A Case for Emerging Memories in DNN Accelerators

Avilash Mukherjee*, Kumar Saurav†, Prashant Nair*, Sudip Shekhar*, and Mieszko Lis*
*The University of British Columbia, Vancouver, Canada †QUALCOMM India
avilash@ece.ubc.ca, saurav@qti.qualcomm.com, {prashantnair,sudip,mieszko}@ece.ubc.ca

***Abstract*—The popularity of Deep Neural Networks (DNNs) has led to many DNN accelerator architectures, which typically focus on the on-chip storage and computation costs. However, much of the energy is spent on accesses to off-chip DRAM memory. While emerging resistive memory technologies such as MRAM, PCM, and RRAM can potentially reduce this energy component, they suffer from drawbacks such as low endurance that prevent them from being a DRAM replacement in DNN applications.**

In this paper, we examine how DNN accelerators can be designed to overcome these limitations and how emerging memories can be used for off-chip storage. We demonstrate that through (a) careful mapping of DNN computation to the accelerator and (b) a hybrid setup (both DRAM and an emerging memory), we can reduce inference energy over a DRAM-only design by a factor ranging from 1.12× on EfficientNetB7 to 6.3× on ResNet-50, while also increasing the endurance from 2 weeks to over a decade. As the energy benefits vary dramatically across DNN models, we also develop a simple analytical heuristic solely based on DNN model parameters that predicts the suitability of a given DNN for emerging-memory-based accelerators.

***Index Terms*—Machine-Learning, Convolutional Neural Networks, Non-Volatile Memories, PCM, RRAM, MRAM**

I. Introduction

Deep Neural Network (DNN) models are extremely compute- and memory-intensive, requiring hundreds of MBs of memory and 50-90K multiply and accumulate operations (MACs) per fetched model weight [1–3]. Accelerators used exclusively for inference reduce energy even further by using short fixed-point values [4]. While this reduces the computation energy, the total inference energy now becomes dominated by memory accesses, since each memory access costs up to two orders of magnitude more energy than a MAC operation for the same bitwidth [5].

DNN accelerators [4, 6, 7] attempt to reduce the memory access energy through an on-chip memory hierarchy. Since on-chip SRAM accesses tend to cost an order of magnitude less energy than off-chip DRAM accesses [5], energy can be saved if memory blocks fetched from DRAM can be reused for multiple computations. For maximizing reuse, a typical DNN accelerator has three levels of memory hierarchy: off-chip DRAM, an on-chip SRAM global buffer, and per-PE register files (RFs) (Fig. 1(a)). However, even with the conventional memory hierarchy, for a ResNet-50 [1] inference, 80% of energy is consumed by off-chip DRAM accesses (Fig. 1(b)).

In this paper, we investigate the potential for reducing memory access energy for DNN accelerators by using emerging memories (eMEMs). These offer much lower read energies, but suffer from low write endurance and bandwidth (BW). Naïvely using eMEMs as a DRAM replacement would result in DNN accelerator chips that have lifetime of a few weeks (Fig. 1(d)).

†Work done while the author was with the University of British Columbia.

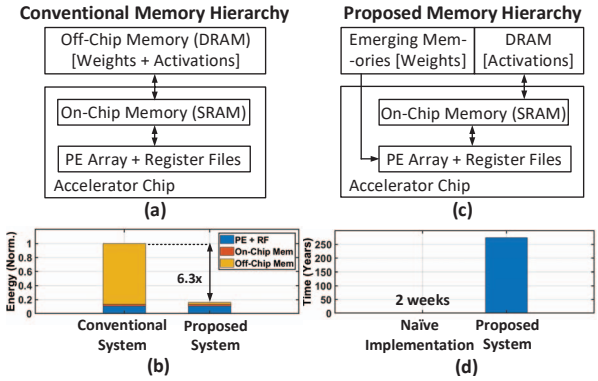

Fig. 1. (a) Conventional memory hierarchy in DNN accelerators — DRAM stores both weights and activations. (b) Energy consumed across the memory hierarchy shows that DRAM consumes more than 80% of the energy in the conventional system for ResNet-50 inference, which we reduce by 6.3× by leveraging low read energy of eMEMs. (c) Proposed memory hierarchy, utilizing a hybrid memory setup to store weights and activations separately. (d) The lifetime of the system on naive usage of eMEM is only two weeks — increasing indefinitely upon utilizing a hybrid memory setup.

We therefore propose a hybrid design that uses a combination of eMEM and DRAM (Fig. 1(c)) as storage for a digital DNN accelerator (in contrast to prior uses of eMEMs for analog in-memory compute [8]). By carefully scheduling the DNN computations, we minimize the writes into the low-endurance eMEM modules. This yields up to 6.3× lower inference energy (Fig. 1(b)) while maintaining chip endurance on the order of decades (Fig. 1(d)). Because the benefits depend on the DNN architecture, we also develop an analytical heuristic based solely on DNN model parameters that predicts the suitability of deploying the model in a hybrid-memory accelerator.

II. Memory Technologies

Owing to its high density and BW, DRAM is commonly employed as the main memory for computing systems. As DRAM accesses are energy-inefficient [5], designers generally create a hierarchy with intermediate memory technologies like SRAM that offer much lower access energies [7].

On the other hand, *resistive* memories use high or low resistance to differentiate stored values and offer low read access energy. MRAM [11, 19, 24] utilizes the magnetic configuration of two ferromagnetic layers; RRAM [13, 21, 23, 25] relies on the formation of conductive filaments in the insulator between two electrodes; and PCM [12, 17, 20] uses chalcogenide materials that switch between crystalline and amorphous resistance states.

Table I compares the best performance measurements reported for each memory technology. MRAM has the lowest read energy, followed by RRAM and PCM; however, the read

TABLE I
COMPARISON OF DIFFERENT MEMORY TECHNOLOGIES

	SRAM	DRAM	eF	MRAM	PCM	RRAM
Write Energy/Bit* (pJ)	0.75 [9]	20 - 40.625 [5]	122 [10]	4.5 [11]	32.7 [12]	7 [13]
Read Energy/Bit* (pJ)	0.954 [9]	20 - 40.625 [5]	2.2 [10]	0.7 [11]	2 [12]	2 [13]
Write Bandwidth (GB/s)	167 [14]	12.8 [15]	0.002 [16]	0.1 [11]	0.001 [17]	1.5 [13]
Read Bandwidth (GB/s)	167 [14]	12.8 [15]	6.25 [16]	0.7 [11]	0.833 [17]	8 [13]
Memory Size (MB)	0.128 [9]	16000 [15]	0.032 [10] - 24 [18]	0.125 [11] - 1024 [19]	0.125 [17] -1024 [20]	2 [13] - 2048 [21]
Area (mm^2)	0.164 [9]	24.192 [15]	0.85 [10] - 8.64 [18]	0.214 [11] - 107.5 [19]	3 [17] - 59.409 [20]	0.0212 [13] - 168 [21]
Cell Size (F^2)	100 [22]	6 [22]	4 - 57.39 [18, 22]	9-75 [11, 19]	4-12 [12, 20]	6 [21]
Endurance (# Writes Cycles)	10^{16} [22]	10^{16} [22]	10^5 [16]	10^8 [11]	$10^8 - 10^9$ [12, 22]	$10^6 - 10^7$ [22, 23]

*Includes macro access energies for all memories. DRAM energy also includes data transfer which accounts for 10-20% of total energy [15]. For fair comparison across all memories, we exclude the data transfer energy from DRAM in our experiments.

BW and endurance of these technologies are significantly worse than SRAM and DRAM. This means that substantial energy savings are on offer *provided* the attendant endurance, bandwidth, and latency challenges can be addressed.

III. Methods

A. DNN Layers and Model Selection

We first examine how commonly used DNN layers, like convolutional (CONV) layers, depthwise convolutional layers (dw-CONV), as well as fully connected (FC) layers can take advantage of eMEMs.

CONV layers are the most common layer in DNNs; the computation can be thought of as a seven-dimensional loop over the following dimensions (using the terminology in [26]): the output tensor dimensions ***P, Q***, the input and output tensor channel counts ***C, K***, the dimensions of the weight kernel ***R, S***, and the number of inputs in the batch ***N***. The weight kernel is convolved over the input image, making reuse of weights possible across the input. Input activations can be reused across different filters to produce different output channels. The maximum possible weight reuse is $P \times Q \times N$, while the maximum possible input reuse is $R \times S \times K$.

dw-CONV layers are used in models for mobile applications [2, 3, 27]. Each kernel is separately applied to each channel rather than across all input channels, saving significant computation but offering far less input reuse.

FC layers are often used as the classifier at the end of DNNs after feature extraction using CONV layers. FC layers reuse weights only across the batch size dimension N and reuse inputs only across the output dimension K.

We selected the representative DNN models by considering top-1 accuracy on ImageNet [28] together with the compute cost. We chose two types of models on the accuracy/#MACs Pareto frontier: in the high-performance category, we chose EfficientNets [3] and ResNet-50 [1]; in the mobile-computing category, we chose ShuffleNet [27] and MobileNet v2 [2]. We used the full models (all layers) for all experiments.

B. Chip Architecture

We modeled a DNN accelerator similar to the Eyeriss architecture [6]: a 2D array of 14×24 processing elements, each containing a single 16b fixed-point MAC unit and separate register files (RFs) for weights (64B W-Reg), input activations (24B InReg), and partial sums (16B PsumReg), together with a 512 kB on-chip SRAM global buffer (GBuF) shared among the PEs for long-term reuse.

The off-chip memory consists of only DRAM in the baseline model (Fig. 1(a)) and a combination of eMEM and DRAM in the hybrid configurations (Fig. 1(c)). We choose the size of eMEM based on the DNN workload, ranging from 4.8 MB for ShuffleNet [27] to 132 MB for EfficientNetB7 [3].

C. Simulation Environment

We used Accelergy [29] to model the hardware and the corresponding energies and BW for the MAC operations and reads/writes to the main memories, GBuF, and RF. To determine the optimal scheduling and use of the available storage, we used Timeloop [26]. We also used Timeloop with the Accelergy model to obtain the energy costs and inference latencies.

IV. Using Emerging Memories in DNN Accelerators

A. Endurance and Architectural Constraints

Because the eMEMs offer significantly lower read energies than DRAM, a designer might be tempted to directly replace DRAM with eMEM of choice as the off-chip memory. However, this means that the eMEM will store both weights and activations, and will burn out quickly: e.g., ShuffleNet [27] needs 7.8M off-chip writes, and will last only a few weeks (Fig. 2).

To be effectively employed, therefore, eMEMs must be read often but written relatively rarely. This consideration motivates the overall architecture shown in Fig 1(c): the off-chip memory is divided into an eMEM weight memory (W-Mem) and a DRAM activation memory (Act-Mem).

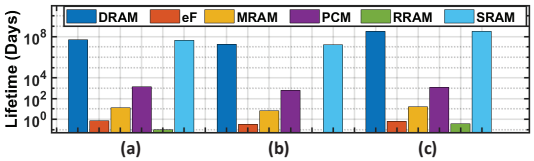

Fig. 2. The lifetime of memory technologies under naive replacement of DRAM for (a) ShuffleNet, (b) MobileNet v2, and (c) ResNet-50 workloads assuming continuous operation.

B. Layer Types and Workload Mappings

Because our design uses eMEMs to store weights, energy savings depend heavily on how many accesses are made to W-Mem compared to Act-Mem. Below, we examine the common DNN layer types and show how to schedule them to support eMEM usage.

*1) **Fully-Connected Layers**:* Because FC layers have a separate weight per combination of input and output activation, they have significantly more weights than activations.

Since all weight reads access external memory, nearly all inference energy goes towards reading the W-Mem. Fig. 3 shows that replacing DRAM with any of the eMEM options therefore offers more than an order of magnitude energy savings.

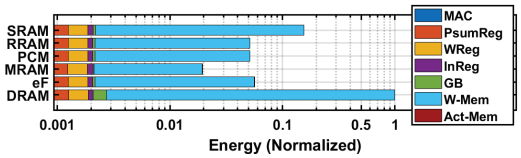

Fig. 3. Energy across different memory levels for an 1536–1000 (input-output) FC layer for different W-Mem, normalized to DRAM. We obtain >95% energy savings when using eMEM in place of DRAM. Note that the SRAM energy is comparatively higher due to the large size of W-Mem required.

*2) **CONV layers with high weight counts**:* For typical image-processing DNNs, relatively few MACs are expended in the initial layers since there are only tens of channels. In contrast, the later layers have many more channels (e.g., 1920 in stage 7 in EfficientNet-B0 [3]), with many more weights than input activations: for example, the conv5-1x1-512-s2 layer in ResNet-50 has 524,288 weights and 50,176 input activations.

Consequently, implementing W-Mem with an eMEM offers substantial savings: the conv5-1x1-512-s2 layer on a hybrid MRAM+DRAM accelerator takes 2.2× less energy than a DRAM-only accelerator using the same dataflow (Fig. 4(c)).

However, a hybrid memory setup allows even more savings since reading one activation from the Act-Mem (DRAM) costs the same energy as multiple weights from W-Mem. Therefore, we propose a *weight-streaming* dataflow, which ensures that activations are only accessed once even if weights must be accessed many more times. While this would consume *more* energy in a DRAM-only accelerator (Fig. 4(b)), in the hybrid accelerator this mapping saves 10% more energy over the DRAM-optimal mapping (Fig. 4(d)), as it decreases GBuF and Act-Mem energies more than it increases the W-Mem energy.

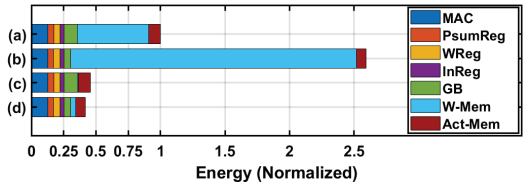

Fig. 4. Energy distribution across different memory levels for conv5-1x1-512-s2 in ResNet-50: (a) DRAM with DRAM-optimal dataflow; (b) DRAM with weight streaming dataflow; (c) MRAM with DRAM-optimal dataflow; (d) MRAM with weight streaming dataflow. Energy normalized to (a).

*3) **CONV layers with low weight counts**:* These layers appear at the start of the DNNs, where the input dimensions are large, but there are relatively few channels. In these layers, W-Mem accesses do not dominate the overall energy, and, using an eMEM for weights yields limited savings of up to 14%.

*4) **Depthwise-CONV Layers**:* Since each filter of a dw-CONV layer acts only on a single channel of the input activations, these layers have few weights, and W-Mem accesses consume an insignificant part of the total energy; savings from using an eMEM are therefore negligible (<1%).

C. Inference Energy

Table II lists single-sample inference energies for a DRAM-only accelerator and hybrid accelerators using various eMEMs. Energy savings can be substantial for models with high reuse (e.g., up to 6.3× over DRAM on ResNet50). Models with less reuse (e.g., MobileNet v2 and EfficienNetB7) still see savings between 1.12× and 2.1×.

D. Inference Latency

The latencies of classifying a single sample on different DNN models are shown in Fig. 5. The maximum read BW demanded from W-Mem occurs for CONV layers with high weight counts in the later stages of these DNNs. MRAM and PCM become BW-bound for these layers, and hence show latency degradation compared to DRAM and SRAM (Fig. 5(a–d)).

In the case of EfficientNetB7, we observe that the maximum read BW demanded from W-Mem is 1.14GB/s; hence changing memories does not affect the latency at all (Fig. 5(e)).

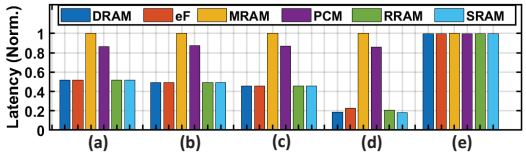

Fig. 5. Latency for processing (a) ShuffleNet, (b) MobileNet v2, (c) EfficientNetB0, (d) ResNet-50, and (e) EfficientNet7, with different W-Mems. Latency normalized to worst latency for each DNN.

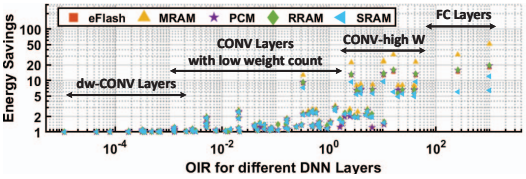

Fig. 6. Energy savings of a hybrid-eMEM vs. DRAM-only accelerator compared to the OIR heuristic.

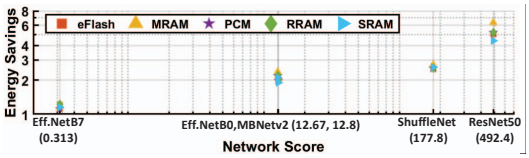

Fig. 7. Energy savings compared to DRAM versus the proposed network score. The energy savings increase with the network score, starting from 1.12× for EfficientNetB7 to 6.3× for ResNet-50.

E. Energy Heuristics

The potential benefits of eMEM hinge upon the number of weights ($R \times S \times K \times C$) compared to activations ($P \times Q \times C$) in each layer. We define the *output channel to input dimension ratio (OIR)* to predict whether a specific layer will be suitable for implementation with a hybrid-memory eMEM accelerator:

$$OIR = K/(P * Q)$$

High OIR indicates high potential savings with eMEM, while low OIR indicates limited savings. Fig. 6 demonstrates this by plotting the OIR against the actual energy savings for all layers from the chosen DNN models.

The layer-wise OIR heuristic estimates the energy savings potential of a *single* layer. To extend this to the entire DNN, we need to consider each layer's energetic contribution to the full

TABLE II
DNN INFERENCE ENERGIES WITH DRAM AND EMEMS (IN MILLIJOULES)

DNN Workloads	DRAM	SRAM	eF	MRAM	PCM	RRAM
ShuffleNet	5.33	2.05	2.12	1.96	2.10	2.09
MobileNet v2	4.75	2.49	2.41	2.29	2.39	2.39
EfficientNetB0	6.41	3.05	2.93	2.72	2.95	2.91
ResNet50	75.89	17.23	15.03	11.97	14.60	14.60
EfficientNetB7	370.8	320	330.87	328.42	330	330

model; the heuristic then becomes a weighted sum of per-layer OIRs. Using MAC counts in each layer as a proxy for its energy contribution, we obtain the network score *NS*:

$$NS = \sum_{L \in layers} \frac{K}{\{(P * Q) * MAC_L\}}$$

NS only depends on the network dimensions and layer type (e.g., CONV, FC, etc), and can hence be easily computed for a specific model. Fig. 7 shows that *NS* is a useful heuristic: it correlates well with the energy savings when using a hybrid DRAM+eMEM design compared to a DRAM-only accelerator.

F. Silicon area

We follow the cell sizes in Table I to find the implementation area for the memories. eF has not yet been implemented in large sizes (e.g., the 132 MB needed for the 66 M parameters in EfficientNetB7), but projecting from [18], 132 MB would take around 48.16 mm^2. For the same capacity, the area required for MRAM [19], PCM [20] and RRAM [21] would be 13.4375 mm^2, 7.426 mm^2, and 10.5 mm^2; meanwhile, SRAM would consume a whopping 168.96 mm^2 (13×–23× more).

V. Discussion

Overall, our results make a strong case that emerging memories (eMEMs) can significantly reduce inference energy — by up to 6.3× for a full DNN model — when used to supplement DRAM in a traditional DNN accelerator with a dataflow that maximally exploits activation reuse by streaming weights.

The key challenge of eMEMs is their limited write endurance. To address this, the hybrid accelerator architecture we propose uses an eMEM only for weights and DRAM for activations, which allows multi-decade lifetimes.

However, the benefits are heavily dependent on the potential for weight and activation reuse, and consequently on the layer geometries within the DNN. To estimate the suitability of specific models, we propose a *Network Score* heuristic that predicts a DNN's suitability for hybrid-eMEM accelerators.

VI. Acknowledgements

This material is based on research sponsored by Air Force Research Laboratory (AFRL) and Defense Advanced Research Project Agency (DARPA) under agreement number FA8650-20-2-7007. The U.S. Government is authorized to reproduce and distribute reprints for Governmental purposes notwithstanding any copyright notation thereon. The views and conclusions contained herein are those of the authors and should not be interpreted as necessarily representing the official policies or endorsements, either expressed or implied, of Air Force Research Laboratory (AFRL), Defense Advanced Research Project Agency (DARPA), or the U.S. Government.

References

[1] K. He *et al.*, "Deep Residual Learning for Image Recognition," in *CVPR*, 2016, pp. 770–778.
[2] M. Sandler *et al.*, "MobileNetV2: Inverted Residuals and Linear Bottlenecks," in *CVPR*, 2018, pp. 4510–4520.
[3] M. Tan *et al.*, "EfficientNet: Rethinking model scaling for convolutional neural networks," in *Proceedings of the 36th International Conference on Machine Learning*, 2019.
[4] N. P. Jouppi *et al.*, "In-datacenter performance analysis of a tensor processing unit," *SIGARCH Comput. Archit. News*, 2017.
[5] M. Horowitz, "Computing's energy problem (and what we can do about it)," in *ISSCC*, 2014.
[6] Y. Chen *et al.*, "Eyeriss: An Energy-Efficient Reconfigurable Accelerator for Deep Convolutional Neural Networks," *JSSC*, 2017.
[7] S. Gudaparthi *et al.*, "Wire-aware Architecture and Dataflow for CNN Accelerators," in *MICRO '52*, 2019.
[8] W. Chen *et al.*, "A 65nm 1Mb nonvolatile computing-in-memory ReRAM macro with sub-16ns multiply-and-accumulate for binary DNN AI edge processors," in *ISSCC*, 2018.
[9] G. Patrigeon *et al.*, "Design and Evaluation of a 28-nm FD-SOI STT-MRAM for Ultra-Low Power Microcontrollers," *IEEE Access*, 2019.
[10] S. Jeloka *et al.*, "An ultra-wide program, 122pJ/bit flash memory using charge recycling," in *2017 Symposium on VLSI Circuits*.
[11] Q. Dong *et al.*, "A 1Mb 28nm STT-MRAM with 2.8ns read access time at 1.2V VDD using single-cap offset-cancelled sense amplifier and in-situ self-write-termination," in *ISSCC*, 2018.
[12] B. C. Lee *et al.*, "Phase-Change Technology and the Future of Main Memory," *IEEE Micro*, 2010.
[13] K. Guo *et al.*, "RRAM Based Buffer Design for Energy Efficient CNN Accelerator," in *ISVLSI*, 2018.
[14] J. Wang *et al.*, "A Compute SRAM with Bit-Serial Integer/Floating-Point Operations for Programmable In-Memory Vector Acceleration," in *ISSCC*, 2019.
[15] "DDR4 SDRAM System-Power Calculator," Micron Technology.
[16] Y. Taito *et al.*, "A 28nm embedded SG-MONOS flash macro for automotive achieving 200MHz read operation and 2.0MB/S write throughput at T_j of 170°C," in *ISSCC*, 2015.
[17] G. De Sandre *et al.*, "A 90nm 4Mb embedded phase-change memory with 1.2V 12ns read access time and 1MB/s write throughput," in *ISSCC*, 2010.
[18] A. Kanda *et al.*, "A 24-MB Embedded Flash System Based on 28-nm SG-MONOS featuring 240-MHz Read Operations and Robust Over-the-Air Software Update for Automotive Applications," *IEEE Solid-State Circuits Letters*, 2019.
[19] K. Rho *et al.*, "A 4Gb LPDDR2 STT-MRAM with compact 9F2 1T1MTJ cell and hierarchical bitline architecture," in *ISSCC*, 2017.
[20] Y. Choi *et al.*, "A 20nm 1.8V 8Gb PRAM with 40MB/s program bandwidth," in *ISSCC*, 2012.
[21] R. Fackenthal *et al.*, "A 16Gb ReRAM with 200MB/s write and 1GB/s read in 27nm technology," in *ISSCC*, 2014.
[22] S. Yu *et al.*, "Emerging Memory Technologies: Recent Trends and Prospects," *IEEE Solid-State Circuits Magazine*, Spring 2016.
[23] G. Sassine *et al.*, "Sub-pJ consumption and short latency time in RRAM arrays for high endurance applications," in *IRPS*, 2018.
[24] Y. Chih *et al.*, "A 22nm 32Mb Embedded STT-MRAM with 10ns Read Speed, 1M Cycle Write Endurance, 10 Years Retention at 150°C and High Immunity to Magnetic Field Interference," in *ISSCC*, 2020.
[25] P. Jain *et al.*, "13.2 A 3.6Mb 10.1Mb/mm2 Embedded Non-Volatile ReRAM Macro in 22nm FinFET Technology with Adaptive Forming/Set/Reset Schemes Yielding Down to 0.5V with Sensing Time of 5ns at 0.7V," in *ISSCC*, 2019.
[26] A. Parashar *et al.*, "Timeloop: A Systematic Approach to DNN Accelerator Evaluation," in *ISPASS*, 2019.
[27] X. Zhang *et al.*, "ShuffleNet: An Extremely Efficient Convolutional Neural Network for Mobile Devices," in *CVPR*, 2018.
[28] J. Deng *et al.*, "ImageNet: A Large-Scale Hierarchical Image Database," in *CVPR*, 2009.
[29] Y. N. Wu *et al.*, "Accelergy: An Architecture-Level Energy Estimation Methodology for Accelerator Designs," in *ICCAD*, 2019.