In-Memory Acceleration of Hyperdimensional Genome Matching on Unreliable Emerging Technologies

Hamza E. Barkam, *Member, IEEE*, Sanggeon Yun, *Member, IEEE*, Paul R. Genssler, *Member, IEEE*, Che-Kai Liu, *Student Member, IEEE*, Zhuowen Zou, *Member, IEEE*, Hussam Amrouch, *Member, IEEE*, and Mohsen Imani, *Member, IEEE*

***Abstract*— Novel computer architectures like Compute-in-Memory (CiM) merge the memory and processing units, mimicking the human brain. Simultaneously, Hyperdimensional Computing (HDC) is emerging as a brain-inspired machine learning (ML) approach. Both developments hold promise for the realm of AI and computing, especially for genome-matching tasks, where large data movements overwhelm traditional von Neumann architectures. FeFET is one of the up-and-coming emerging technologies that promises to enable ultra-efficient and compact CiM architectures. However, the adoption of FeFETs is hindered by their 10 nm-thick Ferroelectric (FE) layer and process variation. Thus, calculations with FeFETs have errors (noise) that traditional ML genome-matching models cannot tolerate. To overcome these challenges, this work is the first one to i) present a reliable HDC framework (HDGIM) for highly-scaled (down to merely 3nm), multi-bit FeFET technology, ii) introduce temperature-thickness modeled noise from FeFET to the HDC system, and iii) extensively define the memorization capacity of HDC hyperparameters in order to evaluate the performance before deployment theoretically. Our novel HDC learning framework iteratively uses two models: a full-precision 32-bit HDC model, an ideal model for training, and a reduced bit-precision by a novel quantization method for validation and inference. Our results demonstrate that highly-scaled FeFET, realizing 3-bit and even 4-bit, can withstand any modeled noise given high dimensionality during inference. Considering the noise during model adjustment improves the inherent robustness by almost 9% on the 4-bit case.**

***Index Terms*— In-memory computing, ferroelectric, FeFET, reliability, hyperdimensional computing.**

Manuscript received 31 March 2023; revised 14 September 2023 and 11 December 2023; accepted 24 December 2023. Date of publication 26 January 2024; date of current version 29 March 2024. This work was supported in part by the Defense Advanced Research Projects Agency (DARPA) Young Faculty Award; in part by the National Science Foundation under Grant 2127780, Grant 2319198, Grant 2321840, and Grant 2312517; in part by Semiconductor Research Corporation (SRC); in part by the Office of Naval Research Young Investigator Program Award under Grant N00014-21-1-2225 and Grant N00014-22-1-2067; in part by the Air Force Office of Scientific Research under Award FA9550-22-1-0253; and in part by the Advantest as part of the Graduate School "Intelligent Methods for Test and Reliability" (GS-IMTR), University of Stuttgart. The work of Che-Kai Liu was supported by the Center for the Co-Design of Cognitive Systems (COCOSYS), one of seven centers in JUMP2.0; and in part by SRC Program Sponsored by DARPA. This article was recommended by Associate Editor Y. Zhang. *(Corresponding author: Hamza E. Barkam.)*

Hamza E. Barkam, Sanggeon Yun, Zhuowen Zou, and Mohsen Imani are with the Bio-Inspired Architecture and Systems Laboratory (BIASLab), UC Irvine, Irvine, CA 92697 USA (e-mail: herrahmo@uci.edu; zhuowez1@uci.edu; m.imani@uci.edu).

Paul R. Genssler is with the Chair of Semiconductor Test and Reliability (STAR), University of Stuttgart, 70569 Stuttgart, Germany (e-mail: genssler@iti.uni-stuttgart.de).

Che-Kai Liu is with the Integrated Circuits and Systems Research Laboratory (ICSRL), Georgia Institute of Technology, Atlanta, GA 30332 USA (e-mail: che-kai@gatech.edu).

Hussam Amrouch is with the Chair of Semiconductor Test and Reliability (STAR), University of Stuttgart, 70569 Stuttgart, Germany, and also with the Chair of AI Processor Design and the Munich Institute of Robotics and Machine Intelligence (MIRMI), Technical University of Munich (TUM), 80333 Munich, Germany (e-mail: amrouch@tum.de).

Color versions of one or more figures in this article are available at https://doi.org/10.1109/TCSI.2024.3351966.

Digital Object Identifier 10.1109/TCSI.2024.3351966

I. Introduction

THIS work is an extension of work presented initially in DATE [1], centered around genome sequence matching, one of the fundamental algorithms in identifying and analyzing genomic data with several applications in identifying and curing diseases, including COVID-19. Sequence matching analyzes essential biological characteristics such as nucleotide or protein sequences and compares their similarity [2]. In sequence matching, a DNA query is searched across large-scale reference DNA strings, which typically comprise over a hundred million DNA bases [3]. Unfortunately, running genome sequence matching on existing hardware is significantly slow and inefficient as it demands significant data movement between the memory and computing cores.

A Computing in-Memory (CiM) architecture is a promising solution to address data movement issues. By integrating basic processing capabilities, the CiM paradigm enables operations directly on the data stored in memory. Hence, CiM significantly reduces power consumption and enables faster computations through less data movement. Non-volatile memories (NVMs) are typically employed to build CiM architectures. However, CiM suffers from NVM's device-to-device variation and inaccuracy from its analog implementations [4], thus degrading the application-level accuracy.

Among different NVM technologies, FeFET has emerged as a promising candidate due to its full CMOS compatibility [5], low read/write energy [6], making its integration into existing manufacturing technology straightforward [5]. However, FeFET devices are often designed with a 10nm-thick ferroelectric layer in the gate stack of the transistor to maintain the state. Such a thick layer has several disadvantages.

The ±4V write voltage is not available on modern ICs, requiring a new dedicated power network inside the chip, incurring a significant overhead. Further, the high write voltage is a significant source of reliability and endurance challenges [7].

Nonetheless, if the write voltage could be reduced to the typical I/O voltage of 1.8V, no additional power network would be required, energy consumption would decrease, and reliability increase. Those challenges can be solved by reducing the thickness of the FE layer. It enables technology scaling, reduces energy consumption, and increases endurance [8], [9]. A 3nm FE-layer was explored in [8], and a write voltage of ±1.85V was sufficient. However, a back gate was added to the FeFET to overcome the high variation during a read operation [9].

The efficient but noisy CiM technology, when combined with high-precision computation that many sequence matching algorithms require [10], [11], calls for a robust computational model. Hyperdimensional computing (HDC) has emerged as an alternative computational model and data representation that mimics important brain functionalities toward high-efficiency and noise-tolerant computation [12]. Recent works have exploited HDC data representation to revisit genome sequence matching algorithms for memory-centric computation [13], [14], [15]. These CiM approaches translate sequencing matching to highly parallel search operations that can be accelerated on content addressable memory (CAM). However, most existing HDC-based genomic algorithms either assume the CiM is ideal, do not work with multi-bit CiM, or the noise modeling is non-faithful to the technology being used. This paper presents a novel design that effectively deals with the constraints of the FeFET-based CiM architecture for the sequence matching problem and provides realistic modeling of the hardware non-idealities. Our work is fundamentally novel and provides the following contributions:

- **Faithful bridge between HDC algorithm and hardware constraints:** We employ physics-based FeFET models to capture their noise and variation accurately. Instead of the simple Gaussian distribution typically employed in other works [16], our FeFET models provide a realistic noise distribution for multi-bit precision, front, and back gate read, and 3nm and 10nm thick ferroelectric layers and temperature, which we take into consideration to design our learning framework.
- Although HDC representation provides robustness to the genome sequence matching process, the enhanced algorithms are still susceptible to technology noise during in-memory computation. Adapting to the non-idealities will result in losing information compared to the ideal case. **With that, we propose a framework that iteratively teaches HDC-based sequence matching algorithms to operate over non-ideal and noisy devices that exist in CiM architecture**. In other words, our framework teaches our HDC-based algorithms not to lose accuracy even under extreme technology noise in CiM architecture.
- **Capacity of memorization tied to hardware characteristics:** To ensure scalability and best fit of an algorithm into the CiM platform, we develop a theoretical framework that expresses HDC memorization capacity as a function of NVM bit precision and dimensionality. This model is a useful measure to determine the best data representation for the algorithm before actual deployment in hardware in any hardware-defined task.

We extensively evaluate the effectiveness of our framework in both theoretical and experimental settings. The evaluation shows that our cross-layer FeFET reliability modeling accurately captures the impact of FE scaling and temperature on errors induced by process variation and inherent stochasticity in multi-bit FeFETs. Our HDC learning framework iteratively adapts using a full-precision, ideal model for training and a quantized, noisy version for validation and inference. Our results demonstrate that highly-scaled FeFET, realizing 3-bit and even 4-bit, can withstand noise given high dimensionality during inference. If we introduce the noise during model adjustment, we can improve the inherent robustness compared to only adding noise during the matching process.

The rest of the paper is structured as follows: Section II illustrates an overview of genome sequence matching, its current HDC-based solution, and gives a CiM background. Section III introduces the FeFET basics and how our modeling is implemented. Section IV explains our HDGIM framework and section V theoretically defines the capacity of memorization dependent on the HDC hyperparameters and hardware constraints. Last, section VI evaluates the performance of our solution and explores its hyperparameters. Section VII gives an overview and conclusion of the work.

II. Related Work

Most modern genome sequence machines can generate a massive amount of data. Such effort is achieved by extracting small random fragments called reads. From a biological standpoint, a protein sequence is translated initially from an mRNA, which can be considered a string over the four alphabets, A, C, G, U. These reads are considered substrings that pass through a computational process known as read mapping, which takes each read, aligns it to one or more possible locations within the reference genome, and finds the matches and differences (i.e., distance) between the read and the reference genome segment at that location [17], [18]. Read mapping is the first critical step in genome sequence analysis.

With the advances in non-volatile memories, several CiM paradigms have been proposed. For instance, crossbars based on FeFET have been employed in neural networks [19] and nearest neighbor search [20]. On the other hand, binary/ternary/analog content addressable memories (CAMs) have been utilized in search-intensive tasks such as IP routers, lookup tables, and associative searches [21], [22]. Specifically, in conjunction with customized sense amplifiers (Fig. 1(c)), 2FeFET CAM (Fig. 1(d)) designs demonstrate great potential as a high-density and energy-efficient associative memory with Hamming and L_2 distance [6], [23].

CiM based on NVM technology has been widely used to accelerate genome sequence matching problems. For example, PIM-Aligner [24] and RAPID [25] are two recent CiM accelerators for alignment based on magnetic and resistive devices. However, the genome sequencing algorithms are not

memory-centric and often suffer from technology noise in CiM architectures. HDC is introduced as a novel computational model for robust and holographic data representation. Revisiting genome sequencing algorithms based on HDC makes the sequencing algorithms memory-centric and compatible with CiM architectures [13], [15]. Recent work in [15] presented a CAM architecture to accelerate a key functionality of HDC-based genome sequence matching. However, the existing platforms assume the CiM or CAM is ideal, do not work with multi-bit CiM, or the noise modeling is non-accurate with the technology being used. This paper presents a novel design that effectively deals with the constraints of the FeFET-based CiM architecture for the sequence matching problem and Provides realistic modeling of the hardware non-idealities.

Important parameters of the implementation are its one-bit precision and highly scalable architecture. HDC assigns a hypervector corresponding to each mRNA base alphabet A, C, G, U. Then, it generates a library of memory hypervectors that save multiple reads. The number of substrings stored on every memory hypervector is tied to the capacity, which is theoretically defined for 1 bit and dimension D. This implementation, however, is limited to one-bit precision CAM architecture. In contrast, FeFET-based CiM needs multi-bit precision to produce effective performance considering the noise.

III. Compute-in-Memory With FeFET Technology

Despite much research and progress at the device level, FeFET is still in the prototype stages and has not yet reached the mass market. Currently, the high write voltage is a limiting factor that makes dense integration with logic difficult. Therefore, the addition of the back gate to reduce the write voltage is a promising direction [26]. At the same time, FeFET devices inherently operate with a limited level of precision, traditionally limited to one bit (single level cell). This would require many FeFETs to present high-precison data which increases area consumption and integration complexity. A multi-bit FeFET can alleviate this complexity but introduces its own challenges such as reduced reliability.

A. FeFET Basics

HfO_2-based FeFET is a competitive candidate in low-power and high-speed edge-computing applications due to its CMOS compatibility [5], comparatively low read/write energy, and short read latency [6]. A FeFET is based on a regular CMOS MOSFET with only one modification to the gate stack depicted in Fig. 1(a). A typically 10nm thick ferroelectric (FE) HfO_2 layer is added, which can be polarized by applying a write voltage pulse to the gate terminal. Applying a positive voltage pulse sets FeFET to the down polarization state; the $I_D - V_G$ current is then high at the read voltage of about 1V as shown in Fig. 1(b). If the polarization is flipped, the $I_D - V_G$ current is low at the read voltage. The difference between the two currents represents two different states, making a FeFET a single device memory. The FE layer contains individual domains, symbolically represented as green and red boxes in Fig. 3, which are flipped by the write pulse. Depending on their direction, the FE-layer exhibits

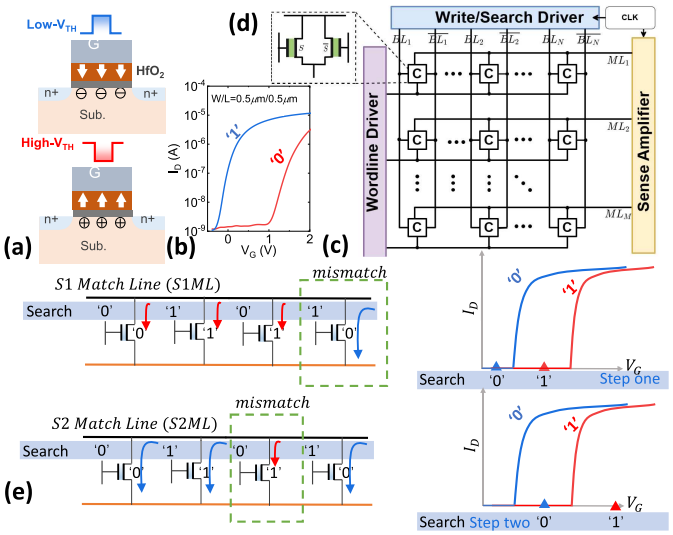

Fig. 1. (a) FeFET operation schematic. (b) SPICE simulation of FeFET $I_D - V_G$ curve for storing 1 and 0. (c) CAM array. (d) Single 2FeFET CAM. (e) Ultra-compact 1 FeFET CAM [27].

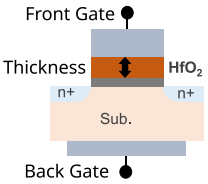

Fig. 2. A back gate is added for the read operation. Although it increases the impact of variability, it increases the memory window even more and thus allows for a reduction in the thickness of the ferroelectric layer.

a different polarization. However, the domains do not flip all simultaneously but are individually based on *stochastic processes*. Hence, the polarization is not purely binary but an analog value. Consequently, the $I_D - V_G$ current is also between the two end states. By applying a short or lower voltage pulse, only a portion of domains is flipped, resulting in an intermediate $I_D - V_G$ current. Such a property can be exploited to create a multi-stage cell to store multiple bits of information [9]. Scaling the thickness of the FE layer down reduces the number of domains. Thus, each domain becomes more impactful on the overall polarization of the device. Because the domain's polarization change is a stochastic process, the variability increases for highly-scaled FeFET devices with fewer domains [9]. This results in a much noisier device with a ferroelectric layer of just 3nm.

To counteract the increase in variation, the typical single access gate can be split into a front and back gate [26] as shown in Fig. 2. The state of the FeFET is written through the front gate, while the read operation is performed through the back gate. However, because of the increased distance between the back gate and the ferroelectric layer, the impact of variation is increased. Nevertheless, the memory window (MW), the difference between the most and least polarized state, is increased. A larger MW provides higher margins against design-time and run-time variation. This increase of the MW is stronger than the additional impact of variation from the back gate. Hence, the overall robustness of the read operation

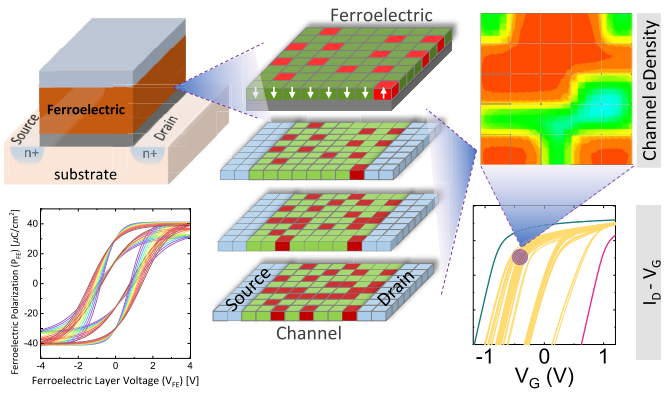

Fig. 3. Our FeFET modeling considers the inherent stochasticity of the domains in the ferroelectric layer. Their up/down polarization determines the electron density of the channel and thus the drain-source current $I_D - V_G$. Adding a back gate to the device allows for disturb-free reading and shrinking of the HfO_2 layer from 10 nm to 3 nm.

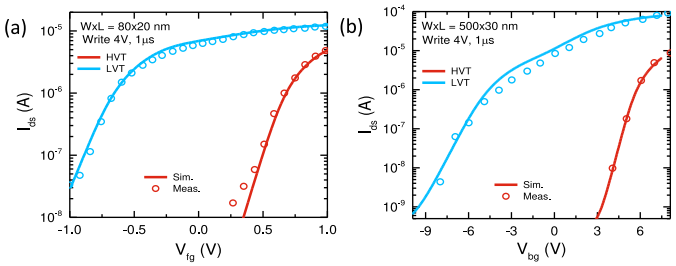

Fig. 4. Our modeling approach matches with experimental measurements from [26] for the 10 nm FE layer.

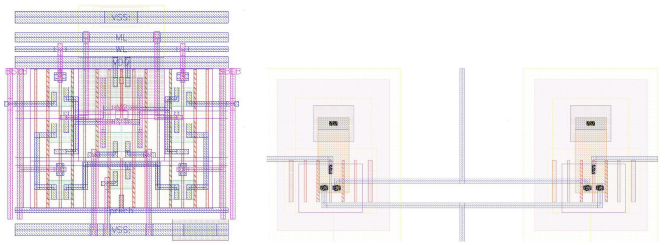

Fig. 5. Layout for an optimized SRAM-based CAM on the right and an FDSOI-base CAM cell that mimics our FE-FDSOI on the left. In this commercial 22 nm FDSOI technology, the back gate requires a large clearance, which increases the area by 1.2x compared to the SRAM-based CAM. However, the FE-FDSOI can store three and four bits per cell, resulting in an area/bit decrease of 2.5x and 3.3x, respectively.

is increased. The improved robustness opens the door for highly-scaled FeFET devices, which otherwise would be too noisy [9].

B. Our FeFET Modeling

In this work, we perform accurate cross-layer reliability modeling with the multi-physics simulator technology CAD (TCAD). We start from the underlying device physics, as follows: First, the FDOSI transistor is carefully calibrated to reproduce experimental measurements of commercial 22 nm FDSOI [28]. The device is a ferroelectric capacitor built from the same materials as a transistor. The parameters of the TCAD models for the incorporated FE layer (remnant polarization, saturation polarization, and coercive field) are calibrated against these measurements [9]. We then replace the HfO_2 layer of an FDSOI transistor model with our calibrated model of the FE material. Although the underlying device is different (capacitor vs. transistor), the model captures the properties of the ferroelectric material itself and thus can be applied to FDSOI transistors. The approach of extracting the FE material properties from a capacitor and applying them to a transistor has been demonstrated in previous work for other technologies and has been validated by experimental measurements [29]. This methodology is applied analogously for different operating conditions to include the temperature impact.

To capture the behavior of a computing system at idle, 27° C is selected, and 80° C for a system under load. Then, the model mentioned above parameters are calibrated to capture different temperatures. Existing TCAD models can capture the impact of temperature in the underlying transistor but not of the ferroelectric material. Hence, our additional modeling is indispensable to account for the degradation in the FE layer induced by an increased temperature. After the calibration, the $I_D - V_G$ characteristic is swept, and different FeFET states are derived from that. Our models have been validated against experimental measurements, and Fig. 4 shows excellent agreement. Monte Carlo simulations are performed to capture variability from FeFET (inherent stochasticity of polarization) and conventional sources (process variation) similar to our previous work [9]. The output is a VT distribution for each stage of the cell (e.g., eight distributions for a 3-bit cell). The mean and variance are extracted and provided to the HDGIM framework for further circuit-level modeling.

C. FeFET-Based Content-Addressable Memory (CAM)

FeFET-based CAMs comprise two FeFETs in contrast to ten CMOS transistors in an SRAM-based design [30]. Thus, significant area reduction is possible, especially for multi-bit FeFET-based CAMs. For a design with 28 nm technology, a 22.6x area reduction per bit compared to an SRAM-based design was reported [30]. At 45 nm, a 7.5x reduction was demonstrated for single-bit TCAM designs [6]. The employed commercial 22 nm FDSOI technology supports the back gate enabling a realistic area assessment. We have performed the layout for an optimized SRAM-based CAM cell and the proposed FE-FDSOI CAM cell under the assumption that the same DRC rules from the baseline FDSOI process apply to the FE-FDSOI. Fig. 5 demonstrates that FE-FDSOI actually increases the area by 1.2x because the back gate requires a large clearance. However, the FE-FDSOI can store three and four bits per cell, resulting in an area/bit decrease of 2.5x and 3.3x, respectively. Future changes in the fabrication process could lead to further reductions.

Regarding power, a FeFET-based multi-bit CAM designs with two FeFETs and one CMOS transistor was manufactured in [30]. They compare against an SRAM-based baseline and report a reduction in energy from 4.13 pJ to 0.87 pJ (4.7x) and area savings of 22.6x per bit. In [31], they manufactured a 2-FeFET CAM cell in a NAND array and report a similar reduction from 4.01 pJ to 0.63 pJ (6.4x) against an SRAM baseline. In our previous work [32], we calibrated our models

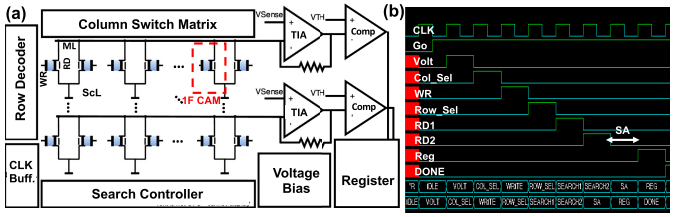

Fig. 6. (a) Circuit-level diagram for the HDGIM framework. (b) Controller signals for all the required operations in HDGIM.

to reproduce measurements from Intel's 14 nm FinFET process. We then modeled a 2-FeFET and a 16-T SRAM-based TCAM cell and connected multiple with a joined match line for Hamming distance computation. In contrast to the other works, we obtained a 0.07x increase in energy because of the higher latency of the FeFET-based TCAM. The major difference of this work to previous works is the addition of the back gate. The back gate enables the reduction of the FE layer, which enables the reduction of the write voltage and, therefore, a reduction in energy [9].

In this work, a CAM is employed to identify the stored sequence that exactly matches the query [6]. With the FeFET-based CAM approach, given a query sequence, one can efficiently search across all the stored sequences in terms of Hamming distance. Specifically, the reference DNA strings are written into CAMs during the writing phase by high/low voltages representing the nucleotides. Then, when a query DNA comes, the high/low voltages are again applied to the FeFET gate terminal to form a high/low I_D current. In this work, by leveraging the above accurate FeFET modeling, we can characterize the non-idealities of CAMs.

D. Threshold Matching With CAMs

We propose HDGIM, a hyperdimensional genome sequence matching framework that adapts to the non-idealities of FeFET CAMs. An area-efficient three to four-bit ultra-compact CAM that performs Hamming distance, i.e., bit-wise XOR operation, is highly desired. In HDGIM, we propose to use the binary CAM (BCAM) from [27]. Unlike conventional single-bit CAM with 2 FeFETs (Fig. 1(d)) and single-step search, single-bit ultra-compact CAM shown in Fig. 1(e) necessitates 1 FeFET with 2 step searches. As the FSDOI-based FeFET CAM is able to handle 4-bit per cell, an N-bit CAM only requires N/4 FeFETs. Specifically, it can be seen that in step one, the "store0search1" (st0sr1) cell is detected, and in step two, the "store1search0" (st1sr0) cell is then detected. Since the total Hamming distance (HD) between the query and a stored vector is simply the number of "store0search1" and "store1search0" cells. By the fact that $I_{S1ML} = I_{ON}N_{st0sr1}$ and $I_{S2ML} = I_{ON}(N_{total} - N_{st1sr0})$, we obtain [27]:

$$Ham.dist. = N_{st0sr1} + N_{st1sr0} \propto I_{S2ML} - I_{S1ML} \quad (1)$$

Then, by changing the reference of the sense amplifier for detecting rows exceeding the threshold, the HDGIM framework is able to handle multi-bit CiM genome sequence matching. Here, we further discuss the required peripherals of the in-memory HDGIM array. Specifically, we choose to leverage a current-based sensing scheme for the multi-bit HDGIM CAM, which bypasses the time-sensitive voltage drop sensing [6]. Unlike previous CAM sensing circuits built with analog-to-digital converters [33], HDGIM only necessitates a threshold-adjustable comparator that outputs the CAM row's label with the highest similarity. As discussed in Section I, with the write voltage of the FDSOI-based FeFET reduced to I/O-compatible $\pm$1.85V, high-cost level shifters [34] and isolation sensing path [35] comprised of high-voltage-tolerant MOSFETs are eliminated. Trans-impedance amplifier (TIA) and comparator serve as the sense amplifiers (SAs), where TIA provides high-resolution current-to-voltage conversion for HD current from a CAM row [36], and the comparator compares the voltage-based HD value with the HDGIM threshold value T [37]. Fig. 6(b) describes the required control signals for the HDGIM array (obtained from system verilog), which comprises the voltage biasing (Volt), column selection (Col_Sel), write from FDSOI-based FeFET front gate (WR), row selection (Row_Sel), step-one search from the back gate (RD1), step-two search from the back gate (RD2), and register enable (Reg).

IV. HDGIM Framework

Fig. 7 shows an overview of genome sequence search in the high-dimensional space. The first step is to encode the genome sequence into a high-dimensional space. It assigns a hypervector corresponding to each base alphabet in $\Sigma = \{A, C, G, T\}$ for DNA. The encoding module depends on the data type and the genomics task [14]. We aggregate all encoded sequences to generate a reference genome, called *HDC Library* that we will consider as our model. An HDC library consists of several reference hypervectors, where each hypervector memorizes thousands of genome sequences in high-dimensional space. During the sequence searching, HDC uses the same encoding to map a query sequence into a hypervector. We perform a similarity computation between a query and each reference hypervector. By searching for an exact or approximate match, it identifies a query's similarity with thousands of memorized patterns stored in each HDC library hypervector.

Our framework consists of two models: the full-precision ideal model and the deployed model, which will be the one that has been adapted in bit-precision and receives the noise effects, in order to be used in a FeFET-based CAM. The framework hyperparameters for initialization consist of the bit-precision for the deployed model B, the number of dimensions in every hypervector D, the chunk size n, and the discharge current matrix M^c, containing mapping values to compute similarities since we do not have available dot product as a similarity function. The mapping values are the current discharges given two symbols being compared.

A. HDGIM Encoding

In this step ❶, the model encodes the given DNA sequence into high-dimensional space. To achieve this, the model first samples $D-$dimensional vectors $\vec{H}_\sigma \in \{x \in \mathbb{R} | -\pi < x < \pi\}^D$ uniformly randomly for each $\sigma \in \{A, C, G, T\}$.

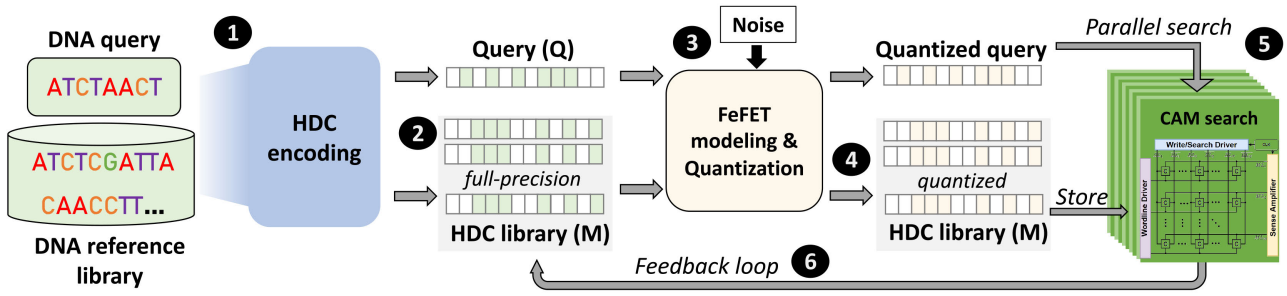

Fig. 7. Overview of HDGIM sequence matching in the hyperdimensional space.

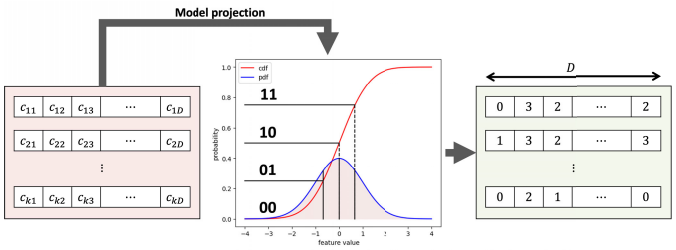

Fig. 8. Overview of model projection where $B = 2$.

Next, the model splits DNA sequence R into small overlapping chunks R_j with length n by window sliding. For example, if we have $R =< A, C, G, G, T >$ with $n = 3$, resulting small chunks would be $< A, C, G >$, $< C, G, G >$, $< G, G, T >$. After this, each chunk is encoded by binding the base hypervectors into high-dimensional space $\vec{S}_i = \vec{H}_{R_j} * \rho\vec{H}_{R_{j+1}} * \rho^2\vec{H}_{R_{j+2}} * \cdots * \rho^{n-1}\vec{H}_{R_{j+n-1}}$, where ρ represents a rotational shift. Lastly, the encoded hypervector of the DNA sequence S is computed by bundling all its chunk hypervectors S_i, $S = \sum_i S_i$ ❷.

B. Hardware Adaptation Components

Before we describe the learning model, we proceed to describe the adaptation of the hardware modeling to our framework, which consists of a modified similarity function based on our designed CAM, bit-precision quantization, to convert our full-precision model to an N-bit precision that fits the FeFET-based CiM used and the noise introduction to the model.

1) Quantization: The quantization is conducted on the query and on the model projection step ❹, where we adapt the ideal model to the bit-precision constraint. It quantizes the components of the hypervector to B bit symbols. Since feature values do not follow uniform distribution in general, it calculates scores of the HDC components and uses the cumulative normal distribution function to quantize feature values. Fig. 8 shows model projection on 2-bits.

2) Noise Modeling: In order for the model to simulate a FeFET-based CiM we use the noise modeling explained in Section III. We use the experiment parameters to generate the distribution. It causes a symbol v to increase or decrease to $v+1$ or $v-1$. Each value in the quantized hypervector has a certain probability of stochastic variation. For instance, if random changing probability is p, a value v in the hypervector will be changed to the value of $v+1$ with probability p_v^{right} and $v-1$ with probability p_v^{left} where $\frac{\sum_{i=0}^{2^B-1} v_i^{left} + v_i^{right}}{2^B} = p$. If $v-1$ is less than 0 or $v+1$ is greater than 2^B-1, only an increase or decrease will be applied with probability p respectively. An example is given in Tab. I. We then apply the noise during inference on the model projection step ❸. The values remain unchanged until the next model projection step.

TABLE I

PROABLISTIC ERROR MODEL AS THE INTERFACE BETWEEN HARDWARE AND SOFTWARE

Intended value v_i	Probability of actually stored value v_a		
	$p_{v_a}^{left}$ (%)	Correct v_a (%)	$p_{v_a}^{right}$ (%)
0	0.00	99.80	0.20
1	0.45	99.32	0.23
2	0.46	99.03	0.51
3	0.45	99.08	0.47
4	0.47	99.05	0.49
5	0.18	99.33	0.50
6	0.13	99.68	0.19
7	0.14	99.86	0.00
Average	0.29	99.39	0.32

Error model for a write of value v_i through the back gate at 27 °C with a 3 nm thick FE layer. The error probabilities are given for the case that the actually stored value v_a is too low ($p_{v_a}^{left}$, $v_a = v_i - 1$) or too high ($p_{v_a}^{right}$, $v_a = v_i + 1$). The model noise value is the average probability of $p_{v_a}^{left}$ and $p_{v_a}^{right}$.

3) Similarity Function: Given two DB-bit(s) hypervectors H_1 and H_2, current similarity $\delta(H_1, H_2)$ is computed using $\vec{d} = |H_1 - H_2|$ and M^c indicating current matrix containing mapping values for computing similarities ❺. Now, the similarity is computed as follows:

$$\delta(H_1, H_2) = \sum_{i=0}^{D-1} M^c_{\vec{d}_i}$$

Note that $0 \leq \vec{d}_i < |M^c| = 2^B$.

C. Iterative Training

For each training step, training data with labels are given to the full-precision model. Each data has a query DNA sequence of n size, which is equal to the size of each chunk used in the encoding step, with a label value indicating whether the query DNA sequence is contained in the model's DNA sequence R or not. Once we reach the validation phase of our training, instead of testing it with the full-precision model, we use

the quantized model. Since this is not a classification but a detection task, the model requires threshold value T in order to decide if a sequence pertains to the HDC library. Once the evaluations are done, we test for multiple threshold values and pick the value with the best accuracy, which is dependent on the dimensionality of the hypervector and error probability.

For each query DNA sequence $\vec{Q}$ with label value, the full-precision model is updated as follows ❻:

$$\begin{cases} \vec{S} \leftarrow \vec{S} - \alpha\vec{Q} & \text{if } \delta(\vec{S}^q, \vec{Q}^q)/D \geq T \text{ and } Q \notin R \\ \vec{S} \leftarrow \vec{S} + \alpha\vec{Q} & \text{if } \delta(\vec{S}^q, \vec{Q}^q)/D < T \text{ and } Q \in R \end{cases}$$

where α indicates the learning rate and T indicates the threshold value. $\vec{Q}^q$ and $\vec{S}^q$ are quantized $\vec{Q}$ and $\vec{S}$ to B-bit(s) respectively. M^c is used in δ which is computing similarity between the given hypervectors. Finally, $\vec{S}^q$ is computed from updated $\vec{S}$ by a model projection that repeats the quantization step ❹.

V. Capacity of Hardware-Based HDC

In this section, we evaluate the memory capacity of our classifier. The notion of capacity in HDC measures the ability of the HDC model to represent and approximate complex functions for learning tasks. Specifically for our matching tasks, it serves as a measure of HDC memorization and establishes some analytical relation between the parameters of the model, the data, and the performance of the model through the lens of information theory. This knowledge facilitates many model deployment tasks, including scaling models for larger datasets, tuning hyperparameters for desired performance, and comparing our model with other ones, including neural networks. In particular, we provide an analysis where the information capacity is evaluated with subsets of symbolic input as memory entries (since each sequence is a bundling of seed hypervectors as opposed to a single one). The analysis consists of four parts:

1) Deriving the distributions of the similarity between a class hypervector and a query DNA hypervector for cases where the DNA is in the class and where is it not.
2) Inferring detection accuracy of the model from said distribution and model parameters, including the threshold.
3) Evaluating the information-theoretic memory capacity of the model based on previous metrics.
4) Discussing the information loss from quantization.

A. Setup

We use dot product as the similarity metric: $\delta(\vec{Q}, \vec{\mathcal{C}}_l) = \vec{Q}^T\vec{\mathcal{C}}_l$. In addition to the notation in Section IV, we assume a fixed number of chunks per DNA sequence m, the number of entries in the classifier V, and the alphabet size $a = |\{A, C, G, T\}| = 4$. In particular, the number of chunks m is introduced to make the analysis more general, where the stride of the encoding can be tuned. In our encoding, stride $s = 1$, meaning that consecutive chunks are displaced by 1 symbol; in this case, $m = \frac{l-n+1}{s} = l - n + 1$. In addition, we assume a static model for memorization, where model $\vec{S}$ is the bundling of all encoded DNA hypervector $\vec{\mathcal{S}}^{(i)}$: $\vec{S} = \sum_{i=1}^{V} \vec{\mathcal{S}}^{(i)}$.

B. Sequence Signal

We first characterize the distribution of similarity between a query and a class hypervector. To do so, we begin by estimating the signal (similarity evaluation with a class hypervector) of each chunk of the query, denoted $\vec{Q}_k$ for the k^{th} chunk. If Q belongs to the class, then $Q = S^{(o)}$ for some $o \in [1, V]$. Applying the linearity of the dot product twice (to the granularity of sequences and then to that of chunks),

$$\begin{aligned} &\delta(\vec{Q}_k, \vec{S}) \\ &= \sum_{i\in\mathcal{C}_l} \delta(\vec{Q}_k, \vec{\mathcal{S}}^{(i)}) = \delta(\vec{Q}_k, \vec{\mathcal{S}}^{(o)}) + \sum_{o\neq i\in\mathcal{C}_l} \delta(\vec{Q}_k, \vec{\mathcal{S}}^{(i)}) \\ &= \delta(\vec{Q}_k, \vec{\mathcal{S}}_k^{(o)}) + \sum_{j\neq k} \delta(\vec{Q}_k, \vec{\mathcal{S}}_j^{(o)}) + \sum_{o\neq i\in\mathcal{C}_l} \sum_{j=1}^{m} \delta(\vec{Q}_k, \vec{\mathcal{S}}_j^{(i)}) \end{aligned}$$

The first term in (V) is clearly D. As for the other $(m-1)+(V-1)m = Vm-1$ terms, we notice that if the chunk underlying the hypervector happens to be the same, then a "spurious", or false positive, the signal of D will be generated; otherwise, the noise follows the normal distribution $N(0, \sigma^2)$ for some σ. While it is certainly the case that consecutive chunks are correlated and their similarity hence depends on the exact sequence of alphabets within the chunk, the total number of chunks within $\vec{S}$ is high enough that this approximation is reasonable. In the case of bipolar hypervector, $\sigma = \sqrt{D}$. Assuming a uniform random distribution of alphabets within DNA sequences, the probability of a false positive chunk is A^{-n}. Thus,

$$\begin{aligned} &\delta(\vec{Q}_k, \vec{S}) \\ &\approx D + \sum_{\substack{i=1 \\ (i,j)\neq(o,k)}}^{V} \sum_{j=1}^{m} (\Pr(\vec{Q}_k \neq \vec{\mathcal{S}}_j^{(i)})N(0, \sigma^2) + \Pr(\vec{Q}_k = \vec{\mathcal{S}}_j^{(i)})D \\ &= (1 + (mV-1)A^{-n})D + (1 - A^{-n})N(0, (mV-1)\sigma^2) \end{aligned}$$

Collectively, a query signal consists of the signal from m chunks. Hence,

$$\begin{aligned} \delta(\vec{Q}, \vec{S}) \sim\ & m(1 + (mV-1)A^{-n})D \\ & + (1 - A^{-n})N(0, m(mV-1)\sigma^2) \end{aligned}$$

Similarly, the similarity between $\vec{Q}'$, a sequence not in the class, and $\vec{S}$ is

$$\delta(\vec{Q}', \vec{S}) \approx m^2VA^{-n}D + (1 - A^{-n})N(0, m^2V\sigma^2)$$

To further simplify the analysis, we consider appropriate parameter settings. In practice, we would like to memorize as many DNA sequences as we can. This means $V \gg 1$; in addition, we would like the probability of a false positive chunk to be low, so $A^n \gg 1$. Finally, This leads to much simpler distributions:

$$\begin{aligned} \delta(\vec{Q}, \vec{S}) &\sim N(m(1 + mVA^{-n})D, m^2V\sigma^2) \\ \delta(\vec{Q}', \vec{S}) &\sim N(m^2VA^{-n}D, m^2V\sigma^2) \end{aligned}$$

C. Detection Accuracy

From the signal and noise distributions derived above, we can estimate the detection accuracy given a threshold, which implies the quality of the memorization (per item). More precisely, we care about both true positive rate and false positive rate, TPR^{θ} and FPR^{θ}:

$$\begin{aligned} TPR^{\theta} &= \Pr(\delta(Q,S) > \theta | Q \in S) \\ &= \Phi\left(\frac{m(1+mVA^{-n})D-\theta}{m\sigma\sqrt{V}}\right) \\ FPR^{\theta} &= \Pr(\delta(Q,S) > \theta | Q \notin S) \\ &= \Phi\left(\frac{m^2VA^{-n}D-\theta}{m\sigma\sqrt{V}}\right) \end{aligned}$$

where Φ is the cumulative distribution function (cdf) of the standard normal distribution. The rates can be simplified further as we rescale the threshold; namely, we set θ' such that $\theta = m\theta' + m^2VA^{-n}D$. We thus derive the true positive rate and false positive rate function tp, fp with respect to scaled threshold θ':

$$tp(\theta') = \Phi\left(\frac{m(D-\theta')}{m\sigma\sqrt{V}}\right) = \Phi\left(\frac{D-\theta'}{\sigma\sqrt{V}}\right) \quad (2)$$

$$fp(\theta') = \Phi\left(\frac{\theta'}{\sigma\sqrt{V}}\right) \quad (3)$$

The detection accuracy Acc can then be derived from the two, with an additional parameter p_s on the probability that a query belongs to the class:

$$\mathrm{Acc}^{p_s}(\theta') = p_s\Phi\left(\frac{D-\theta'}{\sigma\sqrt{V}}\right) + (1-p_s)\left(1-\Phi\left(\frac{\theta'}{\sigma\sqrt{V}}\right)\right) \quad (4)$$

D. Memory Capacity

Following the methodology in [38]. we define the memory capacity as its information content: the mutual information between true inputs and the inputs retrievable from the model $\vec{S}$. Notice that because the model is only for detection, the analysis for the mutual information collapse to an analysis over the distribution with respect to the membership of DNA sequences instead of over that of DNA sequences themselves. Let $\hat{s}$ be the random variable of the detector output and s be the membership of a query. For simplicity, let the support of $\hat{s}, s$ be $\{0, 1\}$, indicating undetected/detected for $\hat{s}$ and not present/present for s, respectively. Therefore, under fixed parameter p_s and threshold θ', the mutual information between the set of DNA sequences $\{\vec{S}^{(i)}\}$ and the model $\vec{S}$ is

$$\begin{aligned} I(\{\vec{S}^{(i)}\}, \vec{S}) &= D_{KL}(\Pr(\hat{s}, s) || \Pr(\hat{s})\Pr(s)) \\ &= \sum_{i,j\in\{0,1\}} \Pr(\hat{s}=i, s=j)\log_2\frac{\Pr(\hat{s}=i, s=j)}{\Pr(\hat{s}=i)\Pr(s=j)} \end{aligned}$$

where D_{KL} is the KL divergence [39]. By definition, $\Pr(s)$ is described succinctly by p_s ($\Pr(s=1) = p_s$). $\Pr(\hat{s}=1) = tp \cdot p_s + fp \cdot (1-p_s)$ is the marginal probability with which the detector outputs "detected". The joint probability $\Pr(\hat{s}, s)$ can be computed by the conditional probability $\Pr(\hat{s}|s) = \Pr(\hat{s}, s)/\Pr(s)$, whose values are the true/false positive/negative rates of the detector. The memory capacity is simplified as

$$\begin{aligned} &I(\{\vec{S}^{(i)}\}, \vec{S}) \\ &= p_s(tp\log tp + (1-tp)\log(1-tp) - \log Z) \\ &\quad + (1-p_s)(fp\log fp + (1-fp)\log(1-fp) - \log(1-Z)) \end{aligned}$$

where $Z = \Pr(\hat{s}=1) = tp \cdot p_s + fp \cdot (1-p_s)$, and θ' is implicit.

E. Effect of Quantization

We now discuss the effect of quantization on our model. Due to the lack of theoretical work on low-fidelity quantization, which would be more proper for our case of quantizing each component to $3 \sim 4$ bits, we analyze the impact of quantization using high-resolution quantization theory.

As mentioned previously, during quantization, we fit a Gaussian distribution over the collection of hypervector components and quantize each bit according to its percentile. As this quantization method takes into account the statistics of all hypervector components (D of them), and each component is quantized to the same amount of bits (encoding rate is fixed), it is classified as a fixed-rate $D-$dimensional quantizer. As a result, the quality of the quantizer $\delta_D(R)$ has an upper bound of

$$\delta_D(R) \cong M_D\beta_D\sigma^2 2^{-2R}$$

where $\delta_D(R)$ is the operational rate-distortion function of the quantizer, and MSE measures the distortion. R is the bit rate, $M_D \xrightarrow{D\to 0} (2\pi e)^{-1}$ is Gersho's constant [40] that accounts for the least normalized moment of inertia of $D-$dimensional tessellating polytopes, $\beta_D \xrightarrow{D\to 0} (2\pi e)$ is Zador's factor [41], and σ is the standard deviation of the source (assumed gaussian). As suggested in [42], the high dimensionality of the quantizer allows close approximation of the constants. Furthermore, the performance of the D-dimentional quantization method converges to that of the optimal D-dimentional quantizer as D approaches infinity. This is the result of Asymptotic Equipartition Property [43], a standard assumption in information theory, which states that when the dimension is large, the dimensional probability density for a stationary, ergodic source with continuous random variables is approximately constant with overwhelming probability. This results in both our HDC quantizer and the optimal quantizer collapsing to a uniform quantizer over the support of the source distribution [42].

One thing worth mentioning is a scalar quantization method typically used for the HDC model: instead of evening out the percentile for each quantization level based on component distribution, this method even out the length for each level based on the range of the components. While the scalar quantization is clearly more efficient, it incurs a space-filling loss of $\frac{\pi e}{6} \approx 1.53dB$[1] [42].

[1]Which is the ratio of the normalized moment of inertia of a cube to that of a high-dimensional sphere.

VI. Evaluation

A. Experimental Setup

The proposed framework has been executed with a software framework. Our software framework is implemented using Pytorch and supports HDC encoding and classification.

Our dataset comprises extensive DNA strings randomly selected from an E. coli [44] dataset to simulate a patient's genome library. Regarding preprocessing, we refrain from applying any modifications to the dataset sequence. Instead, we employ our HDC's sequential encoding algorithm to generate hypervectors, a key design component described in section IV-A. We study the effectiveness of our technique over a randomly selected DNA sequence with 1,000 lengths. To test our model, we have a positive set of 50 samples generated as substrings of DNA sequence and a negative set, which consisted of 50 random queries that did not pertain to the DNA sequence. Each test sequence has a size of 10, and negative samples were sampled from a uniform distribution.

To meticulously assess the efficacy of our algorithm's learning capabilities, we have strategically chosen to emphasize accuracy as our primary evaluation metric, which consists of the percentage of sequences properly tagged either as pertaining or non-pertaining to the DNA library. This choice enables us to maintain a consistent and equitable assessment across various scenarios and environmental settings, all of which are influenced by different configurations of the FeFET circuits. In each evaluation, we executed 10 epochs on the iterative training with a learning rate α of 1.0. Also, to have maximum performance for each evaluation, we evaluated with 100 different threshold values T.

We evaluate our framework with a FeFET model of 3nm and 10nm thickness at 27 °C and 80 °C, where each configuration yields different noise probability distributions. Subsequently, we study the effect of bit-precision, which affects the information loss and is a configuration hyperparameter to the modeling of the noise, dimension of hypervectors, which for HDC, is known to be a main hyperparameter to define accuracy and noise quality. Finally, we compare the effect of adding noise during training and inference in the hopes of proving that introducing the noise during training will allow the model to learn to adapt to the non-idealities of the device. In hardware, we implement and test our method on CiM using TCAD for FeFET device analysis modeling, HSpice for circuit level evaluation, and our in-house cycle-accurate simulator to verify HDGIM functionality in architecture and application levels. All reported results are end-to-end, including the overhead of codebook generation, encoding, HDC library generation, iterative quantization, noise modeling, and model update. Our simulator is connected to PyTorch for easy programmability and maximum efficiency.

In terms of comparison, in the field of genome sequencing, the only significant effort where hyperdimensional computing was used in conjunction with Computing in Memory (CiM) can be found in [15], showing promising results for a 1-bit circuit. However, there was a problem: the CiM circuit they used didn't consider the non-idealities we consider, such as noise during inference or training, and as a result, the device was not realistically accurate.

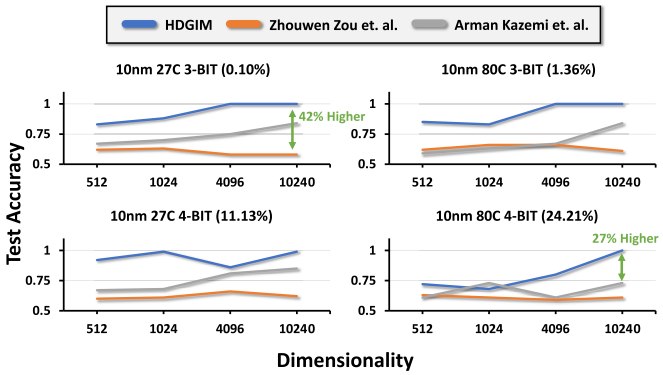

Fig. 9. Performance of HDGIM and the adapted works in [15] and [45] by applying our modeled noise.

Another tried to adapt the same hyperdimensional computing idea for CiM, but they were using it for classification, not matching genome sequences [45]. Additionally, they didn't need to use the same advanced model structure that we used for our application, which consists of two models: one full-precision model that contains the information and a simulated version of the deployed model, with quantization and noise application.

We used both HDC models and applied them to the genome-matching task with our considered non-idealities that can happen in the CiM device we used. Our results, as shown in 9, reveal that our approach performs better than previous attempts. By addressing the imperfections in the CiM device and optimizing the HDC model for genome matching, we've made significant progress in improving the accuracy and effectiveness of this critical task in genomics research, showing improvements for up to 42% against the original work of BioHD and 27%.

Our reference baseline aims for peak precision on a noise-free GPU model, ideal for generating our Computing in Memory (CiM) deployed model, which can be found in Figure 5 ❷. However, it neglects real-world device noise, notably CiM's inherent noise, due to factors like temperature fluctuations and bit-level limitations. This omission maintains model accuracy even after quantization for CiM adaptation, favoring uniform quantization, which handles information loss without noise.

In our unique situation, we lack comparable models for fair analysis. The closest approximation is the HDGIM model, configured with 3nm thickness at a 27Â°C base temperature and a 10nm thickness variation (representing only 0.04% noise probability). Surprisingly, this modest noise consideration sustains our model's 32-bit precision performance, as shown in Figure 8. Subsequent experiments aim to closely mimic this desirable scenario in achieving high accuracies.

B. Highly-Scaled FeFET Performance

In this experiment, we evaluate HDGIM using the highly-scaled FeFET of 3nm thickness and compare it to the 10nm thick one. The exploration was done through several dimensionalities. The noise modeling applied had a 39.7% error probability for the 3nm one and 1.03% for the 10nm. The noise can shift one symbol to a neighboring one and the

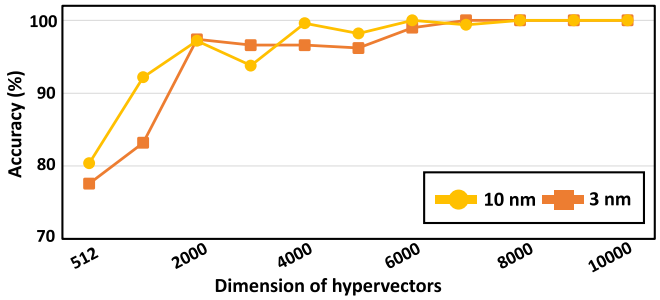

Fig. 10. Performance of HDGIM modeling a 10nm and 3nm thick FeFET.

TABLE II
NOISE LEVELS FOR DIFFERENT THICKNESSES OF THE FERROELECTRIC LAYER, BIT PRECISION, AND TEMPERATURES

HfO_2 Thickness (nm)	Precision (bit)	Front/Back Gate Read	Temperature (°C)	Modeled Noise (%)
10	3	B	27	0.02
10	3	B	80	0.60
10	3	F	27	0.05
10	3	F	80	1.03
10	4	B	27	6.95
10	4	B	80	19.09
10	4	F	27	7.86
10	4	F	80	21.89
3	3	B	27	0.60
3	3	B	80	5.22
3	3	F	27	39.71
3	4	B	27	19.09
3	4	B	80	35.28

probability is not equal for each side contrary to the experiment for the noise exploration where it has equal probability for each side. The noise was only considered during inference. Fig. 10 demonstrates the high accuracy of our framework even for the noisy 3nm cases, only requiring a hypervector dimension of 6000 to achieve perfect performance. This confirms that our framework when deployed to the FeFET-based CiM, will offer great capabilities for a genome sequence task and thus takes full advantage of a computing-in-memory architecture. The results further show that highly-scaled 3nm FeFET can be employed for actual application tasks despite their high variation.

C. High-Temperature FeFET Performance

In this subsection, we extend the study of the performance of HDGIM using the different temperatures on the baseline FeFET and its highly-scaled counterpart. The exploration is structurally the same as the previous experiment, only changing the noise modeling values depending on the temperature. The modeling resulted in Tab. II.

Fig. 12 demonstrates the high accuracy of our framework even for the noisy 3nm cases, only requiring a hypervector dimension of 6000 to achieve perfect performance. This confirms that our framework, when deployed to the FeFET-based CiM, will offer great capabilities for a genome sequence task and thus take full advantage of a computing-in-memory architecture. The results further show that highly-scaled 3nm FeFET can be employed for actual application tasks despite their high variation.

D. Noise Effect on Bit-Precision and Dimension of Hypervectors

Given the results achieved in the previous section, we decided to analyze the impact of manipulating several parameters on the model. We generated multiple instances of HDGIM with different bit-precision and dimensions and applied a probability of noise ranging from 0 to 60%. Fig. 14 shows the results of the exploration.

The results indicate that without noise HDGIM works accurately at any bit-precision. It is able to memorize all the patterns and not lose accuracy, compared to the full-precision model. However, a probability of error in the range of 20-60% impacts the accuracy of all the models. They require higher dimensional hypervectors to maintain the same performance as the full precision model. An exception is a 1-bit model, which fails to improve with higher dimensions for more than 20% error probability.

We observe that a 4-bit precision FeFET-based CAM with at least a dimension of 6000 is able to perform almost as well as the full-precision model. Even though the error probabilities can be high, since the hypervector patterns are highly separated in the hyperspace and the changes are only done to neighboring symbols, these effects are not enough when we have enough representations of values. This is the reason that the 4-bit precision is the only one to successfully surpass the severe error constraint.

Considering that the smallest thickness of our FeFET model is 3nm for 3-bit precision and that the probability of error consists of 39.71%, this shows that we can work with any bit-precision FeFET-based CiM, except the 1-bit, which was considered in the previous implementation for Genome Sequence Matching on CiM [15], and in this case, would be unable to perform under these circumstances.

E. Comparison of Noise During Training and Inference

The next step is considering the impact of adding the noise perturbations during iterative training, to observe if the model is able to learn to adapt to the effect of the noise. We considered only the cases from 2 to 4-bit precision. Fig. 13(a) corresponds to the noise added during inference and (b) during iterative training.

The results show that introducing noise before inference results in better performance overall for the same dimensionality. Most importantly, the 3-bit precision can achieve higher accuracy (lighter zones shown in Fig. 13b) at lower dimensions. The highlighted zones show that for the same area dependent on the dimension and noise values, introducing noise during the model adjustment increases accuracy by 8.4% on average. The 2-bit case starts to perform better for smaller noises but the improvements are smaller for higher perturbations (range 60-100%). Lastly, 4-bit was already robust, and introducing noise during training does not significantly improve the accuracy.

F. Energy and Speedup Hardware Comparison

In this section, we provide an assessment of the energy and speedup costs associated with implementing our algorithm

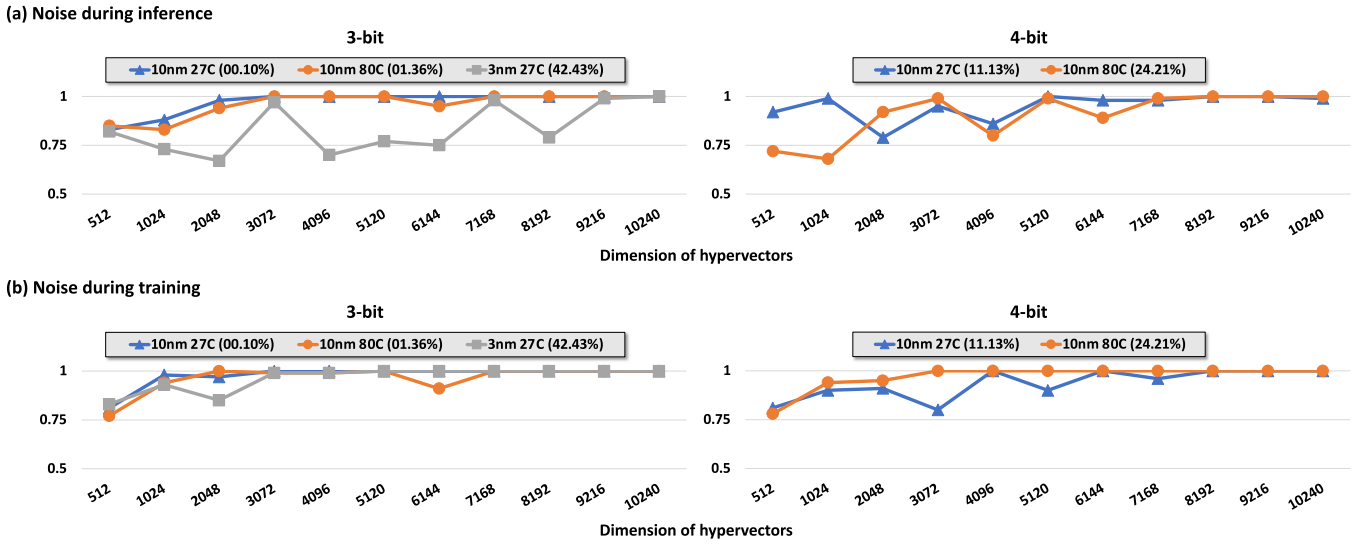

Fig. 11. Performance of HDGIM modeling the temperature for 10nm and 3nm thick on the Front gate FeFET configuration.

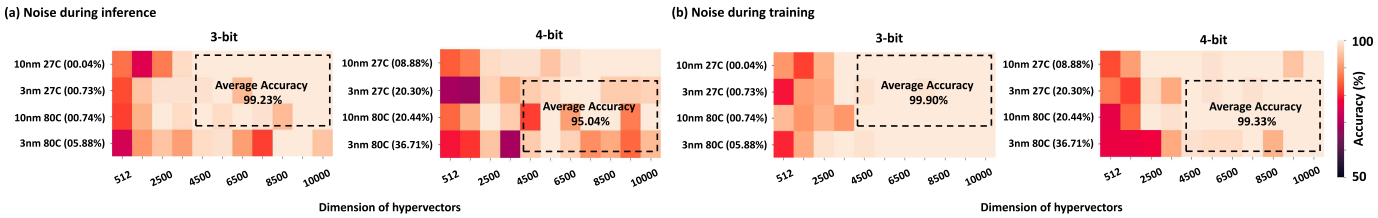

Fig. 12. Performance of HDGIM modeling the temperature for 10nm and 3nm thick on the Back gate FeFET configuration.

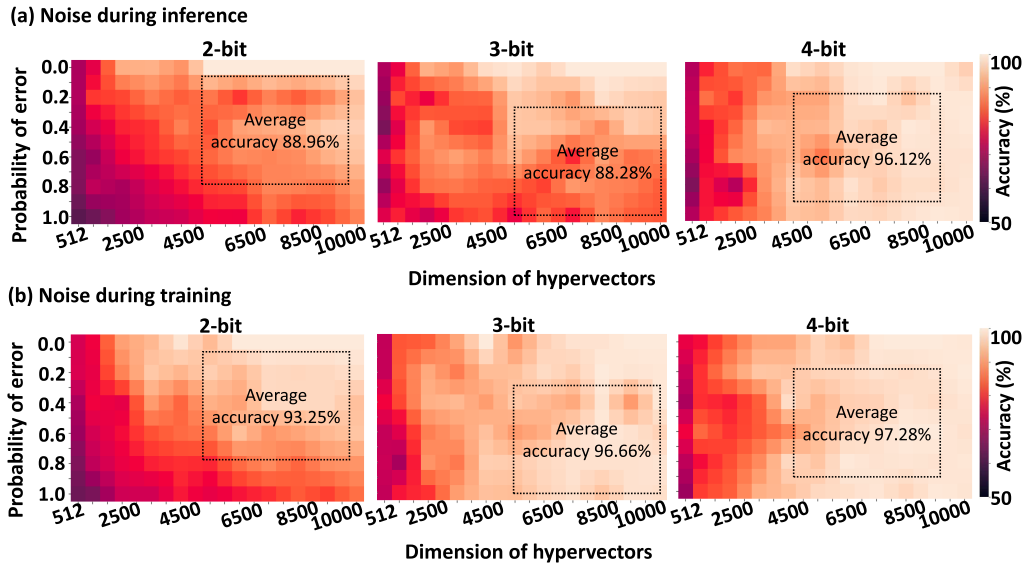

Fig. 13. Heatmap of accuracies for (a) noise introduced during inference and (b) training for 2, 3, and 4 bits.

on cutting-edge hardware designs, comparing it with our FeFET design. While there are various implementations for genome sequence matching using different machine learning algorithms, none involve in-memory computation with the ability to achieve the bit precisions we address. Comparing our work to these other algorithmic implementations would be inappropriate and beyond the scope of this study. Additionally, existing works in this field often neglect to incorporate variability effects due to process variation and temperature into the model from emerging technologies, which is a vulnerability in Deep Neural Network implementations that we have demonstrated in prior work. Our evaluation, previously presented in this section and depicted in Figure 9, illustrates the performance disparities of the algorithm across various Computing Memory FeFET cell configurations.

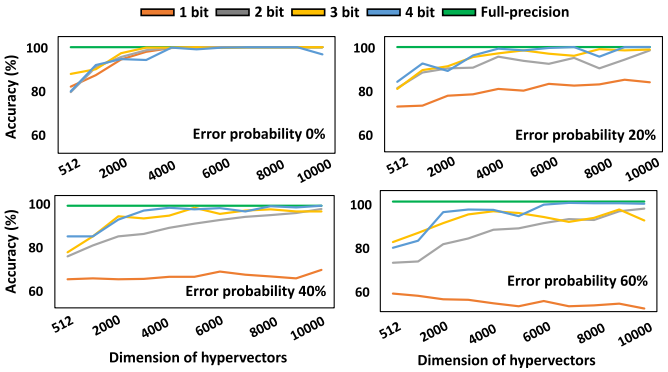

Fig. 14. Exploration of multi-bit HDGIM for different noise probabilities.

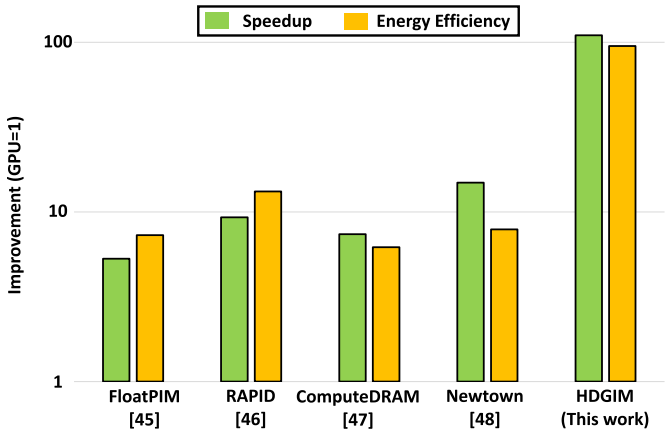

Fig. 15. Speedup and Energy efficiency of HDGIM on different Compute-in-Memory hardware frameworks for 1-bit precision, using GPU on GPU as the baseline.

For a comprehensive analysis, we compared only hardware setups regarding energy and speedup using our algorithm. Our study includes a thorough comparison of speedup and energy efficiency for different state-of-the-art Computing in Memory architectures implementing our HDGIM algorithm. The benchmark for this comparison is a GPU with NVBIO [46], a GPU-accelerated C++ framework designed for High-Throughput Sequence Analysis. We focused our evaluation solely on 1-bit precision since it is not the main emphasis of our work. The evaluation incorporates distinct hardware acceleration technologies, such as ComputeDRAM [47] and Newtown [48] as DRAM-based accelerators, RAPID [25], a recent PIM accelerator for genome tasks based on magnetic and resistive devices, and FloatPIM [49], a Deep Neural Network Compute in Memory Accelerator.

In Figure 15, our CiM architecture, being FeFET-based, demonstrates the most energy-efficient performance, with improvements of up to 95× compared to the GPU baseline and up to 7.2× when contrasted with RAPID. Regarding speedup, our hardware circuit outperforms others, showcasing speedups of up to 110× compared to the GPU baseline and up to 7.3× compared to Newtown. Overall, our results indicate that our architecture using HDC is highly superior to other hardware accelerators used for genomic tasks. If we combine these results and the results of Figure 9, we can prove consistent HDGIM's superiority.

VII. Conclusion

In this paper extension, we revisit genome sequence matching and introduce a framework capable of FeFET-based Computing in Memory. Specifically designed to optimize performance with FeFET non-idealities and in-memory architecture, our results demonstrate equivalent performance to the full-precision model on a highly-scaled FeFET-based CiM. Notably, our approach proves resilient to noise, even at highly-scaled FeFET levels of 3-bit and 4-bit, maintaining robustness with high dimensionality during inference.

Our study thoroughly assesses our framework's performance on advanced hardware, considering often overlooked factors in genomic tasks with compute-in-memory hardware. Focusing on intricacies like temperature and scaling size in FeFET computing, we offer reliable insights for future deployment. Addressing variability from FeFET, including polarization and process variation, enhances fundamental reliability. While not exploring transistor aging in this paper, we acknowledge it as a distinct consideration, actively addressing reliability concerns.

A significant strength is our comprehensive examination of FeFET cell scalability, addressing concerns in the computing-in-memory community. We highlight the model's efficacy, particularly at higher bit-precisions and with elevated temperatures common in handling large genomic datasets. Our research uncovers a trade-off between thickness, scalability, and the required HDC model hyperparameters.

Acknowledgment

The authors would like to thank Yogesh S. Chauhan, Shubham Kumar, Swetaki Chatterjee, Kai Ni, Simon Thomann, and Om Prakash for their support with the FeFET and FDSOI device modeling and calibration, Albi Mema for his support with the CAM cell layouts, and Sam Spetalnick from Georgia Tech for the helpful discussions.

References

[1] H. E. Barkam et al., "HDGIM: Hyperdimensional genome sequence matching on unreliable highly scaled FeFET," in *Proc. Design, Autom. Test Eur. Conf. Exhibition (DATE)*, Apr. 2023, pp. 1–6.

[2] H. Li, "Minimap2: Pairwise alignment for nucleotide sequences," *Bioinformatics*, vol. 34, no. 18, pp. 3094–3100, Sep. 2018.

[3] D. E. Wood, J. Lu, and B. Langmead, "Improved metagenomic analysis with Kraken 2," *Genome Biol.*, vol. 20, no. 1, pp. 1–13, Nov. 2019.

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

[5] S. Beyer et al., "FeFET: A versatile CMOS compatible device with game-changing potential," in *Proc. IEEE Int. Memory Workshop (IMW)*, May 2020, pp. 1–4.

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

[7] P. R. Genssler, V. M. van Santen, J. Henkel, and H. Amrouch, "On the reliability of FeFET on-chip memory," *IEEE Trans. Comput.*, vol. 71, no. 4, pp. 947–958, Apr. 2022.

[8] Z. Jiang et al., "Asymmetric double-gate ferroelectric FET to decouple the tradeoff between thickness scaling and memory window," in *Proc. IEEE Symp. VLSI Technol. Circuits (VLSI Technol. Circuits)*, Jun. 2022, pp. 395–396.

[9] S. Chatterjee, S. Thomann, K. Ni, Y. S. Chauhan, and H. Amrouch, "Comprehensive variability analysis in dual-port FeFET for reliable multi-level-cell storage," *IEEE Trans. Electron Devices*, vol. 69, no. 9, pp. 5316–5323, Sep. 2022.

[10] N. Yuvaraj, K. Srihari, S. Chandragandhi, R. A. Raja, G. Dhiman, and A. Kaur, "Analysis of protein-ligand interactions of SARS-CoV-2 against selective drug using deep neural networks," *Big Data Mining Anal.*, vol. 4, no. 2, pp. 76–83, Jun. 2021.

[11] C. Jain, S. Misra, H. Zhang, A. Dilthey, and S. Aluru, "Accelerating sequence alignment to graphs," in *Proc. IEEE Int. Parallel Distrib. Process. Symp. (IPDPS)*, May 2019, pp. 451–461.

[12] P. Kanerva, "Hyperdimensional computing: An introduction to computing in distributed representation with high-dimensional random vectors," *Cognit. Comput.*, vol. 1, no. 2, pp. 139–159, Jun. 2009.

[13] Y. Kim, M. Imani, N. Moshiri, and T. Rosing, "Geniehd: Efficient DNA pattern matching accelerator using hyperdimensional computing," in *Proc. Design, Automat. Test Eur. Conf. Exhib. (DATE)*. Mar. 2020, pp. 115–120.

[14] P. Poduval et al., "Cognitive correlative encoding for genome sequence matching in hyperdimensional system," in *Proc. 58th ACM/IEEE Design Automat. Conf. (DAC)*, Dec. 2021, pp. 781–786.

[15] Z. Zou et al., "BioHD: An efficient genome sequence search platform using HyperDimensional memorization," in *Proc. 49th Annu. Int. Symp. Comput. Archit.*, Jun. 2022, pp. 656–669.

[16] V. Joshi et al., "Accurate deep neural network inference using computational phase-change memory," *Nature Commun.*, vol. 11, no. 1, p. 2473, May 2020.

[17] C. Alkan et al., "Personalized copy number and segmental duplication maps using next-generation sequencing," *Nature Genet.*, vol. 41, no. 10, pp. 1061–1067, Oct. 2009.

[18] H. Xin, D. Lee, F. Hormozdiari, S. Yedkar, O. Mutlu, and C. Alkan, "Accelerating read mapping with FastHASH," *BMC Genomics*, vol. 14, no. S1, pp. 1–13, Jan. 2013.

[19] X. Chen, X. Yin, M. Niemier, and X. S. Hu, "Design and optimization of FeFET-based crossbars for binary convolution neural networks," in *Proc. Design, Autom. Test Eur. Conf. Exhibition (DATE)*, Mar. 2018, pp. 1205–1210.

[20] C.-K. Liu et al., "COSIME: FeFET based associative memory for in-memory cosine similarity search," in *Proc. IEEE/ACM Int. Conf. Comput. Aided Design (ICCAD)*, Oct. 2022, pp. 1–9.

[21] X. Yin et al., "FeCAM: A universal compact digital and analog content addressable memory using ferroelectric," *IEEE Trans. Electron Devices*, vol. 67, no. 7, pp. 2785–2792, Jul. 2020.

[22] A. F. Laguna, H. Gamaarachchi, X. Yin, M. Niemier, S. Parameswaran, and X. S. Hu, "Seed-and-vote based in-memory accelerator for DNA read mapping," in *Proc. IEEE/ACM Int. Conf. Comput. Aided Design (ICCAD)*, Nov. 2020, pp. 1–9.

[23] A. Kazemi et al., "FeFET multi-bit content-addressable memories for in-memory nearest neighbor search," *IEEE Trans. Comput.*, vol. 71, no. 10, pp. 2565–2576, Oct. 2022.

[24] S. Angizi, N. A. Fahmi, W. Zhang, and D. Fan, "PIM-assembler: A processing-in-memory platform for genome assembly," in *Proc. 57th ACM/IEEE Design Autom. Conf. (DAC)*, Jul. 2020, pp. 1–6.

[25] S. Gupta, M. Imani, B. Khaleghi, V. Kumar, and T. Rosing, "RAPID: A ReRAM processing in-memory architecture for DNA sequence alignment," in *Proc. IEEE/ACM Int. Symp. Low Power Electron. Design (ISLPED)*, Jul. 2019, pp. 1–6.

[26] H. Mulaosmanovic, D. Kleimaier, S. Dünkel, S. Beyer, T. Mikolajick, and S. Slesazeck, "Ferroelectric transistors with asymmetric double gate for memory window exceeding 12 v and disturb-free read," *Nanoscale*, vol. 13, no. 38, pp. 16258–16266, 2021.

[27] X. Yin et al., "An ultracompact Single-ferroelectric field-effect transistor binary and multibit associative search engine," *Adv. Intell. Syst.*, vol. 5, no. 7, Jul. 2023, Art. no. 2200428.

[28] K. Ni, S. Thomann, O. Prakash, Z. Zhao, S. Deng, and H. Amrouch, "On the channel percolation in ferroelectric FET towards proper analog states engineering," in *IEDM Tech. Dig.*, Dec. 2021, pp. 1531–1534.

[29] C.-T. Tung, G. Pahwa, S. Salahuddin, and C. Hu, "A compact model of ferroelectric field-effect transistor," *IEEE Electron Device Lett.*, vol. 43, no. 8, pp. 1363–1366, Aug. 2022.

[30] C. Li et al., "A scalable design of multi-bit ferroelectric content addressable memory for data-centric computing," in *IEDM Tech. Dig.*, Dec. 2020, pp. 2931–2934.

[31] C. Jin et al., "A multi-bit CAM design with ultra-high density and energy efficiency based on FeFET NAND," *IEEE Electron Device Lett.*, vol. 44, no. 7, pp. 1104–1107, Jul. 2023.

[32] S. Thomann, P. R. Genssler, and H. Amrouch, "HW/SW co-design for reliable TCAM-based in-memory brain-inspired hyperdimensional computing," *IEEE Trans. Comput.*, vol. 72, no. 8, pp. 1–14, Feb. 2023.

[33] M. Imani et al., "SearcHD: A memory-centric hyperdimensional computing with stochastic training," *IEEE Trans. Comput.-Aided Design Integr. Circuits Syst.*, vol. 39, no. 10, pp. 2422–2433, Oct. 2020.

[34] X. Peng et al., "Benchmarking monolithic 3D integration for compute-in-memory accelerators: Overcoming ADC bottlenecks and maintaining scalability to 7nm or beyond," in *IEDM Tech. Dig.*, Dec. 2020, pp. 3041–3044.

[35] S. D. Spetalnick et al., "A 40nm 64kb 26.56TOPS/W 2.37Mb/mm^{2}RRAM binary/compute-in-memory macro with 4.23× improvement in density and >75% use of sensing dynamic range," in *Proc. IEEE Int. Solid- State Circuits Conf. (ISSCC)*, vol. 65, Feb. 2022, pp. 1–3.

[36] C. Li et al., "Analogue signal and image processing with large memristor crossbars," *Nature Electron.*, vol. 1, no. 1, pp. 52–59, Dec. 2017.

[37] L. Sumanen, M. Waltari, V. Hakkarainen, and K. Halonen, "CMOS dynamic comparators for pipeline A/D converters," in *Proc. IEEE Int. Symp. Circuits Syst.*, May 2002, pp. 157–160.

[38] E. P. Frady, D. Kleyko, and F. T. Sommer, "A theory of sequence indexing and working memory in recurrent neural networks," *Neural Comput.*, vol. 30, no. 6, pp. 1449–1513, Jun. 2018.

[39] S. Kullback and R. A. Leibler, "On information and sufficiency," *Ann. Math. Statist.*, vol. 22, no. 1, pp. 79–86, 1951.

[40] A. Gersho, "Asymptotically optimal block quantization," *IEEE Trans. Inf. Theory*, vol. IT-25, no. 4, pp. 373–380, Jul. 1979.

[41] P. L. Zador, *Development and Evaluation of Procedures for Quantizing Multivariate Distributions*. Stanford, CA, USA: Stanford Univ., 1964.

[42] R. M. Gray and D. L. Neuhoff, "Quantization," *IEEE Trans. Inf. Theory*, vol. 44, no. 6, pp. 2325–2383, Oct. 1998.

[43] C. E. Shannon, "A mathematical theory of communication," *Bell Syst. Tech. J.*, vol. 27, no. 3, pp. 379–423, Jul. 1948.

[44] *Genome Information by Organism*. Accessed: Jun. 18, 2022. [Online]. Available: https://www.ncbi.nlm.nih.gov/genome/browse/prokaryotes/167/

[45] A. Kazemi et al., "Achieving software-equivalent accuracy for hyperdimensional computing with ferroelectric-based in-memory computing," *Sci. Rep.*, vol. 12, no. 1, p. 19201, Nov. 2022, doi: 10.1038/s41598-022-23116-w.

[46] J. Pantaleoni and N. Subtil. (2013). *NVBIO: GPU-Accelerated C++ Framework for High-Throughput Sequence Analysis*. [Online]. Available: https://developer.nvidia.com/nvbio

[47] F. Gao, G. Tziantzioulis, and D. Wentzlaff, "ComputeDRAM: In-memory compute using off-the-shelf DRAMs," in *Proc. 52nd Annu. IEEE/ACM Int. Symp. Microarchitecture*. New York, NY, USA: Association for Computing Machinery, Oct. 2019, p. 100, doi: 10.1145/3352460.3358260.

[48] M. He et al., "Newton: A DRAM-maker's accelerator-in-memory (AiM) architecture for machine learning," in *Proc. 53rd Annu. IEEE/ACM Int. Symp. Microarchitecture (MICRO)*, Oct. 2020, pp. 372–385.

[49] M. Imani, S. Gupta, Y. Kim, and T. Rosing, "FloatPIM: In-memory acceleration of deep neural network training with high precision," in *Proc. ACM/IEEE 46th Annu. Int. Symp. Comput. Archit. (ISCA)*. New York, NY, USA: Association for Computing Machinery, Jun. 2019, pp. 802–815.

Hamza E. Barkam (Member, IEEE) received the Bachelor of Engineering degree in telecommunications engineering from Universitat Politecnica de Catalunya, Spain, in 2020, and the M.Sc. degree from the University of California Irvine, USA, in 2022. He is currently pursuing the Ph.D. degree with the Engaging Technology and Application Design Laboratory, CaliT2, under the supervision of Prof. Sergio Gago. Since 2021, he has been with the Bio-Inspired Architecture and Systems (BIASLab), with Prof. Mohsen Imani. His research interests include theoretical work on vector symbolic architectures, hyperdimensional computing, compatibility of brain-learning algorithms with emerging technologies, and secure machine learning.

Sanggeon Yun (Member, IEEE) received the B.S. degree in computer science from Kookmin University, Seoul, South Korea, in 2023. He is currently pursuing the Ph.D. degree with the Bio-Inspired Architecture and Systems (BIASLab), University of California, Irvine, under the supervision of Prof. Mohsen Imani. His research interests include hyperdimensional computing, machine learning, natural language processing, human–computer interaction, and information visualization.

Paul R. Genssler (Member, IEEE) received the Dipl.-Inf degree (M.Sc.) in computer science from TU Dresden, Germany, in 2017. He is currently pursuing the Ph.D. degree with the Chair for Embedded Systems (CES), Karlsruhe Institute of Technology, Germany. Since 2020, he continues his Ph.D. with the Semiconductor Test and Reliability (STAR) Chair, Computer Science, Electrical Engineering Faculty, University of Stuttgart. His research interests include emerging technologies, system architecture, and emerging brain-inspired methods for IC test and beyond. He served as a reviewer for the IEEE INTERNET OF THINGS JOURNAL.

Che-Kai Liu (Student Member, IEEE) received the B.Eng. degree in electronic engineering from Zhejiang University in 2023. He is currently pursuing the Ph.D. degree with the Integrated Circuits and Systems Laboratory (ICSRL), Georgia Institute of Technology, USA, under the supervision of Prof. Arijit Raychowdhury. His current research interests include AMS VLSI and architecture for next-generation applications. He served as a reviewer for IEEE JOURNAL ON EMERGING AND SELECTED TOPICS IN CIRCUITS AND SYSTEMS, IEEE INTERNATIONAL SYMPOSIUM ON CIRCUITS AND SYSTEMS, and *ACM Journal on Autonomous Transportation Systems*.

Zhuowen (Kevin) Zou (Member, IEEE) received the Bachelor of Science and Master of Science degrees in computer science from the University of California, San Diego, in 2022. He is currently pursuing the Ph.D. degree with the Bio-Inspired Architecture and Systems (BIASLab), UC Irvine, with a focus on the theory of hyperdimensional computing. His research interests include neuro-symbolic AI, machine learning theory, cognitive neuroscience, and cryptography.

Hussam Amrouch (Member, IEEE) received the Ph.D. degree (summa cum laude) from the Karlsruhe Institute of Technology (KIT) in 2015. He is currently a Professor and heading the Chair of Semiconductor Test and Reliability (STAR), Computer Science, Electrical Engineering Faculty, University of Stuttgart, Germany. Prior to that, he was a Research Group Leader with KIT, where he was leading the research efforts in building dependable embedded systems. He has more than 200 publications in multidisciplinary research areas (including 82 journals) across the entire computing stack, starting from semiconductor physics to circuit design all the way up to computer-aided design and computer architecture. His research in HW security and reliability have been funded by the German Research Foundation (DFG), Advantest Corporation, and the U.S. Office of Naval Research (ONR). His main research interests include design for reliability and testing from device physics to systems, machine learning for CAD, HW security, approximate computing, and emerging technologies with a special focus on ferroelectric devices. He holds eight HiPEAC paper awards and three best paper nominations at top EDA conferences: DAC'16, DAC'17 and DATE'17 for his work on reliability. He served in the technical program committees for many major EDA conferences, such as DAC, ASP-DAC, and ICCAD, and as a reviewer in many top journals, such as *Nature Electronics*, T-ED, IEEE TRANSACTIONS ON CIRCUITS AND SYSTEMS—I: REGULAR PAPERS, IEEE TRANSACTIONS ON VERY LARGE SCALE INTEGRATION, IEEE TRANSACTIONS ON COMPUTER-AIDED DESIGN OF INTEGRATED CIRCUITS AND SYSTEMS, and IEEE TRANSACTIONS ON COMPUTERS. He serves as an Editor for the *Scientific Reports* (Nature).

Mohsen Imani (Member, IEEE) received the Ph.D. degree from the Department of Computer Science, UC San Diego. He is currently an Assistant Professor with the Department of Computer Science, UC Irvine. He is also the Director of the Bio-Inspired Architecture and Systems (BIASLab). His contribution has led to a new direction on brain-inspired hyperdimensional computing that enables ultra-efficient and real-time learning and cognitive support. His research was also the main initiative in opening up multiple industrial and governmental research programs. His research has been recognized with several awards, including the Bernard and Sophia Gordon Engineering Leadership Award, the Outstanding Researcher Award, and the Powell Fellowship Award. He also received the Best Doctorate Research from UCSD and several best paper nomination awards at multiple top conferences, including the Design Automation Conference (DAC) in 2019 and 2020, the Design Automation and Test in Europe (DATE) in 2020, and the International Conference on Computer-Aided Design (ICCAD) in 2020. Furthermore, he received the Best Paper Award in DATE 2022 Conference.