A Variation-Tolerant In-Memory Machine Learning Classifier via On-Chip Training

Sujan K. Gonugondla, *Student Member, IEEE,* Mingu Kang, *Member, IEEE,*
and Naresh R. Shanbhag, *Fellow, IEEE*

***Abstract*—This paper presents a robust deep in-memory machine learning classifier with a stochastic gradient descent (SGD)-based on-chip trainer using a standard 16 kB 6T SRAM array. The deep in-memory architecture (DIMA) enhances both energy efficiency and throughput over conventional digital architectures by reading multiple bits per bit-line (BL) per read cycle, and by employing mixed-signal processing in the periphery of the bit-cell array (BCA). Though these techniques improve the energy efficiency and latency, DIMA's analog nature makes it sensitive to process, voltage, and temperature (PVT) variations especially under reduced BL swings. On-chip training enables DIMA to adapt to chip-specific variations in PVT as well as data statistics thereby further enhancing its energy efficiency. The 65 nm CMOS prototype IC demonstrates this improvement by realizing a on-chip trainable support vector machine (SVM). By learning chip-specific weights, on-chip training enables robust operation under reduced BL swing leading to a 2.4× reduction in energy over an off-chip trained DIMA. The prototype IC in 65 nm CMOS consumes 42 pJ/decision at 32 M decisions/s, corresponding to 3.12 TOPS/W (1 OP = one 8-b×8-b MAC) during inference thereby achieving a reduction of 21× in energy, and 100× in energy-delay product (EDP) as compared to a conventional digital architecture. The energy overhead of training is <26% per decision for SGD batch sizes of 128 and higher.**

***Index Terms*—in-memory, on-chip learning, machine learning, always-on, inference, stochastic gradient descent, mixed-signal, process variations.**

I. Introduction

Energy and delay costs of current day machine learning algorithms inhibit their deployment for real-time inference on sensor-rich platforms such as wearables, UAVs, personal biomedical devices, Internet of Things (IoT), and many others. These applications involve devices that acquire and process data to derive actions and interpretations in order to automate/monitor many tasks without human intervention. These systems need to realize computationally intensive machine learning (ML) algorithms under stringent constraints on energy, latency, and form-factor. It is well-known that the energy and latency cost of realizing ML algorithms is dominated by memory acceses [1]. Recently, a number of energy efficient digital architectures and IC implementations [2]–[7] for ML have been proposed that strive to reduce the number of memory accesses via techniques such as data reuse, minimizing computations, and efficient data-flow.

Sujan K. Gonugondla and Naresh R. Shanbhag are with the Department of Electrical and Computer Engineering, at the University of Illinois at Urbana-Champaign.

Mingu Kang is with the IBM Thomas J. Watson Research Center, Yorktown Heights, NY, USA.

Recently, in-memory architectures [8]–[16] were proposed to directly address memory access costs in ML systems by embedding analog computations in close proximity to the memory bit-cell array (BCA). Such proximity of computation to the BCA makes in-memory architectures inherently and massively parallel, and well-matched to the data-flow of ML algorithms. However, its intrinsic analog nature makes in-memory architectures susceptible to process, voltage, and temperature (PVT) variations. The use of low bit-line (BL) swings and stringent bit-cell pitch matching constraints results in lowered signal-to-noise (SNR) of in-memory computations. These robustness issues were addressed by a combination of: 1) restricting one or both operands to be binary (1-b) [14], [15], 2) using larger (8T or 10T) bit-cells [13], and 3) partitioning the array into sub-banks [13] or using separate right and left word-lines [15]. Specifically, the restriction on 1-b operands makes it difficult for such in-memory architectures to execute multi-bit operations without sacrificing their energy-latency benefits. Furthermore, the use of binary nets [17], the default network realized by these in-memory architectures, leads to increased memory requirements for a fixed accuracy as shown theoretically in [18] and experimentally in [19].

In contrast, the deep in-memory architecture (DIMA) [8]–[12], [20], [21] embeds *multi-bit mixed-signal computations* in the periphery of a conventional 6T BCA to preserve its storage density, and conventional SRAM read-write functionality. DIMA reads access multiple rows of a standard SRAM BCA per precharge to generate a BL discharge ΔV_{BL} proportional to a linear combination of column bits. DIMA then performs scalar operations on the BL outputs via column pitch-matched mixed-signal circuitry in the periphery of the SRAM, followed by a step that aggregates the BL outputs across all the columns of the BCA. Thus, DIMA computations are intrinsically multi-bit, e.g., 8-b in [11], [21] and 5-b in [10], and strives to delay the conversion from analog to digital to allow for SNR budgeting across the different stages of analog processing. It can be shown that this *delayed decision property* of DIMA combined with its cross BL aggregation step effectively compensates for the low SNR problem inherent to in-memory architectures thereby leading to accurate inference [11]. Algorithmic approaches such as boosting (AdaBoost) [10] and ensemble methods (random forest (RF)) [21] have been additionally leveraged to enhance DIMA's robustness to PVT variations.

While previous DIMA realizations have achieved significant gains, e.g., up to $50\times$-to-$175\times$ [10], [11] in energy-delay product (EDP) over conventional (SRAM+fixed-function digital

processor) architectures, these gains are limited by the die-specific nature of PVT variations and the potential for data statistics to vary over time in sensory systems. This paper studies the use of an on-chip SGD-based trainer to adapt and track variations in PVT and data statistics and thereby realize a robust deep in-memory support vector machine (SVM) classifier IC using a standard 6T SRAM array in 65 nm CMOS. SGD-based algorithms [22] are commonly employed to train many ML systems including deep neural networks (DNNs) raising the possibility of other applications of this work. Measurement results show that on-chip learned weights enable accurate inference in presence of scaled BL voltage swing thereby leading to a $2.4\times$ greater energy savings over an off-chip trained DIMA implementation. Compared to a conventional fixed-function digital architecture with identical SRAM array, the prototype IC achieves up to $21\times$ lower energy and $4.7\times$ smaller delay leading to a $100\times$ reduction in the EDP. Preliminary measurement results were reported in [23]. This paper presents the circuit implementation details in realizing signed dot product in DIMA, characterizes the impact of PVT variations on in-memory computations, provides a system rationale for the effectiveness of the SGD-based trainer, and presents additional measured results in terms of learning and receiver operating curve (ROC) that demonstrate the improvement in the quality of inference in presence of variations in PVT and data statistics.

This paper is organized as follow: Section II provides background on DIMA and SGD. Section III motivates the use of SGD to compensate for the impact of process variations on DIMA. The circuit and the architecture implementation details of the prototype IC are described in Section IV. Measurement results showing the learning curves, the ROCs, and the energy consumption are presented in Section V, and Section VI concludes the paper.

II. Background

This section provides background on DIMA, the SVM algorithm, and its training via the SGD algorithm.

A. Deep In-Memory Architecture (DIMA)

DIMA [8], [11] has four sequentially executed processing stages: 1) *functional read* (FR): reads multiple (B) rows of the BCA per access via pulse width and amplitude modulated (PWAM) word-line (WL) access pulses to generate a BL discharge ΔV_{BL} which is proportional to a linearly weighted sum of the bits stored in a column-major format; 2) *BL processing* (BLP): the BL discharge ΔV_{BL} voltages are processed in a massively parallel (single-instruction multiple data (SIMD)) fashion via column pitch-matched mixed-signal circuits that execute scalar arithmetic operations such as addition, subtraction, multiplication, and comparison; 3) *cross BL processing* (CBLP): aggregates the BLP outputs via charge-sharing to obtain a scalar output; and 4) *residual digital logic* (RDL): converts the CBLP's analog output into a digital word and then executes any residual functions digitally to generate the final inference output.

In contrast to a conventional digital architecture, DIMA reads B rows of the BCA per access and avoids the need for a $L:1$ (typically $4:1$ to $16:1$) column multiplexer prior to sense amplification, where L is the number of BLs sharing a single sense amplifier. Therefore, DIMA is able to process the same number of bits with much fewer ($1/LB$ which is typically $1/16$ to $1/64$) read cycles when compared to a conventional digital architecture directly leading to large energy and latency gains. Silicon prototypes of DIMA [10], [11], [21] have shown up to $9.7\times$ reduction in energy and $5.3\times$ improvement [11] in throughput over fixed-function digital implementations. This paper seeks to enhance these gains by compensating for variations in PVT and data statistics using on-chip training.

B. Stochastic Gradient Descent (SGD)

Stochastic gradient descent (SGD) has emerged as one of the most effective training algorithms for machine learning [22]. The SGD algorithm has been used to train a variety of ML algorithms including the support vector machine (SVM) [24]. The SVM is a maximum margin binary classifier which predicts the class label $y_k \in \{\pm 1\}$ as follows:

$$\boldsymbol{W}^T\boldsymbol{X}_k + b \underset{\hat{y}_k=-1}{\overset{\hat{y}_k=+1}{\gtrless}} 0 \tag{1}$$

where $\hat{y}_k$ is the predicted class label, the D-dimensional vectors $\boldsymbol{X}_k = [X_{0,k}, X_{2,k}, ..., X_{D-1,k}]^T$, weights $\boldsymbol{W} = [W_0, W_2, ..., W_{D-1}]^T$, and a scalar bias b are referred to as the parameters of the SVM algorithm. The SVM cost function per sample $Q(\boldsymbol{W}, \boldsymbol{X}_k)$ is given by [22],

$$Q(\boldsymbol{W}, \boldsymbol{X}_k) = \frac{\lambda}{2}(||\boldsymbol{W}||^2 + b^2) + [1 - y_k(\boldsymbol{W}^T\boldsymbol{X}_k + b)]_+ \tag{2}$$

where λ is a regularization factor, y_k is the true label for the data sample $\boldsymbol{X}_k$, and $[x]_+ = \max\{0, x\}$. The SGD algorithm minimizes the cost function $Q(\boldsymbol{W}, \boldsymbol{X}_k)$ averaged over the training set iteratively by updating the weight vector as follows:

$$\boldsymbol{W}_{m+1} = (1 - \lambda\gamma)\boldsymbol{W}_m + \gamma\frac{1}{N}\sum_{n=0}^{N-1}\begin{cases}0 & \text{if } y_n^{(m)}(\boldsymbol{W}_m^T\boldsymbol{X}_n^{(m)} + b_m) > 1\\ y_n^{(m)}\boldsymbol{X}_n^{(m)} & \text{otherwise}\end{cases} \tag{3}$$

$$b_{m+1} = (1 - \lambda\gamma)b_m + \gamma\frac{1}{N}\sum_{n=0}^{N-1}\begin{cases}0 & \text{if } y_n^{(m)}(\boldsymbol{W}_m^T\boldsymbol{X}_n^{(m)} + b_m) > 1\\ y_n^{(m)} & \text{otherwise}\end{cases} \tag{4}$$

where N is the batch size, m is the batch index, $\boldsymbol{W}_m$ is the weight vector in the m^{th} batch, $\boldsymbol{X}_n^{(m)}$ is the n^{th} sample of the m^{th} batch, and γ is the learning rate.

The primary inference computation in the SVM algorithm is the dot product $\boldsymbol{W}^T\boldsymbol{X}$. For SVM implementation on DIMA, the weights $\boldsymbol{W}$ are stored in the BCA and accessed via functional read, while the BLPs execute element-wise multiplication of $\boldsymbol{W}$ and $\boldsymbol{X}$, and the CBLP aggregates the BLP outputs to obtain the final dot product.

III. A Systems Rationale for On-Chip Training

This section provides a systems rationale for using the SGD algorithm on-chip to compensate for PVT variations in DIMA. These variations are caused by [11]: 1) spatial transistor threshold voltage (V_t) variations caused by random dopant fluctuations [29]; 2) BL voltage dependence of the discharge path (access and pull-down transistors in the bit-cell) current; and 3) the finite transition (rise and fall) times of the PWM WL pulses. As a result, the functional read step of DIMA, which accesses weights W_j stored in the j-th column, generates a BL discharge of $\Delta V_{BL,j} = \alpha_j W_j$ on the j^{th} BL. Therefore, even though the BCA stores $\boldsymbol{W} = [W_0, W_1, ..., W_{D-1}]$, DIMA implements the dot product in (1) with weights $\boldsymbol{W'} = [\beta_0 W_0, \beta_1 W_1, ..., \beta_{D-1} W_{D-1}]$, where $\beta_j = \frac{\alpha_j}{\alpha}$ (with $\alpha_j = \alpha(1 + \frac{\Delta\alpha_j}{\alpha})$) is the per dimension proportionality factor. Measured results show that this simple model captures the effects of spatial PVT variations to a first order.

Thus, the SGD algorithm minimizes the modified cost function $Q'(\boldsymbol{W}, \boldsymbol{X}_k)$ given by,

$$Q'(\boldsymbol{W}, \boldsymbol{X}_k) = \frac{\lambda}{2}(||\boldsymbol{W'}||^2 + b^2) + [1 - y(\boldsymbol{W'}^T \boldsymbol{X}_k + b)]_+$$
$$= \frac{\lambda}{2}(\sum_{j=0}^{D-1} \beta_j^2 W_j^2 + b^2) + [1 - y(\sum_{j=0}^{D-1} \beta_j W_j X_{j,k} + b)]_+ \quad (5)$$

The modified cost function $Q'(\boldsymbol{W}, \boldsymbol{X})$ can be shown to be convex in $\boldsymbol{W}$, which implies that the SGD algorithm will converge to the global minimum in the presence of PVT variations. However, as the proportionality factor β_j is unknown in practice and is die-specific, we employ a per dimension learning rate of $\gamma_j = \gamma/\beta_j$ to obtain following SGD update,

$$\boldsymbol{W}_{m+1} = (1 - \lambda\gamma)\boldsymbol{W}_m$$
$$+\gamma \frac{1}{N} \sum_{n=0}^{N-1} \begin{cases} 0 & \text{if } y_n^{(m)}(\boldsymbol{W'}_m^T \boldsymbol{X}_n^{(m)} + b_m) > 1 \\ y_n^{(m)} \boldsymbol{X}_n^{(m)} & \text{otherwise} \end{cases} \quad (6)$$

which is identical to (3) except that it relies upon computation of the dot product in the presence of spatial variations. On-chip training is ineffective in conventional digital architectures in compensating for PVT variations under reduced ΔV_{BL}. This is because PVT variations with reduced ΔV_{BL} leads to increased sense amplifier bit errors including the most significant bit (MSB) errors. These large magnitude errors in turn leads to a large increase in the mean and variance of the misclassification rate over different instances of the architecture. Figure 1 shows that the increase in the misclassification rate cannot be compensated for via on-chip learning. DIMA avoids such errors by reading a weighted function of the bits of W instead of the bits directly.

IV. Implementation

The architecture of the prototype IC in Fig. 2 has four major blocks: (a) the in-memory CORE (IM-CORE) to execute the inference computations, (b) the standard read/write SRAM interface, (c) the digital trainer, and (d) a digital controller (CTRL) to sequence the operations. The prototype IC has three modes of operation: 1) a standard SRAM mode, 2) an in-memory inference mode, and 3) a training mode.

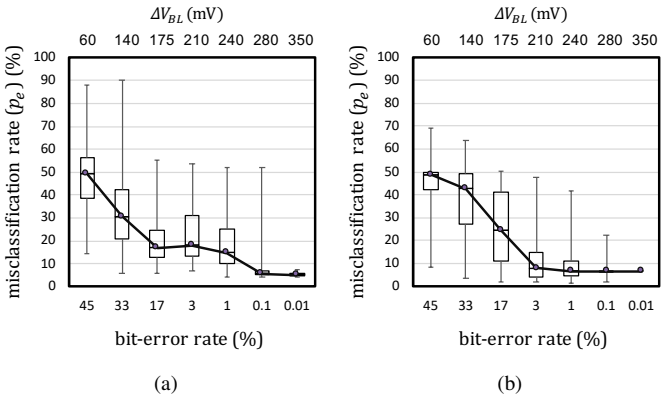


Fig. 1. Misclassification rate of a digital architecture when subject to bit errors during readout: (a) before retraining, and (b) after retraining. The bit error rate (BER) was obtained via measurements from the prototype IC under reduced BL swing (ΔV_{BL}) in the SRAM mode. The misclassification rate was obtained via simulations of 1000 independent instances of an SRAM bank, with each instance processing 858 data samples, in presence of randomly assigned bit-flip locations at a rate based on the BER.

A. The IM-CORE Block

The in-memory inference mode is implemented by the IM-CORE block. The IM-CORE comprises a conventional 512×256 6T SRAM BCA, and in-memory computation circuitry, which includes: (a) pulse width modulated WL drivers to realize functional read, (b) BLPs implementing signed multiplication, (c) CBLP implementing summation, (d) an ADC bank, and (e) a comparator bank to generate final decisions. The 16-b SVM weights $\boldsymbol{W}$ are stored in the BCA. However, only 8-b MSBs of the weights $\boldsymbol{W}$ are used during inference (see Fig. 3). Functional read simultaneously generates ΔV_{BL} discharge voltages on the BLs in one read cycle. The 8-b input samples $\boldsymbol{X}$ are streamed into the input buffer, and are transferred to the BLPs via a 256-b bus for element-wise multiplication of $\boldsymbol{W}$ and $\boldsymbol{X}$, which are then summed in the CBLP to obtain the dot product in (1).

1) Signed functional read: Functional read performs digital-to-analog conversion of the weights stored in the SRAM array such that the discharge on the BL represents the analog values of the weights being read.

The 8-b SVM weights $W \equiv \{w_7, .., w_1, w_0\}$ ($w_i \in \{0, 1\}$ is the i^{th} binary bit of W) are stored in a column-major format split across two adjacent columns (MSB and LSB columns) with 4-b per column in the BCA as shown in Fig. 3. The application of WL access pulses with binary weighted pulse widths (PWM) followed by LSB-MSB merge results in discharge voltages on BL and BLB as follows:

$$\Delta V_{BL} = \frac{V_{pre} T_0}{R_{BL} C_{BL}} \left[\frac{1}{16} \sum_{i=0}^{3} 2^i \overline{w}_i + \sum_{i=4}^{7} 2^i \overline{w}_i \right] \quad (7)$$

$$\Delta V_{BLB} = \frac{V_{pre} T_0}{R_{BL} C_{BL}} \left[\frac{1}{16} \sum_{i=0}^{3} 2^i w_i + \sum_{i=4}^{7} 2^i w_i \right] \quad (8)$$

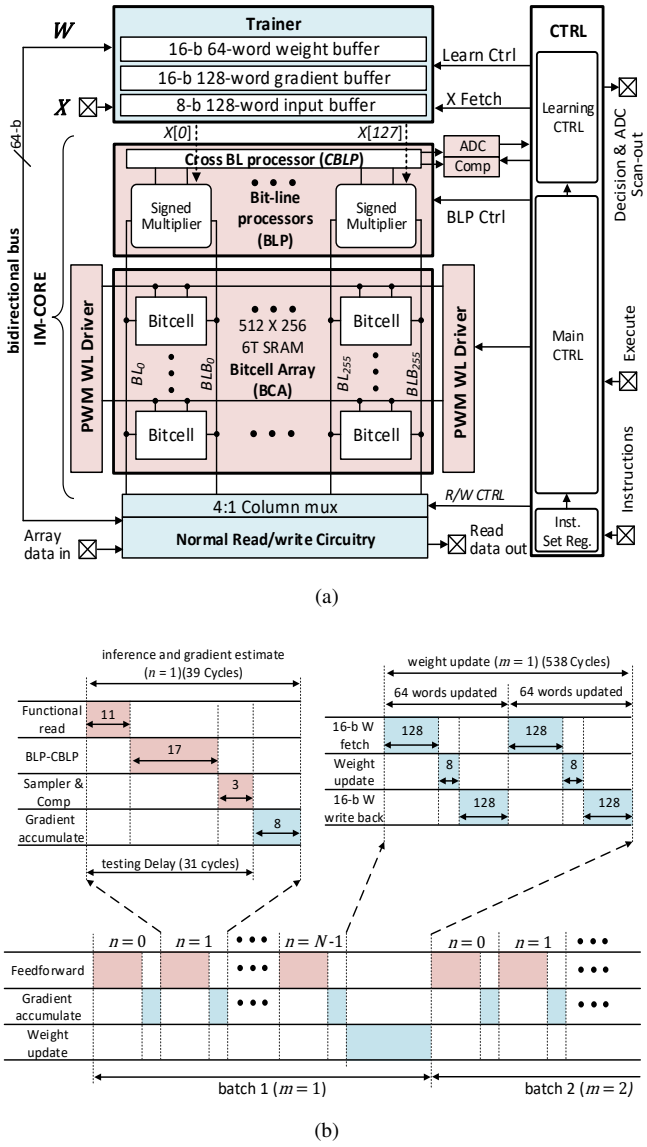


Fig. 2. Proposed system: (a) chip architecture showing IM-CORE, trainer and CTRL, and (b) the timing diagram.

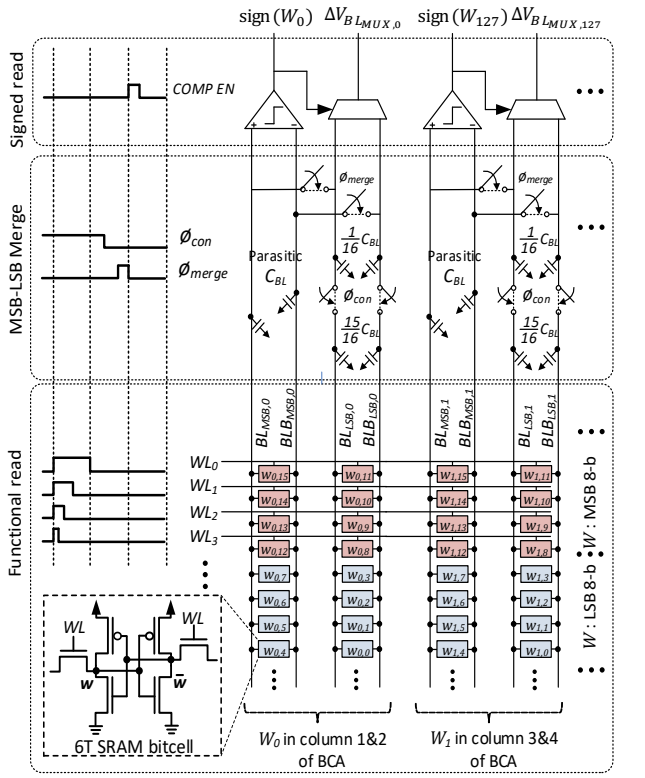


Fig. 3. Implementation of signed functional read. The 8 MSBs of W are stored across two adjacent bit-cell columns (red) and employed for inference, while 8 LSBs of W (blue) are updated during training.

where V_{pre} is the BL precharge voltage, T_0 is the least significant bit (LSB) pulse width, C_{BL} is the BL capacitance, and R_{BL} is the BL discharge path resistance of a bit-cell. We employ two's complement representation for W to account for its sign. The magnitude $|W|$ is obtained by choosing ΔV_{BLB} (ΔV_{BL}) if W is negative (positive) since the two discharges are complementary. The sign of W is obtained by comparing ΔV_{BLB} and ΔV_{BL} as shown in Fig. 3. Thus, the discharge on the output node of functional read $\Delta V_{BL_{MUX}} \propto |W|$. The detected sign and the magnitude are then passed on to the signed multipliers in the BLP.

Equation (8) assumes that R_{BL} is invariant with respect to the time-varying ΔV_{BL} and spatially across the BCA. In practice, the amplitude of word-line voltages V_{WL} is kept sufficiently low (0.45 V to 0.65 V) to alleviate the impact of ΔV_{BL} on R_{BL}. Doing so results in the integral non-linearity (INL) of ΔV_{BLB} to be less than 0.65 LSB (see Fig. 4(a)).

Spatial variations in the access transistor threshold voltages leads to variations in R_{BL}, and hence to variations in ΔV_{BL} across the BCA columns. Figure 4(b) shows that the variance of ΔV_{BLB} at $\Delta V_{BL,\max} = 320\,\mathrm{mV}$ is 60% higher than at $\Delta V_{BL,\max} = 440\,\mathrm{mV}$. Figure 4(c) shows that the misclassification rate $p_e = 4\%$ at $\Delta V_{BL,\max} = 560\,\mathrm{mV}$ is within 1% of the floating point accuracy. However, p_e increases to 16% when $\Delta V_{BL,\max}$ is reduced to 320 mV illustrating the impact of spatial variations in ΔV_{BL} on the accuracy of inference. The impact of spatial variations on ΔV_{BL} can be reduced by increasing the maximum BL discharge voltage $\Delta V_{BL,\max}$ (ΔV_{BLB} when the 4-b $W = 15$) by tuning the WL voltage V_{WL} and T_0.

This sensitivity to chip-specific spatial process variations indicates the need to explore on-chip compensation methods. Additionally, increasing $\Delta V_{BL,\max}$ to reduce the impact of spatial variations incurs an energy cost thereby leading to an interesting trade-off between computational accuracy and energy in DIMA that can be exploited at the architectural level.

2) BLP: Signed multiplication: The BLP block needs to implement a signed 8-b×8-b element-wise product ($W_i X_i$) between $\boldsymbol{W}$ and $\boldsymbol{X}$. The BLP block realizes this using a MSB and LSB multiplier pair with each multiplying the functional read output $\Delta V_{BL_{MUX}} \propto$ 8-b $|W|$ with four MSBs (X_{MSB}) and four LSBs (X_{LSB}) of an 8-b input X. The proposed charge-domain signed multiplier in Fig. 5 is based on the unsigned version in [11].

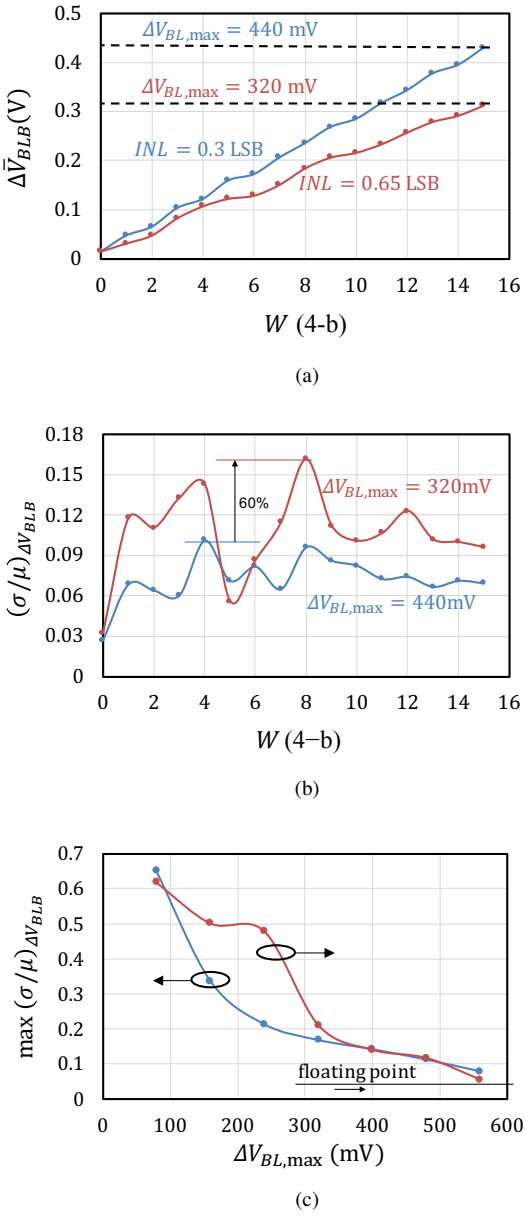


Fig. 4. Impact of spatial variations on functional read obtained by measurements across 30 randomly chosen 4-row groups: (a) average $\Delta\bar{V}_{BLB}$, (b) normalized variance $(\sigma/\mu)_{\Delta V_{BLB}}$, and (c) the impact of spatial variations on $(\sigma/\mu)_{\Delta V_{BLB}}$ and misclassification rate p_e with respect to $\Delta V_{BL,\max}$.

The LSB multiplier receives digital inputs $X_{\text{LSB}} \equiv \{x_3, .., x_0\}$ ($x_i \in \{0,1\}$ is i-th bit of X), the analog $\Delta V_{BL_{MUX}}$, and a digital sign(W) from functional read. Multiplication occurs in four phases P1-P4 with P3 and P4 overlapping in time (see Fig. 5(a)). The multiplier (Fig. 5(b)) employs six equally sized 25 fF capacitors (C_0, C_1, C_2, C_x, C_p, C_n) which are initialized to V_{pre} and either C_p or C_n is chosen as the output capacitor based on sign(W) (P1). Next, the five capacitors nodes are charge shared with node $\Delta V_{BL_{MUX}}$ using the $\phi_{3,i}$ and ϕ_{dump} switches (P2). This is followed by conditionally charging capacitors C_i to V_{pre} using the $\phi_{2,i}$ switches (P3). Capacitors C_x, C_0, C_1, and C_2, are sequentially charge shared using the $\phi_{3,i}$ switches (P4),

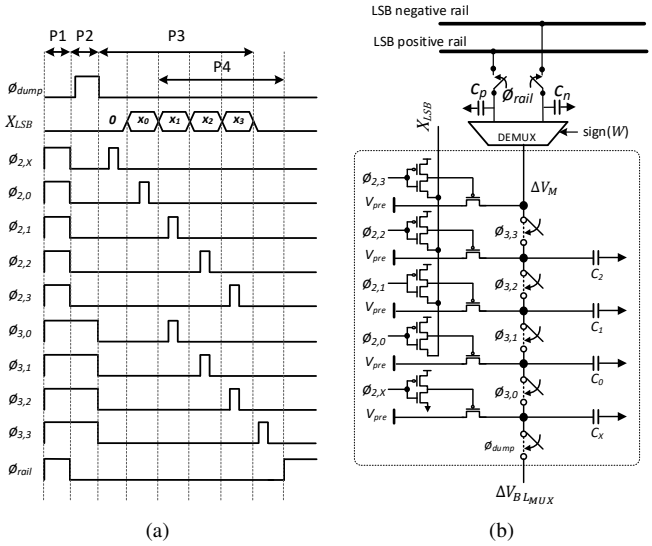


Fig. 5. The signed 4-b×8-b LSB multiplier: (a) timing diagram, and (b) circuit schematic.

resulting in

$$\Delta V_M = \frac{\Delta V_{BL_{MUX}}}{16} \sum_{i=0}^{3} 2^i x_i \propto |X_{\text{LSB}}||W| \tag{9}$$

where ΔV_M is the discharge on the output capacitor C_p or C_n connected to the positive and negative output rails (see Fig. 5(b)). The MSB multiplier operates identically.

3) Aggregation and final decision: The CBLP aggregates the outputs of the MSB and LSB BLP multipliers from across the BCA in order to generate the dot product $\boldsymbol{W}^T\boldsymbol{X}$. This aggregation is accomplished (Fig. 6(a)) by merging the positive (negative) rails from both the MSB and LSB multipliers across the BCA with a 16:1 charge sharing ratio (see Fig. 6(a)). This merging step results in the voltages $\Delta V_{s,p}$ and $\Delta V_{s,n}$ representing the magnitude of the sum of the positive and negative terms, respectively, in the final dot product. Therefore, the dot product is computed as:

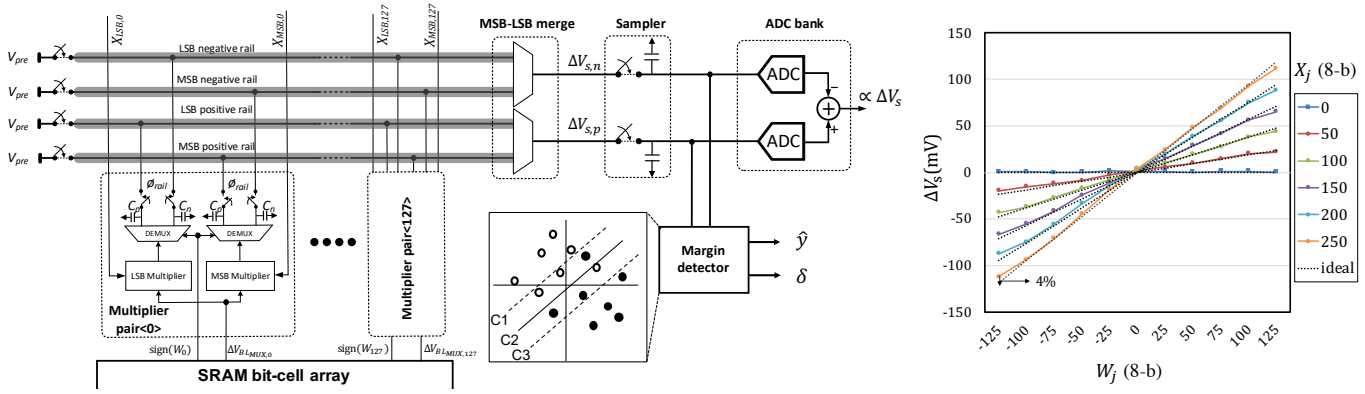
$$\Delta V_s = \Delta V_{s,p} - \Delta V_{s,n} \propto \boldsymbol{W}^T\boldsymbol{X} \tag{10}$$

The voltages $V_{s,p}$ and $V_{s,n}$ are sampled and processed by a bank of three comparators (C1-C3) where C2 generates the predicted label ($\hat{y}$) while C1 and C3 realize the SVM *margin detector* (see Section IV-B). The sampled rail voltages $V_{s,p}$ and $V_{s,n}$ are also converted to digital via a 6-b ADC pair for testing purposes. Measured (Fig. 6(b)) values of ΔV_s are found to lie within $< 4\%$ of the dynamic range when $\Delta V_{BL,\max} = 320\,\text{mV}$.

B. Trainer

As the energy and latency cost of SRAM writes are very high, the trainer which is implemented digitally, writes the updated weights once per batch into the BCA. The trainer (see Fig. 2) implements a reformulated version of the batch update (6) shown below:

$$\boldsymbol{W}_{m+1} = (1-\gamma\lambda)\boldsymbol{W}_m + \frac{\gamma}{N}\Delta_{W,N}^{(m)} \tag{11}$$



(a) (b)

Fig. 6. Aggregation in the CBLP: (a) circuit schematic, and (b) measured output using identical values of W_j and X_j across all the BCA columns, at $\Delta V_{BL,\max} = 320\,\text{mV}$.

where $\Delta_{W,N}^{(m)}$ is the *batch gradient estimate* generated by accumulating the *sample gradient estimate* $\Delta_{W,n}^{(m)}$:

$$\Delta_{W,n+1}^{(m)} = \Delta_{W,n}^{(m)} + \begin{cases} y_n^{(m)} \boldsymbol{X}_n^{(m)} & \text{if } \delta_n^{(m)} \leq 0 \\ 0 & \text{otherwise} \end{cases} \tag{12}$$

where $y_n^{(m)}$ is the true label corresponding to the data sample $\boldsymbol{X}_n^{(m)}$, and $\delta_n^{(m)}$ is computed in the margin detector (see Fig. 6(a)) as follows:

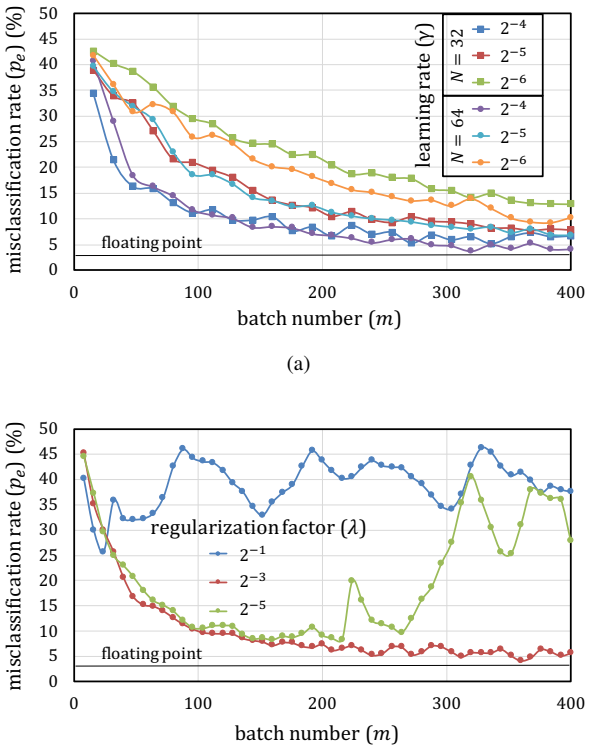
$$\delta_n^{(m)} = y_n^{(m)}[\text{sign}(z_n^{(m)} - 1) + \text{sign}(z_n^{(m)} + 1)] \tag{13}$$

and $z_n^{(m)} = \boldsymbol{W}_m'^T \boldsymbol{X}_n^{(m)} + b$ is the output of the CBLP block. The margin detector also generates the SVM decision $\hat{y}_n^{(m)} = \text{sign}(z_n^{(m)})$.

The trainer consists of an input buffer to store the streamed input $\boldsymbol{X}$ and a gradient buffer to store $\Delta_{W,n}$. The trainer implements (11) and (12) by reusing 64 16-b adders to conserve area. The weight update is performed in two cycles, where 64 words of $\boldsymbol{W}$ are updated per cycle (see Fig. 2(b)). Multiplication with γ/N and $\gamma\lambda$ in (11) are implemented using barrel shifters, thereby restricting them to powers of 2 in the range $[1, 2^{-15}]$. The ability to choose learning rates in powers-of-2 provides a wider tuning range over learning rates chosen on a linear scale. Wider tuning range in fact allows operating with very small learning rates, thereby enabling fine-tuning of the weights to achieve a lower misclassification rate.

1) Trainer precision assignment: The precision of the trainer needs to be chosen carefully in order to minimize the cost of training without compromising the convergence behavior of the SGD algorithm. The minimum bit precision for W and $\Delta_{W,n}$ needs to be set. Additionally, the minimum precision for W during inference (1) and training (11), denoted by B_W and B_{WUD}, respectively, can be substantially different [25]. To avoid gradient accumulator overflow in (12), the precision of $\Delta_{W,n}$ (B_Δ) is bounded by,

$$B_\Delta \geq B_X + \log_2 N \tag{14}$$



(a)

(b)

Fig. 7. Measured learning curves showing the impact of: (a) batch size N and learning rate γ with $\lambda = 2^{-4}$, and (b) regularization factor λ with $\gamma = 2^{-4}$ and $N = 64$.

where B_X is the precision of X, which is fixed at 8-b in this application. We choose $B_\Delta = 16$ as $B_X = 8$ and we wish to accommodate batch sizes of up to $N = 256$.

It can be shown [25] that a necessary condition for convergence (stopping criterion) of the SGD update in (11) is given by,

$$B_{WUD} \geq 1 - \log_2 \gamma \tag{15}$$

We choose $B_{WUD} = 16$ as algorithmic simulations indicate that the algorithm converges with a learning rate $\gamma \geq 2^{-15}$.

Additionally, the batch-mode algorithm offers an interesting trade-off between the batch size N and the learning rate γ which will be studied in Section V-B.

The regularization factor λ has an optimum value that lies in between an upper bound that constrains the magnitude of W and a lower bound needed to avoid overflow in the weight accumulator (11). It can be shown that a sufficient condition to prevent overflow in the weight accumulator is,

$$\lambda \geq \Pr\{y(\boldsymbol{W}'^{T}_{\text{opt}}\boldsymbol{X} + b) < 1\} \tag{16}$$

where $\boldsymbol{W}'_{\text{opt}}$ are the optimal weights after full convergence. Similarly, there exists an upper bound on λ beyond which $|W|$ gets constrained so heavily that the MSBs are zero.

V. Experimental Results

This section describes the measured results from the prototype IC, and evaluates the effectiveness of on-chip learning in enhancing robustness. The 65 nm CMOS prototype IC (see Fig. 11) with a 16 kB SRAM is packaged in a 80-pin QFN.The area overhead of the in-memory computation circuits and the trainer is 15% and 35% of the IM-CORE area, respectively. The overhead of the trainer stems from the need to store intermediate gradients $\Delta_{W,n}$.

A. Training Procedure

The prototype IC is evaluated on the MIT CBCL face detection dataset [26]. The task is a binary classification problem with the dataset consisting separate sets of 4000 training and 858 test images with an equal number of 'face' and 'non-face' images. The input images were scaled down in size to 11×11 and then extended to accommodate the bias term b so that an 8-b 128-dimensional weight vector can be stored in four rows of the BCA.

During training, the batch $\{\boldsymbol{X}_n^{(m)}\}_{n=0}^{N-1}$ is generated from the training set by random sampling with replacement. During convergence, at the end of every 8^{th} batch, the misclassification rate $p_e = Pr\{y \neq \hat{y}\}$ is calculated over the entire test set using the batch weight vector $\boldsymbol{W}_m$.

B. On-chip Learning Behavior

Measured learning curves in Fig. 7(a) show that the convergence is faster at a higher learning rate γ for both $N = 32$ and $N = 64$. Additionally, the learning curves become smoother and converge to a lower p_e with larger batch sizes, e.g., for $\gamma = 2^{-4}$, the misclassification rate p_e at $m = 400$ is lower for the batch size $N = 64$ than for $N = 32$. Figure 7(b) shows that the fixed-point algorithm converges for a regularization factor $\lambda = 2^{-4}$ but diverges for values of $\lambda = 2^{-1}$ and $\lambda = 2^{-5}$. For $\lambda = 2^{-5}$, the weights initially converge and then diverge due to overflow, where as for $\lambda = 2^{-1}$, the algorithm does not converge at all because the MSBs which are used in the SVM dot product (1) remain at zero.

C. Robustness

Figure 8 shows that on-chip learning converges with randomly set initial weights in the BCA. Furthermore, the learning curves converge to within 1% of floating-point accuracy with 400 batch updates for $\gamma = 2^{-3}$ and 2^{-4} with $\Delta V_{BL,\max} = 560\,\text{mV}$. The misclassification rate p_e increases dramatically to 18% when $\Delta V_{BL,\max}$ is reduced to 320 mV at $m = 400$. As discussed in Section IV.A, this increase occurs due to the increased impact of spatial variations. Continued on-chip learning reduces p_e down to 8% for $\gamma = 2^{-4}$ and 2^{-3} within 150 additional batch updates. Similar results are observed when the illumination of the input images changes abruptly at $m = 400$ (see Fig. 8(b)), where p_e increases to 16% and eventually reduces to 6% with further training. The measurements in Fig. 8 indicate the effectiveness of on-chip learning in enhancing the robustness to variations in the process parameters and the input data statistics.

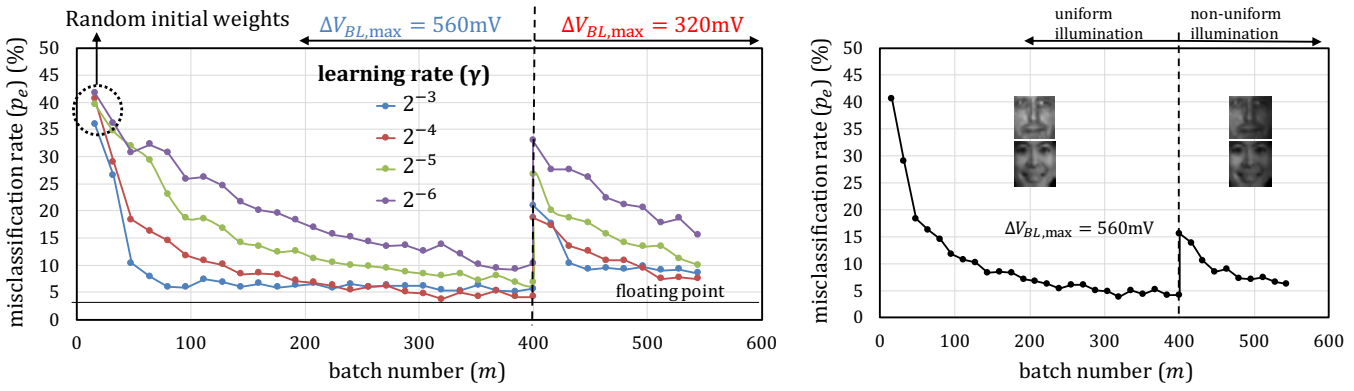
The chip-specific nature of these learned weights can be seen in Fig. 9 which shows the average p_e increases from 8.4% to 43% when weights learned on a different die are used. This result further highlights the need for on-chip learning.

The receiver operating curve (ROC) characterizes the true positive rate p_{tp} (the fraction of positive examples (faces) classified correctly) with respect to false positive rate p_{fp} (the fraction of negative examples (non-faces) classified as positive). The false positive rate p_{fp} was set by digitally adjusting the ADC output offset, and p_{tp} and p_{fp} were measured over the entire test dataset. With off-chip trained weights, the ROC of the classifier (Fig. 10(a)) degrades (moves away from the ideal) with decreasing $\Delta V_{BL,\max}$. Additionally, the default operating point, i.e., without ADC offset cancellation, (black markers in Fig. 10(a)) are away from the optimal (knee of the ROC). However, with on-chip training (see Fig. 10(b)) the ROCs shift towards the ideal (upper left corner), and the default operating point (black markers) also moves automatically to the optimal location (the knee), thereby eliminating the need for offset tuning.

D. Energy Consumption

The minimum $\Delta V_{BL,\max}$ required to achieve a misclassification rate $p_e \leq 8\%$ without compensating for process variations is 520 mV (see Fig. 12(a)). On-chip training enables the IC to achieve a misclassification rate below 8% at a 38% lower $\Delta V_{BL,\max} = 320\,\text{mV}$. Operating with $\Delta V_{BL,\max} = 320\,\text{mV}$ also enables the IC to operate with a lower $V_{DD,\text{IM-CORE}} = 0.675\,\text{V}$ without destructive reads as compared to operating at $V_{DD,\text{IM-CORE}} = 0.875\,\text{V}$ when $\Delta V_{BL,\max} = 520\,\text{mV}$ (see Fig. 12(b)). Thus, on-chip learning enables the reduction in IM-CORE energy by $2.4\times$ at iso-accuracy. This energy gain ranges from $1.5\times$-to-$2.4\times$ for p_e in the range 5%-to-8%.

The energy cost of training is dominated by SRAM writes of the updated weights in (11) at the end of each batch (see Fig. 13). This cost reduces with increasing batch size N, reaching 26% of the total energy cost, for a $N = 128$. At $N = 128$, 60% of the total energy can be attributed to CTRL, whose energy reduces with increasing BCA size.



(a) (b)

Fig. 8. Measured learning curves showing robustness to: (a) process variations, and (b) variations in input statistics with a batch size $N = 64$ and a regularization factor $\lambda = 2^{-4}$.

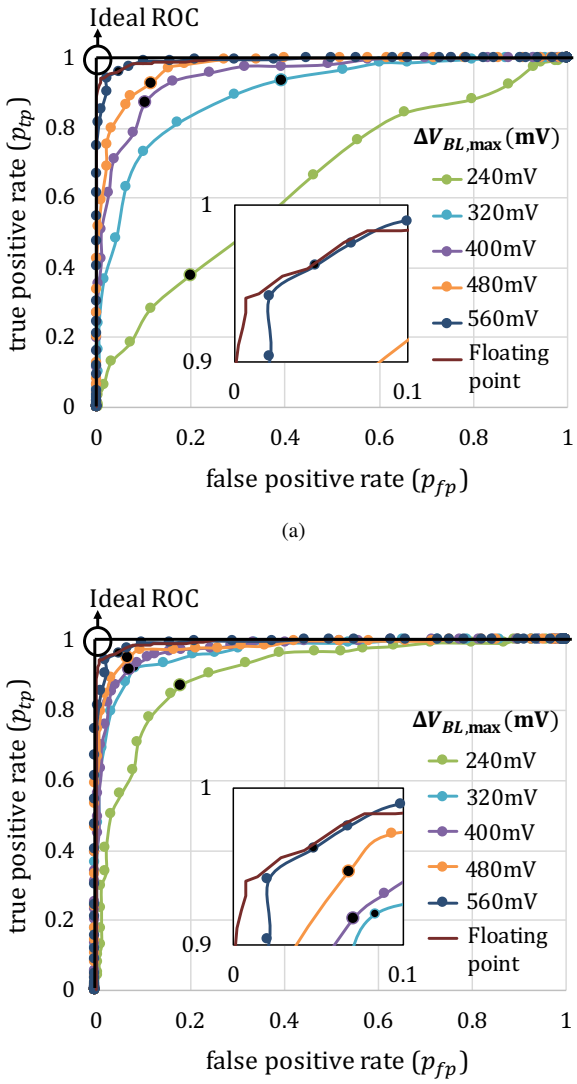
tested on

trained on	Chip1	Chip2	Chip3	Chip4	Chip5
Chip1	8.25	38.3	48.3	51.5	48.8
Chip2	45.8	9	48	49.8	34.5
Chip3	47	51.3	8.5	29.8	49.3
Chip4	51.5	51	17.5	8.25	51.3
Chip5	38.3	18	48.5	48.5	8

Fig. 9. Misclassification rate (p_e) measured across chips when the weights trained on a different die are used.

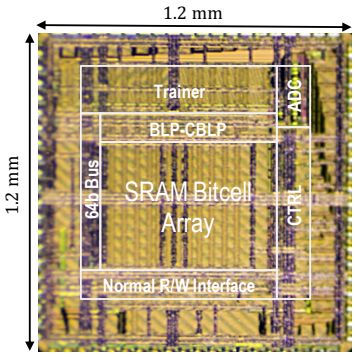
Figure 14 shows the energy breakdown of the prototype IC compared to a conventional digital reference architecture (CONV). CONV is a 2-stage pipelined architecture comprising an SRAM of the same size as in the prototype IC, and a synthesized digital block. The conventional SRAM has a column multiplexer ratio of $4:1$, therefore requiring $16\times$ more read accesses than the DIMA. The energy and delay numbers of CONV were based on measured read energy from the prototype IC and computational energy from post-layout simulations.

The energy and delay benefits of the prototype IC stem from: a) simultaneously reading multiple rows and processing them in low swing analog domain, b) eliminating the 4:1 column mux, and c) by aggressively reducing BL swing enabled by the use of chip-specific weights obtained via on-chip learning. When operating with pre-trained weights at $\Delta V_{BL} = 560\,\text{mV}$, the prototype IC shows a $7.8\times$ reduction in energy compared to CONV, where the contributions due to a) and b) are estimated to be $5\times$ and $1.6\times$, respectively. Use of chip specific weights via on-chip learning increases the energy reduction to $21\times$. Along with the reduction in energy, the prototype IC simultaneously shows an overall $4.7\times$ reduction in delay, thereby achieving an overall $100\times$ reduction in EDP during inference. Due to the use of digital read and write operations during the weight update, the energy gain during training reduces to $6.2\times$ (from $21\times$ during inference) at a



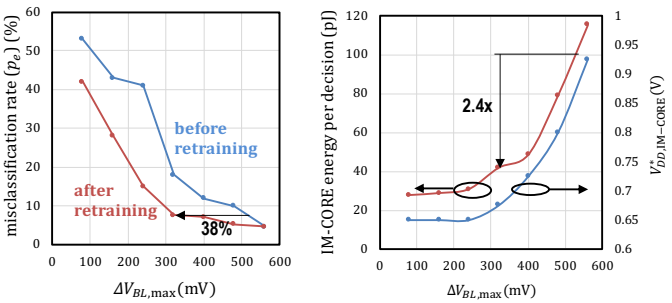
(a) (b)

Fig. 10. Receiver operating curves (ROC) measured with: (a) off-chip trained weights, and (b) on-chip trained weights. Black markers represent default operating points.



Technology	65nm CMOS	
Die size	1.2 mm × 1.2mm	
Memory capacity	16KB (512 × 256)	
Nominal supply	1.0 V	
CTRL operating frequency	1 GHz	
Energy per decision (nJ)	Test	0.21
	Training	0.34
Average throughput (decision/s)	Test	32.3 M
	Training	21 M

Fig. 11. Die photograph and chip summary.



(a) (b)

Fig. 12. Measured misclassification rate p_e showing 38% reduction in BL swing attributed to on-chip learning, leading to a 2.4× reduction in energy ($V^*_{DD,\text{IM-CORE}}$ is the minimum IM-CORE supply to avoid destructive reads).

batch size $N = 64$.

Table I compares the prototype IC to other in-memory dot product implementations. The prototype IC operates with the energy efficiency of 42 pJ/decision at a throughput of 32 M decisions/s, which corresponds to a computational energy efficiency of 3.12 TOPS/W (1 OP = one 8-b×8-b MAC) for inference. DIMA's energy-latency benefits arise primarily from reduced memory access costs, which tend to dominate with large SRAM bank sizes. Furthermore, DIMA is best suited for algorithms that that suffer most from memory access costs

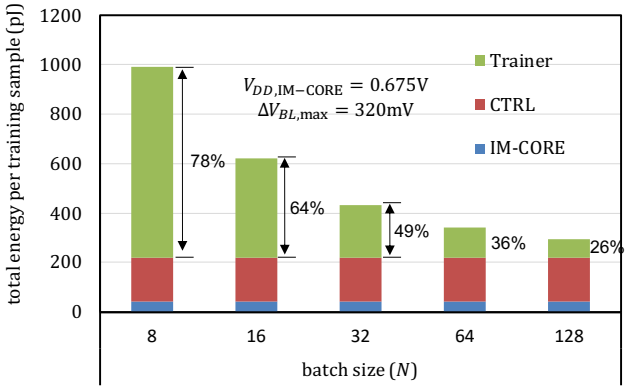


Fig. 13. Energy overhead of on-chip training with respect to batch size.

TABLE I
COMPARISON WITH PRIOR IN-MEMORY DOT PRODUCT IMPLEMENTATIONS

	[10]	[11]	[13]	[15]	This Work
Technology	130nm	65nm	65nm	65nm	65nm
Algorithm	AdaBoost	SVM	CNN	DNN	SVM
Dataset	MNIST	MIT-CBCL	MNIST	MNIST	MIT-CBCL
On-chip memory (kB)	2	16	2	0.5	16
Bit-cell type	6T	6T	10T	6T[1]	6T
Energy/ decision (pJ)	600	400	-	-	42
Decision/s	7.9M	9.2M	-	-	32M
Precision[2] (B_X×B_W)	5×1[s]	8×8	7×1[s]	1×1	8×8[s]
Efficiency[3] (TOPS/W)	350	1.25	14	55.8	3.125 (1.07)[4]
Throughput[3] (GOPS)	819	4.17	5.35	1780	4.13
E_{MAC}[5](pJ)	0.003	0.8	0.071	0.018	0.32 (0.92)[4]
$E_{MAC,p}$[6](fJ)	0.56	12.5	10.2	17.9	4.9 (14.5)[4]

[1]separate left and right word lines (WLs) used
[2]signed number indicated by [s]
[3]1OP = 1 Multiply and accumulate
[4]at ΔV_{BL} = 320 mV (560 mV)
[5]E_{MAC} is the energy of a single MAC operation
[6]precision scaled MAC energy $E_{MAC,p} = E_{MAC}/(B_X \times B_W)$

such as fully connected deep neural networks (FC-DNN). Therefore, we compare with [5] which implements a FC-DNN, and employs aggressive voltage/frequency scaling along with digital error compensation techniques such as RAZOR [27]. Employing reported arithmetic efficiency of 0.09-0.16 TOPS/W [5] and accounting for the difference in the process node (28 nm FD-SOI [5] vs. 65 nm bulk CMOS), we find that the prototype IC achieves a 30× savings in energy accompanied by a 1.8× savings in delay to implement an 8-b 128-wide dot product. These energy savings demonstrate the suitability of the proposed architecture for energy-constrained sensory IoT applications.

VI. CONCLUSION

This paper has presented an IC realization of a deep in-memory classifier for the SVM algorithm with on-chip learning in a 65 nm CMOS process. On-chip training overcomes the impact of variations in both process parameters and data statistics, thereby enabling the IC to operate at lower BL discharges than otherwise possible thereby saving energy.

In this paper, we demonstrate that on-chip learning is effective in compensating for process variations in DIMA. However, to take full advantage of this technique, a number of algorithmic, architectural, and circuit challenges need to be overcome. Unsupervised and semi-supervised learning algorithms are needed to avoid having to store the training set on-chip. Efficient write-back techniques would be required in order to minimize the energy consumption during training especially for always-on systems that need to continuously track their environment. Realizing on-chip training is made

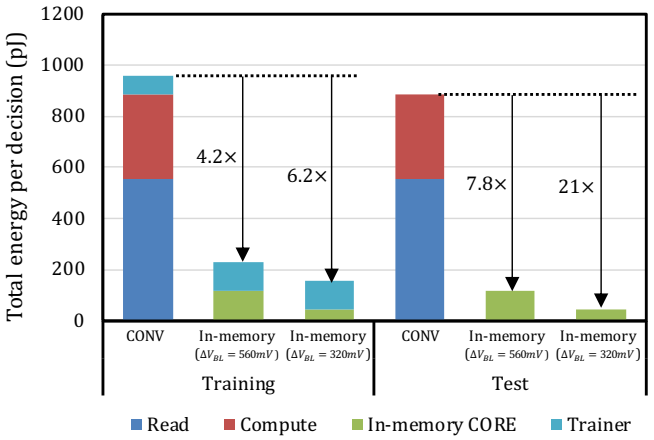


Fig. 14. Measured on-chip energy for training (at $N = 64$) and inference compared to a conventional digital reference architecture (CONV), showing $21\times$ reduction in energy consumption with a simultaneous $4.7\times$ reduction in delay, leading to a $100\times$ reduction in EDP during inference. The supply $V_{DD,\text{IM-CORE}}$ is 1 V and 675 mV when operating with a ΔV_{BL} of 560 mV and 320 mV, respectively.

much more challenging for large-scale deep networks. However, since both inference and trainer computations are based on matrix-vector operations, there is significant potential for employing DIMA to realize both the forward and backward parts of the network. In such cases, the impact of PVT variations on the learning behavior and inference accuracy will be interesting to study.

Acknowledgment

This work was supported in part by Systems On Nanoscale Information fabriCs (SONIC), one of the six SRC STARnet Centers, sponsored by MARCO and DARPA. The authors would like to acknowledge constructive discussions with Professors Pavan Hanumolu, Naveen Verma, Boris Murmann, and David Blaauw.

References

[1] M. Horowitz, "Computing's energy problem (and what we can do about it)," in *IEEE International Solid-State Circuits Conference (ISSCC)*, 2014, pp. 10–14.

[2] Y.-H. Chen, T. Krishna, J. S. Emer, and V. Sze, "Eyeriss: An energy-efficient reconfigurable accelerator for deep convolutional neural networks," *IEEE Journal of Solid-State Circuits*, vol. 52, no. 1, pp. 127–138, Jan 2017.

[3] B. Moons, R. Uytterhoeven, W. Dehaene, and M. Verhelst, "ENVISION: A 0.26-to-10 TOPS/W subword-parallel dynamic-voltage-accuracy-frequency-scalable convolutional neural network processor in 28nm FDSOI," in *IEEE International Solid-State Circuits Conference (ISSCC)*, 2017, pp. 246–247.

[4] D. Shin, J. Lee, J. Lee, and H.-J. Yoo, "DNPU: An 8.1 TOPS/W reconfigurable CNN-RNN processor for general-purpose deep neural networks," in *IEEE International Solid-State Circuits Conference (ISSCC)*, 2017, pp. 240–241.

[5] P. N. Whatmough, S. K. Lee, H. Lee, S. Rama, D. Brooks, and G.-Y. Wei, "A 28nm SOC with a 1.2 GHz 568nJ/prediction sparse deep-neural-network engine with¿ 0.1 timing error rate tolerance for IOT applications," in *IEEE International Solid-State Circuits Conference (ISSCC)*, 2017, pp. 242–243.

[6] J. Lee, C. Kim, S. Kang, D. Shin, S. Kim, and H.-J. Yoo, "UNPU: A 50.6 TOPS/W unified deep neural network accelerator with 1b-to-16b fully-variable weight bit-precision," in *IEEE International Solid-State Circuits Conference (ISSCC)*, 2018, pp. 218–220.

[7] 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," *International conference on Architectural Support for Programming Languages and Operating Systems (ASPLOS)*, vol. 49, no. 4, pp. 269–284, 2014.

[8] M. Kang, M.-S. Keel, N. R. Shanbhag, S. Eilert, and K. Curewitz, "An energy-efficient VLSI architecture for pattern recognition via deep embedding of computation in SRAM," in *IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, May 2014, pp. 8326–8330.

[9] N. Shanbhag, M. Kang, and M.-S. Keel, *Compute Memory*. Issued July 4 2017, US Patent 9,697,877 B2.

[10] 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, April 2017.

[11] M. Kang, S. K. Gonugondla, A. Patil, and N. R. Shanbhag, "A multi-functional in-memory inference processor using a standard 6T SRAM array," *IEEE Journal of Solid-State Circuits*, vol. 53, no. 2, pp. 642–655, Feb 2018.

[12] M. Kang, S. K. Gonugondla, S. Lim, and N. R. Shanbhag, "A 19.4-nJ/decision, 364-K decisions/s, in-memory random forest multi-class inference accelerator," *IEEE Journal of Solid-State Circuits*, vol. 53, no. 7, pp. 2126–2135, July 2018.

[13] A. Biswas and A. P. Chandrakasan, "Conv-RAM: An energy-efficient SRAM with embedded convolution computation for low-power CNN-based machine learning applications," in *IEEE International Solid-State Circuits Conference (ISSCC)*, 2018, pp. 488–490.

[14] W.-H. Chen, K.-X. Li, W.-Y. Lin, K.-H. Hsu, P.-Y. Li, C.-H. Yang, C.-X. Xue, E.-Y. Yang, Y.-K. Chen, Y.-S. Chang *et al.*, "A 65nm 1Mb nonvolatile computing-in-memory ReRAM macro with sub-16ns multiply-and-accumulate for binary DNN AI edge processors," in *IEEE International Solid-State Circuits Conference (ISSCC)*, 2018, pp. 494–496.

[15] W.-S. Khwa, J.-J. Chen, J.-F. Li, X. Si, E.-Y. Yang, X. Sun, R. Liu, P.-Y. Chen, Q. Li, S. Yu *et al.*, "A 65nm 4kb algorithm-dependent computing-in-memory sram unit-macro with 2.3 ns and 55.8 tops/w fully parallel product-sum operation for binary dnn edge processors," in *IEEE International Solid-State Circuits Conference (ISSCC)*, 2018, pp. 496–498.

[16] T. F. Wu, H. Li, P.-C. Huang, A. Rahimi, J. M. Rabaey, H.-S. P. Wong, M. M. Shulaker, and S. Mitra, "Brain-inspired computing exploiting carbon nanotube FETs and resistive RAM: Hyperdimensional computing case study," in *IEEE International Solid-State Circuits Conference (ISSCC)*, 2018, pp. 492–494.

[17] I. Hubara, M. Courbariaux, D. Soudry, R. El-Yaniv, and Y. Bengio, "Binarized neural networks," in *Advances in neural information processing systems*, 2016, pp. 4107–4115.

[18] C. Sakr, Y. Kim, and N. Shanbhag, "Analytical guarantees on numerical precision of deep neural networks," in *International Conference on Machine Learning*, 2017, pp. 3007–3016.

[19] B. Moons, K. Goetschalckx, N. Van Berckelae, and M. Verhelst, "Minimum energy quantized neural networks," in *Asilomar conference on Signals, Systems and Computer*, 2017.

[20] 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 *IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, May 2015.

[21] M. Kang, S. K. Gonugondla, and N. R. Shanbhag, "A 19.4 nJ/decision 364K decisions/s in-memory random forest classifier in 6T SRAM array," in *IEEE European Solid-State Circuits Conference (ESSCIRC)*, 2017, pp. 263–266.

[22] L. Bottou, F. E. Curtis, and J. Nocedal, "Optimization methods for large-scale machine learning," *arXiv preprint arXiv:1606.04838*, 2016.

[23] S. K. Gonugondla, M. Kang, and N. Shanbhag, "A 42pJ/decision 3.12 TOPS/W robust in-memory machine learning classifier with on-chip training," in *IEEE International Solid-State Circuits Conference (ISSCC)*, 2018, pp. 490–492.

[24] C. Cortes and V. Vapnik, "Support-vector networks," *Machine learning*, vol. 20, no. 3, pp. 273–297, 1995.

[25] C. Sakr, A. Patil, S. Zhang, Y. Kim, and N. Shanbhag, "Minimum precision requirements for the SVM-SGD learning algorithm," in *IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, 2017, pp. 1138–1142.

[26] "Center for biological and computational learning (CBCL) at MIT," 2000, `http://cbcl.mit.edu/software-datasets/index.html`.

[27] D. Ernst, N. S. Kim, S. Das, S. Pant, R. Rao, T. Pham, C. Ziesler, D. Blaauw, T. Austin, K. Flautner *et al.*, "Razor: A low-power pipeline based on circuit-level timing speculation," in *IEEE/ACM International Symposium on Microarchitecture*, 2003, p. 7.

Sujan K. Gonugondla (S'16) received the B.Tech. and M.Tech. degrees in Electrical Engineering from Indian Institute of Technology Madras, Chennai, India, in 2014. He is currently pursuing the Ph.D. degree in Electrical and Computer Engineering at the University of Illinois at Urbana- Champaign, Champaign, IL, USA. His current research interest includes low-power integrated circuits specifically algorithm hardware co-design for machine learning systems on resource constrained environments.

Sujan Gonugondla is a recipient of the Dr. Ok Kyun Kim Fellowship 2018-19 from the ECE department at the University of Illinois at Urbana-Champaign and the ADI Outstanding Student Designer Award 2018.

Mingu Kang (M'13) received the B.S. and M.S. degrees in Electrical and Electronic Engineering from Yonsei University, Seoul, Korea, in 2007 and 2009, respectively, and the Ph.D. degree in Electrical and Computer Engineering from the University of Illinois at Urbana-Champaign, Champaign, IL, USA, in 2017.

From 2009 to 2012, He was with the Memory Division, Samsung Electronics, Hwaseong, South Korea, where he was involved in the circuit and architecture design of Phase Change Memory (PRAM). Since 2017, he has been with the IBM Thomas J. Watson Research Center, Yorktown Heights, NY, USA, where he designs machine learning accelerator architecture. His research interests include low-power integrated circuits, architecture, and system for machine learning, signal processing, and neuromorphic computing.

Naresh R. Shanbhag (F'06) is the Jack Kilby Professor of Electrical and Computer Engineering at the University of Illinois at Urbana-Champaign. He received his Ph.D. degree from the University of Minnesota (1993) in Electrical Engineering. From 1993 to 1995, he worked at AT&T Bell Laboratories at Murray Hill where he led the design of high-speed transceiver chip-sets for very high-speed digital subscriber line (VDSL), before joining the University of Illinois at Urbana- Champaign in August 1995. He has held visiting faculty appointments at the National Taiwan University (Aug.-Dec. 2007) and Stanford University (Aug.-Dec. 2014). His research interests are in the design of energy-efficient integrated circuits and systems for communications, signal processing and machine learning. He has more than 200 publications in this area and holds thirteen US patents.

Dr. Shanbhag became an IEEE Fellow in 2006, received the 2010 Richard Newton GSRC Industrial Impact Award, the IEEE Circuits and Systems Society Distinguished Lecturership in 1997, the National Science Foundation CAREER Award in 1996, and multiple best paper awards. In 2000, Dr. Shanbhag co-founded and served as the Chief Technology Officer of Intersymbol Communications, Inc., (acquired in 2007 by Finisar Corporation) a semiconductor start-up that provided DSP-enhanced mixed-signal ICs for electronic dispersion compensation of OC-192 optical links. From 2013-17, he was the founding Director of the Systems On Nanoscale Information fabriCs (SONIC) Center, a 5-year multi- university center funded by DARPA and SRC under the STARnet program.