Cross Layer Design for the Predictive Assessment of Technology-Enabled Architectures

M. Niemier, X.S. Hu, L. Liu, M. Sharifi
University of Notre Dame
Notre Dame, IN USA
{mniemier, shu, lliu24, msharif1}@nd.edu

Ian O'Connor
Ecóle Centrale de Lyon
Lyon, France
Ian.Oconnor@ec-lyon.fr

David Atienza, Giovanni Ansaloni
École Polytechnique Fédérale de Lausanne
Lausanne, Switzerland
{david.atienza, giovanni.ansaloni}@epfl.ch

Can Li
The University of Hong Kong
Hong Kong
canl@hku.hk

Asif Khan
Georgia Institute of Technology
Atlanta, GA USA
akhan40@gatech.edu

Daniel C. Ralph
Cornell University
Ithaca, NY USA
dcr14@cornell.edu

***Abstract*—There is great interest in "end-to-end" analysis that captures how innovation at the materials, device, and/or architectural levels will impact figures of merit at the application-level. However, there are numerous combinations of devices and architectures to study, and we must establish systematic ways to accurately explore and cull a vast design space. We aim to capture how innovations at the materials/device-level may ultimately impact figures of merit associated with both existing and emerging technologies that may be employed for either logic and/or memory. We will highlight how collaborations with researchers at these levels of the design hierarchy – as well as efforts to help construct well-calibrated device models – can in-turn support architectural design space explorations that will help to identify the most promising ways to use new technologies to support application-level workloads of interest. For given compute workloads, we can then quantitatively assess the potential benefits of technology-driven architectures to identify the most promising paths forward. Because of the large number of potentially interesting device-architecture combinations, it is of the utmost importance to develop well-calibrated analytical modeling tools to more rapidly assess the potential value of a given (likely heterogeneous) solution. We highlight recent efforts and needs in this space.**

***Index Terms*—Emerging logic and memory, cross-layer design; FeFETs; RRAM; design space explorations; device modeling; circuit modeling; architectural modeling; application analysis.**

I. Introduction

There is an ever-growing need to build more efficient, higher performance, and/or denser logic and memory. Representative challenges include (1) the scale of language generating artificial intelligence (AI) algorithms, where models like GPT-3 must learn/store 175B parameters [1]; (2) training costs for foundational models, which are approaching 10^{24} compute FLOPS, with training times doubling every 10 months [2]; (3) the computational complexity of homomorphic encryption for privacy preserving computation, where one ciphertext may require megabytes of storage, and training for a 7-layer network may require years on conventional hardware [3].

This work was supported in part by (1) ASCENT, one of six centers in JUMP, a Semiconductor Research Corporation (SRC) program sponsored by DARPA; (2) SUPREME, one of seven centers in JUMP 2.0, a Semiconductor Research Corporation (SRC) program sponsored by DARPA; (3) National Science Foundation grant 1640081; (4) National Science Foundation grant 2212239; (5) EC H2020 WiPLASH (Ga. No. 863337), (6) the EC H2020 FVLLMONTI (Ga. No. 101016776), (7) Swiss NSF ML-Edge (Ga. No. 182009); (8) Early Career Scheme (Grant No. 27210321) from Hong Kong SAR RGC, (9) NSFC Excellent Young Scientist Fund (HK&Macau) (Grant No. 62122005)

When looking toward technology-driven solutions to address the above challenges, a new device may (1) replace an existing technology in an existing architecture (e.g., a new, faster, denser, or more energy efficient memory cell in a traditional array), or (2) serve as an "enabler" of a new circuit architecture and/or compute functionality (e.g., a new memory cell that can natively support an important and/or ubiquitous application-level compute kernels, thereby obviating the need for less efficient conventional hardware and/or excessive data transfer).

Fig. 1 illustrates (1) what is ultimately a large and complex design space, in addition to (2) infrastructure needed to accurately explore it. Numerous research efforts aim to develop new/enhanced logic and memory devices (Fig. 1A) that are superior to the existing state-of-the-art (SOA) across various figures of merit (FOM). Said devices may be used in support of existing computer architectures (e.g., CPUs, GPUs, or TPUs in Fig. 1B), and potentially to realize emerging computer architectures – examples of which include but are not limited to: in-memory computing (IMC) where memory and logic are closely coupled to mitigate expensive data transfer; crossbar arrays to perform efficient matrix-vector multiplication (MVM); associative memory (AM) arrays for efficient search, etc. (again see Fig. 1B). For a given application (Fig. 1C), while some design points may inherently be eliminated (e.g., (a) while flash memory is dense, high write latencies make it ill-suited as main memory for a CPU or GPU, (b) GPUs may be a better baseline for MVM workloads than a CPU, etc.), ultimately there are many combinations of devices/architectures that should be studied to identify the mapping that best meets the computational needs of a given application. Moreover, for a given problem different algorithms may also be employed. Indeed, per Fig. 1D, multi-layer perceptron (MLP) networks, convolutional neural networks (CNNs), hyperdimensional computing (HDC) models, etc. can all perform a classification task [4], but the *computations* associated with said models may be fundamentally different (e.g., CNNs are dominated by MVMs, while HDC is search-based), and are thus better supported by different architectural mappings (e.g., crossbars for MVM and AMs for search), which in turn are better supported by different technologies (e.g., two-terminal resistive memories for crossbars, and three-terminal FeFETs for AMs).

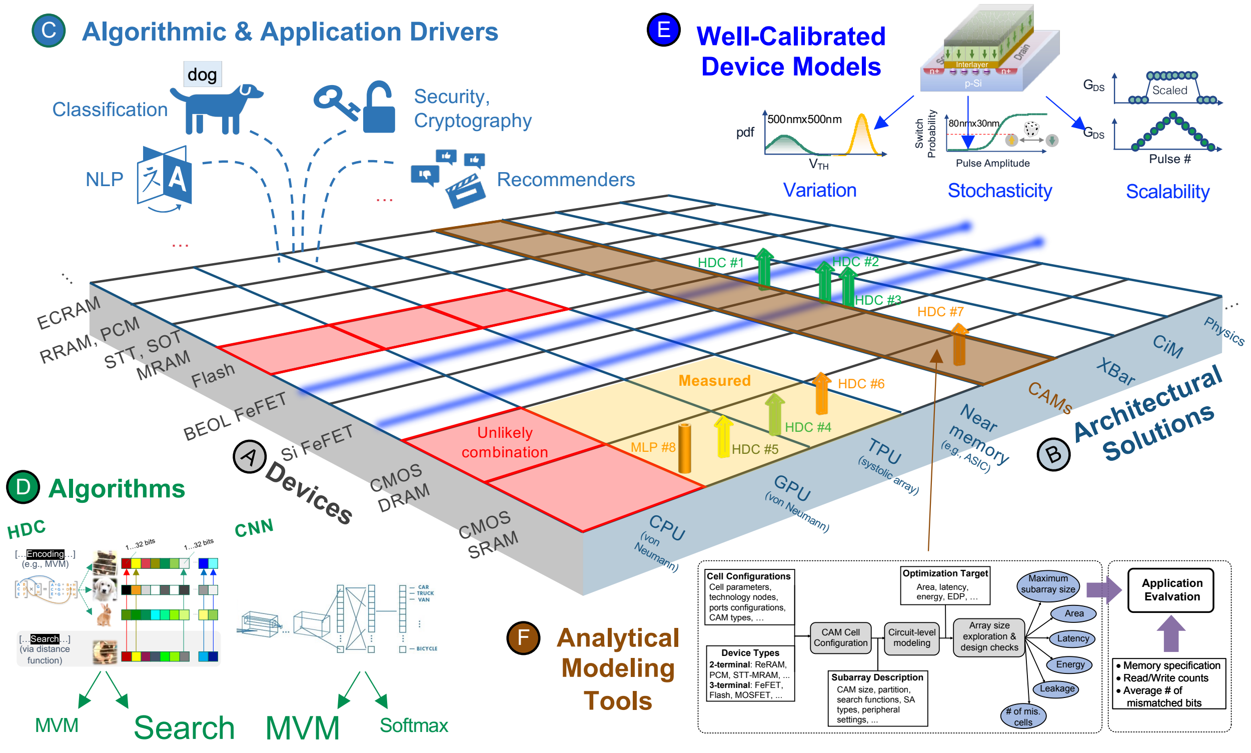

Fig. 1. Technology-circuit-architecture-algorithm-application design space. There are numerous combinations of devices (A), architectures (B), applications (C), and algorithmic possibilities (D) to explore. Explorations must be supported by device models (E) and analytical modeling tools (F) among others.

To properly explore this vast design space, cross-stack modeling efforts are essential. More specifically (1) well-calibrated device models are needed to assess how a given technology may fare when (a) used to construct logic circuits or memory cells, and (b) how variations/non-idealities impact the correctness and fidelity of a circuit or memory cell. Said evaluations will in-turn inform analysis at the architecture/application-levels where (2) existing simulation frameworks such as NVSim [5], NeuroSim [6], etc. may be employed. Among others, questions of interest include but are not limited to: (a) can algorithmic workloads be accelerated by technology-enabled circuit solutions – at iso-accuracy/correctness versus software-based CPU/GPU systems? and (b) how is memory array performance impacted by a new (perhaps multi-level) device? Using ferroelectric field effect transistors (FeFETs) as an example, device modeling efforts [7] will crosscut studies at the circuit and architectural-level per Fig. 1E. Finally, as there are numerous combinations of devices, architectures, algorithms, and applications to consider, (3) a well-validated, analytical modeling/evaluation infrastructure (Fig. 1F) is necessary to rapidly and accurately ”triage” technology-enabled architectures, and prioritize the most promising options for more in-depth investigations.

This paper will (1) outline strategies/best practices to efficiently explore a design space like that illustrated in Fig. 1, (2) identify research needs to accurately accelerate design space exploration (DSE) efforts, and (3) highlight ways in which the design automation community can best support said efforts.

After reviewing relevant technology/architecture background in Sec. II, we will begin by discussing two representative case studies where technology-enabled architectures are evaluated in the context of two different machine learning (ML)/AI models – i.e., HDC algorithms supported by FeFET-based AMs and crossbar arrays [4] (Sec. III), and few-shot learning models supported by RRAM-based crossbars that can universally realize MVM for CNNs, hashing, and AM functions that form the computational workload [8] (Sec. IV). Among others, said case studies illustrate: (1) how device models may be employed; (2) the need to assess different aspects of the design stack – e.g., as degradations from iso-accuracy may stem not just from device variation, but from architectural-level design constraints as well; and (3) the need for comprehensive benchmarking when employing a different algorithmic models (e.g., MLP versus CNN versus HDC), as well as heterogenous architectural solutions for said models (e.g., TPU-GPU hybrids) that may ultimately represent the ideal baseline/software-based solution. At a high-level, the presented case studies illustrate what analysis is needed to determine if industrial investment in a technology-enabled architecture is justifiable, while simultaneously serving as motivation for both (i) architectural modeling efforts and (ii) analytical modeling tools to triage a large design space and identify the most meaningful/plausible points of comparison for a technology-driven architectural solution.

Modeling infrastructure that facilitates the evaluation of integrated, heterogeneous architectures is nascent [9]. Ultimately, we must (1) capture the utility of technology-driven architectures (Fig. 1A, B), (2) determine value from architectural heterogeneity (e.g., the XBar-CAM and GPU-TPU hybrids considered in case studies presented in Secs. III and IV), and (3) determine the best architectural solution in the context of an application-level workload. In this regard, In Sec. V we highlight two complementary approaches. The first is based on open hardware modular platforms [10], which offers a path for the integration of novel technology-enabled solution in systems-on-chip, in order to prototype them and their derived benefits from an the standpoint of an entire application. The use of open hardware modules provides for the reuse of validated hardware components (processors, memories and peripherals) surrounding the accelerator itself. They also afford the compiler infrastructure to map applications expressed in high-level languages on to the resulting platforms. A second approach is based on system simulation [11], which elevates the level of abstraction to the scale of complex systems and software stacks comprised of full-fledged operating systems such as Linux. While systems defined in this way cannot be readily implemented as integrated circuits, the insights gathered by system simulation will provide early and valuable insights on the expected speedup of acceleration, in advance of detailed hardware design.

Then, in Sec. VI, we discuss ongoing work to develop analytical modeling tools that can be employed to accelerate/triage the technology-driven design space exploration efforts captured by Fig. 1. We will briefly summarize/catalog existing approaches with respect to conventional memory [12], content addressable memories [13], crossbars [6], and in-memory computing architectures [14]. These tools can identify the most promising options for deep dives. As before, we present a more detailed case study with respect to analytical modeling tools for AMs.

The paper concludes with a discussion of (1) top-down/bottom-up strategies for evaluating the technology-architectural design space with respect to an application's compute needs, and (2) other aspects of the design space that may be included for evaluations that are of interest to industry.

II. Background

For a self-contained discussion, we review device concepts, IMC architectures, and design tools fundamental to this work.

A. Device Concepts

Ferroelectrics offer unique possibilities for the design of ultra-dense, low-leakage and fast random access memories (RAMs). The structure of an FeFET resembles a MOSFET, except that an additional layer of ferroelectric (FE) oxide is deposited in the gate stack. Due to the coupling between the FE and CMOS capacitances, the threshold (turn-on) voltage of a device can be shifted. This effect can be used to (non-volatilely) store information in the FeFET. FeFETs can store multiple V_{th} levels through partial polarization switching of the FE layer [4]. Silicon FeFETs require relatively high write voltage pulses, may have limited write endurance, and a large read-after-write latency. Low voltage, high speed memory operations with high write endurance have been demonstrated using back end of line (BEOL) compatible FeFET devices by eliminating the defective interlayer between the FE and channel [15]. BEOL FeFETs are promising for logic-compatible, high-performance on-chip memories and multi-bit cells for IMC accelerators. FeFETs could also serve as a solid state synapses for neuromorphic computing models [16], [17].

RRAM devices can store information non-volatilely [18]. An RRAM device is comprised of a top and bottom electrode (TE and BE, respectively), and typically a metal-oxide layer in between. If a voltage is applied across a given RRAM cell, resistive switching occurs. The electric field influences the generation, movement, and recombination of oxygen vacancies (in valence change RRAM) – which in turn causes conductive filaments (CFs) to form and rupture in the oxide. Filaments (or the lack thereof) between the TE and BE result in low (or high) resistance states (LRS and HRS, respectively). Filament formation/rupture is referred to as the SET/RESET process. RRAMs are CMOS compatible, and can be fabricated in the BEOL. They can also be realized in high-density 3D structures [19], which can enable monolithic 3D ICs.

B. IMC Architectures

1) Content Addressable Memories (CAMs): In conventional memory, data is retrieved from a given address. In a CAM, *data* is supplied to memory and all entries that best match a provided query can be returned. Per Fig. 2A, in a CAM, each bit of a query is XNORed with the corresponding bit of every stored entry. With a perfect match, the matchline (MaLi) does not discharge as cell matches function as "open switches". Mismatches function as "closed switches" causing the MaLi to discharge. In a ternary content addressable memory (TCAM), "don't care" states can be stored/searched and cell-level comparisons are treated as a match regardless of values. Search operations are performed directly within the memory itself in O(1) time, eliminating expensive data transfer to a compute unit. CAMs can compute a distance norm for every entry in the memory via a single search. By measuring discharge rate, we can determine a degree of match (or a *Hamming distance*) between the query/stored data [20]. Conventional, CMOS CAMs are volatile, require 16 transistors, occupy a large area and consume high power [21]. More compact/functional CAMs based on emerging technologies are desirable. We can classify AMs according to (1) data representation in a cell (binary, ternary, multi-bit, analog), and (2) type of match performed.

FeFETs may be advantageous for AMs as they have high I_{on}/I_{off} ratios, three terminals, and low read voltage. Fig. 2B shows the most compact FeFET AM cell design that was originally proposed to accomplish exact match (EX), best match (BE), and threshold match (TH) functions (see Fig. 2C) based on a HM distance. (Different match types employ different sensing circuits [22].) The same cell design can also function as a EX/BE/TH-multi-bit CAM (MCAM) when FeFETs are programmed to store multiple bits [22], which improves storage density. The FeFET BE/TH-MCAM can implement sigmoidal and squared Euclidean (SE) distance functions [22], and has utility in ML applications as will be discussed in Sec. III. This design can also function as an EX-analog CAM (ACAM) where the threshold voltage (V_{th}) values in FeFETs define either upper or lower bounds, and an analog input matches stored cell data if it is within the bounds defined by the FeFETs [21]. ACAMs can encode more information per cell than MCAMs but may suffer more from noise and variation effects. Other FeFET AM cell designs have also been considered for reducing

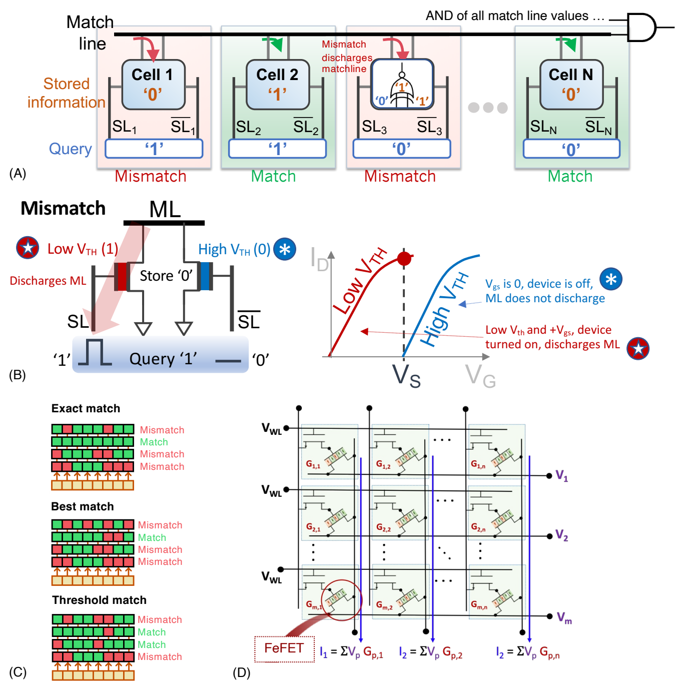

Fig. 2. **(A)** In a CAM, if query and stored data do not match, the matchline is pulled down; **(B)** a CAM cell is realized with 2 FeFETs; if there is a mismatch between the query and stored bit, one FeFET device acts as a closed switch and discharges the matchline; **(C)** different types of CAM-based matches; **(D)** FeFET crossbar for weight storage and in-memory analog MACs [16]

search latency, lowering search energy, simplifying peripherals, and/or improving scalability [22].

RRAM is widely used for crossbar type IMC cores including TCAMs, MCAMs, and ACAMs. There are a variety of RRAM TCAM designs [23] that support EX match and BE/TH match based on HM distance. A 6T2R EX-ACAM [24] was proposed and can directly process analog inputs. This design has high static power consumption and does not support BE/TH search. *PCM* has similar characteristics to RRAM. A PCM EX-TCAM design [25] can perform BE/TH searches based on HM distance with appropriate sensing circuits. *Flash AM Designs*: Floating-gate MOSFET (flash) is a mature non-volatile memory (NVM) technology suitable for AM design. EX-TCAM/MCAM designs based on 3D NAND Flash have been proposed [26]. The same 2-transistor design was implemented in [27] with flash memory where it can perform EX match and BE/TH match based on sigmoidal and SE distance functions. However, flash-based AMs have high write voltages and low endurance.

2) Crossbar Architectures: In NNs/ML, a ubiquitous computation is the multiply and accumulate (MAC) operation (e.g., $y = \sum_{i=1}^{n} w_i x_i$) that multiplies weights by inputs. Projections suggest that the energy to retrieve a weight for a MAC operation from off-chip memory may be two orders of magnitude higher than the integer MAC operation itself [28], and co-locating weight data with logic for MACs is appealing. A resistive processing unit (RPU) [17] was proposed where analog weight values are stored locally in crosspoint devices to minimize data movement during training. A crossbar architecture could perform extremely efficient, *analog* MAC operations. With inputs represented as voltages on horizontal rows, emerging devices (e.g., RRAM, FeFETs, PCM, etc.) can serve as tunable resistors with multiple analog states, and the results of MAC operations are captured by the summation of currents from crosspoint connections in a given column (see Fig. 2D with FeFETs). Crossbars support a wide variety of ML models, and training could be accelerated by >4 orders of magnitude [17].

III. FeFET-based Hyperdimensional Computing

We first present a case study where we aim to leverage FeFET-based AMs to support classification tasks via HDC models [4]. HDC is an emerging neuro-inspired framework based on the mathematical properties of hyper-dimensional spaces associated with human cognition and perception. Computational units are hyper-dimensional vectors, called hypervectors (HVs), which are (pseudo-)random, holographic vectors with independently distributed elements. HDC can learn by looking at a small number of training images. HDC systems are comprised of an encoding module (typically MVM) that maps input data to a high-dimensional space, and an associative search module that stores encoded data and considers its similarity with a given query for inference (Fig. 3A).

Technology-enabled hardware may help to accelerate HDC encoding and search. We consider FeFETs, where one can "tune" the threshold voltage of the device (Fig. 3B). MVM operations for encoding can be performed with crossbar arrays (Fig. 2D) and FeFETs can also be used to perform more efficient searches/distance calculations by realizing compact, content addressable memories (CAMs).

Most work with HDC uses GPUs for encoding and search. Each HV element may have anywhere from 1- to 32-bit precision (Fig. 3A). For search/inference operations, stored HVs must be transferred from a computer's memory to a GPU, such that a query may be compared to all learned HVs to identify the best match (e.g., via cosine distance). Per Fig. 3C, if vector element precision is low (i.e., 1 or 2 bits) classification accuracy drops. That said, software-hardware co-design efforts suggest that 3-to-4 bit vector element precision can be sufficient to match the accuracy of higher precision vector elements.

This analysis illustrates how technology-enabled compute architectures may support emerging compute workloads such as HDC. FeFETs can *enable* CAM cells that store multiple bits of information, thereby addressing an algorithmic need for iso-accuracy HDC workloads. This insight motivated experimental efforts where multi-bit, FeFET CAM cells were demonstrated. Fig. 3D illustrates how a CAM cell that stores 3-bits (8-states) conducts current as input voltage deviates from programmed cell state. If the query matches the stored state, neither transistor turns on, and the matchline that connects the transistors in the CAM cell will remain charged (low conductance point, Fig. 3D). As the query deviates from the stored value, the conductance of the CAM cell increases quadratically, and can mimic the Euclidean squared distance function, which in-turn can serve as a proxy for the Euclidean distance function [29] commonly employed in many ML algorithms. That said, while technology-enabled solutions may naturally implement key steps of an ML model (e.g., search) they may also change an existing algorithm (cosine vs. Euclidean distance) and designers

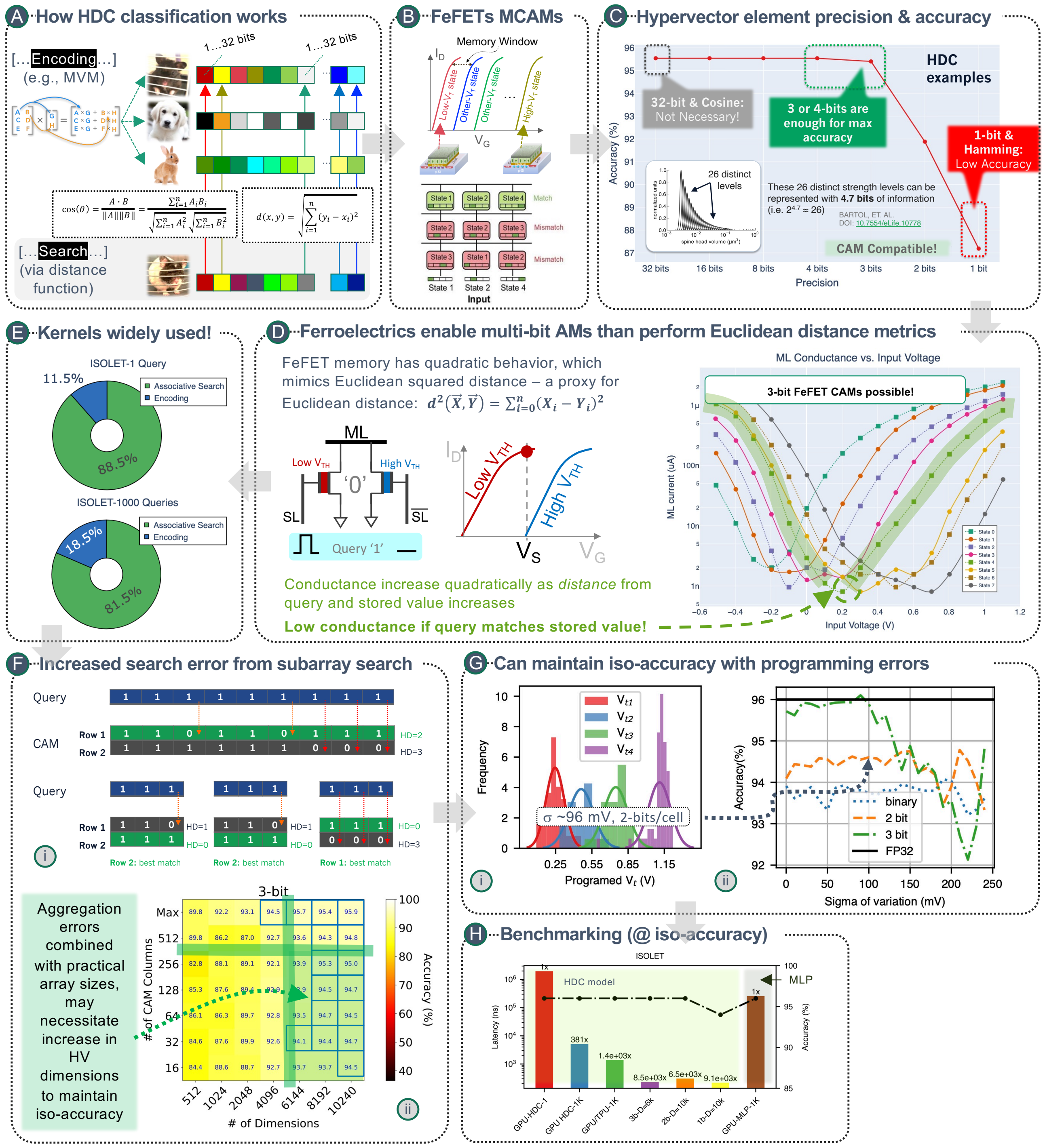

Fig. 3. "End-to-end" Hyperdimensional computing case study – see detailed descriptions of parts (A)-(H) in narrative.

will need to evaluate how new distance functions impact algorithmic accuracy.

Of course, the ultimate utility of a device-driven solution lies in how frequently a compute kernel can actually be used (i.e., Amdahl's Law), and algorithmic workloads/runtimes must be carefully studied. Per Fig. 3E (at least for HDC models), search operations for different datasets can represent a substantial portion of end-to-end compute time, and as such, technology-driven solutions could have substantial application-level impact.

This case study has assumed that a HV (∼1K-10K HV elements) can be compared against all stored HVs "in aggregate." When considering hardware implementations, a single CAM array with even $1K$ columns is not practical, as peripheral circuitry will not be able to accurately differentiate various degrees of match/mismatch. Still, one can partition an N element HV into n elements and perform a search over $\lceil \frac{N}{n} \rceil$ smaller subarrays such that peripheral circuitry can detect the necessary degrees of match at the subarray-level. However, when subarray voting results are tallied to determine a possible best match, per Fig. 3F-i, this approach can induce aggregation-based errors. (The query vector is closest to Row 1; however, if searched "segment-by-segment," the best-match appears to be Row 2.) Fig. 3F-ii provides a snapshot of how, for a given/single dataset, accuracy may be impacted assuming different combinations of HV length and CAM subarray size. As expected, as subarray size increases, accuracy increases as well. (In Fig. 3F-ii, "max" assumes a CAM size that is equal to the HV length listed on the x-axis). A designer may need to compensate for aggregation-based errors by increasing HV length to maintain iso-accuracy, thereby adversely increasing memory capacity. Different datasets/problems may present different tradeoffs.

We must also be cognizant of cell-state distributions as we consider memory devices that store multiple bits/levels. Per Fig. 3G-i, preliminary experimental results indicate that there can be *overlap* between cell states, suggesting that while the intent is to program a cell into a 00 state, it may instead be programmed into a 01 state. As the number of target levels per memory cell increases, the "window" between states (Fig. 3B) will decrease, making overlap more likely. However, *algorithmically*, at least *some* variation may be tolerable. For example, Fig. 3G-ii considers classification accuracy assuming 1-, 2-, and 3-bit CAMs as a function of the sigma of cell state programming variation. Notably, even with sigmas of variation observed in preliminary experiments (94 mV per Fig. 3G-ii), there is no degradation in accuracy (i.e., owing to the relative robustness of the HDC model, no single element has any more weight than any other element). This too may be problem specific, and tradeoffs must be carefully considered across workloads.

Finally, the above efforts must be assessed via comprehensive benchmarking. Fig. 3H considers inference latencies, for the same dataset, assuming different device/architecture platforms. Latency is correlated with accuracy. The leftmost bars assume GPU/HDC-based solutions with inferences that consist of 1 query (real time response needed) and 1000 queries (exploits parallelism, amortizes latency). A TPU-GPU hybrid (third bar) may represent a nominal improvement owing to more efficient MVM for encoding. The next 3 columns represent CAM/HDC solutions assuming 3-bit FeFET, 2-bit FeFET, and 1-bit SRAM cells respectively. 3-bit FeFET-based CAMs represent a superior design point owing to reduced HV dimensions; 2-bit designs only achieve iso-accuracy with larger HVs, and 1-bit HVs offer the lowest latencies, but cannot achieve iso-accuracy. Finally, a GPU/MLP-based solution can achieve iso-accuracy, but offers no latency improvement even with larger batches.

Open questions abound for just this problem. As examples, (1) What is the best baseline architecture to compare to? (i.e., is an HDC model more likely to be deployed "on the edge", making small batches more likely and a GPU less likely to be employed?); (2) What if an existing architecture (e.g., a TPU) is backed by a dense or distributed non-volatile memory? Is this a better way to leverage an emerging technology? In essence, for this technology-driven architecture study, modeling efforts were needed at the device, circuit, architecture, and algorithmic levels to quantify the value proposition for FeFET-based AMs and crossbars, and additional modeling infrastructure is needed to further justify the viability of this approach.

IV. RRAM-based Few Shot Learning

We also consider a few-shot learning problem where the core computations in memory augmented neural networks (MANNs) (CNNs, hashing and associative search) are all realized with RRAM-based crossbars [8]. MANNs can rapidly learn classes from a small number of samples. A software model consists of a regular deep neural network (e.g., a CNN for images) to extract feature vectors (FVs) from the input, followed by an AM that can determine the distance between learned FVs and the FV associated with a new sample/query [30].

While the model is promising with respect to accuracy when "learning," the energy/latency associated with conventional digital hardware may be prohibitive and developing accelerators is of great interest. Profiling suggests that the majority of the runtime in a large MANN model arises from a MANN's AM, as the distance calculation requires that each learned memory be accessed/transferred/processed. FeFET-based CAM solutions have been considered [31], and similar problems have also been studied in [32], [33].

Motivated by FeFET designs, an RRAM-based implementation was realized, where all essential compute tasks for a MANN model (CNN, hashing, and AM) were realized via RRAM crossbars (Fig. 4A). Furthermore, (1) software/hardware co-optimization efforts were employed to chart a path to iso-accuracy when compared to software-based solutions, and (2) all computations for few-shot learning tasks were performed *experimentally* on RRAM-based crossbar arrays (for the Omniglot dataset). Below, we discuss the hashing/AM tasks in more detail; we refer the reader to [8] for more information regarding the well-studied realization of CNN computations.

Looking first at hashing, locality sensitive hashing (LSH) should be more likely to generate the same hashing bits for inputs that are close to each other (e.g., belong to the same class). LSH can be implemented by determining which space a vector belongs to after random projections are made, where the projection of a given vector is in essence a MVM by a random matrix with a zero mean value. Accordingly, the random projection can be implemented by performing the matrix multiplication directly in an RRAM crossbar that is programmed with randomly distributed conductances, and by exploiting intrinsic stochasticity of RRAM devices (Fig. 4B).

In [8], several technology-enabled optimizations were implemented to mitigate the impact of device non-idealities on application-level correctness. First, randomly distributed high resistance states (HRSs) were employed in lieu of low resistance states as device-to-device variation is usually larger with devices in a HRS (better suited for reliable hashing). Additionally, IR drop (and thereby energy consumption) is more prominent when current is higher (i.e., RRAM's resistances are lower). This can lead to a higher probability of generating '1's than '0's and ultimately worse performance.

Additionally, when a result is close to the hashing plane, the hashing bits can be randomly flipped between bits, arising from non-idealities (conductance relaxation). To mitigate the negative impact of this phenomena, a ternary LSH (TLSH) method was proposed. A signature bit is set to the "don't care"

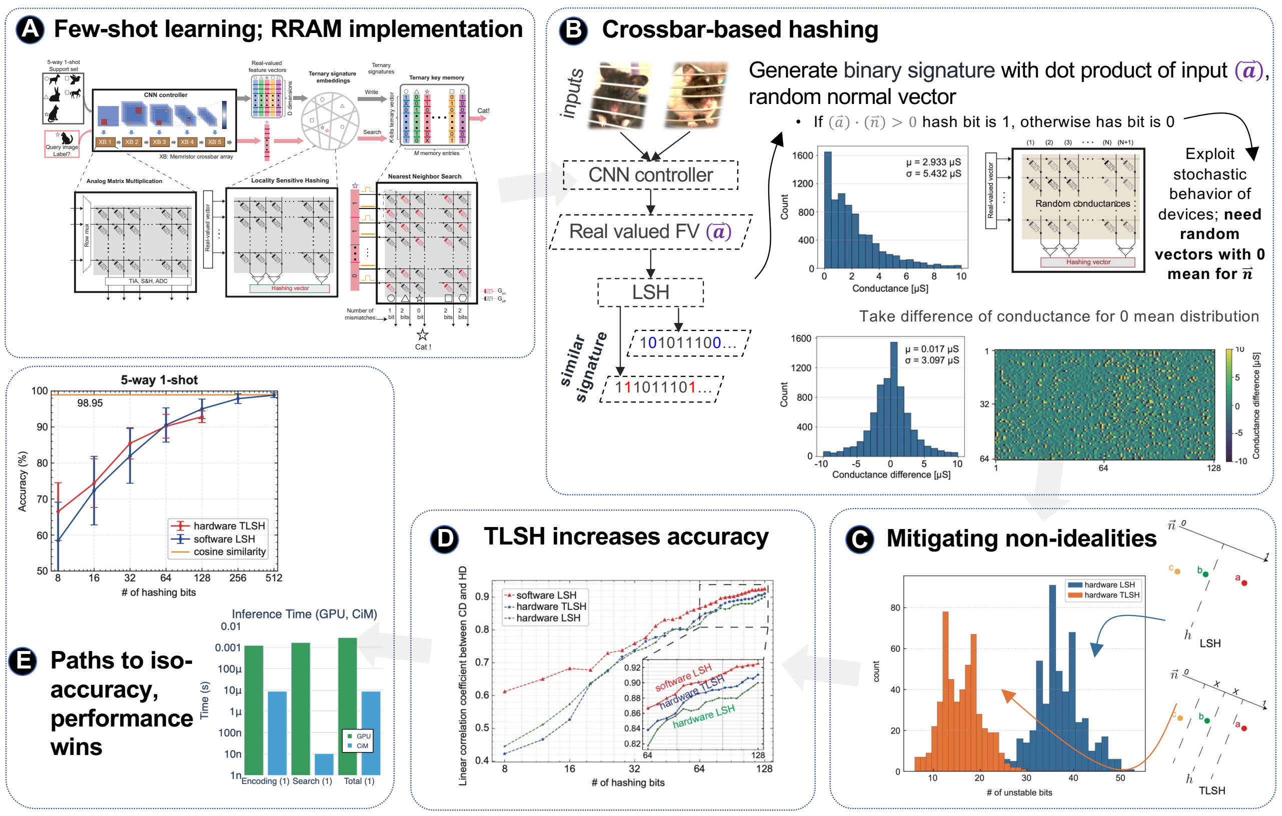

Fig. 4. RRAM-based FS-learning case study – see detailed discussion of A through E in the narrative.

state ’X’ if the difference between adjacent crossbar columns is below a threshold current. As such, the Hamming distance between the ”don’t care” state and an input will always be 0. Per Fig. 4C, this can reduce the impact of unstable bits. Indeed, per Fig. 4D, the linear correlation coefficient between a cosine distance calculation and hashed representation is greater with the TLSH/TCAM, which approaches that of a true, software-based LSH, which should ultimately lead to a positive impact on classification accuracy with the RRAM-based approach.

Crossbars can also be reconfigured to calculate the Hamming distance of the hashed bits, or in other words, the degree of mismatch between a query and other entries stored in a CAM. Several challenges in experimental demonstrations include temporal and spatial variations with RRAM conductance, device ON/OFF ratio, etc. Studies determined that RRAM conductance variation is dependent on the state that it is programmed to, and there exists a conductance region where variation can be more substantial when compared to other conductance regions. While physical mechanisms are under investigation, it was possible to construct a statistical array model that captures the aforementioned phenomena, as well as other array-level non-idealities such as IR-drop from wire resistance, stochastic analog error from peripheral circuits, etc. Such models can facilitate co-optimizations to improve tolerance with respect to device variation [34]. For RRAM crossbar-based TCAMs, conductance states can be mapped away from regions where the conductance variation is large, while simultaneously ensuring negligible impact from IR drop. Ultimately, Hamming distance can be reliably calculated as the output current is linearly dependent on Hamming distance.

By leveraging the aforementioned techniques, few-shot learning with the Omniglot dataset was demonstrated ”end-to-end.” The CNN model employed >65,000 weights that were realized via 130,000 RRAM devices, in 36, 64×64 crossbar arrays. (This was limited by the prototype capacity of 50 nm×50 nm Ta/TaO$_x$/Pt RRAM structures, and devices were re-programmed as needed.) Experimental demonstrations employed 128-bit hashing bits (again, limited by prototype chip capacity).

While experimental demonstrations with 128-bit hashing signatures suggest a degradation in accuracy versus a software-based cosine distance, simulation results suggest iso-accuracy inference is possible with a hashing-based solution if slightly longer hash signatures are employed (Fig. 4E). Similar trends are observed for the more complex/sophisticated miniImageNet dataset (see [8]). Substantial improvements in FOM such as latency (Fig. 4E) and energy (not shown) are also possible.

V. Architectural Modeling Tools

The evaluation of new technology-enabled breakthroughs should not be considered in isolation. Benefits should be assessed in the context of entire systems, when performing realistic applications. The consideration of abstractions from hardware to system-level is extremely challenging, as device- and circuit-level approaches often struggle when attempts are made to elevate exploration scope. The availability of open-hardware and modular platforms is key for the prototyping of novel technological solutions [35]. A recent example of such a framework is provided by the X-HEEP effort [36] [10], an ultra-low power, system-on-chip, architectural template comprised of hardware-validated computing and storage resources, as well as common peripherals (input-output links, timers etc.). X-HEEP is based on the open RISC-V ISA [37], and includes a dedicated software development environment. Instances can be flexibly derived from the template, opening the way for the fast integration of custom accelerators, as exemplified by the coarse grained reconfigurable accelerator in [38] and the in-SRAM computing hardware in [39].

An alternative approach to open hardware platforms is that of system simulators, which waive the availability of a direct path to synthesis, in favor of faster simulation times. Notable examples include Sniper [40] and Simics [41]. The most popular system simulation environment is gem5 [42], partly due to its support for multiple Instruction Set Architectures (ISAs): ARM, x86 and MIPS among others. It can also simulate processors of different complexity and multi-layered memory hierarchies. Gem5 operates in two modes: in syscall mode, only application code is simulated, while in full system mode, the entire hardware/software stack is analyzed, including the contribution of the operating system. In turn, gem5-X [11] builds on gem5 by easing the development effort for system definition, and providing convenience features for accelerating explorations such as advanced profiling and checkpointing, and data migration among guest and host machines. Crucially, gem5-X also allows the easy instantiation of standard and custom components, thereby providing the ability to evaluate the overall benefit of novel technology-enabled accelerators. In [43], this capability was studied to explore the speedup induced by analog crossbars performing MVMs, while the authors of [44] modelled systems featuring in-package wireless communication. Furthermore, in [45] gem5-X was used to investigate tightly-coupled systolic arrays, and in [46] for exploring in-cache computing based on bit-line operations. In all cases, the system level simulation approach allowed for analysis of large machine learning workloads such a CNNs, LSTMs and transformers algorithms. As an example, the authors of [43] showcase that the analog crossbars can speed up the execution of benchmark convolutional networks by up to 20X.

VI. Analytical Modeling Tools

Circuit and array-level evaluation efforts of emerging, technology-driven solutions are essential to ensure that the application-level value proposition of a given technology is maintained as size scales up. While SPICE-based circuit simulations are accurate, they are also time-consuming and have poor scalability in evaluating different memory sizes or technologies. As evidenced by studies in Secs. III and IV, end-to-end evaluations are critical. In said studies, system/application-level evaluation tools will inherently rely on circuit/architecture data that include FOM such as area, latency, and energy to estimate the performance of specific computational tasks.

As examples, consider existing tools that may be employed for array-level evaluations of NVMs: *NVsim* [5] is a circuit-level framework that estimates the performance, energy, and area of a 2D single-level memory with a given organization. Memory organization and cell information are used to obtain array-level FOM. *DESTINY* [47] is based on NVsim and estimates the performance, energy, and area of a 2D/3D single-level cache memory. *NVMExplorer* [12] leverages a modified NVsim (for multi-level memory), a fault model, and DNN network to estimate the application-level accuracy, memory performance, and memory lifetime based on memory traffic. *NeuroSim* [6] is a circuit-level macro model that estimates the performance, energy and leakage power of crossbar arrays. NeuroSim can consider NN topologies from the device-to-algorithmic levels.

Referring back to Fig. 1, said tools may be employed to assess different "lanes" of the architectural design space. For example, NVSim and NVMExplorer can help to assess the impact of memory technology, on a traditional cache/DRAM-based hierarchy, for CPU- and GPU-like architectures. DESTINY may be used – and ultimately modified – to begin to assess the architectural implications of emerging monolithic 3D memories in a similar context. The Neurosim framework may be used to evaluate RPU-like crossbar arrays, and has been validated against implementations based on ferroelectric and various resistive memory technologies.

Analytical modeling infrastructure that addresses other lanes in Fig. 1 is beginning to emerge. For example, Eva-CiM [14] models the performance of different IMC designs from device-to-system, and enables researchers to assess whether a program is IMC-favorable (i.e., can benefit from an IMC architecture), the pros and cons of increased memory size, etc. Eva-CiM uses a multi-level tool chain that leverages existing modeling and simulation tools such as Gem5 [42], McPAT [48], and DESTINY [47]. Eva-CiM can produce system-level energy and performance estimates for a given program, processor architecture, and IMC array assuming a given underlying technology.

Circuit/array-level modeling and evaluation for CAMs is essential given (1) the rapid development of CAM designs per Sec. II and (2) their broad applicability (e.g., per Secs. III and IV, their ability to perform general purpose computation [49], etc.). A dedicated tooling infrastructure is needed as existing modeling tools can only support a subset of memory technologies or structures. For example, NVsim [5] can model NVM-based caches or RAM, but does not support CAMs. NVsim-CAM [50] can help to evaluate NVM-based CAMs, but it does not support (a) CAM designs based on three-terminal devices, such as FeFETs, (b) emerging CAM designs such as ACAMs and MCAMs, or (c) match types beyond EX-match.

As a representative example of an analytical modeling infrastructure, we discuss Eva-CAM [13] – a circuit/architectural-level modeling and evaluation tool that supports a variety of NV-CAMs [13]. Eva-CAM leverages the basic structure of NVsim-CAM, but significantly extends NVsim-CAM functionality. Fig. 1F provides a high-level overview of the Eva-CAM framework. Gray, white, and blue cells represent the major components of Eva-CAM, user-defined inputs, and tool outputs respectively. Currently, Eva-CAM supports (1) both exact and approximate match types, (2) TCAM, and MCAM designs, and (3) two-terminal and three-terminal NVM devices (e.g., FeFETs). Preliminary validations of Eva-CAM assuming

References	FoMs	Actual	Eva-CAM	Error
RRAM 2T2R 40nm	Area* (um^2)	98000	86600	-11.6%
	Search Latency (ns)	≥ 5	2-4.4	–
	Search Energy(pJ)	270	268.5	-1.0%
PCM 2T2R 90nm	Area (um^2)	–	–	–
	Search Latency (ns)	1.9	2.1	9.4%
	Search Energy(pJ)	–	–	–
MRAM 4T2R 90nm	Area (um^2)	17200	18270	6.2%
	Search Latency (ps)	2.5	2.72	8.6%
	Search Energy(pJ)	–	–	–

$*$ The actual area only includes RRAM array and peripherals.

Fig. 5. Validation of Eva-CAM against SOA NV-CAMs with available data.

several state-of-the-art fabricated CAM chips, as well as SPICE simulations are summarized in Fig. 5. Eva-CAM projections are within 20% of measured and/or SPICE simulation data.

Representative enhancements to Eva-CAM should include the evolution of the tool to support new, 3-terminal NVM device models. Currently, 3-terminal NVM devices (FeFETs and flash) are modeled as variable resistors and capacitors. Though an improvement to a simple 2-terminal resistance memory model, this is not intuitive to use, as the states of 3-terminal NVM devices are specified by threshold voltage instead of resistance. However, directly using V_{th} settings presents a challenge in estimating matchline discharge speed and energy.

Eva-CAM can also be extended to consider array size and mismatch limits by incorporating the effects of device variations into circuit/architecture modeling. For a given CAM design, achievable array size is a key consideration for architecture-level design, and impacts area, delay, and energy. However, designs must consider the sense margin (SM) of the matchline (MaLi), especially for BE-/TH-match when determining array size. The relatively small on/off resistance ratios of NVMs can limit the SM of the MaLi [51], and the number of cells on a MaLi. For BE, TH matches, when the least (or threshold) number of mismatched cells is too large, the sensing circuit may not distinguish the BE-match (or TH-match) word from another word due to small differences in a MaLi's discharge speed between the two words (i.e. the *mismatch limit*.) Eva-CAM can determine the maximum number of cells on a MaLi (i.e., the number of columns) in a CAM subarray by comparing the SM of the MaLi and the SM of the sensing circuit, both of which are derived from the circuit model as well as user-specified/default SM values. Device variations impact CAM behavior due to influence on discharge path resistance and capacitance, and potentially reduce CAM array size as larger arrays would suffer more variations on the MaLis and BLs. To properly consider variations, the distributions of device variations will be integrated into circuit models along with array size and mismatch limit prediction formulae. Advances such as these will facilitate accelerated DSEs and triage of a design space like that illustrated in Fig. 1.

VII. Paths Forward

In closing, we briefly discuss additional strategies, visualizations, and potential enhancements for the DSE efforts described herein via top-down and bottom-up efforts. Researchers interested in algorithms and architectures (rightmost column, Fig. 6) should identify application-level needs/problems of interest, and subsequently profile existing algorithms to identify the most significant aspects of computational workloads (see inset). This can in-turn be used to determine if the composition of said workload may map well to an alternative architecture (e.g., binary AM in Fig. 6, other IMC architectures or unit cells, etc.). Said mappings may then be evaluated in a "technology agnostic" fashion – i.e., to determine the potential impact of data precision on algorithmic accuracy, what device metrics are likely to have the most substantial impact on run time (fourth column in Fig. 6), etc. Said studies can serve as a way to rapidly identify what technology may be best suited for an application (e.g., are data traffic patterns write heavy, thereby prioritizing device endurance and/or write latency? are datasets large with frequent reads, thereby prioritizing denser memory?)

The above mappings can then be used to engage with materials scientists and device engineers to prioritize innovation at this level of the design stack. To illustrate a path forward, the three, leftmost columns in Fig. 6 capture possible materials-based innovation for spin devices, and how said levers could enhance device performance. This creates a natural linkage between the third column (materials-based device enhancements) and the fourth column (improved device metrics). Top-down efforts can help to prioritize research at the materials/device-level (bottom-up efforts). Put another way, top-down efforts can provide guidance as to what materials-based innovation can have the most substantial application-level impact. Furthermore, bottom-up efforts can provide a floor/ceiling with respect to possible logic/memory performance, thereby providing realistic constraints at upper-levels of the design stack.

References

[1] T. Brown *et al.*, "Language models are few-shot learners," *Advances in neural information processing systems*, vol. 33, pp. 1877–1901, 2020.

[2] J. Sevilla *et al.*, "Compute trends across three eras of machine learning," *arXiv preprint arXiv:2202.05924*, 2022.

[3] DARPA, "DPRIVE: Building Hardware to Enable Continuous Data Protections," https://www.darpa.mil/news-events/2020-03-02/, 2020.

[4] A. Kazemi *et al.*, "Achieving software-equivalent accuracy for hyper-dimensional computing with ferroelectric-based in-memory computing," *Scientific reports*, vol. 12, no. 1, pp. 1–15, 2022.

[5] X. Dong *et al.*, "NVSim: A Circuit-Level Performance, Energy, and Area Model for Emerging Nonvolatile Memory," *IEEE TCAD*, vol. 31, no. 7, pp. 994–1007, 2012.

[6] P. Chen *et al.*, "Neurosim+: An integrated device-to-algorithm framework for benchmarking synaptic devices and array architectures," in *IEDM*, Dec 2017, pp. 6.1.1–6.1.4.

[7] S. Deng *et al.*, "A comprehensive model for ferroelectric FET capturing the key behaviors: Scalability, variation, stochasticity, and accumulation," in *IEEE Symp. on VLSI*, 2020, pp. 1–2.

[8] R. Mao *et al.*, "Experimentally validated memristive memory augmented neural network with efficient hashing and similarity search," *Nature Communications*, vol. 13, no. 1, pp. 1–13, 2022.

[9] Hill, Mark D and Reddi, Vijay Janapa, "Accelerator-level parallelism," *Communications of the ACM*, vol. 64, no. 12, pp. 36–38, 2021.

[10] F. Ponzina *et al.*, "A hardware/software co-design vision for deep learning at the edge," *IEEE Micro*, vol. 42, no. 6, pp. 48–54, 2022.

[11] Y. M. Qureshi *et al.*, "Gem5-x: A many-core heterogeneous simulation platform for architectural exploration and optimization," *ACM TACO*, vol. 18, no. 4, pp. 1–27, 2021.

[12] L. Pentecost *et al.*, "Nvmexplorer: A framework for cross-stack comparisons of embedded non-volatile memories," *arXiv preprint arXiv:2109.01188*, 2021.

[13] L. Liu *et al.*, "Eva-CAM: A Circuit/Architecture-Level Evaluation Tool for General Content Addressable Memories," in *DATE*, 2022, pp. 1173–6.

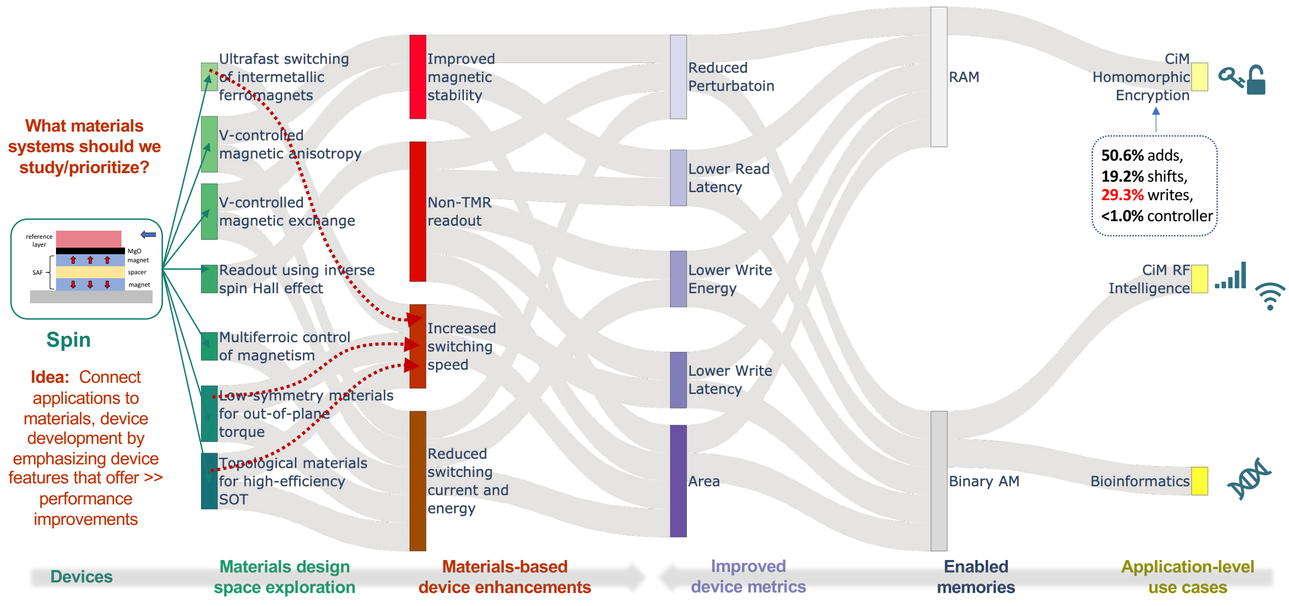
What materials systems should we study/prioritize?
reference layer
MgO
magnet
SAF
spacer
magnet
Spin
Idea: Connect applications to materials, device development by emphasizing device features that offer >> performance improvements
Ultrafast switching of intermetallic ferromagnets
V-controlled magnetic anisotropy
V-controlled magnetic exchange
Readout using inverse spin Hall effect
Multiferroic control of magnetism
Low-symmetry materials for out-of-plane torque
Topological materials for high-efficiency SOT
Improved magnetic stability
Non-TMR readout
Increased switching speed
Reduced switching current and energy
Reduced Perturbatoin
Lower Read Latency
Lower Write Energy
Lower Write Latency
Area
RAM
Binary AM
CiM Homomorphic Encryption
50.6% adds, 19.2% shifts, 29.3% writes, <1.0% controller
CiM RF Intelligence
Bioinformatics
Devices
Materials design space exploration
Materials-based device enhancements
Improved device metrics
Enabled memories
Application-level use cases

Fig. 6. How to connect innovation at the materials/device-levels to application-level use cases.

[14] D. Gao *et al.*, "Eva-CiM: A system-level performance and energy evaluation framework for computing-in-memory architectures," *IEEE TCAD*, vol. 39, no. 12, pp. 5011–5024, 2020.

[15] S. Dutta *et al.*, "Logic compatible high-performance ferroelectric transistor memory," *arXiv preprint arXiv:2105.11078*, 2021.

[16] M. Jerry *et al.*, "A ferroelectric field effect transistor based synaptic weight cell," *J. of Phys. D: App. Phys.*, vol. 51, no. 43, p. 434001, 2018.

[17] G. Tayfun and Y. Vlasov, "Acceleration of Deep Neural Network Training with Resistive Cross-Point Devices," *CoRR*, vol. abs/1603.07341, 2016. [Online]. Available: http://arxiv.org/abs/1603.07341

[18] H. Li *et al.*, "Resistive RAM-Centric Computing: Design and Modeling Methodology," *IEEE TCAS-1*, vol. 64, no. 9, pp. 2263–2273, 2017.

[19] H.-Y. Chen *et al.*, "HfOx based vertical resistive random access memory for cost-effective 3D cross-point architecture without cell selector," in *IEDM*, 2012, pp. 20–7.

[20] M. Imani *et al.*, "Approximate Computing Using Multiple-Access Single-Charge Associative Memory," *IEEE Transactions on Emerging Topics in Computing*, vol. 6, no. 3, pp. 305–316, July 2018.

[21] X. Yin *et al.*, "FeCAM: A Universal Compact Digital and Analog Content Addressable Memory Using Ferroelectric," *IEEE TED*, vol. 67, no. 7, pp. 2785–2792, 2020.

[22] X. S. Hu *et al.*, "In-memory computing with associative memories: a cross-layer perspective," in *IEDM*, 2021, pp. 25–2.

[23] R. Rajaei *et al.*, "Compact Single-Phase-Search Multistate Content-Addressable Memory Design Using One FeFET/Cell," *IEEE Transactions on Electron Devices*, vol. 68, no. 1, pp. 109–117, 2020.

[24] C. Li *et al.*, "Analog content-addressable memories with memristors," *Nat. Commun.*, vol. 11, no. 1, p. 1638, Apr. 2020.

[25] J. Li *et al.*, "1Mb 0.41 µm2 2T-2R cell nonvolatile TCAM with two-bit encoding and clocked self-referenced sensing," in *2013 Symposium on VLSI Circuits*, 2013, pp. C104–C105.

[26] F. Wang *et al.*, "Implementation of data search in multi-level nand flash memory by complementary storage scheme," *IEEE EDL*, vol. 41, no. 8, pp. 1189–1192, 2020.

[27] A. Kazemi *et al.*, "MIMHD: Accurate and Efficient Hyperdimensional Inference Using Multi-Bit In-Memory Computing," in *ISLPED*, 2021, pp. 1–6.

[28] S. Han *et al.*, "EIE: efficient inference engine on compressed deep neural network," in *ISCA*, 2016, pp. 243–254.

[29] Q. Kuang and L. Zhao, "A practical gpu based knn algorithm," in *Proceedings. The 2009 International Symposium on Computer Science and Computational Technology (ISCSCI 2009)*. Citeseer, 2009, p. 151.

[30] A. Santoro *et al.*, "Meta-Learning with Memory-Augmented Neural Networks," in *Proc. of the 33rd International Conference on International Conference on Machine Learning*, vol. 48, 2016, pp. 1842–50.

[31] K. Ni *et al.*, "Ferroelectric ternary content-addressable memory for one-shot learning," *Nature Electronics*, vol. 2, no. 11, pp. 521–529, 2019.

[32] H. Li *et al.*, "Sapiens: A 64-kb rram-based non-volatile associative memory for one-shot learning and inference at the edge," *IEEE TED*, vol. 68, no. 12, pp. 6637–6643, 2021.

[33] M. Hersche *et al.*, "Constrained few-shot class-incremental learning," 2022. [Online]. Available: https://arxiv.org/abs/2203.16588

[34] R. Mao *et al.*, "Experimentally-validated crossbar model for defect-aware training of neural networks," *IEEE TCAS-II*, vol. 69, no. 5, pp. 2468–2472, 2022.

[35] D. Rossi *et al.*, "Pulp: A parallel ultra low power platform for next generation iot applications," in *IEEE Hot Chips 27*, 2015, pp. 1–39.

[36] D. S. et al., "X-HEEP github repository ," https://github.com/esl-epfl/x-heep, 2023.

[37] A. S. Waterman, *Design of the RISC-V instruction set architecture*. University of California, Berkeley, 2016.

[38] B. W. Denkinger *et al.*, "Vwr2a: a very-wide-register reconfigurable-array architecture for low-power embedded devices," in *DAC*, 2022, pp. 895–900.

[39] W. Simon *et al.*, "A fast, reliable and wide-voltage-range in-memory computing architecture," in *DAC*, 2019, pp. 1–6.

[40] T. E. Carlson *et al.*, "Sniper: Exploring the level of abstraction for scalable and accurate parallel multi-core simulation," in *Int. Conf. for High Perf. Comp., Networking, Storage and Analysis*, 2011, pp. 1–12.

[41] P. S. Magnusson *et al.*, "Simics: A full system simulation platform," *Computer*, vol. 35, no. 2, pp. 50–58, 2002.

[42] N. Binkert *et al.*, "The gem5 simulator," *ACM SIGARCH computer architecture news*, vol. 39, no. 2, pp. 1–7, 2011.

[43] J. Klein *et al.*, "Alpine: Analog in-memory acceleration with tight processor integration for deep learning," *IEEE T. Comp.*, pp. 1–14, 2022.

[44] R. Medina Morillas *et al.*, "System-level exploration of in-package wireless communication for multi-chiplet platforms," in *ASPDAC*, 2023.

[45] A. Amirshahi *et al.*, "Tic-sat: Tightly-coupled systolic accelerator for transformers," in *ASPDAC*, 2023.

[46] W. A. Simon *et al.*, "Blade: An in-cache computing architecture for edge devices," *IEEE T. on Computers*, vol. 69, no. 9, pp. 1349–1363, 2020.

[47] M. Poremba *et al.*, "DESTINY: A Tool for Modeling Emerging 3D NVM and eDRAM Caches," in *DATE*, 2015, pp. 1543–1546.

[48] S. Li *et al.*, "McPAT: An integrated power, area, and timing modeling framework for multicore and manycore architectures," in *MICRO*, Dec 2009, pp. 469–480.

[49] H. Caminal *et al.*, "CAPE: A Content-Addressable Processing Engine," in *HPCA*, 2021, pp. 557–569.

[50] S. Li *et al.*, "Nvsim-cam: A circuit-level simulator for emerging nonvolatile memory based content-addressable memory," in *ICCAD*, 2016, pp. 1–7.

[51] J. Li *et al.*, "1 Mb 0.41 μm^2 2T-2R Cell Nonvolatile TCAM With Two-Bit Encoding and Clocked Self-Referenced Sensing," *IEEE JSSC*, vol. 49, no. 4, pp. 896–907, 2014.