Improving DNN Hardware Accuracy by In-Memory Computing Noise Injection

Sai Kiran Cherupally, Jian Meng, and Adnan Siraj Rakin
Arizona State University,
Tempe, AZ 85287 USA

Shihui Yin
Huawei Technologies, Beijing, China

Mingoo Seok
Columbia University,
New York, NY 10027 USA

Deliang Fan and Jae-Sun Seo
Arizona State University,
Tempe, AZ 85287 USA

Editor's notes:
Like any other designs, in-memory computing (IMC) also suffers from operational inaccuracy induced by hardware noise. In this work, the authors propose to take into account hardware noises during the deep neural network (DNN) training in order to improve the DNN inference accuracy.
—Yiran Chen, Duke University

Deep neural networks (DNNs) have been very successful across many applications, but they require a very large amount of computation and storage to achieve high accuracy. On the algorithm side, the arithmetic complexity and storage requirement of DNNs have been aggressively reduced by low-precision quantization techniques [1]. On the hardware side, many digital accelerators efficiently implemented DNNs with specialized dataflows, based on a systolic array of multiply-and-accumulate (MAC) engines and on-chip memory hierarchy. Still, the energy/power breakdown results reported in recent DNN accelerators [2] show that memory access and data communication consume a dominant portion (e.g., two-thirds or higher) of the total on-chip energy/power.

To address such bottlenecks, in-memory computing (IMC) has emerged as a promising technique. IMC performs MAC computation inside the memory (e.g., SRAM) by activating multiple/all rows, whose result is represented by analog bitline voltage/current (V_{BL}/I_{BL}), and subsequently digitized by an analog-to-digital converter (ADC) in the periphery. This substantially reduces data transfer (compared to digital accelerators with separate MAC arrays) and increases parallelism (compared to conventional row-by-row access), significantly improving the energy efficiency of MAC operations. Several IMC SRAM prototype chips [3]–[6] demonstrated high energy efficiency of up to hundreds of TOPS/W by efficiently combining storage and computation.

However, IMC designs achieve higher energy efficiency than digital counterparts by trading off the signal-to-noise ratio (SNR), since analog computation inherently involves variability/noise. As a result,

Digital Object Identifier 10.1109/MDAT.2021.3139047
Date of publication: 27 December 2021; date of current version: 22 June 2022.

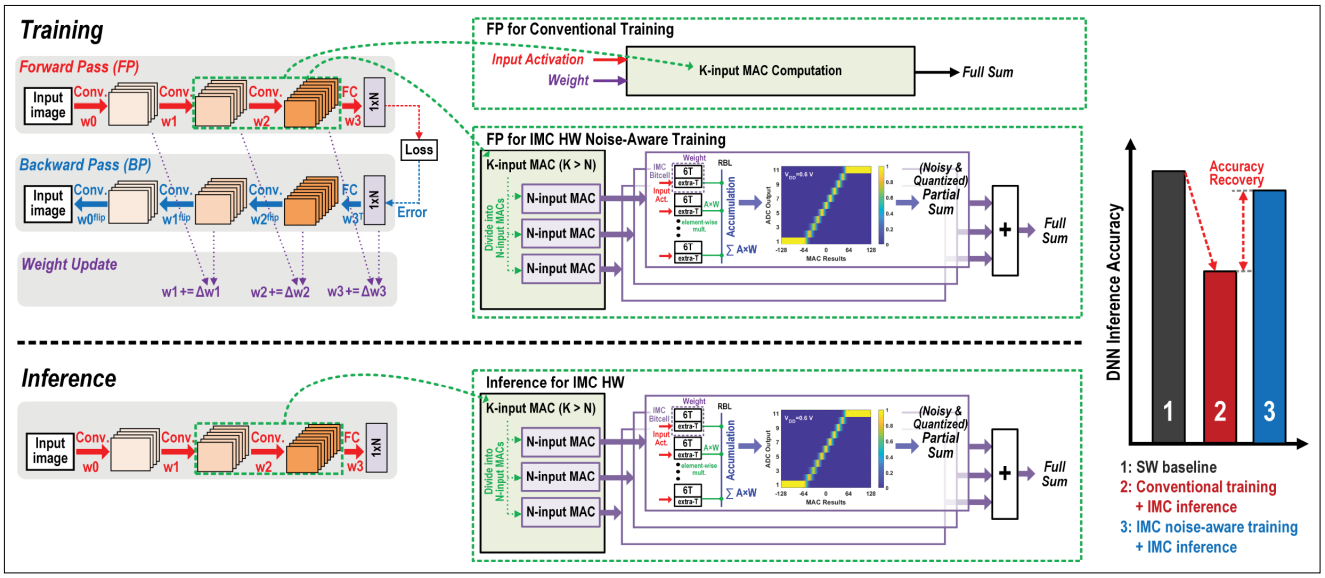

Figure 1. Illustration of the proposed IMC hardware noise-aware training and IMC inference evaluation. Introducing IMC hardware noise to DNN training helps recover most of the accuracy loss, which was validated with IMC prototype chip measurements.

IMC chips show variability in the ADC outputs for the same ideal MAC value and often report accuracy degradation compared to the digital baseline [3], [5], [7], [8]. For example, DNN accuracy degradation of ~7% for the CIFAR-10 data set was reported when baseline DNNs are evaluated on the noisy IMC prototype chip of [3], where all 256 rows of the IMC SRAM array are activated simultaneously.

To mitigate this accuracy loss, some IMC SRAM works attempted to improve the SNR by limiting the number of activated rows for IMC, for example, 36 rows in [9], but this reduces the computing parallelism and energy efficiency. Other works performed DNN training with noise injection at individual weight-level [7], [8] or activation-level [10], which do not consider the IMC crossbar structure and other hardware noise such as bitline/ADC noise.

In this work, we present a new hardware noise-aware DNN training scheme to largely recover the accuracy loss of highly parallel IMC hardware (Figure 1). The novelty of our work lies in that: 1) noise injection is performed at the partial sum level that matches with the IMC crossbar and 2) the injected noise is based on actual hardware noise measured from multiple chips of two recent IMC prototype designs [3], [6]. The actual IMC hardware noise measured at the partial sum level (ADC output) captures individual weight-/activation-level noise, bitline noise, and ADC offset/quantization noise collectively. We report results of hardware noise-aware training and inference for various DNN models with 1-/2-/4-bit precision. Furthermore, by using noise data obtained from five different chips, we also evaluate the effectiveness of our proposed DNN training using individual chip's noise data versus the ensemble noise from multiple chips.

The key contributions and observations of this work are as follows.

- To effectively improve the DNN accuracy of IMC hardware, we inject hardware extracted noise for DNN training at the partial sum level, which matches with the IMC crossbar structure.
- We observe that the IMC hardware noise is different from a Gaussian distribution, and for DNNs trained with Gaussian noise, this causes a mismatch during inference, resulting in suboptimal accuracy.
- We perform noise-injection training and evaluate DNN accuracy for multiple DNNs and precision values with two different IMC designs' measurement results.
- Considering inter/intrachip variations, we evaluate the individual chip data-based training and overall chips' data-based ensemble training methods.

Background and related works

SRAM-based IMC

In IMC systems, DNN weights are stored in a crossbar structure, and analog computation is performed typically by applying activations as voltage on the row side and accumulating the bitwise multiplication result via analog voltage/current on the column side. ADCs at the periphery quantize the analog voltage/current into digital values. This way, vector-matrix multiplication (VMM) of activation vectors and the stored weight matrices is computed in a highly parallel manner without reading out the weights.

Both SRAM-based IMC [3], [4], [6] and nonvolatile memory (NVM)-based IMC [11] have been presented. While NVMs have density advantages, the availability of embedded NVMs in scaled CMOS technologies is limited, and device nonidealities such as low ON/OFF ratio, endurance, relaxation, and so on pose challenges for robust large-scale integration. Conversely, SRAM has a very high ON/OFF ratio and SRAM-based IMC can be implemented in any latest CMOS technology. Therefore, we focus on SRAM IMC designs in this article.

SRAM IMC schemes can be categorized into resistive IMC that uses resistive pull-down/-up transistors [3]–[5] and capacitive IMC [6] that employs capacitive-coupling or charge-sharing for MAC computation. In a resistive IMC design "XNOR-SRAM" [3], binary multiplication (XNOR) between activations and weights is implemented by pull-down/-up transistors. In a capacitive IMC design "C3SRAM" [6], MAC operation is performed via capacitive-coupling with an additional capacitor. For resistive/capacitive IMC designs, each bitcell's multiplication result is accumulated onto the analog V_{BL} by forming a resistive/capacitive divider.

Hardware-aware DNN training for accurate DNN inference with IMC hardware

Accuracy degradation has been reported when software DNNs are deployed on IMC hardware due to quantization, variability, and transistor nonlinearity [3], [6], [7]. To improve the inference accuracy with IMC hardware, several works considered the nonideal hardware characteristics during DNN training [4], [5], [7]–[9], as follows.

ON-chip training circuits for IMC

Considering each chip's variations, [5] implemented an ON-chip gradient descent-based trainer to adapt/track ON-chip variations for support vector machine tasks. However, performance for large DNNs was not reported, and ON-chip training circuits incur a large overhead in area/energy.

Nonlinearity/quantization-aware training for IMC

In [4], the nonlinearity of V_{BL} was compensated in DNN training, but the accuracy was only evaluated for MNIST. A quantization-aware training scheme was proposed in [9] for NVM IMC, but only 36 rows are activated for IMC to maintain SNR, and still >2% accuracy loss is reported on DNNs for CIFAR-10.

Noise-aware training for IMC

Recent works [7], [8] injected noise during training at the individual weight level drawn from Gaussian distributions based on NVM variations. However, the IMC crossbar structure and the variations of wires/ADCs are not accounted for when using weight-level Gaussian distributions.

Our proposed noise-aware training scheme performs noise injection on the partial sum level that matches with the IMC crossbar structure, and the injected noise is directly from IMC chip measurement results on the quantized ADC outputs for different MAC values.

Proposed IMC hardware noise-aware DNN training and inference

IMC hardware and quantization noise

Both XNOR-SRAM [3] and C3SRAM [6] IMC macros activate all 256 rows to perform 256-input MAC with binary activations and weights (–1 and +1). The MAC result in the range from –256 to +256 is represented by the analog V_{BL}, which is digitized by 11-level flash ADCs to one of 11 possible output levels, that is, [–60, –48, –36, –24, –12, 0, 12, 24, 36, 48, 60]. Figure 2 shows the ADC output distributions obtained from XNOR-SRAM chip measurements at three different supply voltages. If the supply voltage changes, noise/variability gets affected, and the IMC chip results change as well. For the resistive IMC design XNOR-SRAM, Figure 2 shows that the measured noise worsened for higher supply voltages due to higher IR drop. These distributions/probabilities are used to transform the partial sums into noisy quantized ADC outputs. Intrachip (e.g., different SRAM columns) and interchip (e.g., different chips) variations exist, which affect the amount of noise

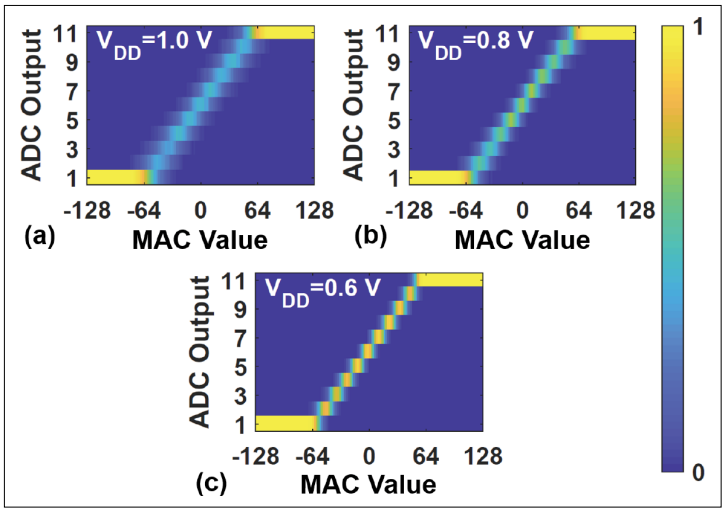

Figure 2. ADC output distributions for each partial sum (MAC) value from XNOR-SRAM chip measurements at (a) 1.0 V, (b) 0.8 V, and (c) 0.6 V supply (adapted from [3]).

introduced to the analog MAC computation and the resultant DNN accuracy.

For multibit DNN evaluation, multibit weights are split across multiple columns of the IMC array and multibit activations are fed to the IMC array over multiple cycles to perform bit-serial processing. The partial sums are then accumulated with proper binary-weighted coefficients depending on the bit positions of the subactivations/weights, and the full sum for a given neuron in the DNN layer is obtained.

DNN inference with IMC hardware emulation

We divide all MAC operations in convolution and fully connected layers of DNNs into multiple 256-input MAC operations (Figure 1). For every 256-input MAC operation, we use IMC chip measurement results with a sampling method described below. Accumulation of such 256-input partial sums and other non-MAC operations are performed via digital simulation.

To characterize the noisy quantization behavior for both XNOR-SRAM and C3SRAM chips, we performed a total of 409,600 measurements for 256-input MACs (409,600 = 1,600 × 25), where 1,600 measurements are obtained for each of the 256 binary MAC values with random activation/weight vectors. With such measurements, 2-D histograms between the MAC value and ADC output are obtained (Figure 2). This data is then converted to a conditional probability table, which reports the probability of each MAC value, resulting in different ADC outputs. Different chips or operating conditions (e.g., supply voltage) will be represented by different probability tables. To evaluate the accuracy of large DNNs, for each 256-input MAC value, we randomly "sample" the ADC output based on the distribution in the probability table, since small IMC chips cannot directly map large DNNs and time-multiplexing the small IMC chip requires excessive testing iterations.

Algorithm 1: Noise-aware DNN training

Input: n binary inputs x_i and weights w_i
Input: IMC row-size r
Input: cumulative noise probability matrix pt
Output: Noisy quantized MAC Q($\sum_1^n x_i \times w_i$)

Initialize: number of chunks $c = ceil(n/r)$
Initialize: Divide the inputs and weights into c chunks
Initialize: MAC, $d = 0$.
cdf.find(cdf, x): identifies the index of the first element in cdf that is less than x
random.uniform(): returns a random float in $[0, 1]$
for $i = 1$ **to** c **do**
  partial sum $ps = \sum_1^r x_i \times w_i$
  level-probs = pt[ps]
  index = cdf.find(level-probs, random.uniform())
  qlevel = levels[index]
  Q(cdp) = qlevel
  d = d + Q(cdp)
end for
return d

IMC hardware noise-aware training

In conventional IMC works, the inference accuracy is affected by the inherent hardware noise and variability. To address this, we performed noise-aware DNN training by injecting the measured IMC hardware noise into the forward pass during DNN training (Figure 1).

The proposed training algorithm is described in Algorithm 1, where we inject the hardware noise by emulating the IMC macro's MAC computation and then use the conditional probability table for each MAC value. We perform random sampling with the probability table and predict the corresponding ADC output value. The DNN is trained while injecting such IMC noise by using a windowed straight-through-estimator for the backward pass.

DNN training noise and DNN inference noise

If P is a partial sum between the inputs and weights in a column of IMC crossbar, we get a quantized partial sum QP at the ADC output as

$$\mathrm{QP} = \mathrm{IF}(P); \mathrm{Loss}_{\mathrm{ideal}} = f(W, \mathrm{IF})$$

where IF() is the ideal ADC transfer function, and the loss corresponding to quantization noise is a function of IF and weights (W). If we use Gaussian noise injection for the quantized partial sum during training, the equation would be

$$\mathrm{GQP} = \mathrm{QP} + N(\mu, \sigma); \mathrm{Loss}_{\mathrm{noise}} = f(W_1, \mathrm{IF}, N(\mu, \sigma))$$

where $N(\mu, \sigma)$ is a Gaussian distribution with mean μ and standard deviation σ, and the loss corresponding to the noisy quantization is a function of N and weights W_1. On the other hand, if we use noisy partial sum quantization scheme obtained from IMC chip measurement data, we get

$$\mathrm{NQP} = \mathrm{NF}(P); \mathrm{Loss}_{\mathrm{noise}} = f(W_2, \mathrm{NF})$$

where NF() is the measured noisy ADC transfer function of the IMC hardware, and the loss corresponding to the noisy quantization is a function of NF and weights W_2.

After training, W_1 and W_2 will converge to different sets of values. Since the real hardware inference exhibits IMC noise NF, W_1 is not matched well and will be suboptimal. More optimal results will be obtained with W_2, since the same IMC noise NF has been used during both training and inference.

Experiment results

We performed IMC hardware noise-aware DNN training with 32-bit floating-point precision and evaluated ResNet18, AlexNet, VGG, and MobileNet DNNs for CIFAR-10 data set. Targeting DNN inference with 1-, 2-, and 4-bit activation/weight precision, we employed quantization-aware training [12]. For the proposed hardware noise-aware training, all DNNs were trained by using a batch-size of 50 and the default hyperparameters in [12]. Furthermore, the reported DNN inference accuracy values are the average values obtained from five inference evaluations of the same DNN under the same noise conditions used during the proposed training process.

We used measurement results from XNOR-SRAM [3] and C3SRAM [6] chips at different supply voltages. Also, we performed ideal ADC-aware training by using the ideal ADC transfer function to quantize the partial sums, and ensemble IMC noise-aware training, by combining the probability tables of five different XNOR-SRAM chips and obtaining a unified probability table that represents the noise from five chips.

We experimented the following four schemes for DNN accuracy evaluation: 1) baseline represents the software DNN baseline; 2) conventional IMC inference represents the scheme with IMC chip measurement-based evaluation on baseline DNNs without noise-aware training; 3) ADC-aware IMC inference represents IMC chip measurement-based evaluation on the new DNNs trained with ideal ADC quantization; and 4) noise-aware IMC inference represents IMC chip measurement-based evaluation on the new DNNs trained with the proposed hardware noise injection.

XNOR-SRAM chip noise-aware training and inference

Using XNOR-SRAM chip [3] measurement results and probability tables, we performed the proposed noise-aware DNN training for different DNN models, with different activation/weight precision and with different noise models (e.g., noise from different supply voltages and different physical chips).

Different DNN models and precisions

We performed DNN training/inference on ResNet-18, VGG, AlexNet, and MobileNet DNNs for CIFAR-10, using the XNOR-SRAM chip measurement at 0.6-V supply. Figure 3a shows the results on the binarized DNNs [12], where the proposed IMC noise-aware training helps restore the IMC hardware accuracy closer to the software baseline in all models. For ResNet-18, the IMC hardware accuracy can be restored to within <1% of the software baseline, compared to ~3.5% accuracy degradation of the conventional scheme.

The accuracy improvement for VGG in Figure 3a is relatively small, with low IMC noise at 0.6 V for XNOR-SRAM chip (Figure 2c). When we applied high IMC noise of XNOR-SRAM chip at 1.0 V (Figure 2a), the conventional IMC inference accuracy was degraded severely to 68.3%, but the proposed IMC noise-aware scheme largely improved the accuracy to 86.4%, similar to the trend of ResNet-18 results, as shown in Figure 3c.

On the other hand, MobileNet have depth/point-wise convolution layers that are shallow and have kernel sizes down to 1×1, which makes the convolution computation sensitive to noise-induced variations, leading to large accuracy degradation.

Figure 3b shows the IMC hardware accuracy improvements in ResNet-18 DNNs for activation/

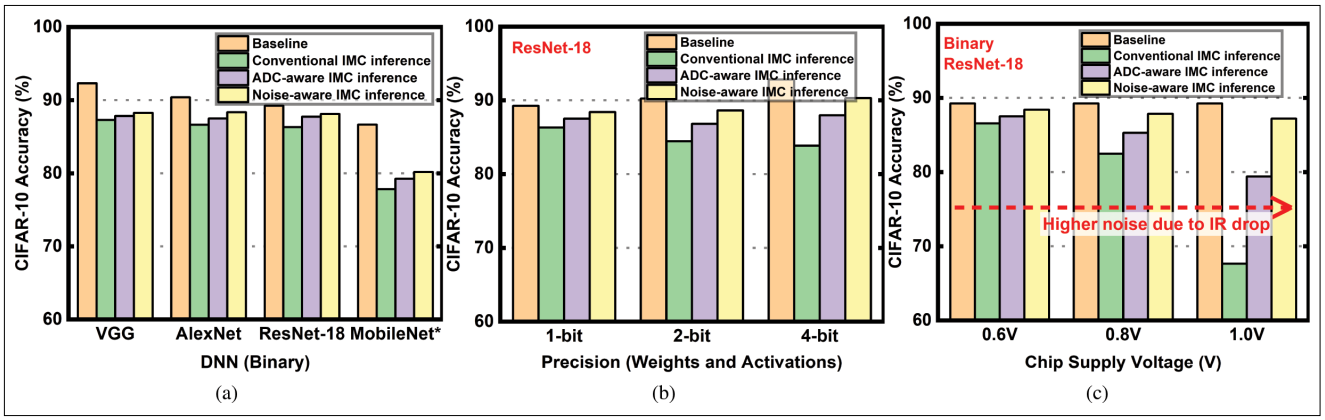

Figure 3. IMC inference accuracy after hardware noise-aware training of (a) different DNN topologies (*MobileNet only binarized convolution layers), (b) different activation/weight precisions for ResNet-18 with XNOR-SRAM chip at 0.6V, and (c) different XNOR-SRAM supply voltages for ResNet-18 DNN.

weight precision values of 1-, 2-, and 4-bit. As we increase the DNN precision, the IMC accuracy without noise-aware training worsens. This is because IMC hardware performs bitwise computations in each column, and as multiple columns' ADC outputs get shifted/accumulated, a higher amount of noise is added to the multibit MAC computation. However, the proposed noise-aware training scheme restores the accuracies for 1-/2-/4-bit ResNet-18 DNNs, close to the software baseline.

Noise measured at different chip voltages

The supply voltage affects the analog IMC operation. XNOR-SRAM measurements reported that higher supply voltages worsened the IMC noise [3], due to a higher IR drop on V_{BL}. Using the XNOR-SRAM measurements at supply voltages of 0.6, 0.8, and 1.0 V, we performed hardware noise-aware training.

Figure 3c shows that the noise-aware IMC accuracy is better than the conventional IMC accuracy in all three supply voltages for binary ResNet-18. IMC accuracy degrades rapidly as the IMC noise worsens, but the proposed noise-aware training largely recovers this severe accuracy loss.

Noise from different chips

We performed the same noise-aware DNN training for binary ResNet-18 by using five different noise probability tables, obtained from five different XNOR-SRAM chips at 0.6 V supply. Table 1 shows the results for 1-, 2-, and 4-bit ResNet-18, where the noise-aware IMC inference achieves consistently higher accuracy than conventional IMC inference across all five chips.

Ensemble of noise from different chips

We also obtained an ensemble probability table by combining the probability data from five XNOR-SRAM chips. To achieve this, 100,000 random samplings of ADC outputs were performed from each chip's probability table for random inputs, and the new ensemble probabilities from the pool of 500,000 samplings were obtained. This ensemble probability table represents a more generalized version of the hardware noise and allows us to test the performance of DNNs when trained with IMC noise averaged from multiple chips.

We trained 1-/2-/4-bit ResNet-18 DNNs by injecting the ensemble IMC noise from five chips and then evaluated the inference by: 1) using each individual chip's probability table and 2) using ensemble probability table from five chips, as shown in the last two columns of Table 1. We performed five inference evaluations for each experiment, and the mean of the five inference accuracies and the average deviation from the mean are reported.

For individual chip's IMC inference, using the trained DNN model with each chip's IMC noise injection shows the best accuracy in Table 1. Employing one ensemble DNN model trained with many chips' noise data could mitigate the chipwise training overhead, while slight accuracy degradation occurs compared to the DNNs trained with individual chip's noise. If we use ensemble noise for inference, the DNN accuracy improves to the level of individual

Table 1. IMC inference accuracies for 1-/2-/4-bit ResNet-18 on CIFAR-10 for different noise-aware training experiments. (a) 1-bit ResNet-18. (b) 2-bit ResNet-18. (c) 4-bit ResNet-18.

(a) 1-bit ResNet-18

Baseline Binary ResNet-18 CIFAR-10 Accuracy: 89.24 ± 1.05 %				
Training Inference	**Conventional IMC** Baseline Individual Chip	**Noise-aware IMC** Individual Chip Individual Chip	**Noise-aware IMC** Ensemble (5 Chips) Individual Chip	**Noise-aware IMC** Ensemble (5 Chips) Ensemble (5 Chips)
Chip 1	85.24% ± 0.29%	88.11% ± 0.61%	87.26% ± 0.71%	
Chip 2	86.15% ± 0.32%	87.63% ± 0.64%	87.32% ± 0.65%	
Chip 3	86.3% ± 0.41%	88.40% ± 0.56%	87.36% ± 0.74%	88.74% ± 0.42%
Chip 4	85.72% ± 0.31%	88.32% ± 0.42%	87.65% ± 0.38%	
Chip 5	84.58% ± 0.52%	88.36% ± 0.61%	88.05% ± 0.62%	
Average	85.60% ± 0.37%	88.16% ± 0.57%	87.53% ± 0.62%	88.74% ± 0.42%

(b) 2-bit ResNet-18

Baseline 2-bit ResNet-18 CIFAR-10 Accuracy: 90.24 ± 0.53 %				
Training Inference	**Conventional IMC** Baseline Individual Chip	**Noise-aware IMC** Individual Chip Individual Chip	**Noise-aware IMC** Ensemble (5 Chips) Individual Chip	**Noise-aware IMC** Ensemble (5 Chips) Ensemble (5 Chips)
Chip 1	84.13% ± 0.32%	88.14% ± 0.72%	88.54% ± 0.64%	
Chip 2	84.28% ± 0.28%	88.34%± 0.43%	87.15% ± 0.73%	
Chip 3	84.45% ± 0.27%	88.29% ± 0.58%	88.26% ± 0.63%	88.94% ± 0.39%
Chip 4	84.86% ± 0.35%	88.62± 0.67%	88.05% ± 0.82%	
Chip 5	84.22% ± 0.31%	88.42± 0.48%	87.19% ± 0.78%	
Average	84.39% ± 0.30%	88.362 ± 0.57%	87.84% ± 0.72%	88.94% ± 0.39%

(c) 4-bit ResNet-18

Baseline 4-bit ResNet-18 CIFAR-10 Accuracy: 92.81 ± 0.32 %				
Training Inference	**Conventional IMC** Baseline Individual Chip	**Noise-aware IMC** Individual Chip Individual Chip	**Noise-aware IMC** Ensemble (5 Chips) Individual Chip	**Noise-aware IMC** Ensemble (5 Chips) Ensemble (5 Chips)
Chip 1	83.92% ± 0.26%	90.32% ± 0.41%	89.11% ± 0.53%	
Chip 2	83.84% ± 0.29%	90.82% ± 0.36%	88.63% ± 0.74%	
Chip 3	84.16% ± 0.33%	91.11% ± 0.31%	89.52% ± 0.58%	89.96% ± 0.52%
Chip 4	84.08% ± 0.26%	90.29 ± 0.53%	88.93% ± 0.64%	
Chip 5	84.11%± 0.37%	90.13± 0.41%	89.26% ± 0.42%	
Average	84.02% ± 0.30%	90.53±0.40%	89.09% ± 0.58%	89.96% ± 0.52%

chip's DNN (or achieves even higher accuracy for 1- and 2-bit ResNet-18 DNNs).

C3SRAM chip noise-aware training and inference

We evaluated the same DNN models as previous section using the C3SRAM chip [6] measurement results. First, we obtained the C3SRAM IMC inference accuracy for baseline ResNet-18, AlexNet, VGG, and MobileNet DNNs trained without noise injection. In addition, we performed noise-aware training for those DNNs with the probability table from C3SRAM measurements and evaluated the C3SRAM inference accuracy.

For the C3SRAM chip, we used the noise data measured at 1.0 and 0.6 V supply voltages. Unlike the XNOR-SRAM chip, where the noise is reduced as the supply voltage decreased, the noise of C3SRAM chip increases as the supply voltage is lowered. This is because, as a resistive IMC, XNOR-SRAM experiences more IR drop at higher supply voltages where the current is large [3]. On the other hand, C3SRAM is a capacitive IMC based on capacitive coupling, so it is not affected by IR drop much, but the V_{BL} range

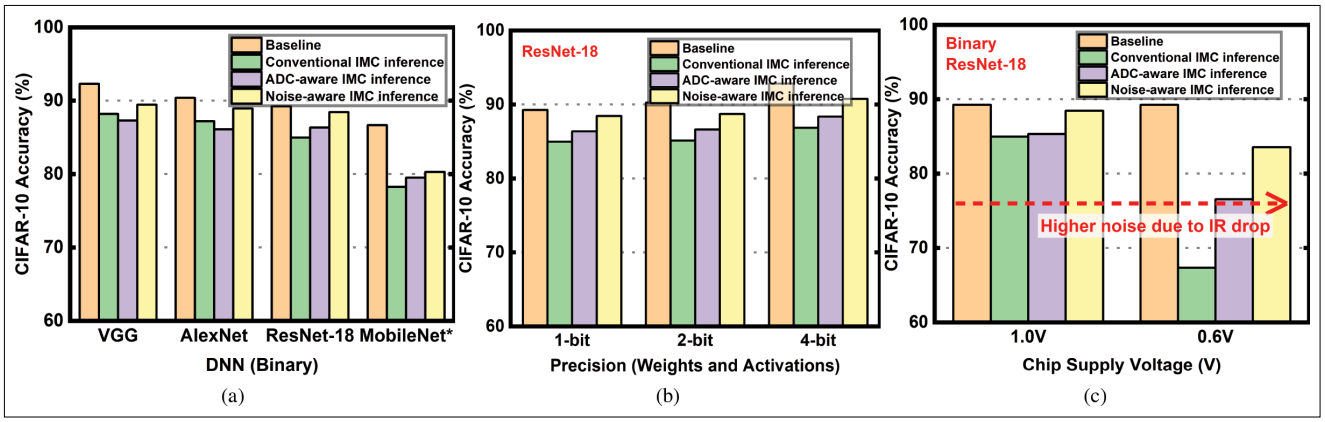

Figure 4. IMC inference accuracy after hardware noise-aware training of (a) different DNN topologies (*MobileNet only binarized convolution layers), (b) different activation/weight precisions for ResNet-18 with C3SRAM chip at 1.0 V, and (c) different C3SRAM supply voltages for ResNet-18 DNN.

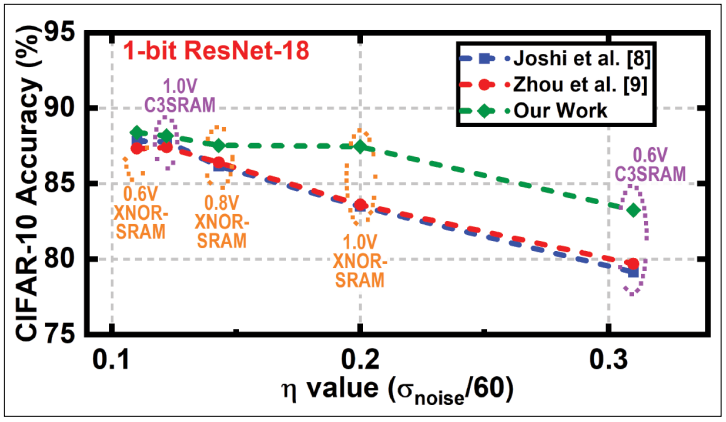

Figure 5. Comparison of our work to [7] and [8] using equivalent noise used for binary ResNet-18 training and inference.

linearly decreases at lower supply voltages, limiting the ADC functionality [6].

Figure 4a shows the IMC hardware inference accuracy improvements obtained for all four DNNs after performing noise-aware training using the C3SRAM chip measurements at 1.0 V supply with low noise. For binary ResNet-18, the IMC hardware accuracy was improved by 3.8% from 84.94% before noise-aware training to 88.74% after noise-aware training.

Figure 4b shows the noise-aware DNN training and inference results on 1-, 2-, and 4-bit ResNet-18 DNNs. In all three cases, the proposed scheme is able to restore the IMC inference accuracy very close to the software baseline, while 4-bit ResNet-18 shows the highest 5.72% accuracy improvement compared to the conventional IMC scheme.

Figure 4c shows the effect of supply voltage of the C3SRAM chip on the IMC accuracy for binary ResNet-18. When a 0.6-V supply with high noise is used for conventional IMC inference without noise-aware training, considerable accuracy degradation of 20.1% is observed. Using the proposed IMC noise-aware training, the DNN accuracy substantially improved from 67.35% to 83.55%.

Comparison to relevant works

We also compared the performance of our work with two relevant works [7], [8]. In an attempt to make an apple-to-apple comparison, we performed noise-aware training using the approaches proposed by each scheme and evaluated DNN inference, where all three schemes employed the same noise data from the XNOR-SRAM chip measurements.

For example, to compare the performance at 0.6 V XNOR-SRAM noise, we performed noise-aware training using the $\eta_{tr} = \eta_{inf}$ combination with a value of 0.11 for the work of [7] and a value of 0.058 for the work of [8]. These values were chosen so that the noise remains the same during training and inference. The maximum and minimum MAC values on which noise is applied (corresponding to the XNOR-SRAM design [3]) are +60 and –60, respectively. If we substitute these values into the noise formula of $\sigma_{noise}/W_{max} = \eta$ provided by [7] and the noise formula of $\sigma^{l}_{noise} = \eta \times (W^{l}_{max} - W^{l}_{min})$ provided by [8], we obtain the aforementioned η values of 0.11 and 0.058.

Figure 5 shows the IMC inference accuracy comparison results for binary ResNet-18 DNNs, which are

trained with three noise models of the XNOR-SRAM chip measured at 0.6, 0.8, and 1.0 V, and two noise models of the C3SRAM chip measured at 0.6 and 1.0 V. Compared to [7] and [8] that used the equivalent amount of noise, our work results in better inference accuracy, especially when the noise amount is high, for example, XNOR-SRAM at 1.0 V and C3SRAM at 0.6 V.

In this work, we presented a new hardware noise-aware DNN training scheme to improve the DNN inference accuracy of IMC hardware. During DNN training, noise injection is performed at the partial sum level, and the injected noise is based on IMC chip measurements. We validated our proposed scheme across different DNN models and precisions, by using measured noise at different supply voltages from multiple chips of two different IMC prototypes. We also examined the effectiveness of using an ensemble of noise from multiple chips. The degraded accuracy of conventional IMC hardware is largely recovered for all experiments that we evaluated by using the proposed noise-aware training, especially when the IMC hardware noise is high. ■

Acknowledgments

This work was supported in part by NSF under Grant 1652866 and Grant 1919147, and in part by C-BRIC, one of the six centers in JUMP, an SRC Program sponsored by DARPA.

References

[1] J. Choi et al., "Accurate and efficient 2-bit quantized neural networks," in *Proc. Conf. Mach. Learn. Syst. (MLSys)*, 2019, pp. 1–12.

[2] B. Zimmer et al., "A 0.32–128 TOPS, scalable multi-chip-module-based deep neural network inference accelerator with ground-referenced signaling in 16 nm," *IEEE J. Solid-State Circuits*, vol. 55, no. 4, pp. 920–932, Apr. 2020.

[3] S. Yin et al., "XNOR-SRAM: In-memory computing SRAM macro for binary/ternary deep neural networks," *IEEE J. Solid-State Circuits*, vol. 55, no. 6, pp. 1733–1743, Jun. 2020.

[4] Q. Dong et al., "A 351 TOPS/W and 372.4 GOPS compute-in-memory SRAM macro in 7 nm FinFET CMOS for machine-learning applications," in *IEEE Int. Solid-State Circuits Conf. (ISSCC) Dig. Tech. Papers*, Feb. 2020, pp. 242–244.

[5] S. K. Gonugondla, M. Kang, and N. R. Shanbhag, "A variation-tolerant in-memory machine learning classifier via on-chip training," *IEEE J. Solid-State Circuits*, vol. 53, no. 11, pp. 3163–3173, Nov. 2018.

[6] Z. Jiang et al., "C3SRAM: An in-memory-computing SRAM macro based on robust capacitive coupling computing mechanism," *IEEE J. Solid-State Circuits*, vol. 55, no. 7, pp. 1888–1897, Jul. 2020.

[7] V. Joshi et al., "Accurate deep neural network inference using computational phase-change memory," *Nature Commun.*, vol. 11, no. 1, pp. 1–13, Dec. 2020.

[8] C. Zhou et al., "Noisy machines: Understanding noisy neural networks and enhancing robustness to analog hardware errors using distillation," 2020, *arXiv:2001.04974*.

[9] W. Wei et al., "A relaxed quantization training method for hardware limitations of resistive random access memory (ReRAM) based computing-in-memory," *IEEE J. Explor. Solid-State Computat. Devices Circuits*, vol. 6, no. 1, pp. 45–52, Jun. 2020.

[10] K. H. Lee et al., "Two-stage noise aware training using asymmetric deep denoising autoencoder," in *Proc. IEEE Int. Conf. Acoust., Speech Signal Process. (ICASSP)*, Mar. 2016, pp. 5765–5769.

[11] C. Xue et al., "A 22 nm 2 Mb ReRAM compute-in-memory macro with 121-28 TOPS/W for multibit MAC computing for tiny AI edge devices," in *IEEE Int. Solid-State Circuits Conf. (ISSCC) Dig. Tech. Papers*, Feb. 2020, pp. 244–246.

[12] I. Hubara et al., "Binarized neural networks," in *Proc. Adv. Neural Inf. Process. Syst.*, 2016, pp. 4107–4115.

Sai Kiran Cherupally is pursuing a PhD with the School of Electrical, Computer and Energy Engineering, Arizona State University, Tempe, AZ, USA. His research interests are machine learning-assisted hardware security and developing defenses against adversarial attacks using noise-injection-based DNN optimization. He is a Student Member of IEEE.

Jian Meng is pursuing a PhD with the School of Electrical, Computer and Energy Engineering, Arizona State University, Tempe, AZ, USA. His research interests are DNN compression optimization, hardware–software codesign with NVM technologies, and event-based object detection. He is a Student Member of IEEE.

Adnan Siraj Rakin is pursuing a PhD with the School of Electrical, Computer and Energy Engineering, Arizona State University, Tempe, AZ, USA. His research interests are secure deployment of DNNs, exploring the attack and defense of adversarial examples and weight attack domain, and efficiency of machine learning algorithms. He is a Student Member of IEEE.

Shihui Yin is a Senior Research Engineer at Huawei Technologies, Beijing, China. His research interests include SRAM/NVM-based IMC and energy-efficient intelligent hardware design. Yin has a PhD from the School of Electrical, Computer and Energy Engineering, Arizona State University, Tempe, AZ, USA. He is a Member of IEEE.

Mingoo Seok is an Associate Professor with the Department of Electrical Engineering, Columbia University, New York, NY, USA. His research interests include VLSI hardware with the foci given to energy efficiency, artificial intelligence, and hardware security. He is a Senior Member of IEEE.

Deliang Fan is an Assistant Professor with the School of Electrical, Computer and Energy Engineering, Arizona State University, Tempe, AZ, USA. His research interests include efficient DNN algorithm/hardware design and security of AI systems. He is a Member of IEEE.

Jae-Sun Seo is an Associate Professor with the School of Electrical, Computer and Energy Engineering, Arizona State University, Tempe, AZ, USA. His research interests are energy-efficient hardware design for deep learning and neuromorphic computing. He is a Senior Member of IEEE.

■ Direct questions and comments about this article to Jae-Sun Seo, Arizona State University, Tempe, AZ 85287 USA; jaesun.seo@asu.edu.