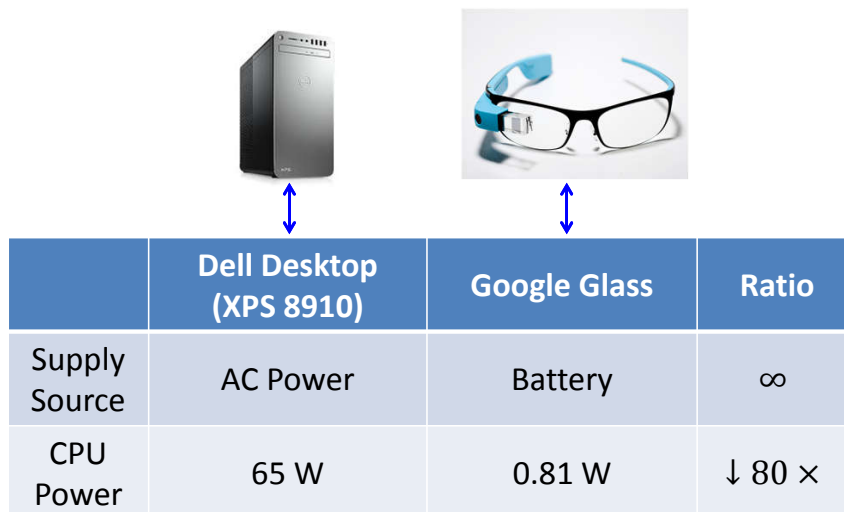


Figure 1.2: Power breakdown in a state-of-the-art 48-core processor at both low and high power modes [5].

there is a grand energy efficiency challenge as described next.

- Energy Efficiency Challenge in the Data Center:** It is reported that US data centers consumed about 70 billion kilowatt-hours of electricity in 2014, representing 2% of the country's total energy consumption [5]. Indeed, the costs of power and cooling are becoming significant factors in the total expenditures of large-scale data centers [8]. In particular, data transfer due to inter-chip, inter-board, inter-shelf and inter-rack communications within data centers is one of the dominant energy costs. For example, the I/O interface consumes about 20% – 70% of the total power in a state-of-the-art 48-core processor [5], as shown in Fig. 1.2. This will be made worse by the growing demand for increased I/O bandwidth of high-performance computing in data centers. For example, a recent projection [9] indicates that the I/O bandwidth demand will exceed 750 TB/s for super-computers by the year 2020, and the I/O power could reach half of the CPU power.
- Energy Efficiency Challenge at the Edge:** Devices at the edge including smart phones, autonomous vehicles, wearable devices, and many others have limited energy, computation and storage resources since they are battery-powered and have a small form factor. For example, the comparison in Fig. 1.3 shows that the CPU power of a Google glass is about one eightieth of a standard desktop [6, 7]. On the other hand,



	Dell Desktop (XPS 8910)	Google Glass	Ratio
Supply Source	AC Power	Battery	∞
CPU Power	65 W	0.81 W	$\downarrow 80 \times$

Figure 1.3: Comparison of available resource between a standard desktop and a Google glass [6, 7].

the required implementation complexity of many ML algorithms is high due to the need to process hundreds of computations and a significant amount of data movement. For example, a state-of-the-art convolutional neural network (CNN), AlexNet, requires 666 million multiplier-accumulators (MACs) per 227×227 image ($13k$ MACs/pixel) and hundreds of megabytes for weight storage [10]. Therefore, the energy efficiency challenge will be aggravated due to the need for ML algorithms to enable inference capability in these platforms. Conventional designs rely on voltage and process scaling for energy efficiency, which have already stagnated as shown in Fig. 1.4 [11].

Therefore, we aim to explore techniques to address energy efficiency challenges in both data centers and resource-constrained platforms at the edge. Specifically, to address the energy efficiency challenge in data centers, we focus on reducing the energy of the I/O interface by exploring the design of analog-to-digital converter (ADC)-based multi-Gb/s serial link receivers. In addition, we also investigate energy-efficient design of complicated ML algorithms such as CNNs for their employment in resource-constrained platforms.

In the remainder of this chapter, we provide an overview of related prior work, and then present the contributions and organization of this dissertation.

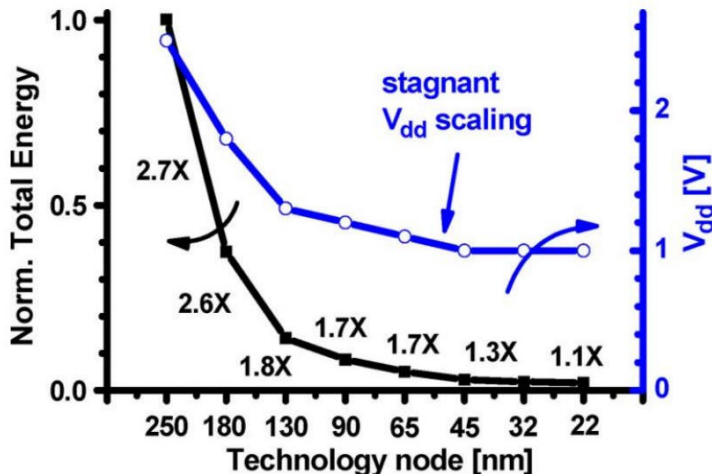


Figure 1.4: The scaling of supply voltage, a quadratic knob for energy efficiency, remains stagnant beyond 45 nm [11].

1.1 Related Work

1.1.1 ADC-based Links

In conventional ADC-based serial links (see Fig. 1.5(a)), the ADC is designed to be a transparent conduit of the input analog waveform $x_c(t)$. In such ADCs, the quantization thresholds t are set uniformly within their full-scale range (FSR). We refer to such an ADC as a *conventional uniform ADC* (CUA). The signal-to-quantization noise ratio (SQNR) of a CUA can be approximated by $SQNR = 6.02B_x + 4.8 - 20 \log_{10}(V_{max}/\sigma_x)$ [12], where σ_x^2 is the average signal power at the ADC input, B_x is the ADC resolution, and V_{max} is the maximum input amplitude. The SQNR is a *signal fidelity* metric as it measures the average squared difference between the ADCs sampled input $x_c(nT)$ and its quantized output $x[n]$. Other signal fidelity based metrics such as signal-to-noise-and-distortion-ratio (SNDR) or effective-number-of-bits (ENOB) are also employed. Such fidelity-based metrics impose overly stringent specifications on the ADC because they ignore the true role of the ADC in a communication link, which is to preserve the *information content* in the input signal $x_c(t)$ in order to recover the transmitted data reliably. One direct consequence of employing fidelity-based metrics is that the ADC needs more resolution B_x than needed. In such ADCs,

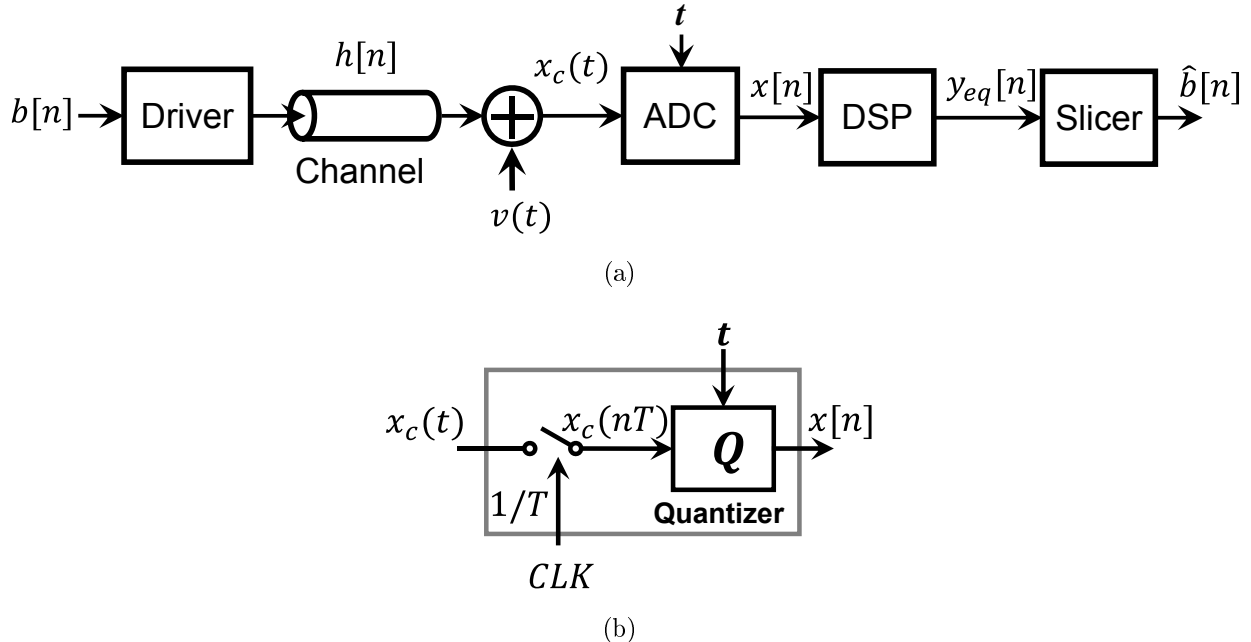


Figure 1.5: Role of an ADC in a serial link: (a) block diagram of a serial link, and (b) idealized model for the ADC in (a).

a single bit reduction in B_x can result in significant power savings. For example, in flash ADCs, the area, power consumption, and input capacitance increase exponentially with B_x . These result in large preamplifier bandwidth and multiple stages of latches which exacerbate the ADC power consumption problem [13, 14]. Therefore, the design of low-power and high-speed ADCs in serial links is a major challenge, which has drawn great interest from both industry and academia.

Recently, there has been research that attempts to employ the link bit-error-rate (BER) as a design metric for energy-efficient link design. Past work on BER-optimal link components includes [15], in which an adaptive minimum BER (AMBER) algorithm is proposed to adapt the equalizer coefficients. It was shown that minimum-BER equalizers outperform conventional minimum mean square error (MMSE) equalizers over a wide variety of channels especially when the BER lies in a regime of rapid descent with the number of equalizer coefficients. Chen et al. [16] demonstrated the benefits of adapting the equalizer coefficients and the sampling phase of the clock-data-recovery (CDR) to minimize the BER in serial links via the design of a prototype IC in 65 nm CMOS for a 6.25 Gb/s serial link. In [17, 18], an

algorithm to determine the BER-optimal ADC (BOA) representation levels was proposed. The *ADC shaping gain* $SG(p_e, B_u, B_o)$ defined below was employed to quantify the benefits of BER optimality:

$$SG(p_e, B_u, B_o) = SNR_u(p_e, B_u) - SNR_o(p_e, B_o), \quad (1.1)$$

where $SNR_u(p_e, B_u)$ and $SNR_o(p_e, B_o)$ are the signal-to-noise ratios (SNRs) needed by a CUA and a BOA, respectively, to achieve a BER equal to p_e with identical receiver processing, and B_u and B_o are the resolutions of the CUA and the BOA, respectively. This ADC shaping gain quantifies the reduction in the required channel SNR to achieve a given BER p_e due to the use of a BOA. The ADC shaping gain is analogous to a coding gain when evaluating links with error control coding. It was shown in [17,18] that $SG(10^{-12}, 3, 3)$ ranged from 2.5 dB to more than 30 dB for highly dispersive channels. We note that a BOA employs representation levels that are dependent on signal statistics and BER, and hence are typically non-uniformly spaced within the FSR of the ADC. The works in [17,18] also showed that the non-uniform BOA representation levels are significantly different from and superior to the non-uniform (also signal statistics-dependent) representation levels obtained from the well-known Lloyd-Max (LM) quantization algorithm [19,20]. This is because the LM algorithm minimizes the SQNR, which is also a fidelity metric. In [21], it was shown that BOA can relax the component specifications of ADCs. In particular, BOA can achieve the same or even better BER while it has less stringent metastability and preamplifier bandwidth requirements on the ADC comparators.

A power-optimized ADC-based 10 Gb/s serial link receiver in 65 nm CMOS was designed in [22] using a low-gain analog and mixed-mode pre-equalizer in conjunction with non-uniform representation levels for the ADC. The works in [22,23] propose to merge slicers whose thresholds are similar into one for loop-unrolled decision feedback equalizers (DFEs) and adjusts a pseudo-BER metric (voltage margin) to minimize BER, which in effect emulates a BOA followed by a DFE. However, this technique is applicable only to loop-unrolled DFEs, and has to rely on a continuous-time linear equalizer to cancel the precursor ISI. Second, as mentioned in [23], their procedure to determine the optimal threshold placement is not

suitable for online calibration. More recently, Son et al. [24] proposed a power efficient equalizing receiver front-end that includes a two-step adaptive BER-minimizing equalizer algorithm. These works mentioned above demonstrate that the use of information-based metrics such as the BER are indeed quite effective in reducing link component power in serial links.

1.1.2 Energy-efficient CNNs

Many emerging applications in pattern recognition and data mining require the use of ML algorithms to process massive data volumes on energy-constrained platforms [25]. CNN is a powerful ML algorithm that achieves state-of-the-art performance in various recognition tasks [2]. For example, the Microsoft ResNet achieved a better-than-human accuracy of 3.57% in the ImageNet Large Scale Visual Recognition Challenge (ILSVRC) 2015 [26], which is a benchmark in object category classification and detection consisting of hundreds of object categories and millions of images. The implementation complexity of CNNs is very high due to the need to compute a large number of convolutions usually taking up over 90% of the total computational cost [27] and to process a significant amount of data movement. This high complexity of CNNs hinders their implementation on power-constrained embedded platforms.

Substantial research efforts have been invested in reducing the complexity of CNNs. One line of research attempts to reduce the precision of weights and activations, and has shown that 8-bit [28] or even binary [29] fixed-point representation is sufficient for evaluating CNNs. Another approach focuses on optimizing the structure of CNN itself. The work in [30] employs a three-step method, where the network is trained to learn important connections, prune redundant connections in pre-trained CNNs, and then retrain the pruned networks to restore the performance. Zhang et al. [31] proposed to replace convolutional layers by several convolutional layers applied sequentially, which have a lower total complexity. Other research thrust exploits sparsity in well-trained CNNs or enhances sparsity in CNNs via regularization, and skips operations with zero entries (zero-skipping) [10, 32]. Recent work [33, 34] showed that it is possible to avoid evaluation of certain computations with a marginal

performance loss. In [33], a linear regression model was trained for each convolutional layer to predict the importance of each convolutional filter and prune low-impact filters at runtime. Panda et al. [34] proposed conditional deep learning (CDL) by adding a linear network to each convolutional layer and monitoring the output to decide whether classification can be terminated at the current stage.

The above mentioned techniques have the potential to reduce both the computational and data movement costs in CNN implementations. In general, data movement (memory access) cost tends to dominate the overall energy consumption in data-intensive computing systems [35]. This is especially true for large-scale CNN implementations [36, 37]. Thus, research focus has been on reducing data movement cost via maximally reusing data locally [36, 37] or in-memory computing [38, 39]. Once these techniques aggressively trim down the data movement cost of large-scale CNNs as well as in the case of small-scale CNNs, the computational cost will be on the same order or even dominate the overall energy cost [40]. In these cases, how to reduce the computational cost of CNNs becomes the primary concern. In line with this direction, one opportunity in CNNs is that matrix-vector multiply (MVM) is the most power hungry kernel and accounts for 90% of the computational cost in state-of-the-art integrated circuit implementations [10]. In a MVM, an input vector \mathbf{x} is projected to a set of weight vectors, i.e.:

$$\mathbf{y} = \mathbf{W}^T \mathbf{x} \tag{1.2}$$

where $\mathbf{W} = [\mathbf{w}_1, \dots, \mathbf{w}_M]$ is the $N \times M$ weight matrix, \mathbf{w}_k is the k^{th} $N \times 1$ weight vector, \mathbf{x} is the $N \times 1$ input vector, $\mathbf{y} = [y_1, \dots, y_M]^T$ is the $M \times 1$ output vector, and y_k is the k^{th} element of \mathbf{y} which can be expressed as a dot product (DP) $y_k = \mathbf{w}_k^T \mathbf{x}$.

As a result, energy-efficient MVM architectures are of great importance for energy-efficient CNN design. Techniques such as low power parallel filter design [41] and common subexpression elimination (CSE) [42] can be applied to MVMs to reduce computational complexity. These techniques exploit the redundancy within a multiplier or a DP. To further reduce energy consumption, near threshold voltage (NTV) designs have been proposed where the supply voltage is reduced to close to the transistor threshold voltage of 0.3V-0.7V. This de-

sign paradigm is well-suited for low throughput sensor based applications such as biomedical monitoring [43], surveillance [44], and structural sensing within critical infrastructures [11]. Past research has shown that NTV designs can achieve up to $10\times$ savings in energy, but suffer from a significant increase in variations, which can be as high as $20\times$ [11]. Error-resilient techniques [45–51] have been employed at various levels of design abstraction to compensate for the resultant timing errors caused by NTV operation. At the logic or circuit level, RAZOR [45], error detection sequential (EDS) [46], and Markov Random Field [47] have been proposed. These techniques either compensate for small error rates ($< 2\%$) or have large overhead ($> 5\times$), limiting their ability to enhance energy efficiency. At the system level, conventional fault-tolerance techniques such as N-modular redundancy (NMR) [48] incur $N\times$ complexity and power overhead, restricting their applicability. Statistical error compensation (SEC) [49–51] has been shown to be a promising solution. SEC employs detection and estimation-based techniques for error compensation. SEC techniques such as algorithmic noise-tolerance (ANT) are able to compensate for error rates of $21\% - 89\%$ while achieving $35\% - 72\%$ energy savings [50].

1.2 Dissertation Contributions and Organization

The design of energy-efficient ML systems is challenging due to the need for intensive computation and massive data movement. In this dissertation, we address this challenge by (1) employing information-based system metrics, as opposed to fidelity circuit metrics, to design power-dominant components in ML systems, (2) making use of inherent redundancy in ML algorithms for reduced complexity, and (3) computing at the limits of energy efficiency and robustness and developing SEC technique to efficiently compensate for the resultant errors. The major contributions and organization of this dissertation are summarized as follows:

Chapter 2 presents an investigation of the use of link BER for designing a BOA based serial link. Channel parameters such as the m -clustering value and the threshold non-uniformity metric h_t are introduced and employed to quantify the BER improvement achieved by a BOA over a CUA in the receiver. Analytical expressions for BER improvement are derived and validated through simulations. A prototype of BOA is designed, fabricated

and tested in a 1.2 V, 90 nm LP CMOS process to verify the results of this study. BOA’s variable-threshold and variable-resolution configurations are implemented via an 8-bit single-core, multiple-output passive digital-to-analog converter (DAC), which incurs an additional power overhead of $< 0.1\%$ (approximately $50 \mu W$). Measurement results show that the BER achieved by the 3-bit BOA receiver can be lower by a factor of 10^9 and 10^{10} , as compared to the 4-bit and 3-bit CUA receivers, respectively, at a data rate of 4-Gb/s and a transmitted signal amplitude of $180 \text{ mV}_{\text{ppd}}$.

Chapter 3 presents a *predictive* CNN (PredictiveNet), which predicts the zero outputs of the nonlinear layers using low-cost predictors thereby bypassing a majority of computations. PredictiveNet skips a large fraction of power-dominant convolutions in CNNs at runtime without modifying the CNN structure or requiring additional branch networks. Analysis supported by simulations validates the proposed PredictiveNet technique. When applied to CNNs for both the MNIST and CIFAR-10 datasets, simulation results show that PredictiveNet is able to achieve up to $2.5\times$ and $1.7\times$ reduction in the computational cost (number of 1-bit full adders) and representational cost (number of bits to represent data and weights), respectively, compared with a state-of-the-art CNN, while incurring only 0.02 classification accuracy degradation.

Chapter 4 presents a variation-tolerant architecture for CNNs capable of operating in NTV regime for energy efficiency. A SEC technique referred to as rank decomposed SEC (RD-SEC) is proposed. The key idea of RD-SEC is to exploit inherent redundancy within a MVM, a power-hungry operation in CNNs, to derive low-cost estimators for error detection and compensation. When evaluated in CNNs for both the MNIST and CIFAR-10 datasets, simulation results in 45 nm CMOS show that the proposed RD-SEC can enable up to $11\times$ improvement in variation tolerance and achieve up to $113\times$ reduction in the standard deviation of classification accuracy while incurring marginal degradation in the median classification accuracy.

Chapter 5 concludes this dissertation and provides directions for future research activities.

Chapter 2

BER-OPTIMAL ADC-BASED RECEIVER FOR SERIAL LINKS

ADC-based multi-Gb/s serial link receivers have gained increasing attention as a promising scheme for data transfer in data centers because they have enabled the application of digital signal processing (DSP) techniques to recover data under severe channel impairments such as channel loss, reflection, and crosstalk, while being constrained by a stringent power budget [22, 52–56]. This chapter presents the effectiveness of employing *information-based system metrics* such as the link BER to reduce the energy consumption of serial link components such as the ADC, which tends to be the most power hungry block. For example, the ADC itself (excluding the clock buffer) consumes 41% of the total receiver power in [22].

The rest of this chapter is organized as follows. Section 2.1 reviews the theory behind the BOA. Section 2.2 discusses how to maximize the benefits of a BOA receiver over a CUA receiver. In Section 2.3, the design of a 4GS/s, 4-bit BOA IC in a 90 nm CMOS process is described. Both stand-alone ADC and link measurement results are summarized in Section 2.4, and a summary of this chapter is provided in Section 2.5.

2.1 Background

In this section, the concept of the BOA is reviewed. Figure 1.5(a) depicts a typical ADC-based serial link, where the ADC is followed by an equalizer prior to detection. When considering an equivalent discrete-time, symbol-spaced, time-invariant channel corrupted by additive white Gaussian noise (AWGN), and 2-PAM modulation, the channel output at time n is given by

$$x_c[n] = x_c(nT) = \sum_{i=0}^{L-1} h[i]b[n-i] + v[n],$$

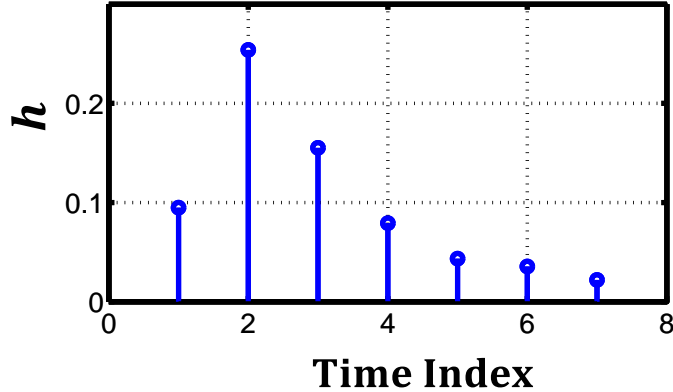


Figure 2.1: Channel response $\mathbf{h} = [0.0949, 0.2539, 0.1552, 0.0793, 0.0435, 0.0356, 0.0220]$ with $L = 7$ of a 20-inch backplane channel carrying 10 Gb/s data [57].

where $b[n] \in \{\pm 1\}$ is the transmitted sequence, $h[n]$ is the equivalent discrete time channel impulse response (see Fig. 2.1 for an example) with memory L , and $v[n]$ is AWGN with variance σ_v^2 . At the receiver, the processor estimates the transmitted symbols from quantized channel outputs through the ADC. A subsequent slicer determines the transmitted bit. Figure 2.1 shows an example of the channel response \mathbf{h} for a 20-inch backplane channel carrying 10 Gb/s data [57].

2.1.1 Comparison Metric

Comparison of a BOA and a CUA requires an appropriate metric to be defined. The ADC shaping gain $SG(p_e, B_u, B_o)$ in (1.1) is one such metric. In applications where it is difficult to measure the underlying circuit and environmental noise, comparing the ratio of the resulting bit-error-rates, or the “BER ratio”, may be of interest. Similarly, when two systems are being compared, and only one of the two experiences an exponential decay in BER with SNR as shown in Fig. 2.2(e), it may not be possible to measure $SG(p_e, B_u, B_o)$. However at a given SNR, the ratio of the measured BERs may be readily measurable. Once again, the BER ratio becomes a quantity of interest. In our application, as the resolution of the ADC may be insufficient to reach the so-called “waterfall” regime of the BER vs. SNR curve, we will use the BER ratio as a metric of comparison. We recognize that this metric may be more susceptible to measurement sensitivities since we are comparing quantities that may differ

by orders of magnitude. However, we proceed with this metric for the above mentioned reasons. Therefore, in this chapter we employ the *BER ratio* (BERR) defined as:

$$BERR(SNR, B_u, B_o) = \frac{p_{eu}(SNR, B_u)}{p_{eo}(SNR, B_o)},$$

where $p_{eu}(SNR, B_u)$ and $p_{eo}(SNR, B_o)$ are the BERs achieved by a B_u -bit CUA and a B_o -bit BOA with identical receiver processing and channel SNR given by $SNR = \sum_{i=0}^{L-1} |h[i]|^2 / \sigma_v^2$.

2.1.2 An Illustrative Example

A BOA [17] exploits signal statistics to maximize the probability of correctly detecting the transmitted bits. To provide insight into the operation of a BOA, we consider an ADC followed by a memoryless symbol-by-symbol maximum likelihood (ML) detector (ADC-ML) receiver, as shown in Fig. 2.2(a), and provide an example to illustrate the point.

First, we define the set of quantization thresholds for the CUA and the BOA.

Definition 1. *The vectors $\mathbf{t}_u = [t_{u,1}, t_{u,2}, \dots, t_{u,M}]$, $\mathbf{r}_u = [r_{u,1}, r_{u,2}, \dots, r_{u,M+1}]$, and the set $\mathbf{I}_u = \{I_{u,1}, I_{u,2}, \dots, I_{u,M+1}\}$ are the thresholds, output representation levels, and interval set of a $\log_2(M + 1)$ -bit CUA, where $I_{u,1} = (-\infty, t_{u,1}]$, $I_{u,k} = [t_{u,k-1}, t_{u,k}]$ for $k = 2, \dots, M$, $I_{u,M+1} = [t_{u,M}, +\infty)$, and the CUA output $x[n] = r_{u,k}$ if $x_c(nT) \in I_{u,k}$ for $k = 1, \dots, M + 1$. Similarly, the vectors $\mathbf{t}_o = [t_{o,1}, t_{o,2}, \dots, t_{o,N}]$, $\mathbf{r}_o = [r_{o,1}, r_{o,2}, \dots, r_{o,N+1}]$, and the set $\mathbf{I}_o = \{I_{o,1}, I_{o,2}, \dots, I_{o,N+1}\}$ are the thresholds, output representation levels, and interval set of a $\log_2(N + 1)$ -bit BOA, where $I_{o,1} = (-\infty, t_{o,1}]$, $I_{o,k} = [t_{o,k-1}, t_{o,k}]$ for $k = 2, \dots, N$, $I_{o,N+1} = [t_{o,N}, +\infty)$, and the BOA output $x[n] = r_{o,k}$ if $x_c(nT) \in I_{o,k}$ for $k = 1, \dots, N + 1$.*

Consider a 4-tap channel with impulse response $\mathbf{h} = [0.08, 0.07, 0.1, 0.04]$. The conditional probability density functions (pdfs) $P(x_c[n]|b[n] = -1)$ and $P(x_c[n]|b[n] = +1)$ corresponding to the marginal pdf of the channel output conditioned on the bit $b[n]$ being either -1 or $+1$ at time n , are illustrated in Fig. 2.2(b) and Fig. 2.2(c), respectively. In a CUA, the thresholds are set uniformly within the ADC's FSR. Assuming that the FSR is $[-0.3, +0.3]$, a 4-bit CUA will have its thresholds at $\mathbf{t}_u = [\pm 0.26005, \pm 0.2290, \pm 0.18575, \pm 0.14875, \pm 0.1145, \pm 0.0743, \pm 0.03715, 0]$ (Fig. 2.2(d)). In contrast, the thresholds in a BOA are positioned at

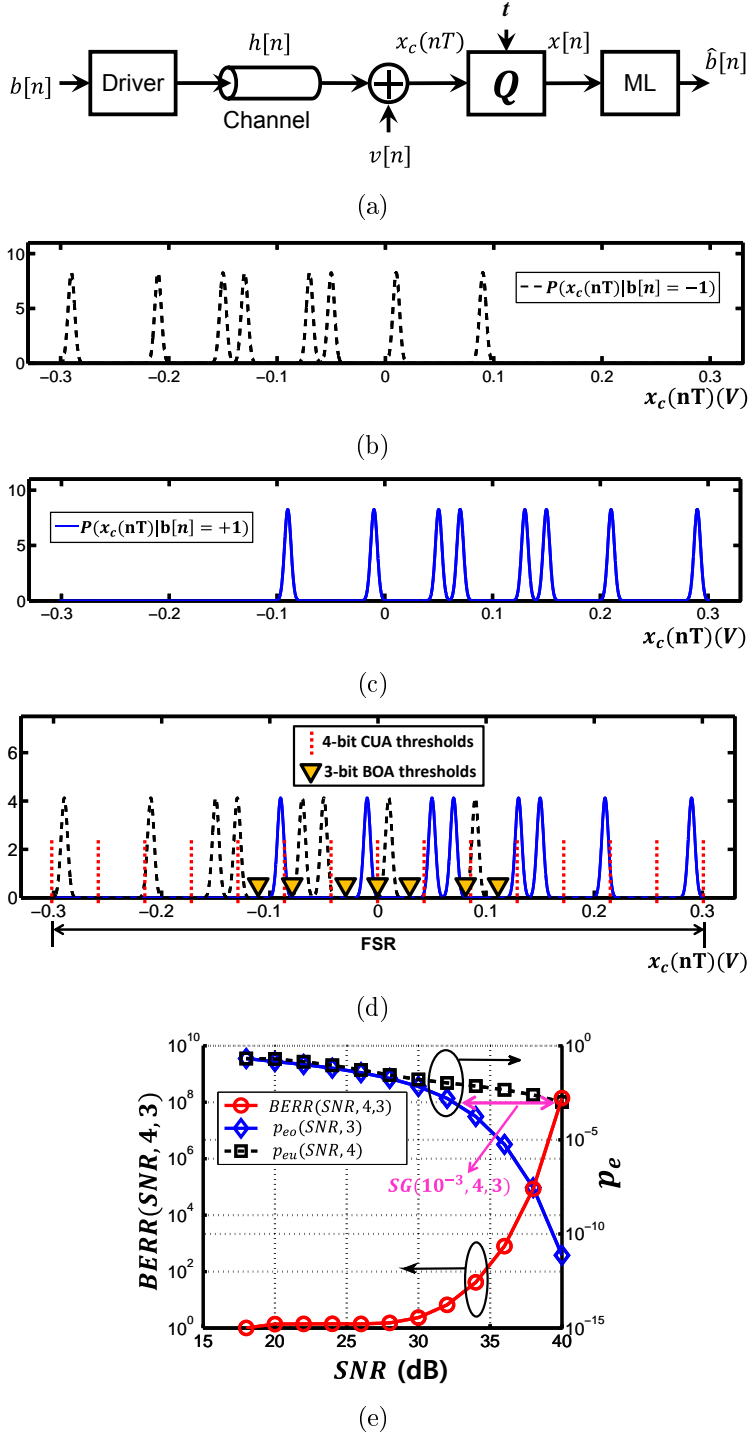


Figure 2.2: An illustrative example: (a) the block diagram of an ADC-ML receiver, (b) the conditional pdf of the channel output given $b[n] = -1$, (c) the conditional pdf of the channel output given $b[n] = +1$, (d) BOA's quantization thresholds (inverted triangles in yellow) and uniform quantization thresholds (dashed lines in red) for channel $\mathbf{h} = [0.08, 0.07, 0.1, 0.04]$ when $SNR = 36$ dB, and (e) the simulated $BERR(SNR, 4, 3)$, $p_{eo}(SNR, 3)$ and $p_{eu}(SNR, 4)$ versus SNR plot.

the crossover points of the two conditional pdfs. For this example, the BOA's thresholds are found to be $\mathbf{t}_o = [\pm 0.11, \pm 0.08, \pm 0.03, 0]$ (see Fig. 2.2(d)) as there are 7 crossover points. Thus, the BOA illustrated here is a 3-bit ADC. Figure 2.2(e) shows that the BOA achieves a 1-bit reduction in the ADC resolution while achieving $BERR(40, 4, 3) \approx 10^8$ and $SG(10^{-3}, 4, 3) \approx 8$ dB.

In order to compute \mathbf{t}_o , we need the following definition of noise-free channel outputs:

Definition 2. The $\boldsymbol{\mu}$ -set of a channel $\mathbf{h} = [h_0, h_1, \dots, h_{L-1}]$ is an ordered set defined as $\boldsymbol{\mu} = \{\boldsymbol{\mu}^+ \cup \boldsymbol{\mu}^-\}$, where both $\boldsymbol{\mu}^+ = \{\mu_l^+\}_{l=1}^{2^{L-1}}$ and $\boldsymbol{\mu}^- = \{\mu_l^-\}_{l=1}^{2^{L-1}}$ are ordered sets of noise-free channel outputs conditioned on the transmitted symbol $b[n]$ taking the value $+1$ and -1 , respectively. The $\boldsymbol{\mu}$, $\boldsymbol{\mu}^+$ and $\boldsymbol{\mu}^-$ sets have elements in ascending order.

In general, the N thresholds of a BOA for an ADC-ML receiver can be obtained [17] as the N solutions for the unknowns $\{t_{o,i}\}$ ($i = 1, \dots, N$) to the following equation:

$$2^{-L+1} \sum_{l=1}^{2^{L-1}} \mathcal{N}(t_{o,i}; \mu_l^+, \sigma_n) = 2^{-L+1} \sum_{l=1}^{2^{L-1}} \mathcal{N}(t_{o,i}; \mu_l^-, \sigma_n), \quad (i = 1, \dots, N), \quad (2.1)$$

where $\mathcal{N}(x; \mu, \sigma_n) = \frac{1}{\sqrt{2\pi\sigma_n^2}} e^{-\frac{(x-\mu)^2}{2\sigma_n^2}}$, $N \leq 2^L - 1$, μ_l^+ and μ_l^- ($1 \leq l \leq 2^{L-1}$) are the 2^{L-1} noise-free channel outputs (see **Definition 2**). In the example shown in Fig. 2.2, $\mathbf{h} = [0.08, 0.07, 0.1, 0.04]$, $L = 4$, $\{\mu_l^+\}_{l=1}^8 = \{-0.09, -0.01, 0.05, 0.07, 0.13, 0.15, 0.21, 0.29\}$ and $\{\mu_l^-\}_{l=1}^8 = \{-0.29, -0.21, -0.15, -0.13, -0.07, -0.05, 0.01, 0.09\}$.

2.1.3 BOA with a Linear Equalizer (LE)

Consider a BOA followed by a K -tap linear equalizer (LE) with taps $\mathbf{w} = [w_0, w_1, \dots, w_{K-1}]$, such that the equalizer output $y_{eq}[n] = \sum_{k=0}^{K-1} w_k x[n-k]$. In a BOA, the representation levels $\mathbf{r}_o = \{r_{o,1}, r_{o,2}, \dots, r_{o,N+1}\}$ and the thresholds \mathbf{t}_o are chosen to minimize the link BER. Obtaining a closed form expression for the BOA's representation levels in the presence of channel ISI and a LE is in general intractable. Therefore, the gradient descent algorithm [17] is employed to compute the representation levels iteratively as follows:

$$BER = f(h, \mathbf{r}_o, \mathbf{t}_o, \mathbf{w}, \sigma_n)$$

$$\Delta BER = f(h, \mathbf{r}_o + \Delta \mathbf{r}_o, \mathbf{t}_o + \Delta \mathbf{t}_o, \mathbf{w}, \sigma_n) - f(h, \mathbf{r}_o, \mathbf{t}_o, \mathbf{w}, \sigma_n), \quad (2.2)$$

$$\begin{aligned} \mathbf{r}_o[j] &= \mathbf{r}_o[j-1] + \mu \left(\frac{\partial BER}{\partial \mathbf{r}_o} \right) \Big|_{\mathbf{r}_o = \mathbf{r}_o[j-1]} \\ &\approx \mathbf{r}_o[j-1] + \mu \left(\frac{\Delta BER}{\Delta \mathbf{r}_o} \right) \Big|_{\mathbf{r}_o = \mathbf{r}_o[j-1]}, \end{aligned} \quad (2.3)$$

where it is assumed that $BER = f(h, \mathbf{r}_o, \mathbf{t}_o, \mathbf{w}, \sigma_n)$ is known, and $\mathbf{r}_o[j] = \{r_{o,1}[j], r_{o,2}[j], \dots, r_{o,N+1}[j]\}$ are the ADC representation levels in the j^{th} iteration of the gradient search. The thresholds in the j^{th} iteration are obtained as follows:

$$t_{o,i}[j] = \frac{r_{o,i}[j] + r_{o,i+1}[j]}{2}, (i = 1, \dots, N). \quad (2.4)$$

The BOA's representation levels adaptation algorithm is as follows. First, the ADC parameters \mathbf{r}_o and \mathbf{t}_o are initialized. Then, the gradient vector is estimated by computing finite differences based on (2.3). The next step is to update \mathbf{t}_o using (2.4). The last two steps are repeated until the BER converges, i.e., when the difference in the BER between adjacent iterations is less than a pre-specified value.

2.2 Achievable BER Improvement via BOA

In this section, we study through analysis and simulations how to maximize the benefits of the BOA over the CUA. Note: a BOA receiver always achieves the same if not better BER as compared to a CUA receiver, given the same number of bits, channel and noise power, because a CUA is a special case of the BOA. An important question to ask is: Under what conditions are the benefits offered by a BOA over a CUA substantial? In particular, we wish to determine channel conditions under which $BERR(SNR, B_u, B_o)$ is say at least $10\times$. $BERR(SNR, B_u, B_o)$ is empirically observed to depend strongly on the difference between the CUA's and the BOA's thresholds, the number of adjacent noise-free channel

outputs with opposing signs, channel SNR and the ADC resolution. We therefore discuss the relationship between these factors on $BERR(SNR, B_u, B_o)$. We restrict our analysis to channels with memory $L < 7$ in order to enable the derivation of useful insights analytically. Note that the performance of BOA for channels with large memory $L > 7$ has been studied in [18].

In this section, for tractability of analysis, we assume that the ADC (BOA or CUA) is followed by a memoryless symbol-by-symbol ML decoder and that binary phase-shift keying (BPSK) signaling is used over a known channel with impulse response \mathbf{h} . Thus, dropping the time index ‘ n ’, and employing the notation X_c and X_u to represent the random variables (RVs) corresponding to $x_c(nT)$ and $x[n]$, respectively, for a CUA, we have:

$$P(X_u = r_{u,k}|b) = P(X_c \in I_{u,k}|b), \quad (k = 1, \dots, M + 1),$$

where $X_u \in \{r_{u,1}, r_{u,2}, \dots, r_{u,M+1}\}$, $r_{u,k}$ is the k^{th} CUA representation level (see **Definition 1**). Then, the memoryless ML decision rule for a CUA is given by:

$$\hat{b} = \begin{cases} +1, & \text{if } \frac{P(X_u|b=+1)}{P(X_u|b=-1)} > 1 \\ -1, & \text{otherwise} \end{cases}.$$

Similarly, let X_o be the RV corresponding to $x[n]$ for a BOA. Then,

$$P(X_o = r_{o,k}|b) = P(X_c \in I_{o,k}|b), \quad (k = 1, \dots, N + 1),$$

where $X_o \in \{r_{o,1}, r_{o,2}, \dots, r_{o,N+1}\}$ and $r_{o,k}$ is the k^{th} BOA representation level (see **Definition 1**). Then, the memoryless ML decision rule for a BOA is given by:

$$\hat{b} = \begin{cases} +1, & \text{if } \frac{P(X_o|b=+1)}{P(X_o|b=-1)} > 1 \\ -1, & \text{otherwise} \end{cases}.$$

2.2.1 BERR Expression

We wish to analytically predict $BERR(SNR, B_u, B_o)$ given its arguments and the channel \mathbf{h} . Such analysis will eliminate the need for expensive Monte Carlo (MC) simulations.

Furthermore, conditions under which a BOA can offer a $BERR(SNR, B_u, B_o)$ of 10^r can be derived.

First, the following definitions are provided.

Definition 3. A channel \mathbf{h} is said to be m -clustered if there are m transitions (μ -transitions) in its $\boldsymbol{\mu}$ -set, where a μ -transition occurs when an element of the $\boldsymbol{\mu}^+$ set is followed by an element of the $\boldsymbol{\mu}^-$ set or vice versa. Note: $m > 0$ and takes odd values only, and $m > N$ at low SNR scenario while $m = N$ at high SNR scenario.

Definition 4. The threshold non-uniformity metric h_t of a $\log_2(N+1)$ -bit BOA is a measure of the difference between \mathbf{t}_o and \mathbf{t}_u , and is defined as:

$$h_t = \frac{-1}{\log_2\left(\frac{N+1}{2}\right)} \sum_{i=2}^{(N+1)/2} \left[\left(\frac{t_{o,i} - t_{o,i-1}}{y_{max}} \right) \log_2 \left(\frac{t_{o,i} - t_{o,i-1}}{y_{max}} \right) + \left(\frac{t_{o,1} + y_{max}}{y_{max}} \right) \log_2 \left(\frac{t_{o,1} + y_{max}}{y_{max}} \right) \right] \quad (2.5)$$

where $[-y_{max}, y_{max}]$ is the ADC FSR, and only the non-positive BOA thresholds are used since the BOA's thresholds are symmetric about the origin. Note: $0 \leq h_t \leq 1$, and the larger the value of h_t the closer are the BOA's thresholds to those of the CUA.

Figure 2.3(a) shows an example when $m = 3$ for a 4-tap channel (thus there are $2^4 = 16$ elements in $\boldsymbol{\mu}$), and Fig. 2.3(b) illustrates two examples when $h_t = 0.7564$ and $h_t = 1$, respectively. Algorithm 1 can be employed to obtain m and h_t for a specific channel.

Definition 5. Let $d_{o,k}^* = \min_{\mu \in \boldsymbol{\mu}} (|t_{o,k} - \mu|)$ be the minimum distance of the k^{th} BOA threshold $t_{o,k}$ ($k = 1, \dots, N$) to the nearest noise-free channel output μ . Then, the minimum BOA distance $d_{o,min} = \min_{1 \leq k \leq N} (d_{o,k}^*)$.

Definition 6. Each of the $(M+1)$ intervals $I_{u,k}$ with $k = 1, \dots, M+1$, in a CUA has a dominant noise-free channel output μ_k^* given by

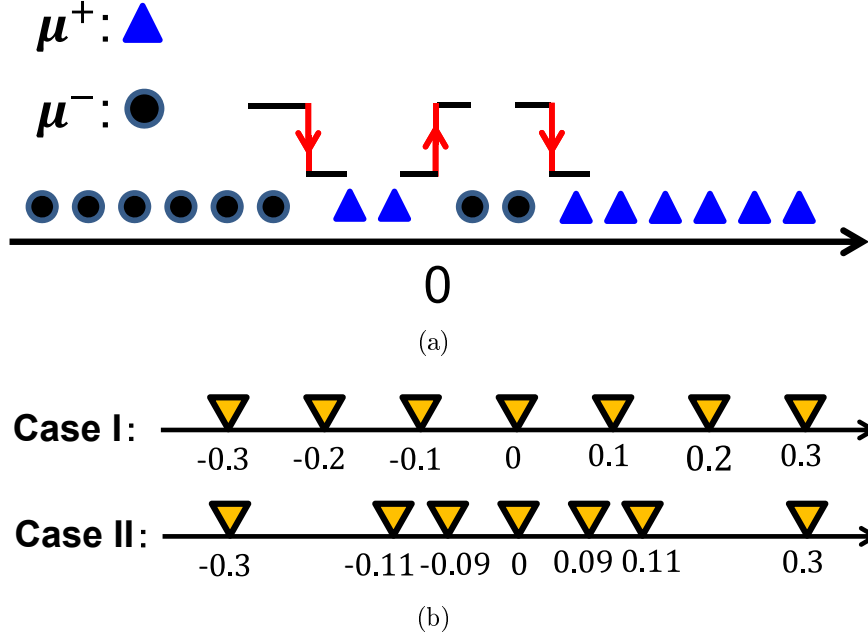


Figure 2.3: Examples of m -clustering and h_t : (a) m -clustering with $m = 3$, and (b) h_t for $\mathbf{t}_o = [\pm 0.3, \pm 0.2, \pm 0.1, 0]$ (case I where $h_t = 1$) and $\mathbf{t}_o = [\pm 0.3, \pm 0.11, \pm 0.09, 0]$ (case II where $h_t = 0.7564$) with $y_{max} = 0.3$.

$$\mu_k^* = \begin{cases} \arg \max_{\mu_l^- \in \mu^-} \left[\int_{I_{u,k}} \mathcal{N}(x; \mu_l^-, \sigma_n) dx \right], & \text{if } \frac{P(X_u=r_{u,k}|b=+1)}{P(X_u=r_{u,k}|b=-1)} > 1 \\ \arg \max_{\mu_l^+ \in \mu^+} \left[\int_{I_{u,k}} \mathcal{N}(x; \mu_l^+, \sigma_n) dx \right], & \text{otherwise} \end{cases}$$

Definition 7. Let $d_{u,k}^* = -\min(\mu_k^* - t_{u,k-1}, t_{u,k} - \mu_k^*)$ be the minimum distance of the k^{th} CUA's dominant noise-free channel output μ_k^* from the boundaries of the k^{th} interval $I_{u,k}$. Then, the minimum CUA distance $d_{u,min} = \min_{1 \leq k \leq (M+1)} (d_{u,k}^*)$.

Figure 2.4 shows an example of the marginal pdf of the channel output, illustrating the

Algorithm 1 Algorithm to obtain m -clustered value and h_t for an ADC-ML receiver.

1. Initialize the channel \mathbf{h} and the SNR, calculate noise variance σ_v^2 based on \mathbf{h} and the SNR.
 2. Define the main cursor of \mathbf{h} , calculate $\mu^+ = \{\mu_i^+\}_{i=1}^{2^L-1}$ and $\mu^- = \{\mu_i^-\}_{i=1}^{2^L-1}$, respectively, and obtain the ordered set $\mu = \{\mu^+ \cup \mu^-\}$.
 3. Count the number of transitions in μ , which is the m -clustered value.
 4. Obtain \mathbf{t}_0 using equation (2).
 5. Calculate h_t using equation (6).
-

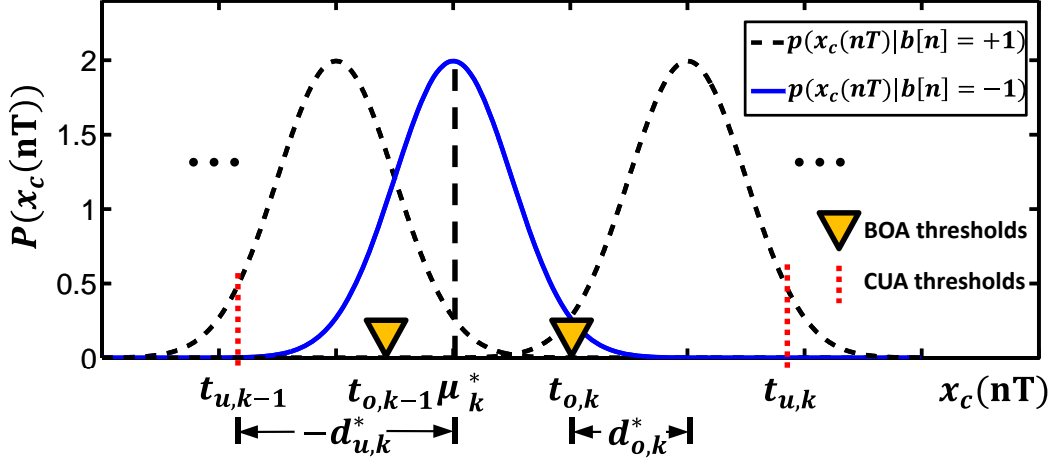


Figure 2.4: An illustrative example for $d_{o,k}^*$, μ_k^* and $d_{u,k}^*$.

corresponding $d_{o,k}^*$, μ_k^* and $d_{u,k}^*$.

In Section 2.6, we show that $BERR(SNR, B_u, B_o)$ is given by:

$$\begin{aligned}
 & BERR(SNR, B_u, B_o) \\
 & \approx \begin{cases} \frac{d_{o,min}}{2d_{u,min}} e^{\frac{d_{o,min}^2 - d_{u,min}^2}{2\sigma_n^2}}, & \text{if } d_{u,min} > 0 \\ \sqrt{\frac{\pi}{2}} \frac{d_{o,min}}{\sigma_n} e^{\frac{d_{o,min}^2}{2\sigma_n^2}}, & \text{if } d_{u,min} < 0. \\ \sqrt{\frac{\pi}{8}} \frac{d_{o,min}}{\sigma_n} e^{\frac{d_{o,min}^2}{2\sigma_n^2}}, & \text{if } d_{u,min} = 0 \end{cases} \quad (2.6)
 \end{aligned}$$

Equation (2.6) indicates that $BERR(SNR, B_u, B_o)$ increases with $d_{o,min}^2$ or $(d_{o,min}^2 - d_{u,min}^2)$. Furthermore, $BERR(SNR, B_u, B_o)$ can be predicted given the channel \mathbf{h} (thus $\mu^-, \mu^+, d_{o,min}^2$ and $d_{u,min}^2$) and SNR .

MC simulations of link BER were run with 10^8 (for $SNR \leq 36$ dB) or 10^{12} (for $SNR > 36$ dB) samples and for SNR ranging from 18 dB to 40 dB for channels $\mathbf{h} = [0.09, 0.1, 0.08, -0.05]$ and $\mathbf{h} = [0.08, 0.07, 0.1, 0.04]$, respectively. Figure 2.5 indicates that the analytical expressions (2.9) and (2.12) can predict the results of the MC simulations to within an order of magnitude, and thus can be employed to estimate the p_{eo} and p_{eu} . Furthermore, as expected, the expressions (2.9) and (2.12) become more accurate at high SNRs. Finally, it can be seen in Fig. 2.5 that 3-bit BOA achieves a shaping gain of 2 dB (8 dB) over a 6-bit

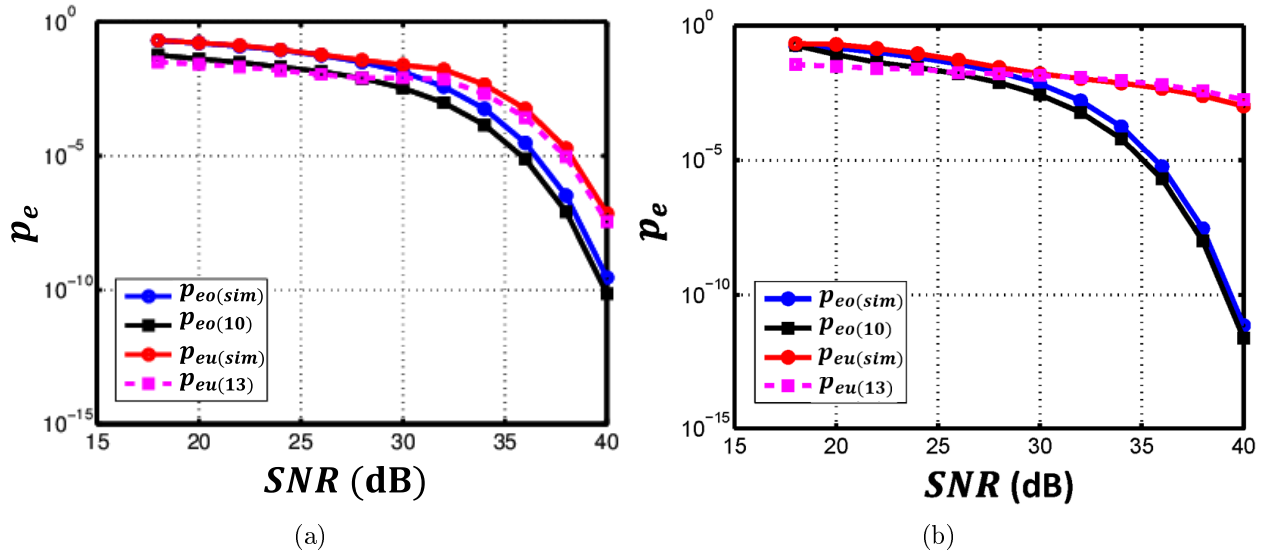


Figure 2.5: Comparison among p_{eo} from MC simulation ($p_{eo(sim)}$), p_{eo} estimated using (2.9) ($p_{eo(10)}$), p_{eu} from MC simulation ($p_{eu(sim)}$), and p_{eu} estimated using (2.12) ($p_{eu(13)}$), for channels (a) $\mathbf{h} = [0.09, 0.1, 0.08, -0.05]$ when $B_u = 6$ and $B_o = 3$, and (b) $\mathbf{h} = [0.08, 0.07, 0.1, 0.04]$ when $B_u = 4$ and $B_o = 3$, respectively.

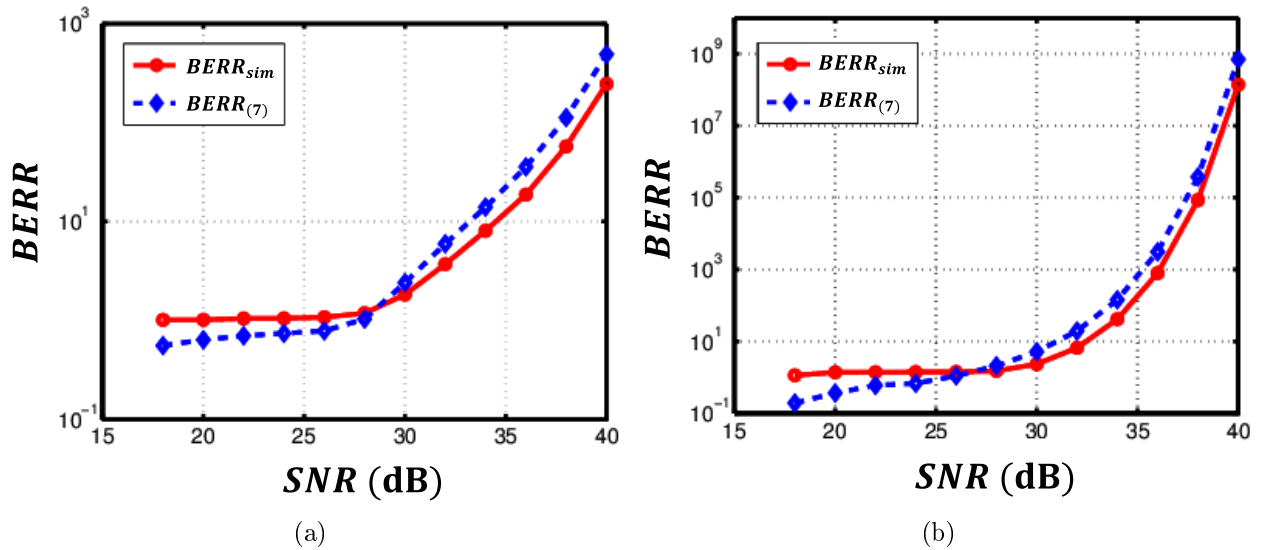


Figure 2.6: Comparison between $BERR(SNR, B_u, B_o)$ from MC simulation ($BERR_{sim}$), and $BERR(SNR, B_u, B_o)$ estimated using (2.6) ($BERR_{(7)}$) for channels (a) $\mathbf{h} = [0.09, 0.1, 0.08, -0.05]$ when $B_u = 6$ and $B_o = 3$, and (b) $\mathbf{h} = [0.08, 0.07, 0.1, 0.04]$ when $B_u = 4$ and $B_o = 3$, respectively.

(4-bit) CUA at a BER of 10^{-5} (10^{-3}). Figure 2.6 shows that BERR can also be predicted via (2.6) to within an order of magnitude of MC simulations, and that it increases with SNR.

2.2.2 BER Improvement vs. Channel ISI

This subsection presents an empirical study of BERR as a function of channel ISI. In the rest of this subsection, we consider the special case of $\log_2(m+1)$ -bit BOA and CUA, i.e., $B_u = B_o = \log_2(m+1)$. Specifically, we study the relationship between $BERR(SNR, B_u, B_o)$ and m -cluster and h_t , for a set of 4-tap channels $\mathbf{h} = [1, a_1, a_2, a_3]$ ($a_i = 0.1 : 0.1 : 0.9$, $i = 1, 2, 3$) using an ADC-ML receiver. The corresponding results are shown in Fig. 2.7, where $BERR(SNR, B_u, B_o)$ is calculated using BER expressions (2.9) and (2.12) when $SNR = 38$ dB. Note: the results when $p_{e,u} > 10^{-1}$, which occurs because $m = 1$ or the channel ISI is too large for the given SNR, are removed in Fig. 2.7 for better illustration. The $BERR(SNR, B_u, B_o)$ and h_t for the channel (under which the measured results are shown in Fig. 2.17) is also shown in Fig. 2.7, which has $\log_2(m+1) = 2.6 \approx 3$ and $h_t \approx 0.46$. Figure 2.7 indicates that smaller h_t and larger m combinations are likely to result in larger $BERR(SNR, B_u, B_o)$. Furthermore, Fig. 2.7 shows that $BERR(SNR, B_u, B_o) > 10^6$, when $m \geq 5$ and $h_t \leq 0.8$.

2.2.3 BER Improvement vs. Number of Bits B_x in the ADC

In this subsection, we claim that for any given realization of the type considered in this section, there exists an optimal ADC resolution in the sense that it achieves the maximum of the function $BERR(SNR, B_x, B_x)$. To see this, note that $p_{e,u}$ decreases with B_x , which is a consequence of $p_{eu}(SNR, B_x + 1)$ being defined through a maximization over the set of decision regions that includes those used to achieve $p_{eu}(SNR, B_x)$ as a subset. Except on a set of measure zero (for which $p_{eu}(SNR, B_x) = p_{eo}(SNR, B_x)$), $p_{eu}(SNR, B_x + 1) < p_{eu}(SNR, B_x)$, for all B_x . On the other hand, we note that p_{eo} decreases with B_x for $B_x < \log_2(N+1)$, following the same reasoning. However, once $B_x = \log_2(N+1)$, further increases in B_x provide no additional improvement in p_{eo} (for the memoryless symbol-by-

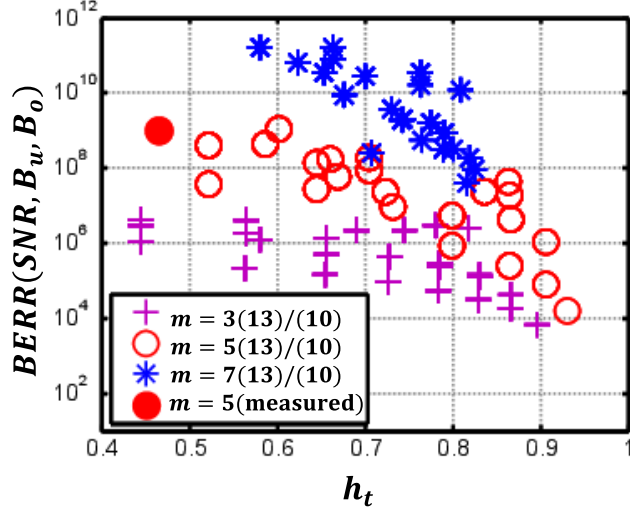


Figure 2.7: $BERR(SNR, B_u, B_o)$ vs. h_t , where $B_u = B_o = \log_2(m + 1)$, for channels $\mathbf{h} = [1, a_1, a_2, a_3]$ ($a_i = 0.1 : 0.1 : 0.9$, $i = 1, 2, 3$) using an ADC-ML receiver when $SNR = 38$ dB. The value of h_t and measured $BERR(SNR, 4, 3)$ for a FR4 channel using an ADC-LE receiver (described in section 2.3) are also shown.

symbol detector used in this analysis). As a result, the ratio $BERR(SNR, B_x, B_x)$ will monotonically decrease for $B_x > \log_2(N + 1)$. Hence, it must achieve a maximal value for one (or more) $B_x \leq \log_2(N + 1)$. The $BERR(SNR, B_x, B_x)$ versus B_x plot in Fig. 2.8(a) shows that $BERR(36 \text{ dB}, B_x, B_x)$ is maximized when $B_x = 3$.

The intuition gained from the analysis of the ADC-ML receiver holds for the ADC-LE receivers. This is confirmed by the simulated $BERR(SNR, B_x, B_x)$ versus B_x plot shown in Fig. 2.8(b), which is obtained for a FR4 channel at 10 Gb/s in Fig. 2.1, when $SNR = 36$ dB. Figure 2.8(b) also shows, under the given condition, $B_x = 3$ maximizes the $BERR(SNR, B_x, B_x)$.

2.3 Implementation of BOA Receiver

In this section, a prototype IC implementation of BOA receiver is described. Figure 2.9 shows the block diagram of the receiver, which consists of a BOA chip and an Altera FPGA board. The FPGA board implements the back-end DSP blocks. The ADC chip includes a reconfigurable 4-bit 4GS/s flash ADC and an 8-bit digital-to-analog converter (DAC). The 8-bit DAC enables variable-threshold and variable-resolution ADC configurations, so that

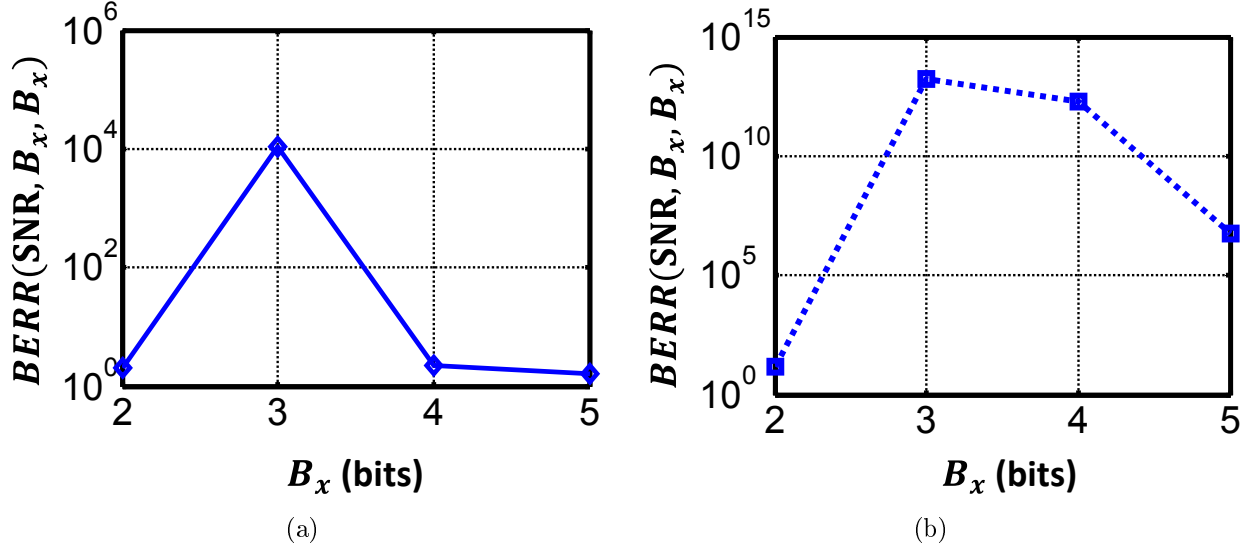


Figure 2.8: $BERR(SNR, B_x, B_x)$ vs. B_x of (a) an ADC-ML receiver, when the channel impulse response is $\mathbf{h} = [0.09, 0.1, 0.08, 0.04]$ and $SNR = 36$ dB, and (b) an ADC-LE receiver, when the channel impulse response is equal to the FR-4 channel (see Fig. 2.1) and $SNR = 36$ dB.

the performance of a CUA receiver can be compared with that of the BOA receiver. The back-end DSP block includes a LE and QL-UD. In the BOA receiver, the ADC quantizes the channel outputs and provides these into the back-end DSP block, which implements an adaptive LE. Once the equalizer coefficients converge, the quantization levels \mathbf{r}_o that minimize link BER are obtained using gradient descent search algorithm. The updated quantization thresholds \mathbf{t}_o are then fed back into the ADC chip.

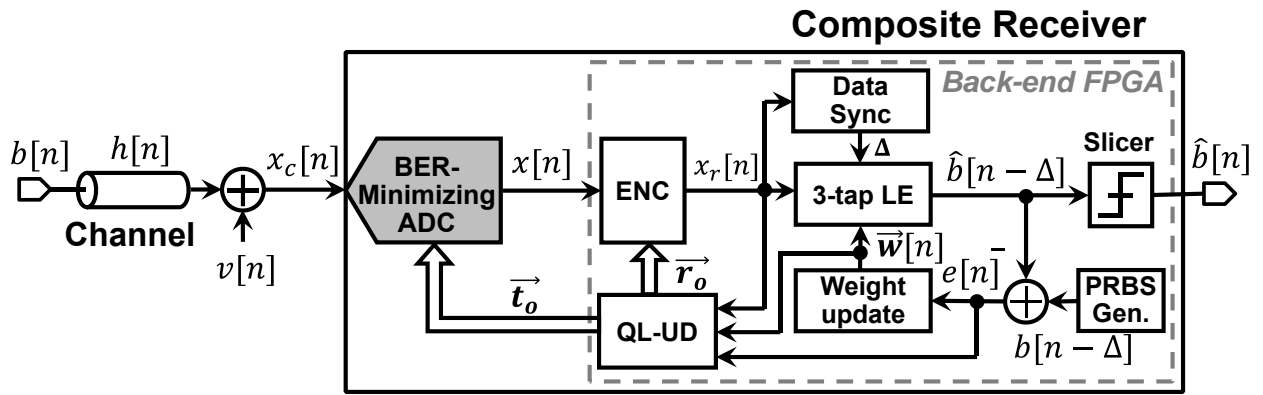


Figure 2.9: Block diagram of the BOA receiver.

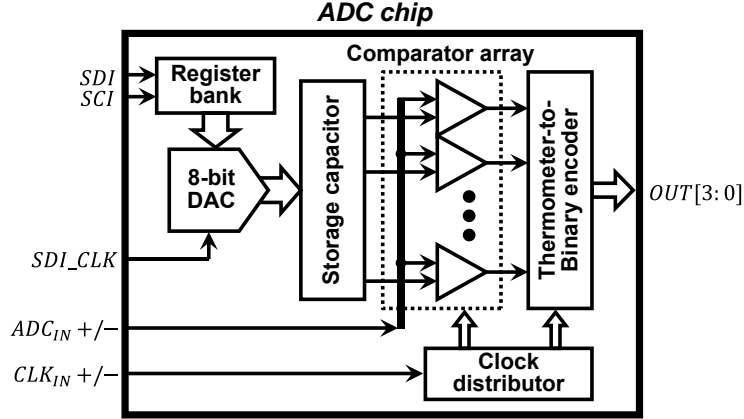


Figure 2.10: Block diagram of the BOA chip.

2.3.1 ADC Full-chip Block Diagram

The ADC chip consists of an 8-bit DAC, storage capacitor array, and a 4-bit flash ADC, as illustrated in Fig. 2.10. In a conventional flash ADC, the threshold voltages are generated by a resistor ladder. A BOA, however, requires variable thresholds. Thus, a DAC is employed for threshold generation. System analysis shows that an 8-bit DAC is required to ensure that the 3-bit BOA receiver can achieve similar or better BER performance compared to a 4-bit CUA receiver. In principle, 30 DACs are needed for 4-bit ADC threshold generation. To minimize power and area overhead, a single-core, multiple-output passive DAC, which is an extension of the single-core single-output passive DAC presented in [58], is proposed to generate the variable threshold voltages. The threshold voltage generator has a power overhead of 10% ($\sim 50 \mu\text{W}$) compared to a fixed resistor string for a CUA.

Figure 2.11 illustrates the operation of the 8-bit DAC. A single voltage threshold update occurs over two phases of non-overlapping clocks ϕ_1 and ϕ_2 . In phase I ($\phi_1 = 1, \phi_2 = 0$), the 4 MSBs of the 8-bit DAC input selects a 4-bit section of the resistor ladder to charge the 4-bit unit capacitor (C_u) array. In phase II ($\phi_1 = 0, \phi_2 = 1$), C_u and C_{ref} are connected together. The resulting charge sharing shifts the threshold voltage toward the desired value. The nominal, post-layout extracted operating frequency of the DAC core is 12 MHz, resulting in an update frequency per C_{ref} of 375 kHz. This is more than sufficient to compensate for leakage. The 8-bit DAC updates the thresholds of the comparator array

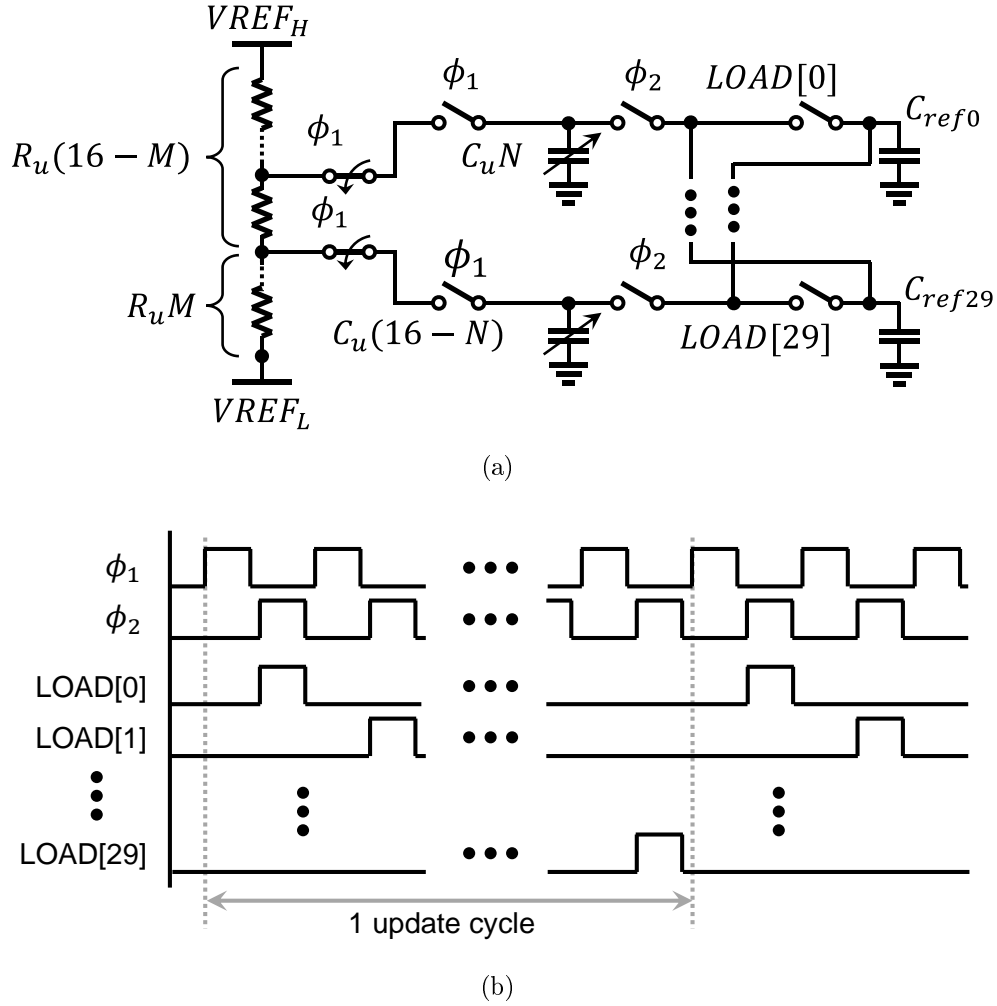


Figure 2.11: The 8-bit single-core multiple-output DAC: (a) circuit schematic, and (b) timing diagram.

sequentially. Therefore, only the threshold of one comparator is updated at each moment. The storage capacitors hold the thresholds for the other comparators. In a flash ADC, one input of each comparator is connected to the analog input, while the other input is connected to the corresponding threshold. The comparator array compares the analog input with the threshold voltages simultaneously and produces comparison results in the form of a thermometer code. Thus, a binary encoder following the comparator array is needed for the ease of back-end digital processing.

2.3.2 Back-end DSP in the FPGA

The back-end DSP units are implemented in an Altera Transceiver Signal Integrity Development FPGA board [59], in which the DSP units operate at a frequency of 100 MHz. Thus, 40 parallel channels are used to handle the 4 Gb/s outputs from the ADC chip. As shown in Fig. 2.9, the back-end DSP block consists of an encoder (ENC), data synchronization unit (Data Sync), LE, and QL-UD. The binary encoder converts the ADC output $x[n]$ into two's complement number representation $x_r[n]$, while the data synchronization unit provides the start position of the pseudorandom binary sequence (PRBS). A 3-tap least mean squares (LMS) adaptive equalizer is designed to compensate for channel ISI, while the QL-UD unit adjusts the ADC quantization thresholds \mathbf{t}_o and representation levels \mathbf{r}_o to achieve the minimum BER. As other blocks are standard DSP functional blocks, we focus on the implementation of the QL-UD unit. The equalizer computes an estimate of the transmitted symbol $b[n - \Delta]$ based on the encoder output $[x_r[n], \dots, x_r[n - M + 1]]^T$ as:

$$\hat{b}[n - \Delta] = \sum_{k=0}^{M-1} w[k]x_r[n - k].$$

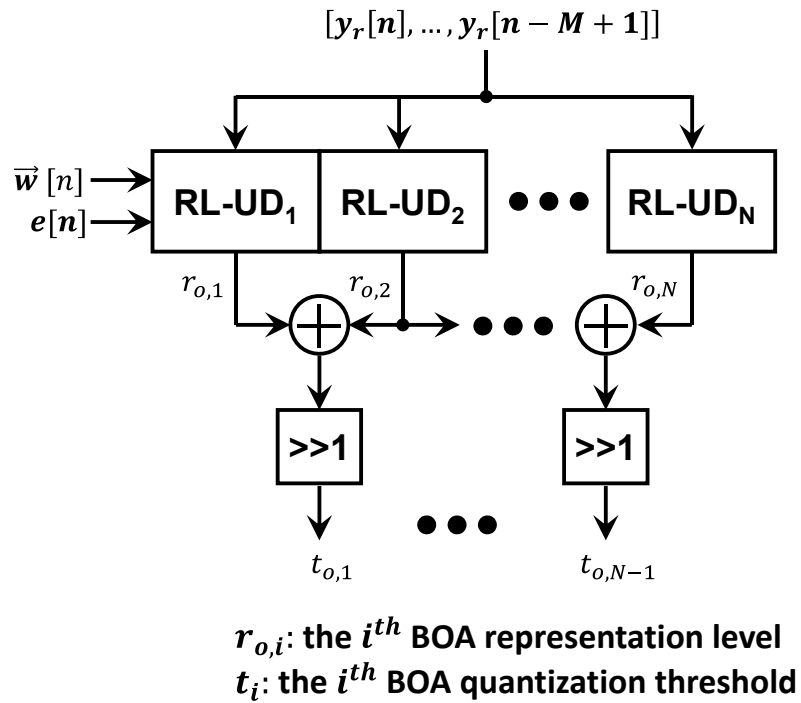
The estimation error $e[n] = b[n - \Delta] - \hat{b}[n - \Delta]$ is used to adjust the equalizer coefficients \mathbf{w} . Once \mathbf{w} converges, $e[n]$ can be used to update the ADC representation levels. Fixing the quantization thresholds \mathbf{t}_o , the optimal representation levels \mathbf{r}_o that minimize the MSE between $b[n]$ and $\hat{b}[n]$ can be obtained from gradient descent search. The LMS update of the quantization levels is obtained by approximating the gradient of the MSE by its instantaneous value,

$$r_{o,i}[n + 1] = r_{o,i}[n] + \mu_r e[n] \sum_{k: x_r[n-k]=r_{o,i}} \mathbf{w}[k].$$

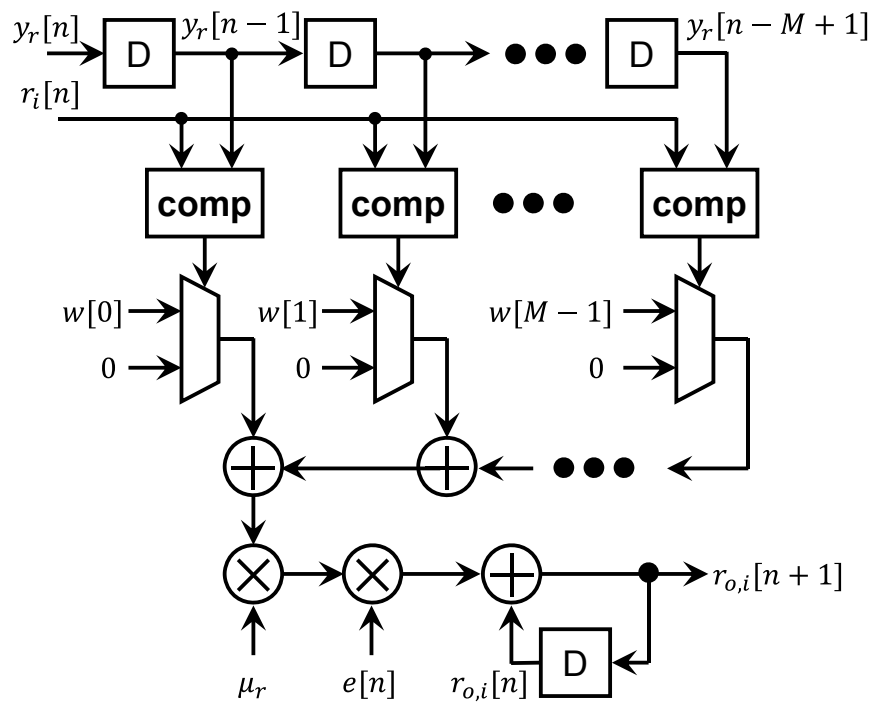
The LMS update can be further modified to the AMBER algorithm [15]:

$$r_{o,i}[n + 1] = r_{o,i}[n] + \mu_r I[n] \text{sgn}(e[n]) \sum_{k: x_r[n-k]=r_{o,i}} \mathbf{w}[k],$$

where $I[n]$ is the bit error indicator function. Figure 2.12 is the block diagram of the quantization level update unit and the individual RL-UD block, which update each quantization level based on the current ADC output, equalizer estimation error, and equalizer coefficients. The architecture of the update unit for each representation level $r_{o,i}$ is shown in Fig. 2.12(b).



(a)



(b)

Figure 2.12: Architectures of: (a) the QL-UD unit, and (b) the i^{th} RL-UD unit.

A total of 2^{B_o} such units are needed to update the representation levels \mathbf{r}_o . The total number of gates and power consumption of the QL-UD unit are estimated to be about 90K (NAND gates equivalent) and 12.3 mW, respectively. This power accounts for about 44% of the total power of the back-end DSP, and is expected to reduce with technology scaling. Note: the QL-UD unit can be turned off once BOA representation levels are obtained.

2.3.3 Comparator Design

The comparator consists of a preamplifier and 3 cascaded latches, and Fig. 2.13 illustrates the schematics of the preamplifier and latches. The preamplifier subtracts the analog input from the threshold and provides polarity of the comparison result. The cascaded latches amplify the preamplifier output to reduce the occurrence of meta-stability. The preamplifier design is shown in Fig. 2.13, which is widely employed for high-speed ADCs [12,13]. The first latch is a current-mode latch [12], which is composed of an input differential pair (M_1, M_2) and a cross-coupled regenerative latch pair (M_3, M_4) sharing the same resistive load, R_D . When CLK is high, the circuit is in tracking mode with low gain and large bandwidth. When the CLK is low, the circuit shifts to the regenerative mode, and the sampled signal from the tracking mode is amplified and delivered to the next stage. The second and third latches do not consume static power once they fully regenerate, and are referred to as dynamic latches [60], [61]. In this design, only the first latch uses a current mode latch because it is the most critical to guarantee accurate comparison results. For example, if the first latch does not have a large enough bandwidth to follow the updated polarity of the preamplifier output, the comparator may generate an incorrect output, regardless of how well the following latches perform. It should be noted that the cascaded latches operate in a pipelined manner for speed consideration. In particular, when the preceding latch is working in a tracking mode, the subsequent latch is working in regeneration mode, and vice versa.

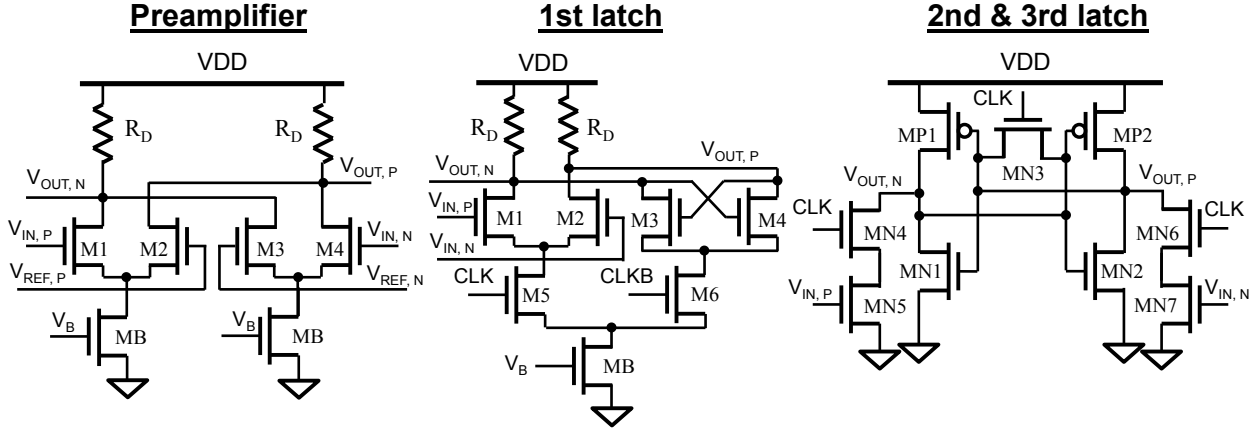


Figure 2.13: Schematics of the preamplifier and latches.

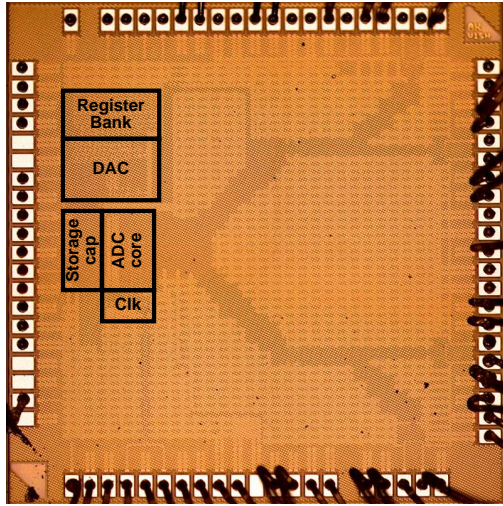
2.3.4 Encoder Design

A Gray encoder is used because it is more compact and faster than a summing encoder [13]. Since less pipelining is required for multi-gigahertz operation, the Gray encoder consumes less power than a summing encoder. There are three steps to encoding. First, the thermometer code is converted to a 1-of-N code. And then, the 1-of-N code is converted to a Gray code to suppress bubble errors. The Gray code is finally converted to binary code by XOR gates, for the purpose of further DSP processing in the subsequent blocks. In this chip, pipelined D flip-flops (DFFs) are used between adjacent logic gates in the encoder to guarantee 4 Gb/s operations.

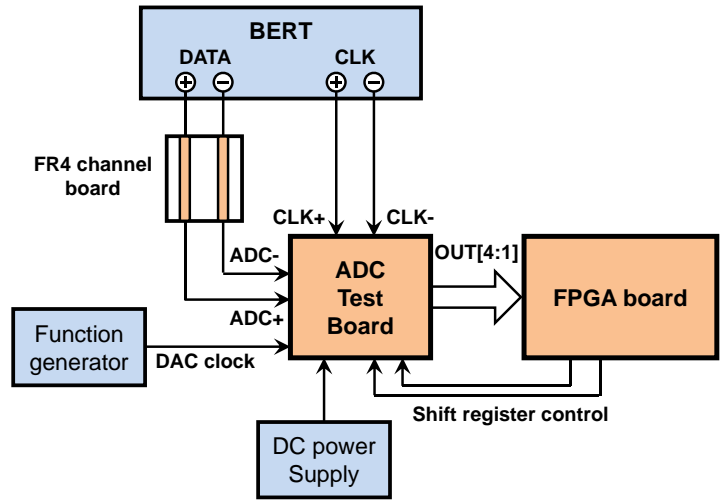
2.4 Measurement Results

This section summarizes the measurement results of both the stand-alone ADC chip and the ADC-based receiver. A 4-bit 4 GS/s ADC chip is fabricated in a 90 nm low power (LP) CMOS technology with an active area of 0.33 mm² and tested in a chip-on-board assembly. The chip's micrograph is shown in Fig. 2.14(a).

First, we ensure the standalone ADC performance is sufficient to support link operation. Since the ADC chip includes an 8-bit DAC for variable-threshold and variable-resolution ADC configuration, we utilize it to configure the ADC's threshold voltages for calibration.

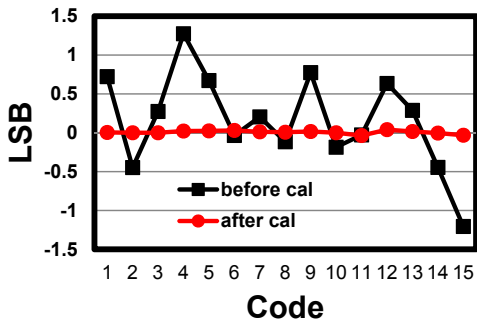


(a)



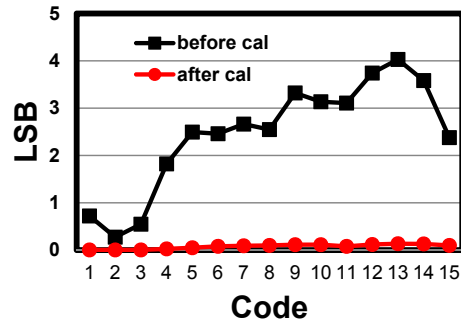
(b)

Figure 2.14: (a) Micrograph of the BOA chip, and (b) the test set-up.



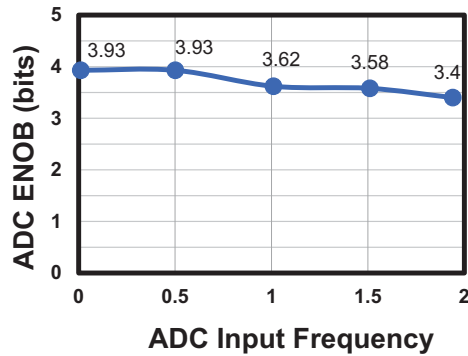
Before cal: +/- 1.3 LSB
After cal: +/- 0.04 LSB

(a)



Before cal: 4 LSB
After cal: 0.14 LSB

(b)



(c)

Figure 2.15: Standalone ADC measurement results: (a) DNL and (b) INL characteristics before/after calibration, and (c) measured ENOB vs. input frequency.

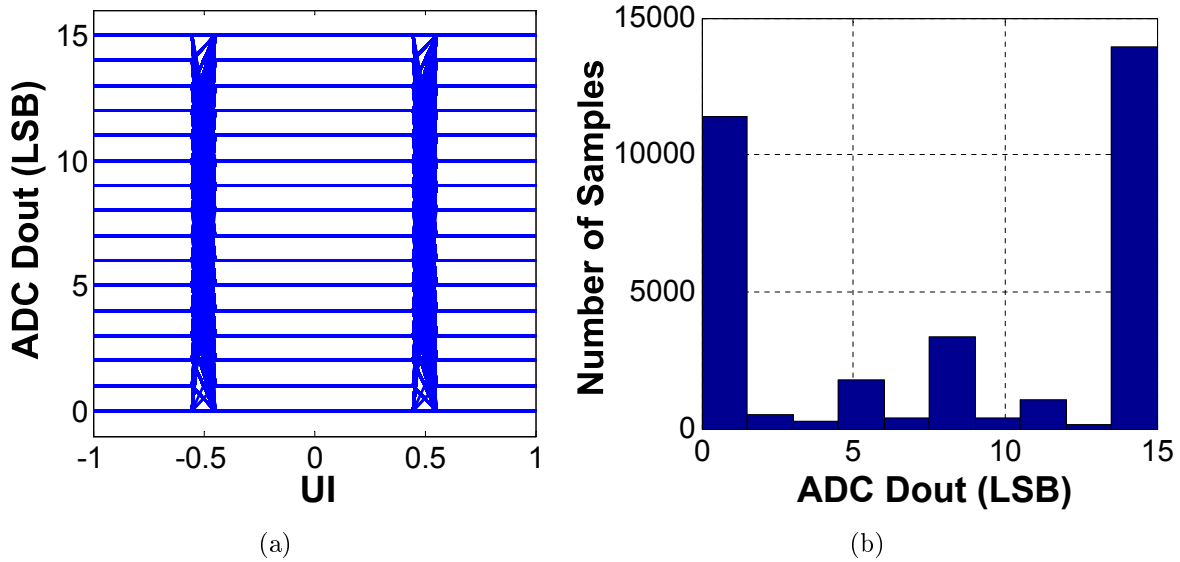


Figure 2.16: Measured ADC output: (a) eye diagram, and (b) histogram for a 20-inch FR4 channel at 4 Gb/s when TX amplitude is $180\text{ mV}_{\text{ppd}}$ and ADC FSR is $100\text{ mV}_{\text{ppd}}$.

The measured DNL and INL characteristics before and after calibration are shown in Fig. 2.15(a) and (b), respectively. The DNL and INL are reduced from ± 1.3 LSB and 4 LSB to ± 0.04 LSB and 0.14 LSB, respectively, after calibration, indicating the effectiveness of the calibration process. Figure 2.15(c) illustrates that the ADC can achieve up to 3.4 bits of ENOB at near-Nyquist rate input frequency of 1.9375 GHz. The figure of merit (FOM) of the ADC is 1.42 pJ/conv.step at 4 GS/s excluding clock buffers, which is comparable to the state-of-art using a similar technology [13, 14].

Figure 2.14(b) shows the block diagram of the link test set-up, which mainly consists of a BER tester (BERT), channel board, ADC PCB board, and FPGA board. The BERT provides 4 Gb/s synchronous data and clock, with the data passing through a 20-inch FR4 channel before entering the ADC board. The ADC chip quantizes the incoming analog signal and its outputs are fed into the FPGA board. The back-end DSP units in the FPGA then perform equalization and optimal ADC representation levels search. Finally, the updated representation levels are fed back to the ADC chip. However, in our experiment, the BOA's representation levels are obtained off-line due to a synchronization problem in the interface between the BOA chip and the FPGA board.

Link tests were conducted at 4 Gb/s over a 20-inch channel with $2^{23} - 1$ PRBS data.

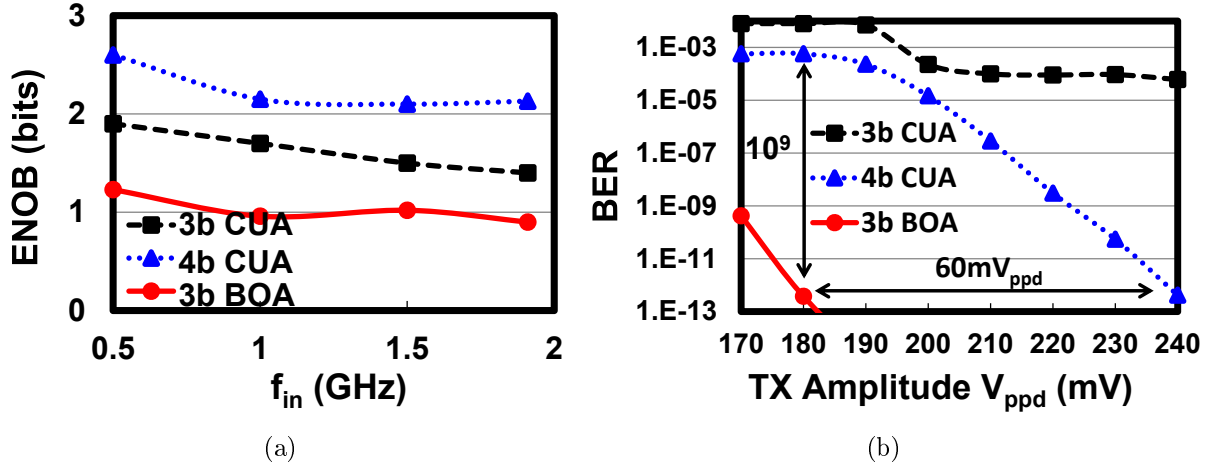


Figure 2.17: ENOB and BER measurements: (a) ENOB vs. input frequency, and (b) BER vs. TX amplitude at 4Gb/s when the FSR of the CUA is $100mV_{ppd}$.

The BOA's representation levels were obtained by first extracting the converged equalizer coefficients with the ADC IC in a 4-bit uniform mode, followed by an off-line adaptive channel estimation and gradient search procedure [17]. The ADC representation levels were then manually set in the lab. Figure 2.16 shows the post-ADC eye diagram and histogram of the ADC code at a 20-inch FR4 channel, with TX amplitude of $180mV_{ppd}$ and a CUA FSR of $100mV_{ppd}$, which indicate that the received eye is closed. In particular, the channel loss at Nyquist rate is about -22 dB.

Figure 2.17(a) compares the measured ENOB when the FSR of the CUA is $100mV_{ppd}$. The FSR of the 4-bit CUA was adjusted to achieve the best BER under the given TX amplitude and channel loss. The 3-bit BOA has the lowest ENOB. The ENOB difference between the 4-bit CUA and the 3-bit BOA is in the range of 1.37-bit to 1.08-bit, while the difference between the 3-bit CUA and the 3-bit BOA is in the range of 0.74-bit to 0.48-bit. Figure 2.17(b) illustrates that the BER achieved by the 3-bit BOA receiver is lower by a factor of 10^9 and 10^{10} , as compared to the 4-bit and 3-bit CUA receivers, respectively, at a TX amplitude of $180mV_{ppd}$. This is in spite of the 3-bit BOA having a poorer ENOB than both the 3-bit and 4-bit CUA. Furthermore, the 3-bit BOA requires a $60mV_{ppd}$ lower TX swing compared to a 4-bit CUA to achieve $BER < 10^{-12}$. Thus, Fig. 2.17 indicates that ENOB is not the best ADC design metric for serial links. The bathtub curve in Fig. 2.18

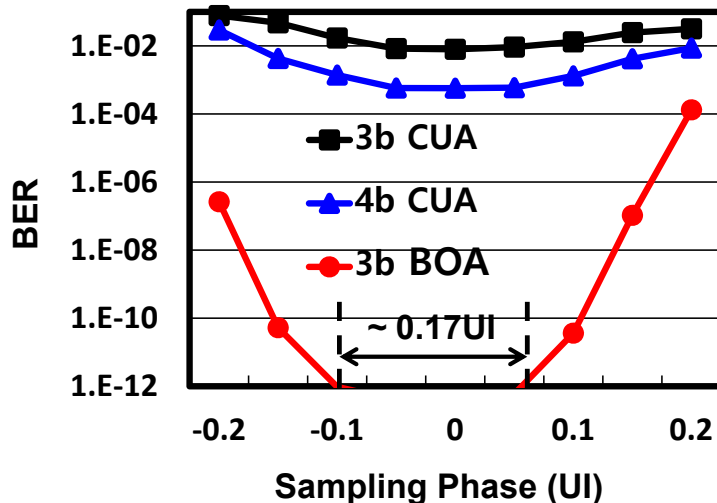


Figure 2.18: Measured BER vs. sampling phase using a 20-inch channel when TX amplitude is $180\text{ mV}_{\text{ppd}}$ and the FSR of the CUA is $100\text{ mV}_{\text{ppd}}$.

Table 2.1: Performance Summary of the ADC-based Receivers

ADC operating mode		3-bit BOA	4-bit CUA
Technology		90 nm LP CMOS (1P8M)	
Core die area		0.38 mm^2	
Supply voltage		1.2 V for analog, 1.28 V for digital & clock	
Data rate		4 Gb/s	
Power consumption	ADC [mW]	30.7	59.7
	B/E digital [mW]	*(30/**17.7)	*16.4

*Digital back-end power estimated from synthesis in 90 nm LP CMOS.

**The power of the QL-UD unit is excluded.

shows that the 3-bit BOA can tolerate a peak-to-peak jitter of about 43 ps ($\sim 0.17\text{ UI}$) at $\text{BER}=10^{-12}$ with a TX amplitude of $180\text{ mV}_{\text{ppd}}$, while the 4-bit and 3-bit CUAs are unable to achieve $\text{BER} < 10^{-4}$ and 10^{-3} , respectively, under identical conditions.

Table 2.1 summarizes the performance of the proposed BOA receiver, and Table 2.2 compares this work against state-of-the-art ADC-based receivers [22, 53–55] and analog receivers [62, 63] in CMOS. The ADC IC consumes 59.7 mW in 4-bit CUA mode and 30.7 mW in 3-bit non-uniform mode excluding the clock buffers. The clock buffers in our design accept external clocks and have to drive a long interconnect before they reach the ADC comparators, as the ADC occupies a small fraction ($< 9\%$) of the die area. Furthermore, the power

Table 2.2: Performance Comparison with State-of-the-art ADC-based Receivers and Analog Receivers in CMOS

	ADC-based Receivers					Analog Receivers		
	This work	[22]	[53]	[54]	[55]	[62]	[63]	[64]
Process	90 nm LP	65 nm	65 nm	65 nm	65 nm	40 nm	40 nm	90 nm
Sampling rate [GS/s]	4	2.5	2.575	2.5	1.2875	5.1563	7.0125	3.125
Number of bits	3	3	6	5	8	N/A	N/A	N/A
BER@Channel	$< 10^{-12}$	$< 10^{-7}$	$< 10^{-15}$	$< 10^{-12}$	N/A	$< 10^{-12}$	$< 10^{-15}$	$< 10^{-15}$
Channel loss	-22 dB	-17 dB	-26 dB	34''	N/A	~ -28 dB	-26 dB	-15 dB
RX power [mW]	60.7	106	500	192	1600	87	410	8.0
Efficiency [pJ/bit]	* (15.2/**12.1)	10.6	48.5	38.4	155.3	***8.4	****29.23	1.28
FOM2	* (10.9/**13.7)	18.8	10.3	0.5568	N/A	74.8	13.6	24.7

*Digital back-end power estimated from synthesis in 90 nm LP CMOS.

**The power of the QL-UD unit is excluded.

***With both Tx and Rx.

****With both Tx and Rx and under the worst case PVT conditions.

Note: $FOM2 = DR(\text{Gb/s}) \times 10^{\frac{\text{loss}}{10}} / \text{Power}(\text{mW})$ [65], where DR stands for data rate.

consumption of the clock buffers for the ADC core alone could not be measured because the power pins of the clock buffers driving the interconnect and the ADC core are shared. The power of the clock buffers driving the ADC core alone, extracted from post-layout simulations, is about 10 mW when the ADC operates at 4 GS/s, 4-bit CUA mode. The digital back-end power including all the functional blocks in the FPGA was estimated to be 30 mW and 17.7 mW via synthesis when including and excluding the QL-UD unit, respectively. The energy efficiency of this receiver excluding the clock buffers is 15.2 pJ/bit and 12.1 pJ/bit when including and excluding the QL-UD unit, respectively.

The presented receiver achieves a BER of less than 10^{-12} with the lowest ADC resolution (3-bit non-uniform; Note: a 3-bit CUA was not able to achieve $BER < 10^{-3}$ under the same conditions) at the highest ADC sampling rate (4 GS/s) while achieving more than $2\times$ higher energy efficiency compared with [53], [54] and [55]. Taking channel loss into account, our solution achieves higher (better) figure-of-merit FOM2 (proposed in [65]) than [53] and [54]. Implemented in a more advanced technology and combining several low power circuit techniques, [22] achieves a better energy efficiency than this work. However, [22] only showed measured BER of 10^{-7} although an extrapolated BER of 10^{-15} was reported. A 2.3-bit (5 comparators) BOA is sufficient if the target BER is relaxed from 10^{-12} to 10^{-7} based on simulations, which translates to about $2/7$ power savings in the ADC. As a result,

the efficiency is improved to 13.0 pJ/bit from 15.2 pJ/bit. Furthermore, it should be noted that the power consumption of a flash ADC is mostly determined by its sampling rate and the process technology. Compared to [22], our ADC has higher sampling rate (4 GS/s vs. 2.5 GS/s in [22]) while being implemented in a slower technology (90 nm LP vs. 65 nm in [22]). Therefore, it is expected that our solution will achieve comparable or better energy efficiency if the sampling rate and process technology are identical. On the other hand, Table 2.2 shows that energy efficiency and FOM2 of ADC-based receivers need to be further improved compared with analog receivers [62–64].

A key outcome of this work is the demonstration of information-based metric benefits, such as the BER, in reducing the ADC precision requirements, and the identification of conditions that maximize the BER improvement offered by a BOA receiver over a CUA receiver under the same condition. Although we demonstrate the BOA concept via a flash ADC, BOA is in principle applicable to different ADC architectures because it adjusts the ADC thresholds but does not change the ADC architecture. However, the power savings when designing BOA using other ADC architectures will not be as great as with flash ADCs. In particular, for flash ADC every bit decrease in resolution almost halves the size of the ADC core circuitry and the power. In contrast, for a SAR, pipelined, or sigma-delta ADC, the die size and power will decrease linearly with a decrease in resolution.

2.5 Summary

This chapter describes our study on the benefits of BOA for serial links. First, we discuss conditions that maximize BER improvement by a BOA receiver over a CUA receiver, and propose two channel-dependent parameters to quantify these conditions. Furthermore, a 4 Gb/s BOA receiver, which employs the true system BER to adjust the ADC representation levels and a linear equalizer, was implemented in 90 nm LP CMOS to show that a 3-bit BOA needs a lower SNR than a 4-bit CUA at a BER $< 10^{-12}$. This study demonstrates that the use of information-based system metrics such as the BER are very effective in reducing the component power of information transfer in ML systems. It inspires us to extend such a design principle to address the challenge of energy-efficient information processing in ML

systems as described in the following chapters.

2.6 Derivation of BERR

In this section, we derive (2.6).

In an ADC-ML receiver, the BER of a $\log_2(N + 1)$ -bit BOA is given by:

$$p_{eo}(SNR, B_o) = \sum_{k=1}^N p_{eo,k}, \quad (2.7)$$

where $p_{eo,k}$ is the BER contribution from the k^{th} μ -transition. In particular, $p_{eo,k}$ includes the BER contribution from all the peaks $\mathcal{N}(x; \mu_l^+, \sigma_n)$ with $\mu_l^+ \in \boldsymbol{\mu}^+$ ($\mathcal{N}(x; \mu_l^-, \sigma_n)$ with $\mu_l^- \in \boldsymbol{\mu}^-$) to the interval $[t_{o,k}^l, t_{o,k})$ and the BER contribution from all the peaks $\mathcal{N}(x; \mu_l^-, \sigma_n)$ with $\mu_l^- \in \boldsymbol{\mu}^-$ ($\mathcal{N}(x; \mu_l^+, \sigma_n)$ with $\mu_l^+ \in \boldsymbol{\mu}^+$) to the interval $[t_{o,k}, t_{o,k}^r)$ if the memoryless ML decision for the interval $[t_{o,k}^l, t_{o,k})$ is -1 ($+1$), where $t_{o,k}^l$ and $t_{o,k}^r$ are defined as follows:

$$t_{o,k}^l = \begin{cases} -\infty, & \text{if } k = 1 \\ \frac{t_{o,k-1} + t_{o,k}}{2}, & \text{if } 1 < k \leq N \end{cases}$$

$$t_{o,k}^r = \begin{cases} \frac{t_{o,k} + t_{o,k+1}}{2}, & \text{if } 1 \leq k < N \\ +\infty, & \text{if } k = m \end{cases}.$$

Note: if the decision for the interval $[t_{o,k}^l, t_{o,k})$ is $+1$ (or -1) then the decision for the interval $[t_{o,k}, t_{o,k}^r)$ is -1 (or $+1$). At high SNR, this contribution is well-approximated by:

$$p_{eo,k} \approx 2^{-(L-1)} Q\left(\frac{d_{o,k}^*}{\sigma_n}\right), \quad (2.8)$$

where $d_{o,k}^*$ (see **Definition 5**) is the minimum distance of the k^{th} BOA threshold to the nearest noise-free channel output μ .

Substituting (2.8) into (2.7), employing the high SNR approximation for the Q-function ($Q(y) \approx \frac{1}{y\sqrt{2\pi}} e^{-\frac{y^2}{2}}$, for $y > 0$), and the approximation $\sum_k e^{a_k} \approx e^{\frac{\max(a_k)}{k}}$, we get:

$$\begin{aligned}
p_{eo}(SNR, B_o) &= \sum_{k=1}^N p_{eo,k} \approx \sum_{k=1}^N \left[2^{-(L-1)} Q \left(\frac{d_{o,k}^*}{\sigma_n} \right) \right] \\
&= \sum_{k=1}^N \left[2^{-(L-1)} \frac{\sigma_n}{\sqrt{2\pi} d_{o,k}^*} e^{-\frac{1}{2} \left(\frac{d_{o,k}^*}{\sigma_n} \right)^2} \right]. \\
&\approx 2^{-(L-1)} \frac{\sigma_n}{\sqrt{2\pi} d_{o,min}} e^{-\frac{1}{2} \left(\frac{d_{o,min}}{\sigma_n} \right)^2}
\end{aligned} \tag{2.9}$$

Similarly, the BER of a $\log_2(M + 1)$ -bit CUA receiver is given by:

$$p_{eu}(SNR, B_u) = \sum_{k=1}^{M+1} p_{eu,k}, \tag{2.10}$$

where $p_{eu,k}$ denotes the BER contributed by the k^{th} interval $I_{u,k}$. Specifically, $p_{eu,k}$ includes BER contribution from all the peaks $\mathcal{N}(x; \mu_l^+, \sigma_n)$ with $\mu_l^+ \in \boldsymbol{\mu}^+$ ($\mathcal{N}(x; \mu_l^-, \sigma_n)$ with $\mu_l^- \in \boldsymbol{\mu}^-$) to the interval $I_{u,k}$ if the memoryless ML decision for the interval $I_{u,k}$ is -1 ($+1$). At high SNR, this contribution is well-approximated by:

$$p_{eu,k} \approx 2^{-L} Q \left(\frac{d_{u,k}^*}{\sigma_n} \right), \tag{2.11}$$

where $d_{u,k}^*$ (see **Definition 7**) is the minimum distance of the k^{th} dominant noise-free output μ_k^* from the boundaries of the interval $I_{u,k}$.

Substituting (2.11) into (2.10), employing the high SNR approximation for the Q-function, the approximation $\sum_k e^{a_k} \approx e^{\frac{\max(a_k)}{k}}$, and the relationship $Q(y) = 1 - Q(-y)$ for $y < 0$, we get:

$$\begin{aligned}
& p_{eu}(SNR, B_u) \\
&= \sum_{k=1}^{M+1} p_{eu,k} \approx \sum_{k=1}^{M+1} \left[2^{-L} Q \left(\frac{d_{u,k}^*}{\sigma_n} \right) \right] \\
&\approx 2^{-L} Q \left(\frac{d_{u,min}}{\sigma_n} \right) \\
&= \begin{cases} 2^{-L} \frac{\sigma_n}{\sqrt{2\pi}d_{u,min}} e^{-\frac{1}{2} \left(\frac{d_{u,min}}{\sigma_n} \right)^2}, & \text{if } d_{u,min} > 0 \\ 2^{-L} \left[1 + \frac{\sigma_n}{\sqrt{2\pi}d_{u,min}} e^{-\frac{1}{2} \left(\frac{d_{u,min}}{\sigma_n} \right)^2} \right], & \text{if } d_{u,min} < 0 \\ 2^{-(L+1)}, & \text{if } d_{u,min} = 0 \end{cases} \quad (2.12)
\end{aligned}$$

Therefore, from (2.9) and (2.12), we obtain:

$$\begin{aligned}
& BERR(SNR, B_u, B_o) \\
&= \frac{p_{eu}(SNR, B_u)}{p_{eo}(SNR, B_o)} \\
&\approx \begin{cases} \frac{d_{o,min}}{2d_{u,min}} e^{\frac{d_{o,min}^2 - d_{u,min}^2}{2\sigma_n^2}}, & \text{if } d_{u,min} > 0 \\ \frac{d_{o,min}}{2d_{u,min}} e^{\frac{d_{o,min}^2 - d_{u,min}^2}{2\sigma_n^2}} (1 + \chi), & \text{if } d_{u,min} < 0 \\ \sqrt{\frac{\pi}{8}} \frac{d_{o,min}}{\sigma_n} e^{\frac{d_{o,min}^2}{2\sigma_n^2}}, & \text{if } d_{u,min} = 0 \end{cases} \quad (2.13)
\end{aligned}$$

where $\chi = \frac{\sqrt{2\pi}d_{u,min}}{\sigma_n} e^{\frac{d_{u,min}^2}{2\sigma_n^2}}$. Note: $d_{o,min} \geq d_{u,min}$ and $d_{o,min} \geq 0$. Applying the approximation $\sum_k e^{a_k} \approx e^{\frac{\max(a_k)}{k}}$ to the case when $d_{u,min} < 0$, (2.13) can be further simplified to (2.6).

Chapter 3

PREDICTIVENET

In the previous chapter, the energy efficiency of information transfer in ML systems is improved by using information-based metrics such as the BER to design power dominant components. In this chapter, we explore a similar approach to reduce the implementation complexity of information processing such as CNNs in ML systems. As such, the power dominant convolutions in CNNs are designed to maintain the information-based system metric (i.e. classification accuracy) rather than fidelity circuit metrics such as SNR. Specifically, we make use of CNN structure to propose a technique referred to as PredictiveNet which predicts zero activations using low-cost predictors to skip a significant amount of convolutional operations. PredictiveNet first evaluates the most significant bit (MSB) part of the convolution to predict whether the nonlinear layer output corresponding to the current convolution is zero, and then decides if the remaining least significant bit (LSB) part's computation can be skipped or not. PredictiveNet takes advantage of the fact that the MSB part has an exponentially larger contribution to the final output and well-trained CNNs have high sparsity.

The rest of this chapter is organized as follows. Section 3.1 provides the relevant background. Section 3.2 presents the PredictiveNet technique and analysis to justify PredictiveNet's effectiveness. Simulation results are shown in Section 3.3. Section 3.4 provides conclusions.

3.1 Background

3.1.1 Convolutional Neural Networks (CNNs)

CNNs are a class of multi-layer neural networks [66]. A CNN consists of a cascade of multiple convolutional layers (C-layers), subsampling layers (S-layers) (feature extractor), and fully-connected layers (F-layers) (classifier). Figure 3.1 illustrates a state-of-the-art CNN for object recognition [66]. In a C-layer, DPs between receptive fields and weight vectors are computed, to which a bias term is added, and passed through a nonlinear function to generate the output feature maps (FMs). The computation of one output pixel for the C-layer is described as follows:

$$z_m[j] = f(y_m[j] + \delta_m), (m = 1, \dots, M) \quad (3.1)$$

$$y_m[j] = \sum_{l=1}^L \mathbf{w}_{ml}^T \mathbf{x}_{jl}, (m = 1, \dots, M), \quad (3.2)$$

where L and M are the number of input and output FMs, respectively. \mathbf{w}_{ml} is the N -tuple weight vector connecting the l^{th} input FM $\mathbf{X}_l = [\mathbf{x}_{1l}, \dots, \mathbf{x}_{Jl}]$ (where \mathbf{x}_{jl} is the j^{th} receptive field in \mathbf{X}_l) to the m^{th} convolutional output \mathbf{y}_m , δ_m is the bias term, and \mathbf{z}_m denotes the m^{th} output FM in the C-layer. Equation (3.2) shows that the j^{th} pixel $y_m[j]$ of the m^{th} convolutional output \mathbf{y}_m is obtained by first performing DPs between the L input vectors \mathbf{x}_{jl} and the weight vectors \mathbf{w}_{ml} , and summing up the result. The nonlinear function $f(\cdot)$ typically takes a sigmoid or hyperbolic form. However, a rectified linear unit (ReLU) has emerged recently as increased evidence shows that it improves performance of CNNs [67]. The S-layer reduces the dimension of its input FMs via either an average or a max pooling.

3.1.2 Sparsity in CNNs

Sparsity has become a concept of interest in the fields of neuroscience, machine learning, and signal processing. It was first introduced in the context of sparse coding in visual systems [69], which seeks to find an overcomplete basis set and represent images as a linear superposition

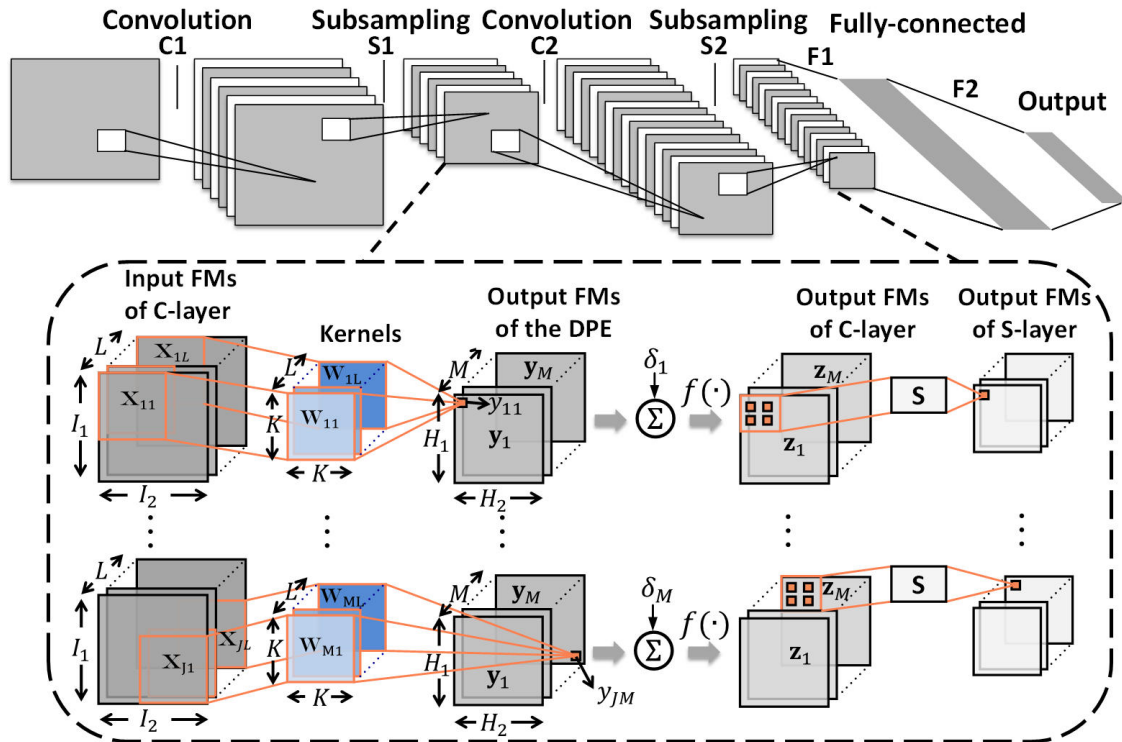


Figure 3.1: Illustration of a state-of-the-art CNN [68] showing a convolutional layer (C-layer), a subsampling layer (S-layer), feature maps (FMs), and the squashing function $f(\cdot)$.

of basis functions from the resulted set. *Overcomplete* means that the number of basis functions M is greater than the effective dimensionality of the image space L , which gives rise to sparsity as only L out of M nonzero coefficients are needed to represent arbitrary L -dimensional images. As similar sparse overcomplete representation was observed in biological neurons, it becomes a plausible model for the visual cortex [70, 71]. In ML models, sparse overcomplete representation has been claimed to be a fundamental reason behind the success of deep neural networks, such as CNNs [72]. Specifically, it has a number of theoretical and practical advantages [72–74], including 1) greater flexibility in capturing inherent structure of underlying data; 2) increased robustness to small perturbations of the data; and 3) better separability because the information is represented in a high-dimensional space. State-of-the-art CNN models can obtain sparsity from 50% to 85% in their activations [67]. It is worth noting that while conventional hyperbolic tangent or sigmoid nonlinear function generates sparse activations taking small but non-zero values, the recently emerged ReLU is able to

produce real zeros of activations and thus truly sparse representations while achieving better classification accuracy.

3.2 The Proposed PredictiveNet Technique

This section describes the PredictiveNet technique and its analytical justification.

3.2.1 Principle and Architecture

Without loss of generality, we drop the indices for j and m in (3.1) and (3.2) and assume $f(\cdot)$ is a ReLU, i.e.,

$$z = \max \left(\sum_{l=1}^L \mathbf{w}_l^T \mathbf{x}_l + \delta, 0 \right), \quad (3.3)$$

where $\mathbf{w}_l^T \mathbf{x}_l = \sum_{i=1}^N w_l[i] x_l[i]$.

We first decompose $x_l[i]$, $w_l[i]$, and δ into MSB and LSB parts. If we assume that B_{msb} is the precision of the MSB part of \mathbf{w}_l , \mathbf{x}_l , and δ , then:

$$z = \max(y, 0) = \max \left(\sum_{l=1}^L \mathbf{w}_l^T \mathbf{x}_l + \delta, 0 \right) = \begin{cases} y_{\text{msb}} + y_{\text{lsb}} 2^{-(B_{\text{msb}}-1)} & \text{if } \sum_{l=1}^L \mathbf{w}_l^T \mathbf{x}_l + \delta > 0 \\ 0 & \text{otherwise} \end{cases}, \quad (3.4)$$

where y_{msb} and y_{lsb} can be expressed as follows:

$$y_{\text{msb}} = \sum_{l=1}^L \mathbf{w}_{l,\text{msb}}^T \mathbf{x}_{l,\text{msb}} + \delta_{\text{msb}} \quad (3.5)$$

$$y_{\text{lsb}} = \sum_{l=1}^L (\mathbf{x}_{l,\text{lsb}}^T \mathbf{w}_{l,\text{msb}} + \mathbf{x}_l^T \mathbf{w}_{l,\text{lsb}}) + \delta_{\text{lsb}}, \quad (3.6)$$

where $\mathbf{x}_{l,\text{msb}}$, $\mathbf{w}_{l,\text{msb}}$, and δ_{msb} denote the MSB parts of \mathbf{x}_l , \mathbf{w}_l , and δ , respectively. Also,

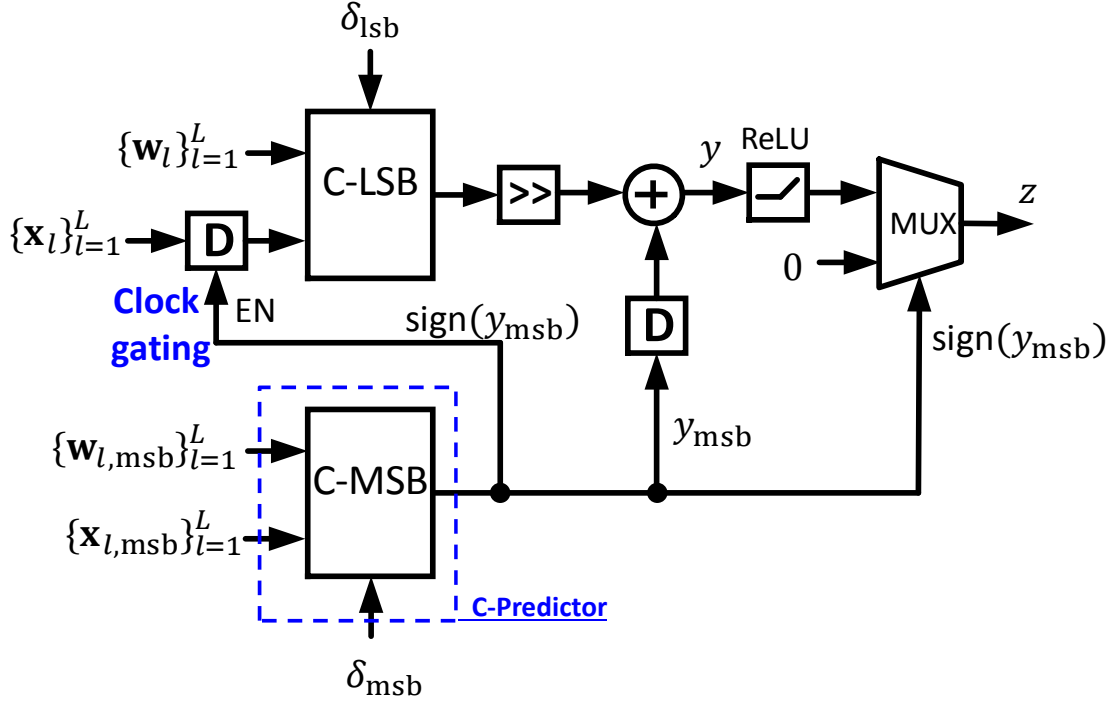


Figure 3.2: An architecture implementing (3.4) in PredictiveNet.

$\mathbf{x}_{l,\text{lsb}}$, $\mathbf{w}_{l,\text{lsb}}$, and δ_{lsb} denote the LSB parts of \mathbf{x}_l , \mathbf{w}_l , and δ , respectively.

The PredictiveNet architecture (Fig. 3.2) includes a C-MSB block that predicts the sign of y by computing only y_{msb} in (3.5). If $y_{\text{msb}} < 0$ (i.e., $\text{sign}(y_{\text{msb}}) = 1$), then we set $z = 0$ without computing y_{lsb} in (3.6) and $y_{\text{msb}} + y_{\text{lsb}}2^{-(B_{\text{msb}}-1)}$ according to (3.4). If $y_{\text{msb}} \geq 0$ (i.e., $\text{sign}(y_{\text{msb}}) = 0$), then C-LSB computes (3.6) and sets $z = y_{\text{msb}} + y_{\text{lsb}}2^{-(B_{\text{msb}}-1)}$ in (3.4). By doing so, PredictiveNet avoids evaluating a significant number of convolutions while incurring only marginal accuracy loss.

The reasons for the accuracy loss to be marginal in PredictiveNet are as follows: 1) the contribution of y_{lsb} for calculating z is $2^{-(B_{\text{msb}}-1)}$ smaller than y_{msb} as shown in (3.4); 2) the specific value of $y_{\text{msb}} + y_{\text{lsb}}2^{-(B_{\text{msb}}-1)}$ is not important if it is negative due to the rectification effect of ReLU; and 3) the high sparsity in CNNs as mentioned in Section 3.1.2 implies that the term $y_{\text{msb}} + y_{\text{lsb}}2^{-(B_{\text{msb}}-1)}$ is very likely to be negative, which will result in zero C-layer outputs after being passed through the ReLU function.

Table 3.1: Errors of MSB-CNN and PredictiveNet with Respect to FP-CNN

Event	Condition	MSB-CNN error	PredictiveNet error
H_0	$y_{\text{msb}} \leq 0, y \leq 0$	0	0
H_1	$y_{\text{msb}} \leq 0, y > 0$	y	y
H_2	$y_{\text{msb}} > 0, y \leq 0$	$-y_{\text{msb}}$	0
H_3	$y_{\text{msb}} > 0, y > 0$	$y - y_{\text{msb}}$	0

3.2.2 Analysis

In this subsection, analysis and empirical simulation results are presented to justify why PredictiveNet incurs marginal accuracy loss while greatly decreasing the computational cost. Our analysis is based on the trade-offs between accuracy and precision. Recently, such a trade-off has been analytically characterized for simple ML algorithms such as support vector machine (SVM) [75]. Such insights have not yet been leveraged for complex algorithms such as CNNs.

Assume that $B_{\mathbf{w}}$, $B_{\mathbf{x}}$ and B_{δ} denote the required precisions of \mathbf{w}_l , \mathbf{x}_l , and δ , respectively. Also, let $B_{\mathbf{w},\text{msb}}$, $B_{\mathbf{x},\text{msb}}$ and $B_{\delta,\text{msb}}$ denote the precisions of $\mathbf{w}_{l,\text{msb}}$, $\mathbf{x}_{l,\text{msb}}$, and δ_{msb} in (3.5), respectively. For convenience, we term the CNN comprising only C-MSB as MSB-CNN, and the CNN implemented using $B_{\mathbf{w}}$, $B_{\mathbf{x}}$ and B_{δ} as the full precision CNN (FP-CNN), respectively.

In Table 3.1, we compare the ReLU output errors of MSB-CNN and PredictiveNet with respect to the outputs of FP-CNN (i.e., z) in (3.3) for four disjoint events from H_0 to H_3 . Note that each possible outcome is included in exactly one of these events. The MSE at the outputs of the ReLU with respect to FP-CNN are:

$$\begin{aligned}
 MSE_{\text{MSB-CNN}} &= E[Y^2|H_1]P(H_1) + E[Y_{\text{msb}}^2|H_2]P(H_2) \\
 &\quad + E[|Y - Y_{\text{msb}}|^2|H_3]P(H_3)
 \end{aligned} \tag{3.7}$$

$$MSE_{\text{PredictiveNet}} = E[Y^2|H_1]P(H_1), \tag{3.8}$$

where upper case letter denotes random variables. By comparing (3.7) and (3.8), we see

that:

$$MSE_{\text{PredictiveNet}} < MSE_{\text{MSB-CNN}}. \quad (3.9)$$

Furthermore, $P(H_1)$ has been found to be small in practice and can be upper bounded as follows:

$$P(H_1) \leq \Delta_w^2 E_1 + \Delta_x^2 E_2 + \Delta_\delta^2 E_3, \quad (3.10)$$

where $\Delta_w = 2^{-(B_{\mathbf{w}, \text{msb}}-1)}$, $\Delta_x = 2^{-(B_{\mathbf{x}, \text{msb}}-1)}$ and $\Delta_\delta = 2^{-(B_{\delta, \text{msb}}-1)}$ are the quantization noise step sizes of $\mathbf{w}_{l, \text{msb}}$, $\mathbf{x}_{l, \text{msb}}$ and δ_{msb} , respectively, and E_1 , E_2 , and E_3 are given in Section 3.5 along with the proof of (3.10).

Similarly, the $E[Y^2|H_1]$ can be upper bounded as follows:

$$E[Y^2|H_1] \leq \Delta_w^2 E_4 + \Delta_x^2 E_5 + \Delta_\delta^2, \quad (3.11)$$

where $E_4 = E\left[\sum_{l=1}^L \|\mathbf{X}_l\|^2 | H_1\right]$ and $E_5 = \sum_{l=1}^L \|\mathbf{w}_l\|^2$. The proof of (3.11) can also be found in Section 3.5.

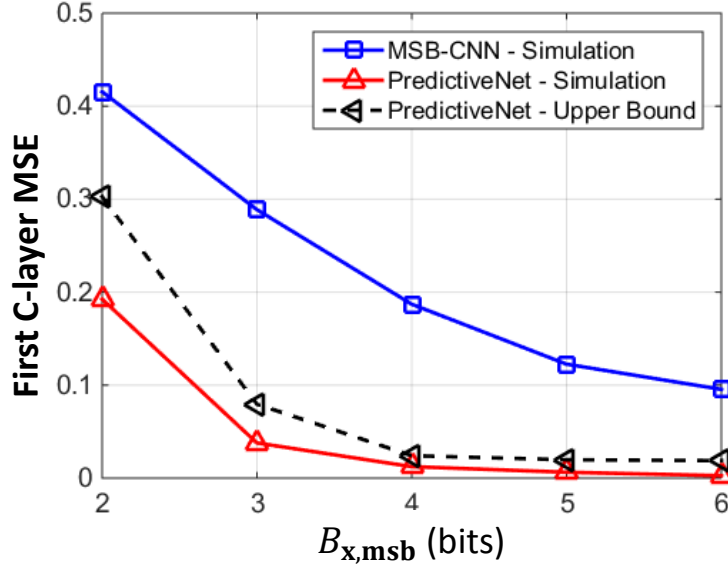
Combining (3.10) and (3.11), we can obtain an upper bound on $MSE_{\text{PredictiveNet}}$:

$$\begin{aligned} MSE_{\text{PredictiveNet}} &\leq \Delta_w^4 E_6 + \Delta_x^4 E_7 + \Delta_\delta^4 E_8 + (\Delta_w \Delta_x)^2 E_9 \\ &\quad + (\Delta_w \Delta_\delta)^2 E_{10} + (\Delta_x \Delta_\delta)^2 E_{11}, \end{aligned} \quad (3.12)$$

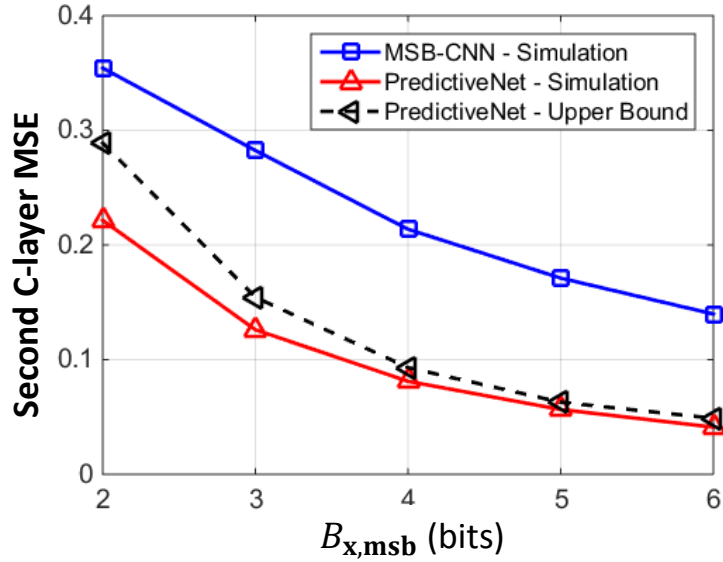
where E_6, \dots, E_{11} are the cross product terms associated with the product of (3.10) and (3.11).

We observe that every term in (3.12) is a fourth order multiplicative combination of quantization steps. Each quantization step is of the order of $2^{-B_{\text{msb}}}$. Hence, the upper bound in (3.12) is of the order of $2^{-4B_{\text{msb}}}$.

Figure 3.3 shows empirical values for $MSE_{\text{MSB-CNN}}$ and $MSE_{\text{PredictiveNet}}$ for the two C-layers in a CNN designed for handwritten digit recognition [76]. Figure 3.3 supports (3.9) and shows that the MSE of PredictiveNet is much smaller than the MSE of MSB-CNN. This results from the exponentially larger weighting factor of y_{msb} contributed to y over that of



(a)



(b)

Figure 3.3: Illustration of the empirical values of $MSE_{\text{MSB-CNN}}$ and $MSE_{\text{PredictiveNet}}$ and the upper bound on $MSE_{\text{PredictiveNet}}$ with respect to $B_{x,msb}$ for: the (a) first and (b) second C-layers over FP-CNN where $B_{w,msb} = 5$ bits, $B_w = 8$ bits, $B_x = B_\delta = 7$ bits, and $B_{\delta,msb} = B_{x,msb}$. Both curves are obtained by averaging over all pixels of the two C-layers' output FMs.

y_{lsb} and the high sparsity of the C-layer outputs in well trained CNNs.

3.3 Simulation Results

In this section, we evaluate the performance of PredictiveNet on two datasets: MNIST and CIFAR-10, which are benchmark datasets for handwritten digit and object recognition, respectively.

3.3.1 System Set-up

The term δ_m and kernel \mathbf{w}_{ml} in (3.2) are trained using the back propagation algorithm [66]. The following four architectures are considered: 1) FP-CONV: a conventional FP-CNN; 2) FP-ZS: a full-precision input zero skipping CNN; 3) PredictiveNet; and 4) MSB-CNN: a predictor-only CNN. These architectures are evaluated in terms of the following metrics:

- Classification error rate p_e : $p_e = P\{\hat{T} \neq t\}$, where \hat{T} and t are the decision of the evaluated CNN and the true label, respectively.
- Computational cost: the total number of full adders (FAs) in the network, where an FA is a basic building block of arithmetic units. We assume that the evaluated CNNs are implemented using the commonly used Baugh-Wooley multiplier and ripple carry adder (RCA) designed using FAs. Therefore, the number of FAs to compute an R -dimensional DP between the kernel weights and the activations is [77]:

$$RB_{\mathbf{w}}B_{\mathbf{x}} + (R - 1)(B_{\mathbf{x}} + B_{\mathbf{w}} + \lceil \log_2(R) \rceil - 1). \quad (3.13)$$

- Representational cost: the total number of bits associated with non-zero activations and weights in the network, which represents the data storage and movement costs. For a fixed-point network, it is defined as:

$$|\mathcal{X}| B_{\mathbf{x}} + |\mathcal{W}| B_{\mathbf{w}}, \quad (3.14)$$

Table 3.2: Parameters Summary of the CNN for the MNIST Dataset

Parameter Definition		CNN Parameter Summary				
Parameter	Description	Layer	L	M	$I_1 \times I_2$	$K \times K$
L/M	# of input/output FMs	C1	1	16	28×28	5×5
$K \times K$	size of kernels	C2	16	32	12×12	5×5
$I_1 \times I_2$	size of input FMs	F1	100	10	4×4	4×4

where \mathcal{X} and \mathcal{W} are the sets of all non-zero activations and weights in the network, respectively. Together, the computational and representational costs capture the implementation complexity of a CNN, and are equally important metrics.

3.3.2 Evaluation on CNNs for MNIST

The parameters of the CNN for the MNIST dataset are summarized in Table 3.2, which is developed based on the CNN architecture in [66]. The precision B_x and B_w are set to be 7 bits and 8 bits, respectively, ensuring the error-free fixed-point p_e increases by only 4×10^{-3} compared with the floating-point p_e of 0.016.

Figure 3.4 compares FP-CNN (FP-CONV and FP-ZS), PredictiveNet, and MSB-CNN in terms of their classification error rates, computational and representational costs normalized over that of FP-CONV. Figure 3.4 shows that PredictiveNet is able to achieve a classification error rate that is only 1.9×10^{-2} larger than that of FP-CNN while reducing the computational cost by $2.5\times$ compared to the state-of-the-art FP-ZS. Furthermore, PredictiveNet achieves $1.7\times$ reduction in the representational cost over that of FP-ZS. On the other hand, when compared to MSB-CNN, PredictiveNet reduces the classification error rates by $12\times$ ($0.475/0.039$) at the cost of only $1.6\times$ greater computational cost.

It is interesting to observe that PredictiveNet has even 19% smaller representational cost than its predictor-only counterpart, i.e., MSB-CNN (see Fig. 3.4 (c)). This can be justified by the higher sparsity observed in the PredictiveNet than the latter. In particular, the computational and representational costs for a CNN applied on top of FP-ZS depend not only on the precision requirement associated with B_x and B_w but also the sparsity in the

Table 3.3: Computational and Representational Cost Comparison among CNNs for the MNIST Dataset

	FP-CONV	FP-ZS	PredictiveNet	MSB-CNN
Computational cost (million)	77.13	40.88	16.43	10.18
Representational cost (million)	0.5376	0.2640	0.1586	0.1951

Table 3.4: Parameters Summary of the CNN [78] for the CIFAR-10 Dataset

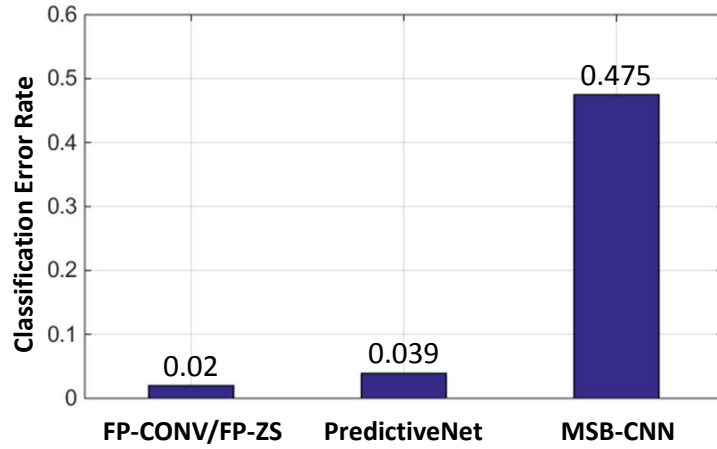
Parameter Definition		CNN Parameter Summary				
Parameter	Description	Layer	L	M	$I_1 \times I_2$	$K \times K$
L/M	# of input/output FMs	C1	3	192	32×32	5×5
		C2	192	160	32×32	1×1
		C3	160	96	32×32	1×1
$K \times K$	size of kernels	C4	96	192	15×15	5×5
		C5	192	192	15×15	1×1
		C6	192	192	15×15	1×1
$I_1 \times I_2$	size of input FMs	C7	192	192	7×7	3×3
		C8	192	192	7×7	1×1
		C9	192	10	7×7	1×1

C-layer inputs. For example, Fig. 3.5 (a) shows that the input sparsity of PredictiveNet’s C2 and F1 layers are 14.6% and $1.6\times$ higher than those of the MSB-CNN, respectively. As the representational cost of the C2 and F1 layers accounts for $> 90\%$ of the total representational cost, the higher sparsity in PredictiveNet’s C2 and F1 layers explains its smaller representational cost over the MSB-CNN.

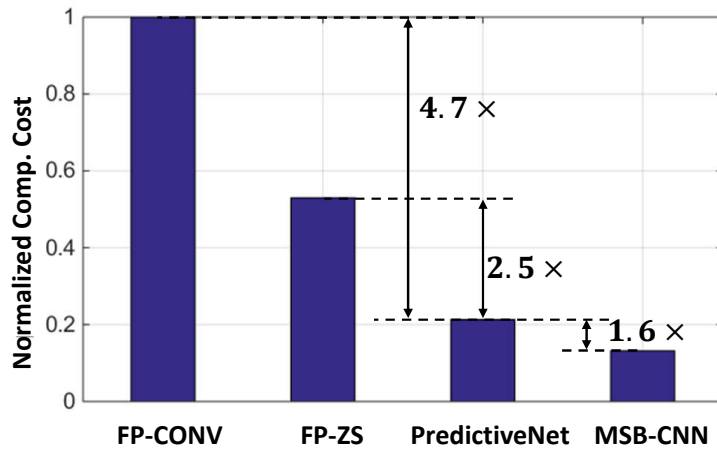
Table 3.3 summarizes the computational and representational costs to implement the four CNNs. Figure 3.4 and Table 3.3 show that PredictiveNet’s accuracy is slightly worse than FP-CNN (FP-CONV or FP-ZS) but with significantly lower complexity.

3.3.3 Evaluation on CNNs for CIFAR-10

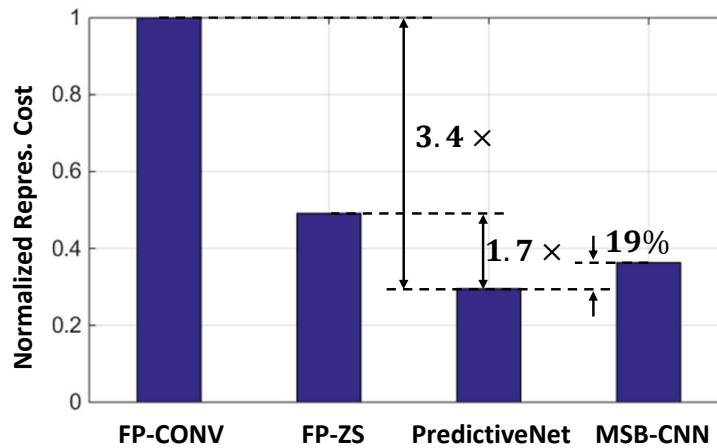
To demonstrate the generality of the proposed PredictiveNet technique, it is also applied to the CIFAR-10 dataset [79]. The parameters of the CNN for the CIFAR-10 dataset are



(a)



(b)



(c)

Figure 3.4: Simulation results for the MNIST dataset comparing FP-CNN (FP-CONV and FP-ZS), PredictiveNet, and MSB-CNN in terms of: (a) classification error rates, (b) normalized computational cost (# of full adders (FAs)), and (c) normalized representational cost (# of bits), where $B_{\mathbf{x},\text{msb}} = B_{\delta,\text{msb}} = 4$ bits and $B_{\mathbf{w},\text{msb}} = 5$ bits.

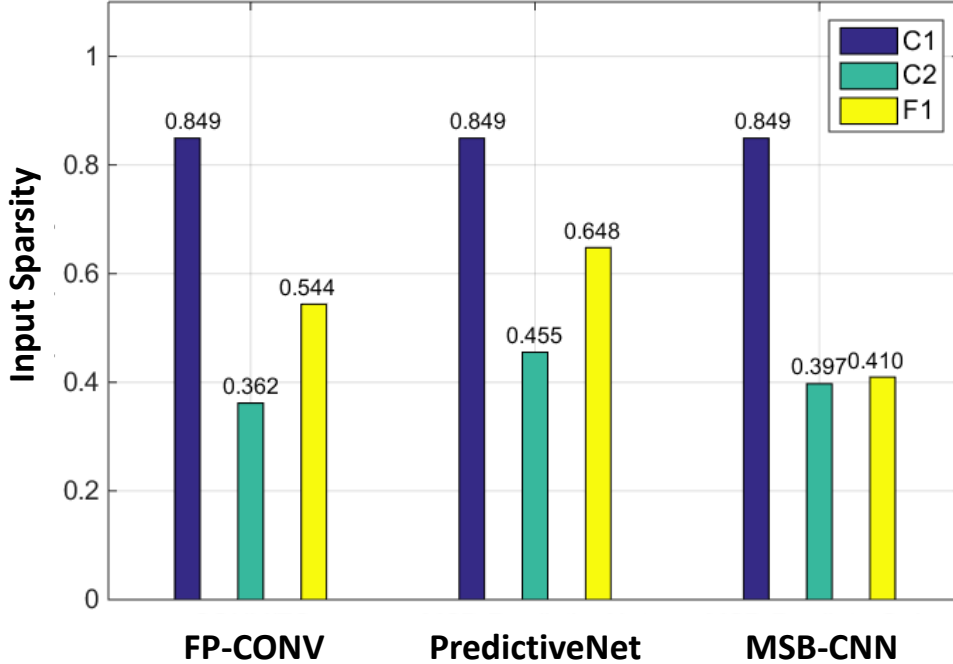
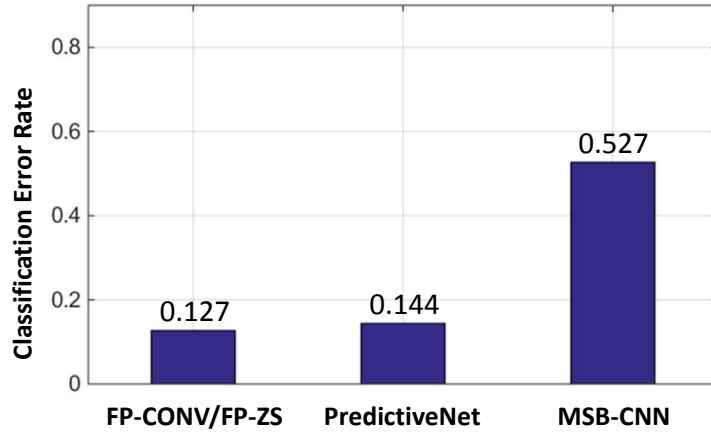


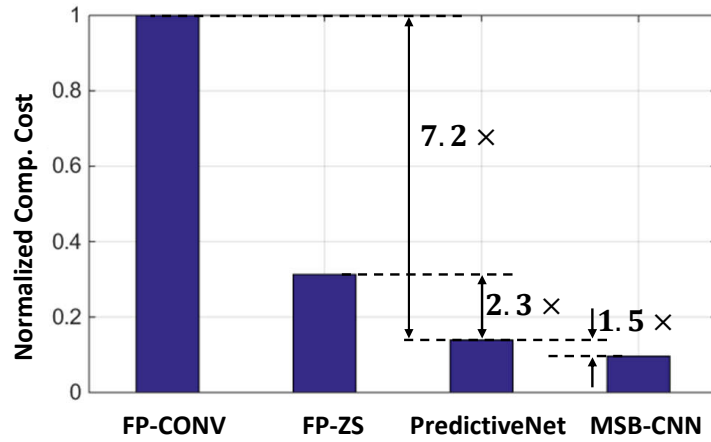
Figure 3.5: Comparison on the C-layer input sparsity of FP-CONV/FP-ZS, PredictiveNet and MSB-CNN for the MNIST dataset.

summarized in Table 3.4 [78]. The precision B_x and B_w are set to be 9 bits and 8 bits, respectively, ensuring the error-free fixed-point p_e to be within 2.4×10^{-2} of the floating-point p_e of 0.124. Although both the MNIST and CIFAR-10 datasets contain data of 10 categories, the data of the latter are more diverse and thus the data statistics are more complex. As a result, it can be seen from Tables 3.2 and 3.4 that the CNN architecture for the CIFAR-10 dataset is more complicated and the achievable classification error rates are higher than those of the CNNs for the MNIST dataset.

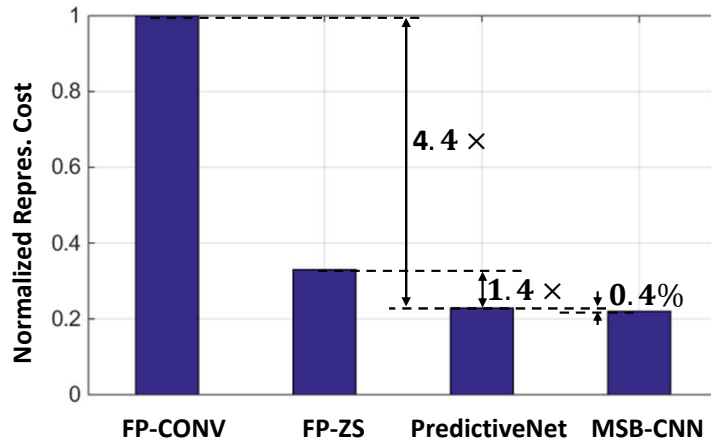
Figure 3.4 compares the performance of FP-CNN (FP-CONV and FP-ZS), PredictiveNet, and MSB-CNN in terms of classification error rates, computational and representational costs normalized over that of FP-CONV. It can be seen from Fig. 3.4 that PredictiveNet is able to maintain a classification error rate that is only 1.7×10^{-2} larger than that of FP-CNN while achieving a $2.3 \times$ reduction in the computational cost over the state-of-the-art FP-ZS. Furthermore, PredictiveNet reduces the representational cost by $1.4 \times$ compared to FP-ZS. On the other hand, when compared to MSB-CNN, PredictiveNet shrinks the classification error rates by $3.7 \times$ ($0.527/0.144$) at the cost of $1.5 \times$ greater computational cost.



(a)



(b)



(c)

Figure 3.6: Simulation results for the CIFAR-10 dataset comparing FP-CNN (FP-CONV and FP-ZS), PredictiveNet, and MSB-CNN in terms of: (a) classification error rates, (b) normalized computational cost (# of full adders (FAs)), and (c) normalized representational cost (# of bits), where $B_{\mathbf{x},\text{msb}} = B_{\delta,\text{msb}} = 6$ bits and $B_{\mathbf{w},\text{msb}} = 5$ bits.

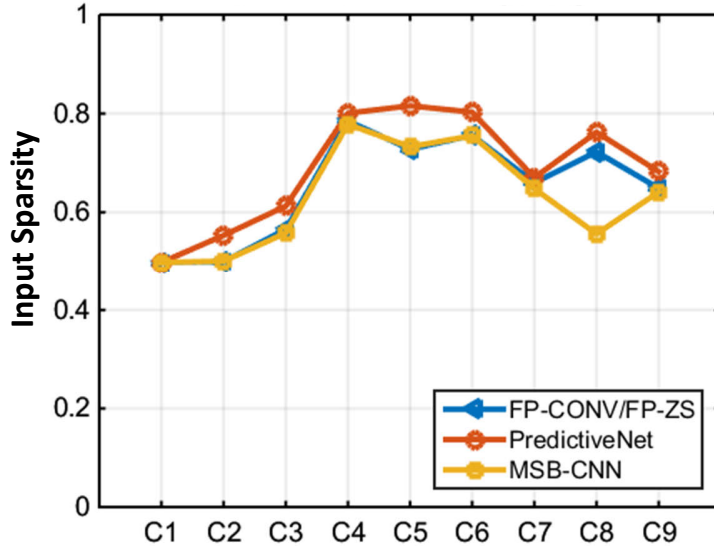


Figure 3.7: Comparison on the C-layer input sparsity of FP-CONV/FP-ZS, PredictiveNet and MSB-CNN for the CIFAR-10 dataset.

Table 3.5: Computational and Representational Cost Comparison among CNNs for the CIFAR-10 Dataset

	FP-CONV	FP-ZS	PredictiveNet	MSB-CNN
Computational cost (billion)	17.60	5.52	2.45	1.69
Representational cost (million)	11.95	3.94	2.74	2.63

Similar to the case in CNNs for the MNIST dataset, it is observed in Fig. 3.6(c) that PredictiveNet requires negligible (0.4%) higher representational cost than its predictor-only counterpart, i.e., MSB-CNN. Again, this can be explained by the higher sparsity observed in the PredictiveNet than the latter as shown in Fig. 3.7. Specifically, the input sparsity of the C2-C9 layers in the PredictiveNet is higher than those of the MSB-CNN, and the representational cost corresponding to these layers accounts for $> 90\%$ of the total representational cost, therefore justifying the lower representational cost of PredictiveNet over MSB-CNN.

Table 3.5 lists the number of FAs and bits associated with the four CNNs for the CIFAR-10 dataset. Similarly, it can be observed from Fig. 3.6 and Table 3.5 that PredictiveNet is able to significantly reduce the complexity while maintaining classification accuracy close to that of the FP-CNN (FP-CONV or FP-ZS).

3.4 Summary

In this chapter, we propose a new technique, PredictiveNet, which predicts sparse nonlinear outputs and skips corresponding convolution operations for reduced complexity CNN design. Analysis is performed to justify the effectiveness of PredictiveNet and predict the behavior of CNNs with respect to precision of its predictors. PredictiveNet takes advantage of the fact that the weighting factors in fixed-point representation decrease exponentially and high sparsity is commonly observed in well trained CNNs. This work opens up a new research dimension to greatly reduce CNNs' implementation cost without degrading their detection accuracy. Future work includes the application of PredictiveNet to other ML algorithms such as multilayer perceptron and spiking neural networks, where high sparsity is also commonly observed. Imposing additional constraints that favor the reduction of prediction errors in PredictiveNet into the training algorithms is also an interesting research topic.

3.5 Derivation of (3.10) and (3.11)

We provide a detailed derivation of (3.10) and (3.11).

$$\begin{aligned}
P(H_1) &= P(Y_{\text{msb}} \leq 0, Y > 0) = P(Y > 0) P(Y_{\text{msb}} \leq 0 | Y > 0) \\
&= \frac{1}{2} P(Y > 0) P(|q_y| > Y | Y > 0) \\
&= \frac{1}{2} P(Y > 0) \int f_{\mathbf{X}|Y>0}(\mathbf{x}) P(|q_y| > Y | Y > 0, \mathbf{X} = \mathbf{x}) d\mathbf{x} \\
&\leq \frac{P(Y > 0)}{24} \int f_{\mathbf{X}|Y>0}(\mathbf{x}) \frac{\sum_{l=1}^L [\Delta_w^2 \|\mathbf{x}_l\|^2 + \Delta_x^2 \|\mathbf{w}_l\|^2] + \Delta_\delta^2}{\left| \sum_{l=1}^L \mathbf{w}_l^T \mathbf{x}_l + \delta \right|^2} d\mathbf{x} \\
&= \frac{1}{24} P(Y > 0) E \left[\frac{\sum_{l=1}^L [\Delta_w^2 \|\mathbf{X}_l\|^2 + \Delta_x^2 \|\mathbf{w}_l\|^2] + \Delta_\delta^2}{\left| \sum_{l=1}^L \mathbf{w}_l^T \mathbf{X}_l + \delta \right|^2} \middle| Y > 0 \right] \\
&= \frac{1}{24} E \left[\frac{\sum_{l=1}^L [\Delta_w^2 \|\mathbf{X}_l\|^2 + \Delta_x^2 \|\mathbf{w}_l\|^2] + \Delta_\delta^2}{\left| \sum_{l=1}^L \mathbf{w}_l^T \mathbf{X}_l + \delta \right|^2} \cdot \mathbb{1}_{Y>0} \right] \\
&= \frac{1}{24} \Delta_w^2 \sum_{l=1}^L E \left[\frac{\|\mathbf{X}_l\|^2 \cdot \mathbb{1}_{Y>0}}{\left| \sum_{l=1}^L \mathbf{w}_l^T \mathbf{X}_l + \delta \right|^2} \right]
\end{aligned}$$

$$\begin{aligned}
& + \frac{1}{24} \left(\Delta_x^2 \sum_{l=1}^L \|\mathbf{w}_l\|^2 + \Delta_\delta^2 \right) E \left[\frac{\mathbb{1}_{Y>0}}{\left| \sum_{l=1}^L \mathbf{w}_l^T \mathbf{X}_l + \delta \right|^2} \right] \\
& = \Delta_w^2 E_1 + \Delta_x^2 E_2 + \Delta_\delta^2 E_3,
\end{aligned}$$

where $f_{\mathbf{X}|Y>0}(\mathbf{x})$ is the conditional distribution of \mathbf{X} given $Y > 0$ and $q_y = \sum_{l=1}^L (\mathbf{q}_{\mathbf{w}_l}^T \mathbf{x}_l + \mathbf{q}_{\mathbf{x}_l}^T \mathbf{w}_l) + q_\delta$. Note that $\mathbf{q}_{\mathbf{w}_l}$, $\mathbf{q}_{\mathbf{x}_l}$, and q_δ are the quantization noise terms of $\mathbf{w}_{l,\text{msb}}$, $\mathbf{x}_{l,\text{msb}}$, and δ_{msb} , respectively. $\mathbb{1}_A$ denotes the indicator function of the event A . The $\frac{1}{2}$ in the second step is due to the symmetric distribution of q_y . The fourth step comes from Chebyshev's inequality. Note that

$$\begin{aligned}
E_1 &= \frac{1}{24} \sum_{l=1}^L E \left[\frac{\|\mathbf{X}_l\|^2 \cdot \mathbb{1}_{Y>0}}{\left| \sum_{l=1}^L \mathbf{w}_l^T \mathbf{X}_l + \delta \right|^2} \right] \\
E_2 &= \frac{1}{24} \sum_{l=1}^L \|\mathbf{w}_l\|^2 E \left[\frac{\mathbb{1}_{Y>0}}{\left| \sum_{l=1}^L \mathbf{w}_l^T \mathbf{X}_l + \delta \right|^2} \right] \\
E_3 &= \frac{1}{24} E \left[\frac{\mathbb{1}_{Y>0}}{\left| \sum_{l=1}^L \mathbf{w}_l^T \mathbf{X}_l + \delta \right|^2} \right].
\end{aligned}$$

Furthermore, under H_1 we have $y_{\text{msb}} = y + q_y \leq 0$ and $y > 0$ which means that $0 < y \leq -q_y$ (i.e., $|y|^2 \leq |q_y|^2$). Hence,

$$\begin{aligned}
E[Y^2|H_1] &\leq E[q_y^2|H_1] \\
&= \Delta_w^2 E \left[\sum_{l=1}^L \|\mathbf{X}_l\|^2 | H_1 \right] + \Delta_x^2 \sum_{l=1}^L \|\mathbf{w}_l\|^2 + \Delta_\delta^2.
\end{aligned}$$

Chapter 4

RANK DECOMPOSED STATISTICAL ERROR COMPENSATION

In previous chapters, the energy efficiency of both information transfer and processing in ML systems is improved by employing information-based system metrics rather than fidelity circuit metrics to aggressively reduce component power of information transfer or complexity of ML algorithms for information processing. This new dimension of energy efficiency vs. robustness trade-off is made possible by taking advantage of accuracy relaxation on circuit level operations offered by the probabilistic nature of information-based system metrics or the inherent structure in ML algorithms.

Such energy efficiency vs. robustness trade-off can also be leveraged to address the robustness challenge of implementing information processing subsystems on stochastic unreliable fabrics such as NTV for more aggressively reduced computational cost. Aligning with this thought, this chapter proposes a new SEC technique referred to as RD-SEC that enables robust CNNs operating in the NTV regime. The opportunity is that one commonly employed operation in signal processing and ML applications is MVMs in which the same input vector is projected to a set of weight vectors. Examples include CNNs [66], filter banks for feature extraction [80], principal component analysis (PCA) [81], and wavelet transforms [82]. RD-SEC exploits inherent structure within MVMs for low-cost error detection and compensation.

The remainder of this chapter is organized as follows. Section 4.1 provides background on low power design techniques, ANT and rank decomposition. Section 4.2 presents the proposed MVM-based CNN architecture and RD-SEC technique to enhance robustness. The error model generation and validation are presented in Section 4.3. Simulation results are shown in Section 4.4. Finally, conclusions and future work are presented in Section 4.5.

4.1 Background

4.1.1 Low Power Design Techniques

Various low power techniques can be used to reduce the energy consumption of MVMs. At the logic level, programmable CSE [42] is a low power technique, where common subexpressions (CSs) in the coefficients are first computed using shift and add, and then summed up to obtain the final product. Programmability is enabled via a look-up table.

In order to further reduce energy, NTV was proposed to operate the devices at or near their threshold voltage (V_{th}), and has shown an energy reduction on the order of $10\times$ [11]. However, the energy efficiency of NTV comes at a cost of exponential increase in the normalized delay variation, leading to an increased functional failure. Specifically, circuit simulations in a commercial 45 nm CMOS show that the delay variation of an 8-bit ripple-carry adder (RCA) increases by $8.5\times$ at $V_{dd} = 0.35\text{V}$ (NTV) compared with that at the nominal $V_{dd} = 1.1\text{V}$ due to process variations. To address the variation challenge, the traditional approach is to add design margin, which substantially reduces the benefits of NTV [11]. For example, it is estimated that the employing of voltage margining to ensure error-free operation results in $3.1\times$ energy overhead for the 8-bit RCA operating at 0.35V . Techniques such as body biasing [83] or variable pipeline stage latency [84] have been proposed. Although these techniques demonstrated some degree of effectiveness, they can incur significant overheads due to the local nature of variations.

4.1.2 Algorithmic Noise-Tolerance (ANT)

ANT is an algorithmic technique that employs error statistics to perform error compensation, and has been shown to be effective for signal processing and ML kernels [49]. Specifically, ANT incorporates a main block (**M**-block) and an estimator block (**E**-block) which is an approximate version of the **M**-block (see Fig. 4.1(a)). The **M**-block is subject to large magnitude errors η (e.g., timing errors which typically occur in the MSBs) while the **E**-block is subject to small magnitude errors e (see Fig. 4.1(b), e.g., due to quantization noise in the LSBs), i.e.:

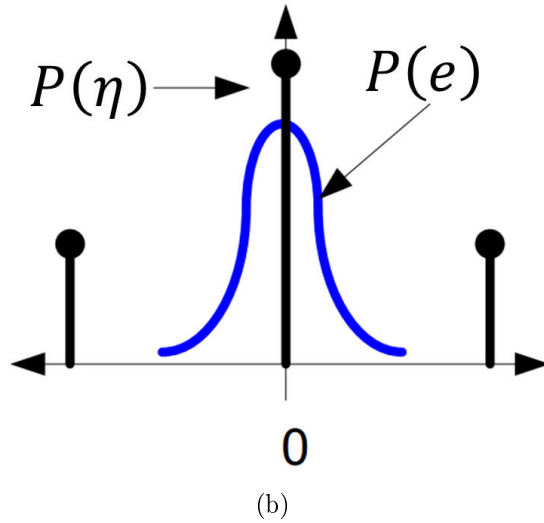
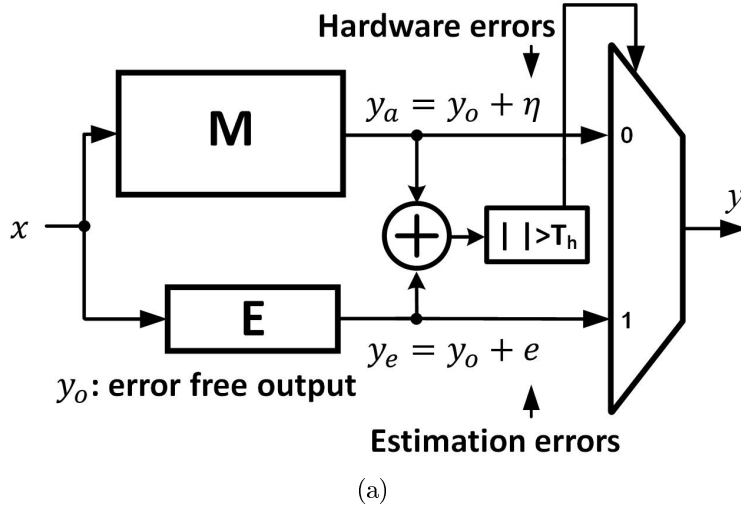


Figure 4.1: Algorithmic noise-tolerance (ANT): (a) architecture, and (b) the error statistics in the **M**-block and **E**-block [50].

$$y_a = y_o + \eta \quad (4.1)$$

$$y_e = y_o + e \quad (4.2)$$

where y_o , y_a , and y_e are the error-free, the **M**-block and **E**-block outputs, respectively. ANT exploits the difference in the error statistics of η and e to detect and compensate for errors and obtain the final corrected output \hat{y} as follows:

$$\hat{y} = \begin{cases} y_a & \text{if } |y_a - y_e| \leq T_h \\ y_e & \text{otherwise} \end{cases}, \quad (4.3)$$

where T_h is an application dependent threshold parameter chosen to maximize the performance of ANT.

4.1.3 Rank Decomposition

Rank decomposition exists for every finite-dimensional matrix [85]. Assume \mathbf{A} is an $N \times M$ matrix ($N < M$) whose rank is R , then $R \leq N$ and there exist R linearly independent rows in \mathbf{A} . A rank decomposition of \mathbf{A} is a product $\mathbf{A} = \mathbf{BC}$, where $\mathbf{B} = [\mathbf{b}_1, \dots, \mathbf{b}_R]$ is a $N \times R$ basis matrix, \mathbf{b}_r ($r = 1, \dots, R$) is the r^{th} $N \times 1$ basis vector, and $\mathbf{C} = [\mathbf{c}_1, \dots, \mathbf{c}_M]$ is a $R \times M$ coefficient matrix. Every column vector of \mathbf{A} is a linear combination of the columns in matrix \mathbf{B} . That is, the j^{th} column \mathbf{a}_j in the matrix $\mathbf{A} = [\mathbf{a}_1, \dots, \mathbf{a}_M]$ can be expressed as $\mathbf{a}_j = \mathbf{B}\mathbf{c}_j = c_{1j}\mathbf{b}_1 + \dots + c_{Rj}\mathbf{b}_R$ with $\mathbf{c}_j = [c_{1j}, \dots, c_{Rj}]^T$.

4.2 The Proposed RD-SEC Technique

This section describes the proposed error compensation technique RD-SEC to enable robust CNN design in the NTV regime. First, we reformulate the C-layer computation in terms of the MVM.

4.2.1 MVM-based CNNs

Equation (3.2) can be rewritten in a vector form as follows:

$$\begin{bmatrix} y_1[j] \\ \vdots \\ y_m[j] \\ \vdots \\ y_M[j] \end{bmatrix} = \begin{bmatrix} \sum_{l=1}^L \mathbf{w}_{1l}^T \mathbf{x}_{jl} \\ \vdots \\ \sum_{l=1}^L \mathbf{w}_{ml}^T \mathbf{x}_{jl} \\ \vdots \\ \sum_{l=1}^L \mathbf{w}_{Ml}^T \mathbf{x}_{jl} \end{bmatrix} = \sum_{l=1}^L \mathbf{W}_l^T \mathbf{x}_{jl}, \quad (4.4)$$

where $\mathbf{W}_l = [\mathbf{w}_{1l}, \dots, \mathbf{w}_{Ml}]$. It can be seen that (4.4) is the sum of L MVMs, where the l^{th} MVM is given by $\mathbf{W}_l^T \mathbf{x}_{jl}$. A single stage of an MVM-based CNN in Fig. 4.2 consists of input and weight buffers, an MVM-based C-layer, and an S-layer. Specifically, the input vectors and weight matrices are streamed from the input and weight buffers, respectively. The MVM-based C-layer accepts the input vectors and weight matrices, and obtains the M outputs according to (3.1) and (3.2). In the S-layer, the spatial resolution of the C-layer output FMs is reduced by either averaging or max pooling.

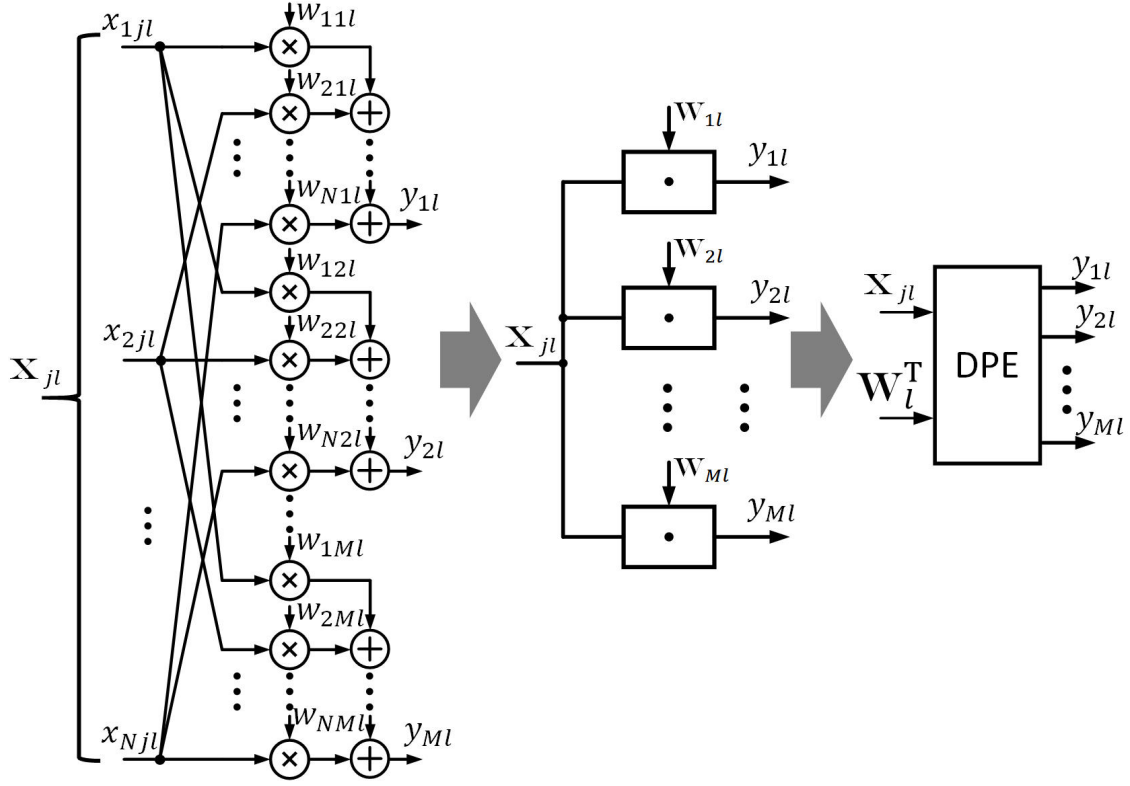
4.2.2 Rank Decomposed SEC (RD-SEC): Principle and Architecture

The formulation of an MVM-based CNN in Section 4.2.1 enables us to exploit redundancy within an MVM for statistical error compensation. The proposed approach RD-SEC employs low-cost estimators from a set of basis vectors in the $N \times M$ weight matrix \mathbf{W} (see (1.2)). To do so, we make use of the rank decomposition of \mathbf{W} [85]:

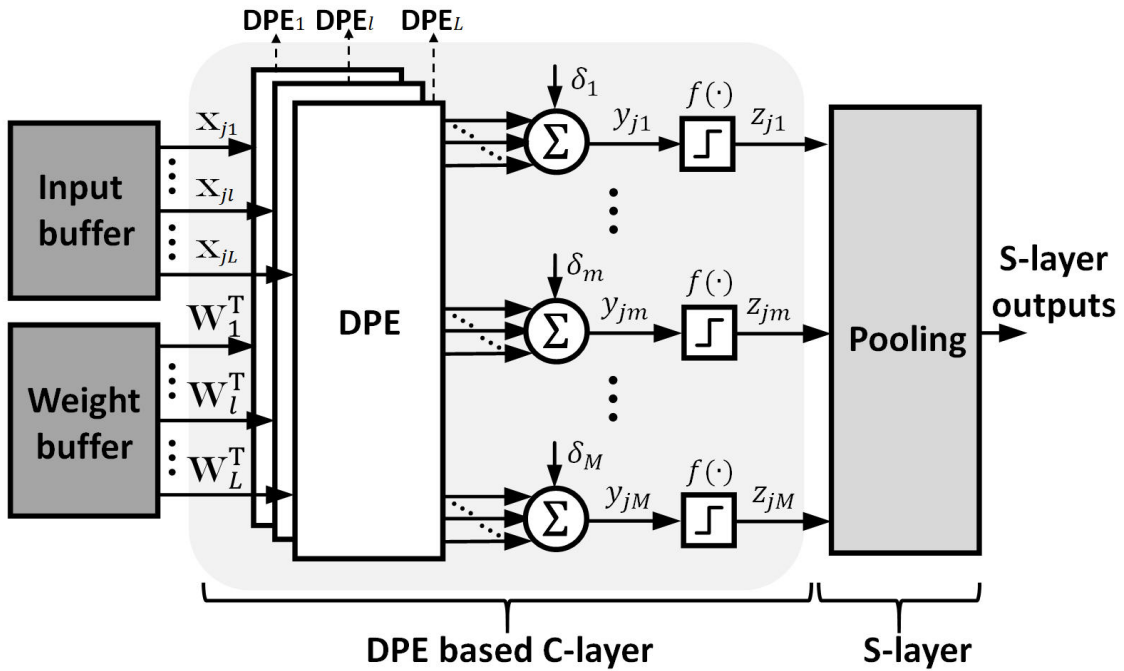
$$\mathbf{W} = \mathbf{BC}, \quad (4.5)$$

where $\mathbf{B} = [\mathbf{b}_1, \dots, \mathbf{b}_R]$ is an $N \times R$ basis matrix with $R = \text{rank}(\mathbf{W})$ (assume $M > N$, then $R \leq N$), \mathbf{b}_r ($r = 1, \dots, R$) is the r^{th} $N \times 1$ basis vector, $\mathbf{C} = [\mathbf{c}_1, \dots, \mathbf{c}_M]$ is an $R \times M$ coefficient matrix, and \mathbf{c}_m ($m = 1, \dots, M$) is the m^{th} $R \times 1$ coefficient vector. We choose $\mathbf{b}_i = \mathbf{w}_i$ ($i = 1, \dots, R$) so that

$$\mathbf{W} = \mathbf{BC} = \mathbf{B} \begin{bmatrix} \mathbf{I}_R & \mathbf{C}_e \end{bmatrix}, \quad (4.6)$$



(a)



(b)

Figure 4.2: Architecture of: (a) a (N, M) dot product ensemble (MVM), where $\mathbf{w}_{ml} = [w_{1ml}, \dots, w_{Nml}]$ and $\mathbf{W}_l = [\mathbf{w}_{1l}, \dots, \mathbf{w}_{Ml}]$, and (b) one stage MVM-based CNN consisting of a C-layer and an S-layer.

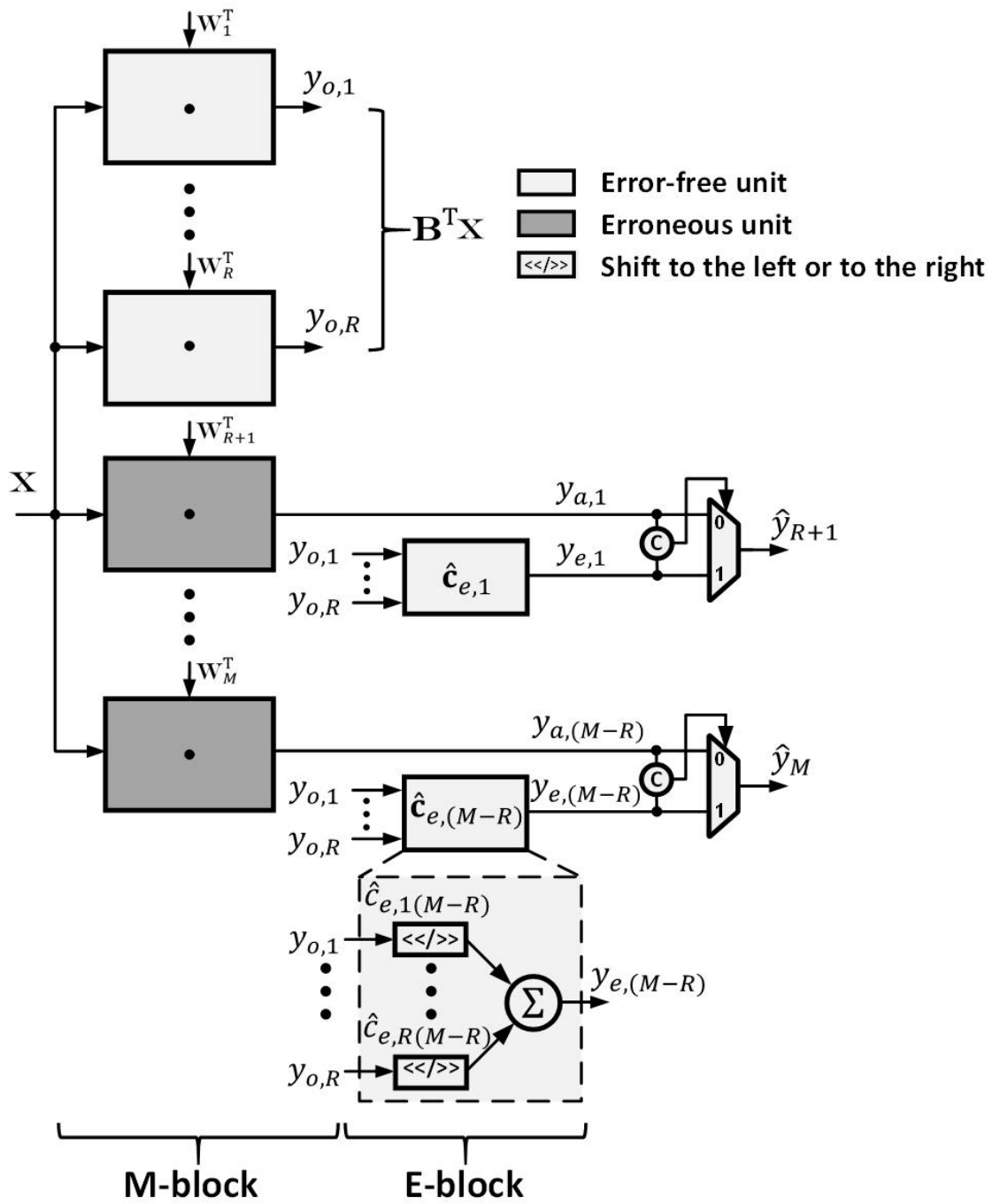


Figure 4.3: RD-SEC applied to an MVM.

where \mathbf{I}_R is an $R \times R$ identity matrix, and $\mathbf{C}_e = [\mathbf{c}_{e,1}, \dots, \mathbf{c}_{e,M-R}]$ is an $R \times (M - R)$ matrix. Substituting (4.6) into (1.2), we have:

$$\begin{aligned} \mathbf{y} &= [\mathbf{I}_R \quad \mathbf{C}_e]^\top (\mathbf{B}^\top \mathbf{x}) \\ &= [\mathbf{I}_R \quad \mathbf{C}_e]^\top \mathbf{y}_o \\ &= \begin{bmatrix} \mathbf{y}_o \\ \mathbf{y}_a \end{bmatrix}, \end{aligned} \tag{4.7}$$

where $\mathbf{y}_o = \mathbf{B}^\top \mathbf{x} = [y_{o,1}, \dots, y_{o,R}]^\top$ is the error-free $R \times 1$ vector, and $\mathbf{y}_a = \mathbf{C}_e^\top \mathbf{y}_o = [y_{a,1}, \dots, y_{a,(M-R)}]^\top$ is an $(M - R) \times 1$ output vector from the \mathbf{M} -block subject to errors. In RD-SEC, we derive a low-cost estimator of \mathbf{y}_a using the error-free output \mathbf{y}_o and a rounded coefficient matrix $\hat{\mathbf{C}}_e^\top$, i.e.:

$$\mathbf{y}_e = \hat{\mathbf{C}}_e^\top \mathbf{y}_o = [\hat{\mathbf{c}}_{e,1}, \dots, \hat{\mathbf{c}}_{e,(M-R)}]^\top \mathbf{y}_o, \tag{4.8}$$

where $\mathbf{y}_e = [y_{e,1}, \dots, y_{e,(M-R)}]^\top$ is an $(M - R) \times 1$ estimation vector, $\hat{\mathbf{C}}_e^\top = \text{round}(\mathbf{C}_e^\top)$ where the $\text{round}(\cdot)$ operator rounds an element to the nearest power of 2, and $\hat{\mathbf{c}}_{e,m} = [\hat{c}_{e,1m}, \dots, \hat{c}_{e,Rm}]^\top$ is the m^{th} $R \times 1$ coefficient vector corresponding to $y_{e,m}$. Equation (4.8) indicates that $y_{e,m}$ can be implemented using only shifts and adds. Finally, the m^{th} error compensated output \hat{y}_m is obtained as follows:

$$\hat{y}_m = \begin{cases} y_{o,m} & \text{if } m \leq R \\ y_{a,(m-R)} & \text{if } m > R \ \& \ |y_{a,(m-R)} - y_{e,(m-R)}| \leq T_h, \\ y_{e,(m-R)} & \text{otherwise} \end{cases} \tag{4.9}$$

where the threshold T_h is an application dependent parameter chosen to maximize system performance [49]. The RD-SEC architecture is shown in Fig. 4.3.

4.2.3 RD-SEC Overhead

The overhead of an RD-SEC-based CNN can be approximated relative to the **M**-block in an MVM. The computational overhead γ of RD-SEC relative to the **M**-block is defined as:

$$\gamma = \frac{N_{\text{P}} - N_{\text{conv}}}{N_{\text{conv}}} = \frac{(M - R)}{M} \alpha, \quad (4.10)$$

where N_{P} and N_{conv} denote the complexities of the RD-SEC-based MVM and the conventional MVM in terms of the number of full adders (FAs), respectively, and α quantifies the ratio of the complexities of one **E**-block and **M**-block (only $M - R$ out of M channels have **E**-blocks (see Fig. 4.3)). The detailed expression for α is provided in Section 4.6.

The γ_{C} and γ_{F} in Fig. 4.4 correspond to the computational overhead of the C1/C2 layers and the F1 layer of the CNN in [68], respectively. Figure 4.4 shows that γ_{C} increases with N for $N \leq 5$, and then decreases with N . This is because α increases with N due to the increased number of adders in (4.8), while at the same time, the number of **E**-blocks $M - R$ reduces since $R = N$. Similar results were obtained for γ_{F} . This indicates that RD-SEC overhead reduces with N for large vector length (i.e., $N \geq 10$). Specifically, $\gamma_{\text{C}} \approx 5\%$ and $\gamma_{\text{F}} \approx 15\%$ when RD-SEC is applied to the CNN in [68].

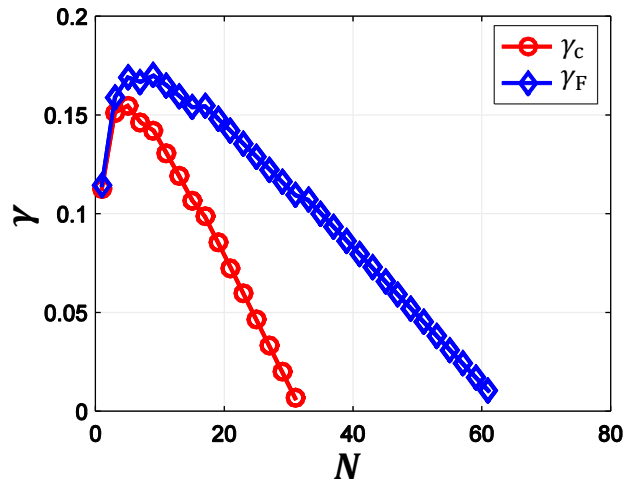


Figure 4.4: Overhead of the RD-SEC-based MVM: computational overhead γ vs. N , where the corresponding parameters are summarized in Table 4.1.

Table 4.1: Parameters for the γ_C and γ_F in Fig. 4.4

	Parameters
γ_C	$B_{in} = 7, B_w = 8, R = N, M = 32$
γ_F	$B_{in} = 7, B_w = 8, R = N, M = 100$

4.3 Error Model Generation and Validation

This section presents the timing error model generation methodology [86] and the validation of this timing error model in a commercial 45 nm CMOS.

4.3.1 Error Model Generation

The error model generation methodology is shown in Fig. 4.5, and described below:

1) Characterize the gate delay distribution vs. operating voltage V_{dd} of basic gates such as AND and XOR using HSPICE. Specifically, the gate delay d is modeled as a Gaussian random variable, i.e., $d \sim \mathcal{N}(\hat{\mu}_d, \hat{\sigma}_d)$, where the mean $\hat{\mu}_d$ and standard variation $\hat{\sigma}_d$ are estimated from HSPICE Monte Carlo (MC) simulations with 1000 iterations.

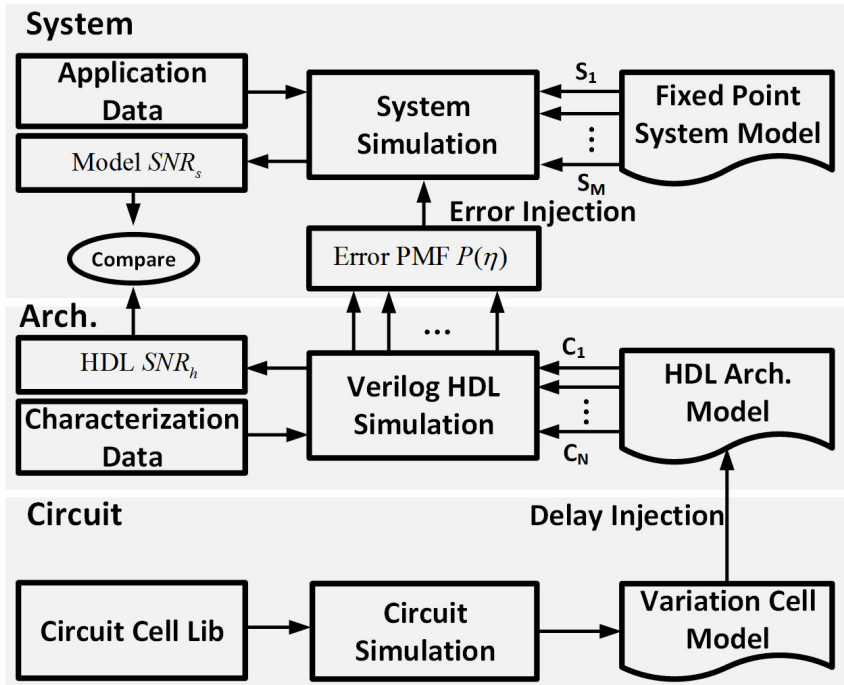
2) Implement the MVM architecture shown in Fig. 4.2(a) using structural Verilog HDL and the basic gates characterized in Step 1.

3) Emulate process variations at NTV by generating multiple (30) architectural instances and assigning random gate delays obtained via sampling the gate delay distributions obtained in Step 1.

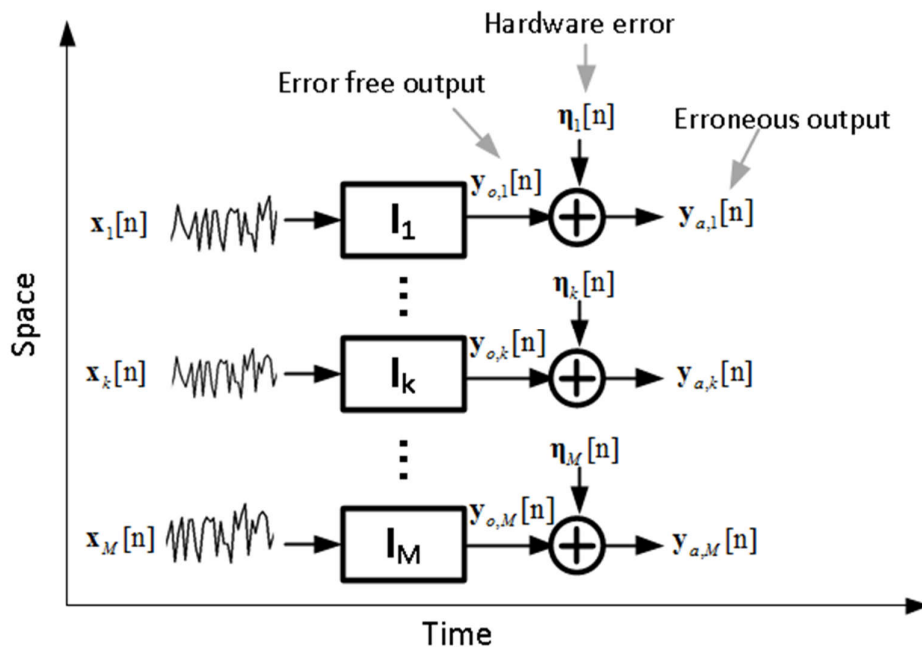
4) Generate the error PMF $P(\eta)$ employing the procedure in [86].

Specifically, Steps 3-4 are performed according to Algorithm 2. During system level simulations, the system performance (i.e. the probability of detection) is evaluated by performing error injection using the HDL error PMF $P(\eta)$.

Figure 4.6 shows that the extent of within-the-die (WID) delay variation $(\hat{\sigma}/\hat{\mu})_d$ of an AND gate increases from 0.03 at the supply voltage $V_{dd} = 1.2V$ (see Fig. 4.6(a)) to 0.24 at $V_{dd} = 0.4V$ (see Fig. 4.6(b)), indicating an $8\times$ increase in the WID delay variation at NTV compared with that of the super-threshold regime. The worst-case normalized confidence intervals (with a 99% confidence level) for $\hat{\mu}_d$ and $\hat{\sigma}_d$ of the AND gate delays are 1% and

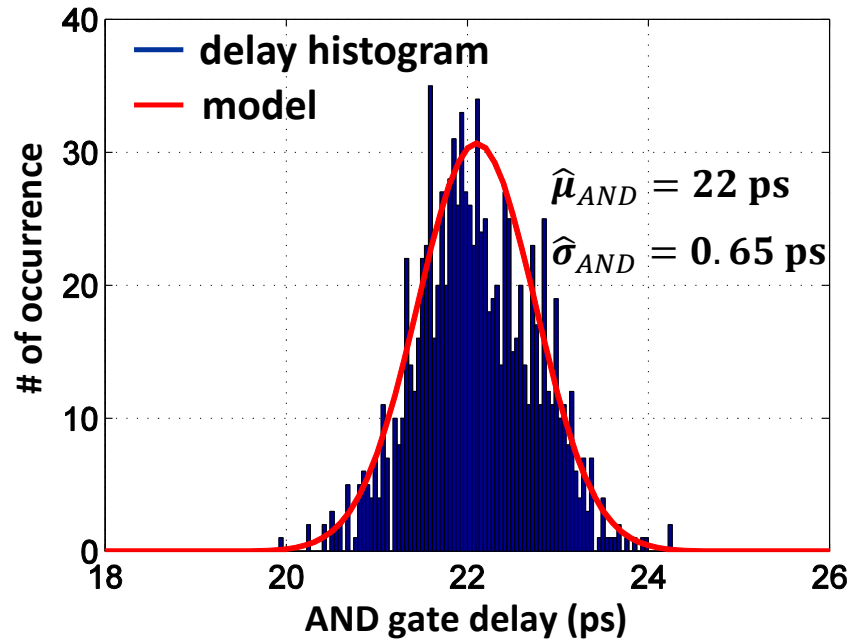


(a)

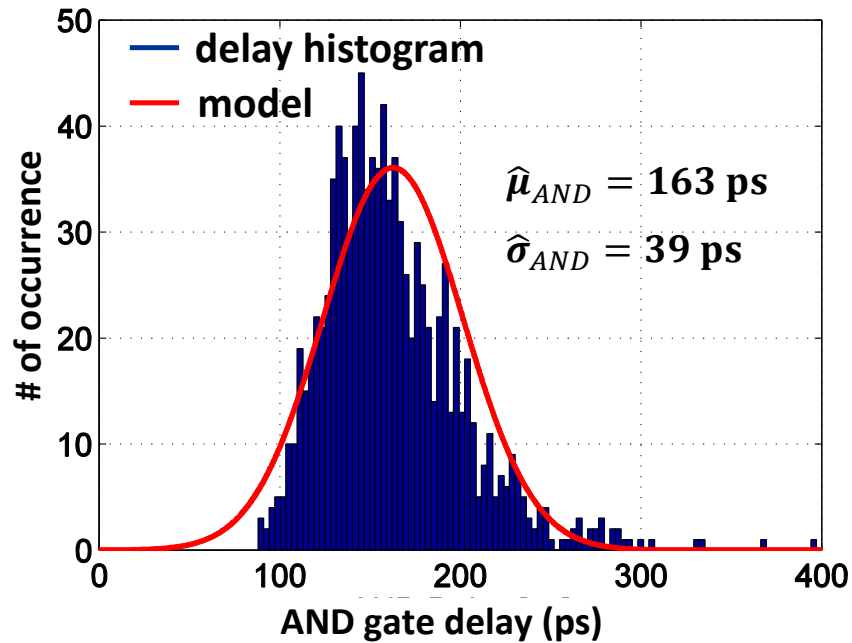


(b)

Figure 4.5: Block diagram of: (a) model generation methodology, and (b) error modeling framework.



(a)



(b)

Figure 4.6: Illustration of the AND gate within-the-die (WID) delay histograms from HSPICE Monte Carlo (MC) simulations and the AND gate delay model at: (a) $V_{dd} = 1.2V$, (b) $V_{dd} = 0.4V$, with 1000 MC iterations.

Algorithm 2 Algorithm to obtain the kernel error PMF $P(\eta)$ under each operating voltage V_{dd} .

- 1: **Initialize** the frequency to be the maximum error free frequency with delay of basic gates set to their estimated means, and obtain the error free output \mathbf{y}_o for $N = 10^5$ random inputs
 - 2: **for** each kernel instance i **do**
 - 3: instantiate die-to-die (D2D) delay d_{D2D} via sampling the D2D delay distribution
 - 4: **for** each gate within the instance i **do**
 - 5: instantiate WID gate delay d_{WID} by sampling the WID delay distribution, and set $d_g = d_{D2D} + d_{WID}$
 - 6: **end for**
 - 7: **for** each of the n -th random input **do**
 - 8: obtain kernel output $\mathbf{y}_i(n)$ and error $\epsilon_i(n) = \mathbf{y}_o(n) - \mathbf{y}_i(n)$
 - 9: **end for**
 - 10: obtain the error PMF: $P(\eta)_i = hist(\epsilon_i)/N$
 - 11: **end for**
-

5%, respectively, where a confidence interval with a p ($0 \leq p \leq 1$) confidence level implies that the probability of the (random) confidence interval contains the true percentage is at least p [87]. These results indicate that the 1000 MC iterations are sufficient to provide high accuracy estimation for the gate level delay models.

4.3.2 Error Model Validation

This subsection validates the error model generation methodology in Section 4.3.1. A complete HDL simulation for the entire CNN is infeasible due to the large amount of the MVMs; thus, we validate the model for a single MVM employing the circuit-level SNR of the main block (see Fig. 4.1 and (4.1)) as follows:

$$SNR = 10 \log_{10} \left(\frac{\sigma_{y_o}^2}{\sigma_{\eta}^2} \right), \quad (4.11)$$

where $\sigma_{y_o}^2$ and σ_{η}^2 are the variances of the error-free output y_o and the timing error η , respectively.

The validation procedure is as follows. First, HDL (bit and clock accurate) simulations of

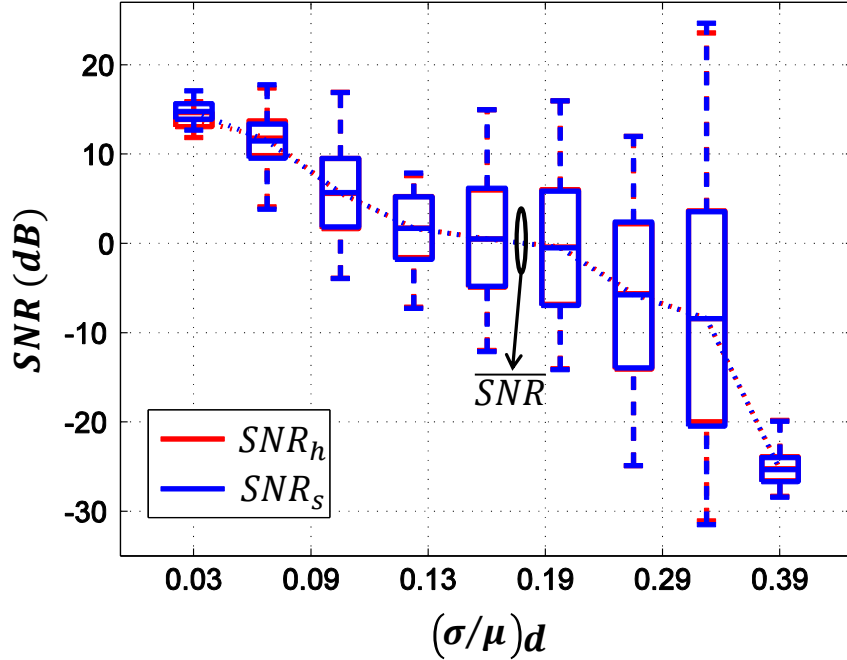


Figure 4.7: Validating the error model generation methodology by comparing SNR from HDL simulations and the NTV methodology based on 30 MVM instances with 10^5 random input samples for each instance operating at gate level delay variation of 3%-39%.

each instance in Step 3 are run to obtain error samples and circuit-level SNR SNR_h . Second, fixed-point MATLAB simulations using the PMF from Step 4 to inject errors for the MVMs are run to obtain circuit-level SNR SNR_s . Third, we compare SNR_s with SNR_h .

Figure 4.7 plots SNR_h obtained via HDL simulations using the characterized gate delay distributions and SNR_s obtained via MATLAB simulations using error PMF as a function of the gate level delay variation $(\sigma/\mu)d$. It is found that the difference between the median SNR_h ($\overline{SNR_h}$) and SNR_s ($\overline{SNR_s}$) is no more than 5% when $(\sigma/\mu)d$ increases from 3% to 39%. Figure 4.7 shows that the variation of SNR increases for $3\% \leq (\sigma/\mu)d \leq 34\%$, and then decreases because all the instances are subject to large timing errors. Figure 4.7 further shows that the maximum and minimum values of SNR_h and SNR_s differ by no more than 6% and 4%, respectively. These results indicate that the timing error is well-modeled by its PMF.

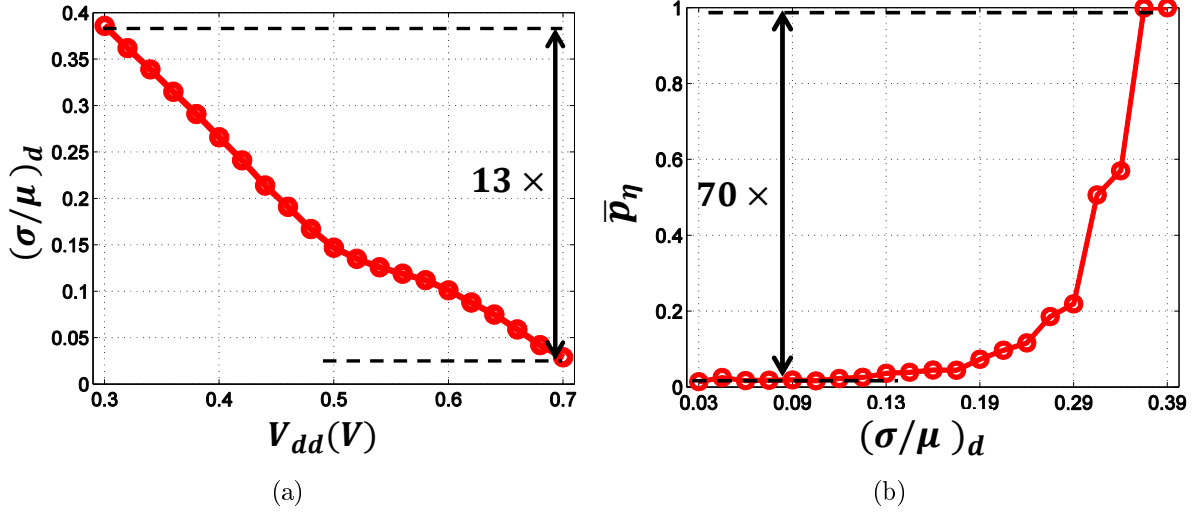


Figure 4.8: Characterization of: (a) process variations in terms of $(\sigma/\mu)_d$ vs. V_{dd} , and (b) the impact of process variations on MVM error rate \bar{p}_η based on 30 MVM instances.

4.4 Simulation Results

In this section, we evaluate the performance of RD-SEC-based CNNs employing the error PMFs from Section 4.3 for the MNIST [68] and CIFAR-10 datasets [79].

4.4.1 System Set-up

Similar to the case in PredictiveNet, the bias term δ_m in (3.1) and kernel \mathbf{w}_{ml} in (3.2) of the CNNs being studied are trained using the back-propagation algorithm [66]. The following two architectures are considered: 1) a slow CNN architecture with RD-SEC applied to the C-layers and F1 layer (denoted as RD-SEC CNN), where the multipliers and adders are implemented using Baugh-Wooley (BW) multiplier and RCA, respectively; 2) an uncompensated fast CNN architecture (denoted as Conv CNN), where the multipliers and adders are implemented using the programmable CSE technique in [42] and Kogge-Stone adder, respectively. The fast architecture is chosen for comparison because it will result in the largest energy savings in the error-free case when voltage scaling is employed.

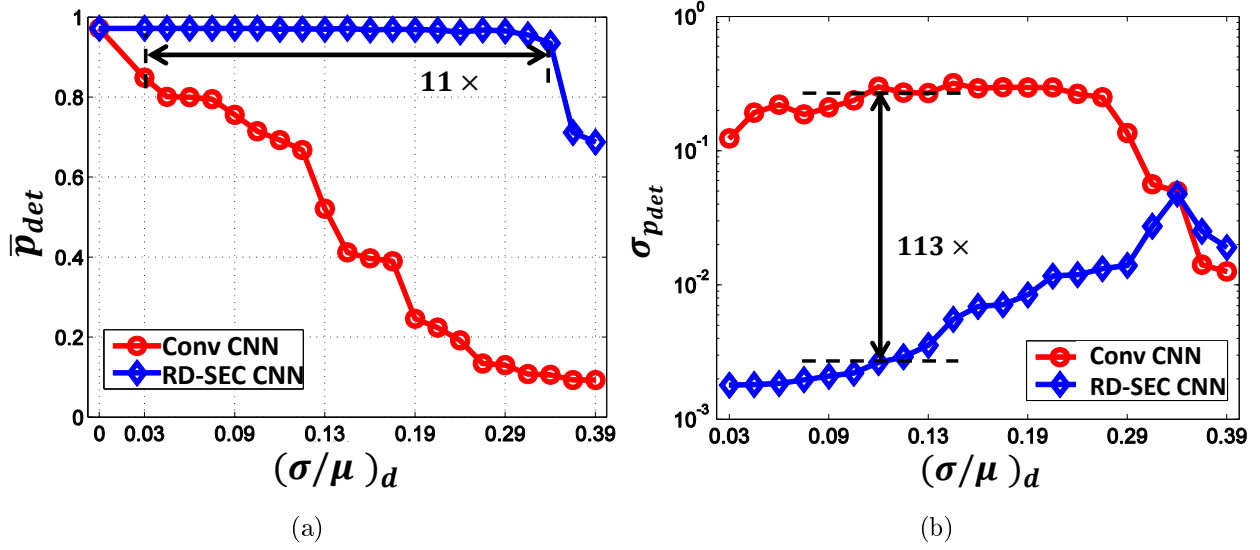


Figure 4.9: Simulation results for the MNIST dataset: (a) \bar{p}_{det} vs. $(\sigma/\mu)_d$, and (b) $\sigma_{p_{det}}$ vs. $(\sigma/\mu)_d$, based on 30 CNN instances in the presence of process.

4.4.2 Characterization

First, the extent of process variation in NTV is characterized in terms of $(\sigma/\mu)_d$. Figure 4.8(a) shows that $(\sigma/\mu)_d$ increases by $13\times$ from 3% to 39% as the supply voltage V_{dd} decreases from 0.7 V to 0.3 V. Note that process variation makes the detection accuracy $p_{det} = P\{\hat{T} = t\}$ (\hat{T} and t are the classifier decision and the true label, respectively) a random variable, which is denoted as P_{det} . Figure 4.8(b) shows that the median error rate \bar{p}_η (where the error rate is defined as $p_\eta = P\{\eta \neq 0\}$) increases by $70\times$ from 1.4×10^{-2} to 0.99 as V_{dd} decreases from 0.7 V to 0.3 V. At a $(\sigma/\mu)_d = 34\%$, the median error rate $\bar{p}_\eta = 0.57$.

Next, we employ the error PMFs obtained from Step 4 of the NTV error modeling methodology (see Section 4.3.1) to inject errors in fixed-point MATLAB simulations of CNN architectures to evaluate their robustness to timing errors in NTV. We compare the two architectures in terms of median (\bar{p}_{det}) and standard deviation ($\sigma_{p_{det}}$) of the detection accuracy P_{det} . This is because p_η and p_{det} are spatially distributed random variables in the presence of process variations, where the path delay distribution and the timing violations (hence p_η and p_{det}) are different for each MVM or CNN instance.

Table 4.2: Summary of CNN Parameters from [68]

Parameter Definition		CNN Parameter Summary				
Parameter	Description	Layer	L	M	$I_1 \times I_2$	$K \times K$
L/M	# of input/output FMs	C1	1	32	28×28	5×5
$K \times K$	size of kernels	C2	32	64	12×12	5×5
$I_1 \times I_2$	size of input FMs	F1	64	100	4×4	2×1

4.4.3 Comparison of \bar{p}_{det} and $\sigma_{p_{det}}$ for CNNs using MNIST

The parameters of the CNNs for the MNIST dataset are summarized in Table 4.2 [68]. The precision B_{in} and B_w are set to 7 bits and 8 bits, respectively, ensuring the error-free fixed-point detection accuracy to be within 0.2% of the floating-point detection accuracy of 0.98.

Figure 4.9(a) shows that RD-SEC CNN is able to maintain $\bar{p}_{det} \geq 0.9$ (the worst-case confidence interval with a 95% confidence level is [0.92, 0.95]) for $(\sigma/\mu)_d \leq 34\%$, whereas Conv CNN can only maintain the same performance for $(\sigma/\mu)_d \leq 3\%$ (the worst-case confidence interval with a 95% confidence level is [0.80, 0.90]). Thus, RD-SEC CNN is able to deliver a high detection accuracy in the presence of high error rate of $\bar{p}_\eta \leq 0.57$ (see Fig. 4.8(b)). This indicates an $11\times$ improvement compared with the Conv CNN. Figure 4.9(b) shows that the RD-SEC CNN can achieve $\sigma_{p_{det}} = 2.6 \times 10^{-3}$ (with a 95% level confidence interval of $[2.1 \times 10^{-3}, 3.6 \times 10^{-3}]$), indicating an $113\times$ reduction in $\sigma_{p_{det}}$ as compared to that of the Conv CNN $\sigma_{p_{det}} = 0.3$ (with a 95% level confidence interval of [0.24, 0.4]) at $(\sigma/\mu)_d = 11\%$. Figure 4.9(b) also shows that $\sigma_{p_{det}}$ of the RD-SEC CNN is no more than 4.8×10^{-2} , whereas the maximum $\sigma_{p_{det}}$ of the Conv CNN is 0.32 for $3\% \leq (\sigma/\mu)_d \leq 39\%$.

Furthermore, Fig. 4.9(b) demonstrates that $\sigma_{p_{det}}$ of the RD-SEC CNN increases from 1.8×10^{-3} to 4.8×10^{-2} when $(\sigma/\mu)_d$ increases from 3% to 34%, and then decreases. When $(\sigma/\mu)_d > 34\%$, $\sigma_{p_{det}}$ of the RD-SEC CNN is larger than that of the Conv CNN because all the instances of the Conv CNN achieve a low $P_{det} \approx 0.1$, whereas some instances of the RD-SEC CNN can still achieve a $P_{det} \geq 0.9$, leading to a larger $\sigma_{p_{det}}$.

To understand the robustness improvement achieved by RD-SEC, the input, C1 FMs (12 out of 32), the output vector and the final decision \hat{T} are analyzed (see Fig. 4.10). Note

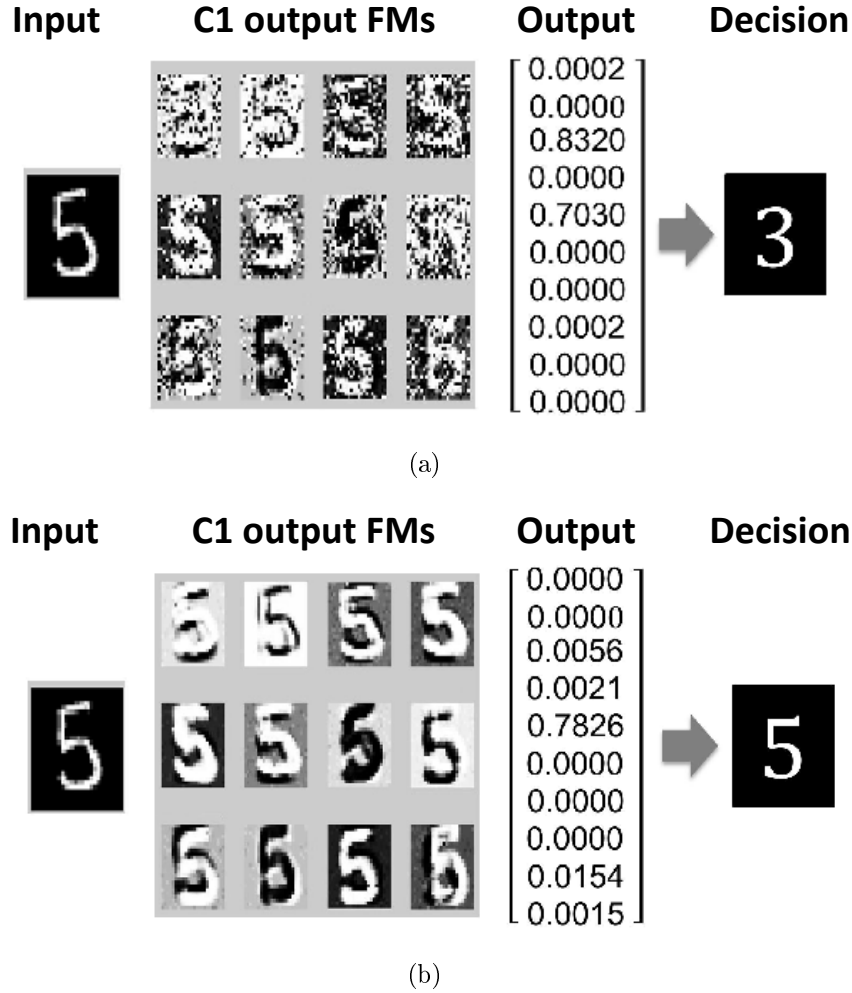


Figure 4.10: An example of the C1 FMs and the output vector from: (a) the Conv CNN, and (b) the RD-SEC CNN, when the input digit is “5” and $(\sigma/\mu)_d = 27\%$.

that \hat{T} is chosen as the index of the maximum element in the output vector. Figure 4.10(a) shows that the timing errors contaminate the extracted features in the Conv CNN, leading to classification failure. Specifically, the output vector has two peaks (at positions “3” and “5”) due to the contaminated features, resulting in a wrong decision “3” instead of the correct one “5”. On the other hand, RD-SEC is able to compensate for timing errors, and thus enables the RD-SEC CNN to extract correct features for correct classification even in the presence of a large number of timing errors.

Table 4.3: Summary of CNN Parameters for CIFAR-10 Dataset [79]

Parameter Definition		CNN Parameter Summary				
Parameter	Description	Layer	L	M	$I_1 \times I_2$	$K \times K$
L	# of input FMs	C1	3	3×32	32×32	5×5
M	# of output FMs	C2	3×32	3×32	16×16	5×5
$K \times K$	size of kernels	C2	3×32	3×64	8×8	5×5
$I_1 \times I_2$	size of input FMs	F1	3×64	3×64	4×4	2×1

4.4.4 Comparison of \bar{p}_{det} and $\sigma_{p_{det}}$ for CNNs using CIFAR-10

To demonstrate the generality of the proposed RD-SEC technique, RD-SEC based CNN is also applied to the CIFAR-10 dataset [79], which contains three C-layers and S-layers and one F-layer. The parameters of the CNNs for the CIFAR-10 dataset are summarized in Table 4.3, which is developed based on the LeNet-5 CNN in [66]. The precision B_{in} and B_w are set to 8 bits and 7 bits, respectively, ensuring the error-free fixed-point detection accuracy to be within 0.3% of the floating-point detection accuracy of 0.8.

Figure 4.11(a) shows that RD-SEC CNN is able to maintain $\bar{p}_{det} \geq 0.8$ (the worst-case confidence interval with a 95% confidence level is [0.78, 0.80]) for $(\sigma/\mu)_d \leq 29\%$, whereas Conv CNN can only maintain the same performance for $(\sigma/\mu)_d \leq 6\%$ (the worst-case confidence interval with a 95% confidence level is [0.63, 0.80]). This indicates a $5\times$ improvement in error rate tolerance compared with the Conv CNN. Figure 4.11(b) shows that the RD-SEC CNN can achieve $\sigma_{p_{det}} = 2.6 \times 10^{-3}$ (with a 95% level confidence interval of $[2.1 \times 10^{-3}, 3.6 \times 10^{-3}]$), indicating an $85\times$ reduction in $\sigma_{p_{det}}$ as compared to that of the Conv CNN $\sigma_{p_{det}} = 0.229$ (with a 95% level confidence interval of [0.18, 0.30]) at $(\sigma/\mu)_d = 13\%$. Figure 4.11(b) also shows that $\sigma_{p_{det}}$ of the RD-SEC CNN is no more than 2.3×10^{-2} , whereas the maximum $\sigma_{p_{det}}$ of the Conv CNN is 0.23 for $3\% \leq (\sigma/\mu)_d \leq 39\%$.

Similar to the observation for CNNs using the MNIST dataset in Fig. 4.9(b), Fig. 4.11(b) demonstrates that $\sigma_{p_{det}}$ of the RD-SEC CNN increases from 1.6×10^{-3} to 2.3×10^{-2} when $(\sigma/\mu)_d$ increases from 3% to 34%, and then decreases. When $(\sigma/\mu)_d > 34\%$, $\sigma_{p_{det}}$ of the RD-SEC CNN is larger than that of the Conv CNN because all the instances of the Conv CNN achieve a low $P_{det} \approx 0.1$, whereas some instances of the RD-SEC CNN can still achieve

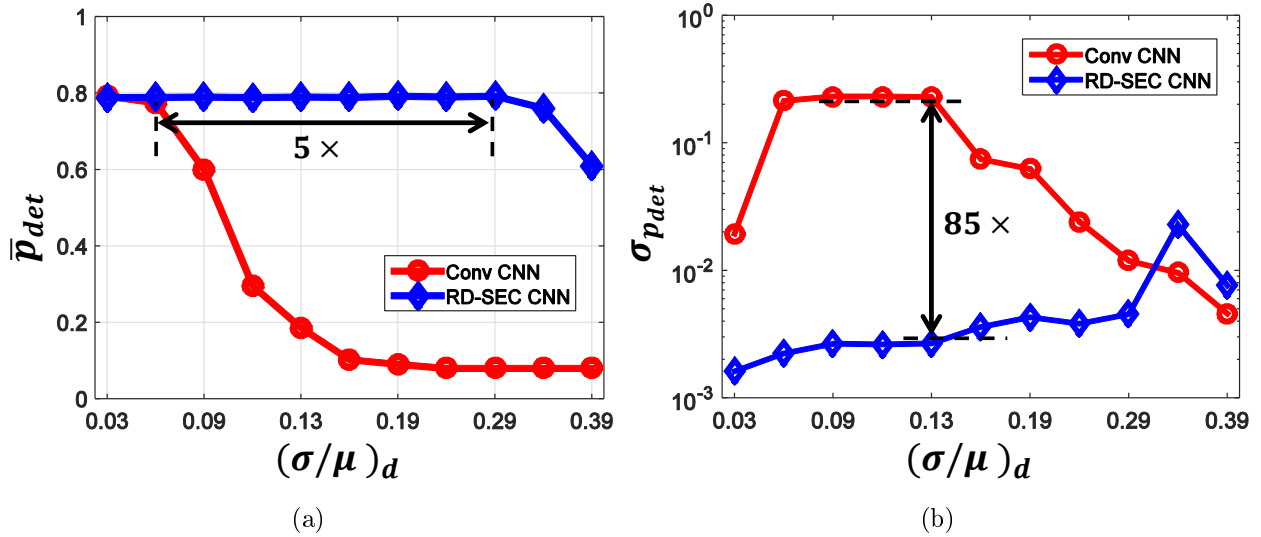


Figure 4.11: Simulation results for the CIFAR-10 dataset: (a) \bar{p}_{det} vs. $(\sigma/\mu)_d$, and (b) $\sigma_{p_{det}}$ vs. $(\sigma/\mu)_d$, based on 30 CNN instances in the presence of process variations.

a $P_{det} \geq 0.8$, leading to a larger $\sigma_{p_{det}}$.

Comparing the simulation results for the MNIST (see Fig. 4.9) and CIFAR-10 (see Fig. 4.11) datasets, there are three observations. First, the Conv CNN for the CIFAR-10 dataset can tolerate larger process variation in terms of $(\sigma/\mu)_d$ ($\leq 6\%$) than that of the MNIST dataset ($\leq 3\%$). This could result from its deeper structure and thus better inherent robustness. Second, the Conv CNN for the CIFAR-10 dataset fails more abruptly as $(\sigma/\mu)_d$ increases than the Conv CNN for the MNIST dataset. This is likely due to the fact that data statistics in the CIFAR-10 dataset are more diverse and thus more sensitive to computation errors. Third, RD-SEC can effectively enhance robustness of the CNNs for both the MNIST and CIFAR-10 datasets even in the presence of a large number of computation errors, indicating its generality as a low power error resiliency algorithmic technique.

4.5 Summary

In this chapter, a new SEC technique named RD-SEC is proposed for MVMs, which is a power-hungry and commonly employed kernel in many signal processing and ML algorithms. RD-SEC is able to significantly enhance the robustness of information processing when op-

erating in NTV for energy efficiency. Therefore, RD-SEC has the potential to enable the deployment of powerful but power-hungry ML algorithms on power-constrained platforms. This work opens the possibility to exploit inherent redundancy or structure within signal processing and ML algorithms to develop low-cost SEC techniques that enable robust computing on unreliable stochastic fabrics for significant improvement in energy efficiency.

4.6 Derivation of α in (4.10)

In this section, we provide a detailed expression for α in (4.10). The complexity is calculated in terms of the number of FAs. From (4.10), α is given by:

$$\alpha = \frac{N_E}{N_M} = \frac{N_{add-R} + N_{MUX}}{N_{DP}}, \quad (4.12)$$

where N_E and N_M denote the complexities of one **E**-block and **M**-block, respectively, N_{add-R} denotes the complexity of the summer in (4.8), N_{MUX} denotes the complexity of MUX-based shifter in (4.8), N_{DP} denotes the complexity of one DP implemented using a Baugh-Wooley (BW) multiplier and ripple-carry adder (RCA). Specifically,

$$N_{add-R} = (R - 1)(B_{out} + \lceil \log_2(R) \rceil - 1) \quad (4.13)$$

$$N_{MUX} = B_{out}(\lceil \log_2(B_{out} + 1) \rceil r_{M2F})R \quad (4.14)$$

$$N_{DP} = NB_w B_{in} + (N - 1)(B_{in} + B_w + \lceil \log_2(N) \rceil - 1), \quad (4.15)$$

where r_{M2F} denotes the normalized complexity of a 2 : 1 MUX over a FA and we use $r_{M2F} = 3.5/9$ [88], the $\lceil a \rceil$ is the ceiling operation, and B_{in} , B_{out} and B_w denote the precision for the input/output and weights, respectively.

Chapter 5

CONCLUSIONS AND FUTURE WORK

With ML systems increasingly becoming woven into our daily lives, energy efficiency will be the key enabler for their pervasive applications. The design for energy-efficient ML systems is made challenging by the need for intensive computation and massive data movement. This dissertation explores techniques to address this challenge for energy-efficient information transfer and processing in ML systems.

5.1 Dissertation Contributions

Current ML systems adopt either centralized cloud computing or distributed edge computing. In both cases, there is an imperative challenge of energy efficiency which will only be made worse with the growing demand for increased I/O bandwidth of high-performance computing in data centers as well as the increasing need to embed complicated ML algorithms to local devices.

To address the energy efficiency challenge in data centers, this dissertation has presented our study on the use of link BER for designing a BOA-based serial link. First, we study, through analysis and simulations, the benefits of the BOA over the CUA in a serial link receiver. In particular, we propose two channel-dependent parameters to quantify these benefits: 1) m -clustering value, and 2) the threshold non-uniformity metric h_t . Furthermore, we show that the BER improvement is greater than 10^6 when $m \geq 5$ and $h_t \leq 0.8$ for a family of channels. Second, we present the design of a 4GS/s, 4-bit BOA IC in a 90 nm CMOS process that includes a single-core, multiple-output passive DAC to enable a variable-threshold and variable-resolution ADC configuration and verify the aforementioned analysis. Measured results demonstrate that a 3-bit BOA has lower SNR requirement than

a 4-bit CUA, thereby supporting the BOA idea in the presence of non-idealities such as finite sampling bandwidth and metastability. In the process, we demonstrate conclusively that ENOB is not the best metric when designing ADCs for serial links. Third, we propose architectures to implement the gradient descent algorithm to compute the representation levels of BOA iteratively. In particular, the architectures for the QL-UD and the individual RL-UD block are proposed.

For the problem of resource-constrained computing at the edge, this dissertation focuses on energy-efficient implementation of ML algorithms, particularly CNNs, for their application on power-constrained embedded platforms. This dissertation develops two techniques for energy-efficient CNN design:

First, this dissertation proposes PredictiveNet which predicts the zero activations (zero prediction) and thereby avoids computing those. In this way, a significant reduction in the number of convolutional operations is achieved without altering the structure or introducing additional side networks. Thus, PredictiveNet has negligible overhead and can easily be applied on top of existing techniques to obtain an even greater reduction in implementation complexity. When applied to CNNs for the MNIST and CIFAR-10 datasets, simulation results show that PredictiveNet can achieve up to $7.2\times$ and $4.4\times$ reduction in the computational and representational costs, respectively, compared to a conventional CNN, and up to $2.5\times$ and $1.7\times$ reduction in the computational and representational costs, respectively, compared to a zero-skipping CNN, while incurring only 0.02 degradation in classification accuracy.

Second, this dissertation proposes a new SEC technique referred to as RD-SEC that is particularly well-suited for MVMs, which is a commonly used signal processing and ML kernel. RD-SEC makes use of the fact that a large fraction of computation inside a MVM can be derived from a small subset, and employs these for low-cost error detection and correction. Simulation results in 45 nm CMOS for an RD-SEC-based CNN architecture show that RD-SEC enables robust CNNs operating in the NTV regime for aggressive energy savings. Specifically, when applied to CNNs for the MNIST dataset, the proposed architecture can achieve a median classification accuracy $P_{det} \geq 0.9$ in the presence of gate level delay variation of up to 34%. This represents an improvement in variation tolerance of $11\times$ as

compared to a conventional CNN. We further show that RD-SEC-based CNN enables up to $113\times$ reduction in the standard deviation of P_{det} compared to the conventional CNN. When applied to CNNs for the CIFAR-10 dataset, the proposed architecture improves variation tolerance by $5\times$ and reduces variation in CNN classification accuracy (P_{det}) by $85\times$ compared with a conventional CNN.

5.2 Future Work

Since ML systems have their unique properties, it is important to consider non-traditional approaches and explore innovations at various levels of design abstraction to address the challenge of designing energy-efficient ML systems. This is because the design space of ML systems is complex due to their interlinked challenges and opportunities at system, architecture, circuit and device levels.

5.2.1 System Level

Designing ML systems in many emerging applications leads to new problems compared to those in the mature areas of signal processing and communication systems. The current practice of ML systems design is being conducted in an ad-hoc manner. Therefore, systematic design methodology and system innovations are critical for realizing ML systems with optimal energy efficiency. ML algorithms are essentially optimization problems and try to minimize certain loss functions. Furthermore, many ML algorithms such as CNNs have been shown to achieve satisfactory performance even when the training reaches only a local minimum rather than the global minimum in the search space. In addition, there are usually more than one local minimum that would lead to the specified system performance. This provides a system-level opportunity to improve energy efficiency: the original optimization problem can be reformulated to include additional constraints on architecture, circuit, and data movement for energy efficiency. Such energy-constrained reformulation aims to achieve a holistic optimal realization of the entire information gathering and processing stack in ML systems. In line with this direction, the PredictiveNet and RD-SEC techniques proposed

in this dissertation offer two future directions. First, one natural next step is to constrain the ML training algorithms such that the zero predictors in PredictiveNet can be shared by many kernels thereby further reducing both the computational and representational costs significantly. Second, imposing additional constraints that favor the reduction of estimation errors in the RD-SEC technique could suppress the estimation errors. Possibly, the low-cost estimators themselves can guarantee marginal system performance loss and eliminate the need for power-hungry implementations.

5.2.2 Architecture Level

Conventional computing architectures separate sensing, computation and storage units. Such architectures may not be energy-efficient for ML systems due to the required large amounts of data movement. New architectures should reduce the need for costly massive data movement in the entire information gathering and processing chain of ML systems and make use of the probabilistic nature of performance metric in ML algorithms for significant energy reduction. The recently proposed in-memory computing architecture [38] is one such example. The MVM-based architecture presented in this dissertation offers another direction to be extended. In the case when data movement is much more expensive than computation, one energy-efficient architecture taking advantage of conventional computing architectures is to store only the basis weights and then derive computations associated with the non-basis weights from those associated with the basis weights. The energy efficiency of such architecture is achieved by trading the more costly data access with relatively low-cost computation. Another possible direction is to develop energy-efficient sensing, storage and processing combo units, and distribute many of such in a systematic and energy-minimizing manner.

5.2.3 Circuit and Device Level

Each decision of many ML algorithms involves hundreds of operations; therefore, the correctness of final decision may not require each operation to be always accurate. This inherent

robustness of ML algorithms can be leveraged to design circuits using non-traditional information metrics such as classification accuracy. The resultant new circuits would be more energy-efficient as they have more relaxed mismatch, precision or linearity requirements than those designed using traditional fidelity metrics such as SQNR. The BOA technique presented in Chapter 2 is one such example. Essentially, new techniques should bridge the probabilistic nature of ML systems and the statistical behavior of circuits in scaled CMOS and emerging technologies for energy efficiency. On the device side, the performance benefits of CMOS scaling have become stagnant if adopting conventional designs. Although emerging technologies such as spin [89] have been shown to have a potential to achieve large energy savings, they have various robustness issues. A promising future direction is to investigate SEC techniques to address those robustness issues and thus enable energy-efficient and robust ML systems on scaled CMOS or emerging beyond-CMOS technologies. In fact, the RD-SEC technique in Chapter 4 is one such example where SEC is shown to enable robust ML systems designed in unreliable stochastic circuit/device fabrics for aggressive improvement in energy efficiency.

REFERENCES

- [1] “The fourth industrial revolution, by Klaus Schwab.” [Online]. Available: <https://www.weforum.org/pages/the-fourth-industrial-revolution-by-klaus-schwab/>
- [2] A. Krizhevsky, I. Sutskever, and G. E. Hinton, “Imagenet classification with deep convolutional neural networks,” in *Advances in Neural Information Processing Systems*, 2012, pp. 1097–1105.
- [3] D. Silver, A. Huang, C. J. Maddison, A. Guez, L. Sifre, G. Van Den Driessche, J. Schrittwieser, I. Antonoglou, V. Panneershelvam, M. Lanctot et al., “Mastering the game of go with deep neural networks and tree search,” *Nature*, vol. 529, no. 7587, pp. 484–489, 2016.
- [4] J. Baliga, R. W. Ayre, K. Hinton, and R. S. Tucker, “Green cloud computing: Balancing energy in processing, storage, and transport,” *Proceedings of the IEEE*, vol. 99, no. 1, pp. 149–167, 2011.
- [5] J. Howard, S. Dighe, S. R. Vangal, G. Ruhl, N. Borkar, S. Jain, V. Erraguntla, M. Konow, M. Riepen, M. Gries et al., “A 48-core IA-32 processor in 45 nm CMOS using on-die message-passing and DVFS for performance and power scaling,” *IEEE Journal of Solid-State Circuits*, vol. 46, no. 1, pp. 173–183, 2011.
- [6] “Google glass,” Google Technology Company, 2013. [Online]. Available: <https://en.wikipedia.org/wiki/Google-Glass/>
- [7] “XPS tower,” Dell Computer Company, 2016. [Online]. Available: <http://www.dell.com/en-us/shop/productdetails/xps-8910-desktop/>
- [8] D. Abts, M. R. Marty, P. M. Wells, P. Klausler, and H. Liu, “Energy proportional data center networks,” in *ACM SIGARCH Computer Architecture News*, vol. 38, no. 3. ACM, 2010, pp. 338–347.
- [9] M. Mansuri, J. E. Jaussi, J. T. Kennedy, T.-C. Hsueh, S. Shekhar, G. Balamurugan, F. O’Mahony, C. Roberts, R. Mooney, and B. Casper, “A scalable 0.128-1 Tb/s, 0.8–2.6 pJ/bit, 64-lane parallel I/O in 32-nm CMOS,” *IEEE Journal of Solid-State Circuits*, vol. 48, no. 12, pp. 3229–3242, 2013.
- [10] Y.-H. Chen, T. Krishna, J. Emer, and V. Sze, “Eyeriss: An energy-efficient reconfigurable accelerator for deep convolutional neural networks,” in *2016 IEEE International Solid-State Circuits Conference (ISSCC)*. IEEE, 2016, pp. 262–263.

- [11] R. Dreslinski, M. Wieckowski, D. Blaauw, D. Sylvester, and T. Mudge, "Near-threshold computing: reclaiming Moore's law through energy efficient integrated circuits," *Proceedings of the IEEE*, vol. 98, no. 2, 2010.
- [12] R. Plassche, *Integrated Analog-to-Digital and Digital-to-Analog Converters*. Kluwer Academic Publishers Dordrecht, The Netherlands, 1994.
- [13] S. Park, Y. Palaskas, A. Ravi, R. Bishop, and M. Flynn, "A 3.5 GS/s 5-b flash ADC in 90nm CMOS," in *2006. CICC IEEE Custom Integrated Circuits Conference*, Sept 2006, pp. 489–492.
- [14] S. Sheikhaei, S. Mirabbasi, and A. Ivanov, "A 43 mW single-channel 4GS/s 4-bit flash ADC in 0.18 μ m CMOS," in *2007. CICC IEEE Custom Integrated Circuits Conference*, Sept 2007, pp. 333–336.
- [15] C.-C. Yeh and J. Barry, "Adaptive minimum bit-error rate equalization for binary signaling," *IEEE Transactions on Communications*, vol. 48, no. 7, pp. 1226–1235, Jul 2000.
- [16] E.-H. Chen, W. Leven, N. Warke, A. Joy, S. Hubbins, A. Amerasekera, and C.-K. Yang, "Adaptation of CDR and full scale range of ADC-based SerDes receiver," in *2009 Symposium on VLSI Circuits*, June 2009, pp. 12–13.
- [17] M. Lu, N. Shanbhag, and A. Singer, "BER-optimal analog-to-digital converters for communication links," in *Proceedings of 2010 IEEE International Symposium on Circuits and Systems (ISCAS)*, May 2010, pp. 1029–1032.
- [18] R. Narasimha, M. Lu, N. Shanbhag, and A. Singer, "BER-optimal analog-to-digital converters for communication links," *Signal Processing, IEEE Transactions on*, vol. 60, no. 7, pp. 3683–3691, 2012.
- [19] S. Lloyd, "Least squares quantization in PCM," *IEEE Transactions on Information Theory*, vol. 28, no. 2, pp. 129–137, Mar 1982.
- [20] J. Max, "Quantizing for minimum distortion," *IRE Transactions on Information Theory*, vol. 6, no. 1, pp. 7–12, March 1960.
- [21] Y. Lin, A. Xu, N. Shanbhag, and A. Singer, "Energy-efficient high-speed links using ber-optimal ADCs," in *Electrical Design of Advanced Packaging and Systems Symposium (EDAPS), 2011 IEEE*, 2011, pp. 1–4.
- [22] E.-H. Chen, R. Yousry, and C.-K. Yang, "Power optimized ADC-based serial link receiver," *Solid-State Circuits, IEEE Journal of*, vol. 47, no. 4, pp. 938–951, April 2012.
- [23] J. Kim, E.-H. Chen, J. Ren, B. Leibowitz, P. Satarzadeh, J. Zerbe, and C.-K. Yang, "Equalizer design and performance trade-offs in ADC-based serial links," *Circuits and Systems I: Regular Papers, IEEE Transactions on*, vol. 58, no. 9, pp. 2096–2107, 2011.

- [24] S. Son, H.-S. Kim, M.-J. Park, K. Kim, E.-H. Chen, B. Leibowitz, and J. Kim, “A 2.3-mW, 5-Gb/s low-power decision-feedback equalizer receiver front-end and its two-step, minimum bit-error-rate adaptation algorithm,” *IEEE Journal of Solid-State Circuits*, vol. 48, no. 11, pp. 2693–2704, Nov 2013.
- [25] A. Pullini, F. Conti, D. Rossi, I. Loi, M. Gautschi, and L. Benini, “A heterogeneous multi-core system-on-chip for energy efficient brain inspired vision,” in *Circuits and Systems (ISCAS), 2016 IEEE International Symposium on*, 2016, pp. 2910–2910.
- [26] K. He et al., “Deep residual learning for image recognition,” *CoRR*, vol. abs/1512.03385, 2015.
- [27] J. Cong and B. Xiao, “Minimizing computation in convolutional neural networks,” in *International Conference on Artificial Neural Networks*. Springer, 2014, pp. 281–290.
- [28] Y. Wang, L. Xia, T. Tang, B. Li, S. Yao, M. Cheng, and H. Yang, “Low power convolutional neural networks on a chip,” in *Circuits and Systems (ISCAS), 2016 IEEE International Symposium on*, 2016, pp. 129–132.
- [29] M. Courbariaux and Y. Bengio, “BinaryNet: Training deep neural networks with weights and activations constrained to + 1 or -1,” *arXiv preprint arXiv:1602.02830*, 2016.
- [30] S. Han, J. Pool, J. Tran, and W. Dally, “Learning both weights and connections for efficient neural network,” in *Advances in Neural Information Processing Systems*, 2015, pp. 1135–1143.
- [31] X. Zhang, J. Zou, K. He, and J. Sun, “Accelerating very deep convolutional networks for classification and detection,” *arXiv preprint arXiv:1505.06798*, 2015.
- [32] P. Knag, C. Liu, and Z. Zhang, “A 1.40mm² 141mW 898GOPS sparse neuromorphic processor in 40nm CMOS,” in *2016 IEEE Symposium on VLSI Circuits (VLSI-Circuits)*, June 2016, pp. 1–2.
- [33] S. Leroux, S. Bohez, C. De Booem, E. De Coninck, T. Verbelen, B. Vankeirsbilck, P. Simoens, and B. Dhoedt, “Lazy evaluation of convolutional filters,” *arXiv preprint arXiv:1605.08543*, 2016.
- [34] P. Panda, A. Sengupta, and K. Roy, “Conditional deep learning for energy-efficient and enhanced pattern recognition,” in *2016 Design, Automation & Test in Europe Conference & Exhibition (DATE)*, 2016, pp. 475–480.
- [35] M. Horowitz, “Computing’s energy problem (and what we can do about it),” in *Solid-State Circuits Conference Digest of Technical Papers (ISSCC), 2014 IEEE International*. IEEE, 2014, pp. 10–14.
- [36] T. Chen, Z. Du, N. Sun, J. Wang, C. Wu, Y. Chen, and O. Temam, “DianNao: A small-footprint high-throughput accelerator for ubiquitous machine-learning,” in *ACM Sigplan Notices*, vol. 49, no. 4. ACM, 2014, pp. 269–284.

- [37] Y.-H. Chen, J. Emer, and V. Sze, “Eyeriss: A spatial architecture for energy-efficient dataflow for convolutional neural networks,” in *Computer Architecture (ISCA), 2016 ACM/IEEE 43rd Annual International Symposium on*. IEEE, 2016, pp. 367–379.
- [38] M. Kang, S. K. Gonugondla, M.-S. Keel, and N. R. Shanbhag, “An energy-efficient memory-based high-throughput vlsi architecture for convolutional networks,” in *Acoustics, Speech and Signal Processing (ICASSP), 2015 IEEE International Conference on*. IEEE, 2015, pp. 1037–1041.
- [39] J. Zhang, Z. Wang, and N. Verma, “In-memory computation of a machine-learning classifier in a standard 6T SRAM array,” *IEEE Journal of Solid-State Circuits*, vol. 52, no. 4, pp. 915–924, 2017.
- [40] Z. Du et al., “ShiDianNao: shifting vision processing closer to the sensor,” in *Computer Architecture (ISCA), 2015 ACM/IEEE 42nd Annual International Symposium on*, 2015, pp. 92–104.
- [41] J.-G. Chung and K. K. Parhi, “Frequency spectrum based low-area low-power parallel FIR filter design,” *EURASIP J. Appl. Signal Process.*, vol. 2002, pp. 944–953, 2002.
- [42] R. Mahesh and A. Vinod, “New reconfigurable architectures for implementing FIR filters with low complexity,” *Computer-Aided Design of Integrated Circuits and Systems, IEEE Transactions on*, vol. 29, no. 2, 2010.
- [43] X. Liu, J. Zhou, X. Liao, C. Wang, J. Luo, M. Madihian, and M. Je, “Ultra-low-energy near-threshold biomedical signal processor for versatile wireless health monitoring,” in *2012 IEEE Asian Solid State Circuits Conference (A-SSCC)*, Nov 2012, pp. 381–384.
- [44] Y. Kim, I. Hong, and H. J. Yoo, “A 0.5V 54 uW ultra-low-power recognition processor with 93.5% accuracy geometric vocabulary tree and 47.5% database compression,” in *2015 IEEE International Solid-State Circuits Conference - (ISSCC) Digest of Technical Papers*, Feb 2015, pp. 1–3.
- [45] S. Das, D. Blaauw, D. Bull, K. Flautner, and R. Aitken, “Addressing design margins through error-tolerant circuits,” in *Design Automation Conference (DAC), 46th ACM/IEEE*, 2009, pp. 11–12.
- [46] J. Tschanz, K. Bowman, C. Wilkerson, S.-L. Lu, and T. Karnik, “Resilient circuits: enabling energy-efficient performance and reliability,” in *Computer-Aided Design (ICCAD), IEEE/ACM International Conference on*, 2009.
- [47] R. Bahar, J. Mundy, and J. Chen, “A probabilistic-based design methodology for nanoscale computation,” in *Computer Aided Design (ICCAD), IEEE/ACM International Conference on*, 2003, pp. 480–486.
- [48] N. Vaidya and D. Pradhan, “Fault-tolerant design strategies for high reliability and safety,” *Computers, IEEE Transactions on*, vol. 42, no. 10, pp. 1195–1206, 1993.

- [49] B. Shim, S. Sridhara, and N. Shanbhag, "Reliable low-power digital signal processing via reduced precision redundancy," *Very Large Scale Integration (VLSI) Systems, IEEE Transactions on*, vol. 12, no. 5, pp. 497–510, 2004.
- [50] J. Choi, E. P. Kim, R. A. Rutenbar, and N. R. Shanbhag, "Error resilient MRF message passing architecture for stereo matching," in *Signal Processing Systems (SiPS), IEEE Workshop on*, 2013, pp. 348–353.
- [51] R. A. Abdallah and N. R. Shanbhag, "Error-resilient systems via statistical signal processing," in *Signal Processing Systems (SiPS), IEEE Workshop on*, 2013.
- [52] H.-M. Bae, J. Ashbrook, J. Park, N. Shanbhag, A. Singer, and S. Chopra, "An MLSE receiver for electronic dispersion compensation of OC-192 fiber links," *IEEE Journal of Solid-State Circuits*, vol. 41, no. 11, pp. 2541–2554, Nov 2006.
- [53] J. Cao, B. Zhang, U. Singh, D. Cui, A. Vasani, A. Garg, W. Zhang, N. Kocaman, D. Pi, B. Raghavan, H. Pan, I. Fujimori, and A. Momtaz, "A 500mW digitally calibrated AFE in 65nm CMOS for 10Gb/s serial links over backplane and multimode fiber," in *2009 IEEE International Solid-State Circuits Conference Digest of Technical Papers (ISSCC)*, Feb 2009, pp. 370–371,371a.
- [54] B. Abiri, A. Sheikholeslami, H. Tamura, and M. Kibune, "A 5Gb/s adaptive DFE for 2x blind ADC-based CDR in 65nm CMOS," in *2011 IEEE International Solid-State Circuits Conference Digest of Technical Papers (ISSCC)*, Feb 2011, pp. 436–438.
- [55] O. Agazzi, D. Crivelli, M. Hueda, H. Carrer, G. Luna, A. Nazemi, C. Grace, B. Kobeissy, C. Abidin, M. Kazemi, M. Kargar, C. Marquez, S. Ramprasad, F. Bollo, V. Posse, S. Wang, G. Asmanis, G. Eaton, N. Swenson, T. Lindsay, and P. Voois, "A 90nm CMOS DSP MLSD transceiver with integrated AFE for electronic dispersion compensation of multi-mode optical fibers at 10Gb/s," in *2008 IEEE International Solid-State Circuits Conference Digest of Technical Papers (ISSCC)*, Feb 2008, pp. 232–609.
- [56] M. Harwood, N. Warke, R. Simpson, T. Leslie, A. Amerasekera, S. Batty, D. Colman, E. Carr, V. Gopinathan, S. Hubbins, P. Hunt, A. Joy, P. Khandelwal, B. Killips, T. Krause, S. Lytollis, A. Pickering, M. Saxton, D. Sebastio, G. Swanson, A. Szczepanek, T. Ward, J. Williams, R. Williams, and T. Willwerth, "A 12.5Gb/s SerDes in 65nm CMOS using a baud-rate ADC with digital receiver equalization and clock recovery," in *Solid-State Circuits Conference, 2007. ISSCC 2007. Digest of Technical Papers. IEEE International*, 2007, pp. 436–591.
- [57] [Online]. Available: http://www.ieee802.org/3/ap/public/channel_model/index.html#Backplane%20Models
- [58] C.-M. Hsu, M. Straayer, and M. Perrott, "A low-noise wide-BW 3.6-GHz digital fractional-N frequency synthesizer with a noise-shaping time-to-digital converter and quantization noise cancellation," *IEEE Journal of Solid-State Circuits*, vol. 43, no. 12, pp. 2776–2786, Dec 2008.

- [59] “Stratix II GX edition transceiver signal integrity development FPGA board,” Altera Corporation, 2003. [Online]. Available: http://www.altera.com/products/devkits/altera/kit-signa_integrity_s2gx.html
- [60] G. Al-Rawi, “A new offset measurement and cancellation technique for dynamic latches,” in *Circuits and Systems, 2002. ISCAS 2002. IEEE International Symposium on*, vol. 5, 2002, pp. V-149–V-152 vol.5.
- [61] J. He, S. Zhan, D. Chen, and R. Geiger, “Analyses of static and dynamic random offset voltages in dynamic comparators,” *Circuits and Systems I: Regular Papers, IEEE Transactions on*, vol. 56, no. 5, pp. 911–919, 2009.
- [62] M. Ramezani, M. Abdalla, A. Shoval, M. van Ierssel, A. Rezayee, A. McLaren, C. Hold-enried, J. Pham, E. So, D. Cassan, and S. Sadr, “An 8.4mW/Gb/s 4-lane 48Gb/s multi-standard-compliant transceiver in 40nm digital CMOS technology,” in *Solid-State Circuits Conference Digest of Technical Papers (ISSCC), 2011 IEEE International*, 2011, pp. 352–354.
- [63] F. Zhong, S. Quan, W. Liu, P. Aziz, T. Jing, J. Dong, C. Desai, H. Gao, M. Garcia, G. Hom, T. Huynh, H. Kimura, R. Kothari, L. Li, C. Liu, S. Lowrie, K. Ling, A. Malipatil, R. Narayan, T. Prokop, C. Palusa, A. Rajashekara, A. Sinha, C. Zhong, and E. Zhang, “A 1.0625-14.025 Gb/s multi-media transceiver with full-rate source-series-terminated transmit driver and floating-tap decision-feedback equalizer in 40 nm CMOS,” *Solid-State Circuits, IEEE Journal of*, vol. 46, no. 12, pp. 3126–3139, 2011.
- [64] J. Poulton, R. Palmer, A. Fuller, T. Greer, J. Eyles, W. Dally, and M. Horowitz, “A 14-mW 6.25-Gb/s Transceiver in 90-nm CMOS,” *Solid-State Circuits, IEEE Journal of*, vol. 42, no. 12, pp. 2745–2757, 2007.
- [65] S. Saxena, R. Nandwana, and P. Hanumolu, “A 5 Gb/s 3.2 mW/Gb/s 28 dB loss-compensating pulse-width modulated voltage-mode transmitter,” in *Custom Integrated Circuits Conference (CICC), 2013 IEEE*, 2013, pp. 1–4.
- [66] Y. Lecun et al., “Gradient-based learning applied to document recognition,” *Proceedings of the IEEE*, vol. 86, no. 11, pp. 2278–2324, 1998.
- [67] X. Glorot, A. Bordes, and Y. Bengio, “Deep sparse rectifier neural networks,” in *Proceedings of the Fourteenth International Conference on Artificial Intelligence and Statistics*, 2011, pp. 315–323.
- [68] K. Jarrett, K. Kavukcuoglu, M. Ranzato, and Y. LeCun, “What is the best multi-stage architecture for object recognition?” in *Computer Vision, IEEE 12th International Conference on*, 2009, pp. 2146–2153.
- [69] B. A. Olshausen and D. J. Field, “Sparse coding with an overcomplete basis set: A strategy employed by V1?” *Vision research*, vol. 37, no. 23, pp. 3311–3325, 1997.

- [70] C. Poultney, S. Chopra, Y. L. Cun et al., “Efficient learning of sparse representations with an energy-based model,” in *Advances in neural information processing systems*, 2007, pp. 1137–1144.
- [71] H. Lee, C. Ekanadham, and A. Y. Ng, “Sparse deep belief net model for visual area V2,” in *Advances in neural information processing systems*, 2008, pp. 873–880.
- [72] D. Arpit, Y. Zhou, H. Ngo, and V. Govindaraju, “Why regularized auto-encoders learn sparse representation?” in *International Conference on Machine Learning*, 2016, pp. 136–144.
- [73] X. Glorot, A. Bordes, and Y. Bengio, “Deep sparse rectifier neural networks.” in *Aistats*, vol. 15, no. 106, 2011, p. 275.
- [74] Y. Bengio, G. Mesnil, Y. Dauphin, and S. Rifai, “Better mixing via deep representations,” in *Proceedings of the 30th International Conference on Machine Learning (ICML-13)*, 2013, pp. 552–560.
- [75] C. Sakr, A. Patil, S. Zhang, Y. Kim, and N. Shanbhag, “Understanding the energy and precision requirements for online learning,” *arXiv preprint arXiv:1607.00669*, 2016.
- [76] R. B. Palm, “Prediction as a candidate for learning deep hierarchical models of data,” M.S. thesis, 2012.
- [77] Y. Lin, S. Zhang, and N. R. Shanbhag, “Variation-tolerant architectures for convolutional neural networks in the near threshold voltage regime,” in *2016 IEEE International Workshop on Signal Processing Systems (SiPS)*, Oct 2016, pp. 17–22.
- [78] M. Lin, Q. Chen, and S. Yan, “Network In Network,” *ArXiv e-prints*, Dec. 2013.
- [79] A. Krizhevsky and G. Hinton, “Learning multiple layers of features from tiny images,” 2009.
- [80] J. Yoo, L. Yan, D. El-Damak, M. Altaf, A. Shoeb, and A. Chandrakasan, “An 8-Channel Scalable EEG Acquisition SoC With Patient-Specific Seizure Classification and Recording Processor,” *Solid-State Circuits, IEEE Journal of*, vol. 48, no. 1, pp. 214–228, Jan 2013.
- [81] V. Perlibakas, “Distance measures for PCA-based face recognition,” *Pattern Recogn. Lett.*, vol. 25, no. 6, pp. 711–724, Apr. 2004.
- [82] H. He and J. Starzyk, “A self-organizing learning array system for power quality classification based on wavelet transform,” *Power Delivery, IEEE Transactions on*, vol. 21, no. 1, pp. 286–295, 2006.
- [83] R. Teodorescu, J. Nakano, A. Tiwari, and J. Torrellas, “Mitigating parameter variation with dynamic fine-grain body biasing,” in *Microarchitecture (MICRO), 40th Annual IEEE/ACM International Symposium on*, 2007.

- [84] X. Liang, G.-Y. Wei, and D. Brooks, “Revival: a variation-tolerant architecture using voltage interpolation and variable latency,” *Micro, IEEE*, vol. 29, 2009.
- [85] G. Strang, *Introduction to Linear Algebra*, 3rd ed. Wesley-Cambridge Press, 2003.
- [86] S. Zhang and N. Shanbhag, “Probabilistic error models for machine learning kernels implemented on stochastic nanoscale fabrics,” in *Design, Automation Test in Europe (DATE)*, 2016.
- [87] D. P. Bertsekas and J. N. Tsitsiklis, *Introduction to Probability. 2nd ed.* Athena Scientific, 2008.
- [88] J. M. Rabaey, A. Chandrakasan, and B. Nikolic, *Digital Integrated Circuits: a Design Perspective.* Upper Saddle River (N.J.): Prentice-Hall, Inc., 2003.
- [89] K. Roy, M. Sharad, D. Fan, and K. Yogendra, “Beyond charge-based computation: Boolean and non-boolean computing with spin torque devices,” in *Low Power Electronics and Design (ISLPED), 2013 IEEE International Symposium on.* IEEE, 2013, pp. 139–142.