HDGIM: Hyperdimensional Genome Sequence Matching on Unreliable highly scaled FeFET

Hamza Errahmouni Barkam[1], Sanggeon Yun[1,2], Paul R. Genssler[4], Zhuowen Zou[1], Che-Kai Liu[3]
Hussam Amrouch[4] and Mohsen Imani[1⋆]
[1]University of California Irvine, [2]Kookmin University, [3]Zhejiang University, [4]University of Stuttgart
⋆Corresponding Author: m.imani@uci.edu

***Abstract*—This is the first work to present a reliable application for highly scaled (down to merely 3nm), multi-bit Ferroelectric FET (FeFET) technology. FeFET is one of the up-and-coming emerging technologies that is not only fully compatible with the existing CMOS but does hold the promise to realize ultra-efficient and compact Compute-in-Memory (CiM) architectures. Nevertheless, FeFETs struggle with the 10nm thickness of the Ferroelectric (FE) layer. This makes scaling profoundly challenging if not impossible because thinner FE significantly shrinks the memory window leading to large error probabilities that cannot be tolerated. To overcome these challenges, we propose HDGIM, a hyperdimensional computing framework catered to FeFET in the context of genome sequence matching. Genome Sequence Matching is known to have high computational costs, primarily due to huge data movement that substantially overwhelms von-Neuman architectures. On the one hand, our cross-layer FeFET reliability modeling (starting from device physics to circuits) accurately captures the impact of FE scaling on errors induced by process variation and inherent stochasticity in multi-bit FeFETs. On the other hand, our HDC learning framework iteratively adapts by using two models, 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 any noise given high dimensionality during inference. If we consider the noise during model adjustment, we can improve the inherent robustness compared to adding noise during the matching process.**

I. Introduction

Genome sequence matching is one of the key algorithms in identifying and analyzing genomic data with several applications in identifying and curing diseases, including COVID-19. Sequence matching consists of analyzing essential biological characteristics such as nucleotide or protein sequences and comparing them in terms of their similarity [1]. In sequence matching, a DNA query is searched across large-scale reference DNA strings which typically comprise more than a hundred million DNA bases [2]. Unfortunately, running genome sequence matching on existing hardware is significantly slow and inefficient as it demands a large amount of 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 [3], thus degrading the application-level accuracy.

Among different NVM technologies, FeFET has emerged as a promising candidate due to its full CMOS compatibility [4], low read/write energy [5], etc. However, FeFET devices are often designed with a 10nm-thick ferroelectric layer in the gate stack of the transistor in order to maintain the state. Such a thick layer has several disadvantages. The $\pm$4V write voltage is not available on modern ICs, requiring a new dedicated power network inside the chip, incurring a large overhead. Further, the high write voltage is a major source of reliability and endurance challenges [6].

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 decreases, 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 [7], [8]. A 3nm FE-layer was explored in [7] and a write voltage of $\pm$1.85V was sufficient. However, an additional back gate was added to the FeFET to overcome the extremely high variation during a read operation [8]. Such an extra gate increases the complexity and area of a circuit.

The efficient but noisy CiM technology, when combined with high-precision computation that many sequence matching algorithms require [9], [10], 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 [11], [12]. Multiple recent works have exploited HDC data representation to revisit genome sequence matching algorithms for memory-centric computation [13]–[18]. These CiM approaches translate sequencing matching to highly parallel search operations that can be accelerated on content addressable memory (CAM) [15], [19]. 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:

978-3-9819263-7-8/DATE23/© 2023 EDAA

- **Faithful bridge between HDC algorithm and hardware constraints:** We employ physics-based FeFET models to accurately capture their noise and variation. Instead of the simple Gaussian distribution typically employed in other works [20], our FeFET models provide a realistic noise distribution for multi-bit precision, 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 highly sensitive to technology noise during in-memory computation. 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. a

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 on errors induced by process variation and inherent stochasticity in multi-bit FeFETs. Our HDC learning framework iteratively adapts by using two models, 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 any 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.

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. 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 [21], [22].

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 [23] and nearest neighbor search [24]. 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 [25], [26]. Specifically, in conjunction with customized sense amplifiers (Figure 1(c)), 2-FeFET CAM (Figure 1(d)) designs demonstrate great potential as a high-density and energy-efficient associative memory with Hamming and L_2 distance [5], [27].

CiM based on NVM technology has been widely used to accelerate genome sequence matching problems. For example, PIM-Aligner [28] and RAPID [29] 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

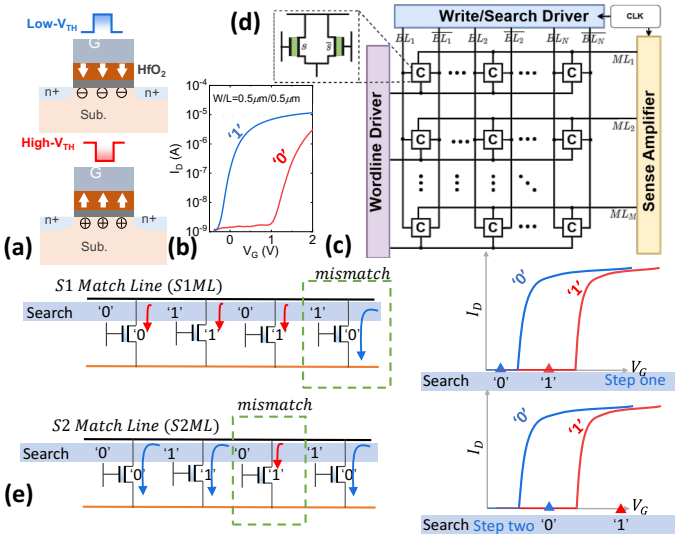

Fig. 1. (a) FeFET operation schematic. (b) SPICE simulation of FeFET $I_D - V_G$ curve for storing 1 and 0. (c) CAM array. (d) Single 2FeFET CAM. (e) Ultra-compact 1 FeFET CAM [30].

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 either assume the CiM or CAM is ideal, do not work with multi-bit CiM or the noise modeling is non-accurate 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.

III. Compute-in-Memory with FeFET Technology

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 [4], comparatively low read/write energy, and short read latency [5]. A FeFET is based on a regular CMOS MOSFET with only one modification to the gate stack depicted in Figure 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 the FE layer to the down polarization state and $I_D - V_G$ becomes high at the read voltage of about 1V as shown in Figure 1(b). A negative voltage pulse results in an up polarization and a low $I_D - V_G$. The two different 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 Figure 2, which are flipped by the write pulse, and define the polarization. However, flipping is a *stochastic process* over time and thus the polarization (i.e., $I_D - V_G$) is an intermediate value for short or lower voltage pulse. Such a property can be exploited to create a multi-stage cell to store multiple

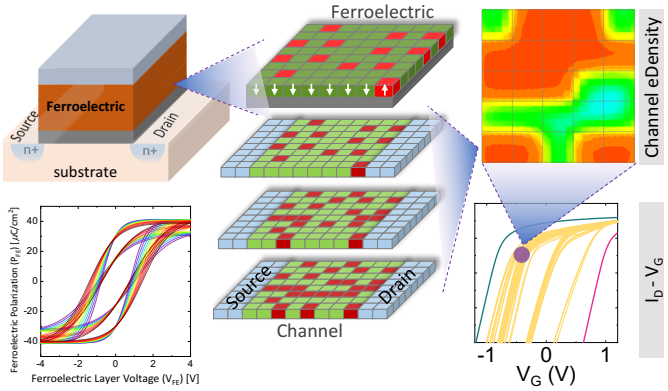

Fig. 2. Our FeFET modeling considers the inherent stochasticity of the domains in the ferroelectric layer. Their up/down polarization determines the electron density of the channel and thus the drain-source current $I_D - V_G$.

bits of information [8]. Scaling the thickness of the FE layer down reduces the number of domains, which become more impactful on the overall polarization. Because of the inherent stochasticity, the variability increases for highly scaled FeFET devices with fewer domains [8].

Our FeFET modeling: In this work, we perform accurate cross-layer reliability modeling with the multi-physics simulator TCAD. We start from the underlying device physics, as follows: First, the FDOSI transistor is carefully calibrated to reproduce experimental measurements of commercial 22nm FDSOI [31]. Then, the parameters of the incorporated FE layer (remnant polarization, saturation polarization, and coercive field) are also calibrated against measurements [8]. Then, the $I_D - V_G$ characteristic is swept, and different FeFET states are derived from that. To capture process variation and the inherent stochasticity, Monte Carlo simulations are performed resulting in a VT distribution for each stage of the cell. The mean and variance are extracted and provided to the HDGIM framework for further modeling at circuit level.

FeFET-based Content-Addressable Memory (CAM): CAM can identify the stored sequence that exactly matches the query [5]. 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, During the writing phase, the reference DNA strings are written into CAMs 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. In this work, by leveraging the above accurate FeFET modeling, we can characterize the non-idealities of CAMs.

B. 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 [30]. Unlike conventional single-bit CAM with 2 FeFETs (Figure 1(d)) and single step search, single-bit ultra-compact CAM shown in Figure 1(e) necessitates 1 FeFET with 2 step searches. As a result, an N-bit CAM performing bit-wise XOR only requires N 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 between the query and a stored vector is simply the number of "store0search1" cells and "store1search0" cells. By the fact that $I_{S1ML} = I_{ON} N_{st0sr1}$ and $I_{S2ML} = I_{ON}(N_{total} - N_{st1sr0})$, we obtain [30]:

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

Then, by changing the reference of the sense amplifier for detecting rows exceeding the threshold, HDGIM framework, illustrated in Section IV, is able to handle multi-bit CiM genome sequence matching.

IV. HDGIM Framework

Figure 3 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 $\sigma \in \{A, C, G, T\}$. 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 $< A, C, G >, < C, G, G >, < G, G, T >$.

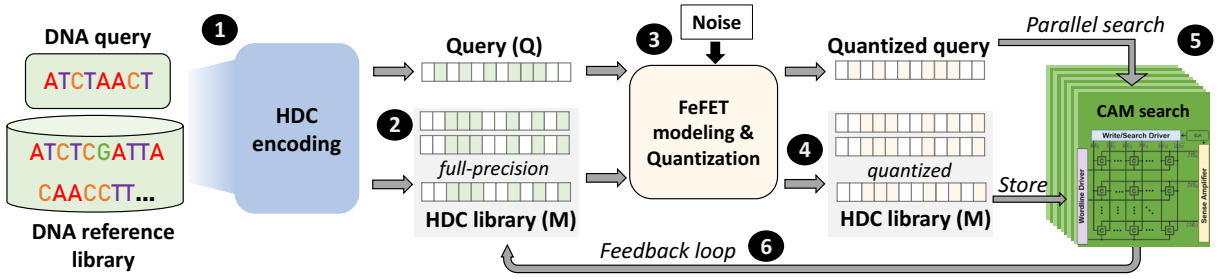

Fig. 3. Overview of HDGIM sequence matching in the hyperdimensional space.

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.

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 are not following uniform distribution in general, it calculates scores of the HDC components and uses the cumulative normal distribution function to quantize feature values.

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. We then apply the noise during inference on the model projection step (❸). The values remain unchanged until the next model projection step.

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).

D. Capacity of hardware-based HDC

Following the methodology in [32], 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

$$\begin{aligned} I(\{\vec{S}^{(i)}\}, \vec{S}) &= D_{KL}(\Pr(\hat{s}, s) || \Pr(\hat{s})\Pr(s)) \\ &= \sum_{i,j\in\{0,1\}} \Pr(\hat{s}=i, s=j) \log_2 \frac{\Pr(\hat{s}=i, s=j)}{\Pr(\hat{s}=i)\Pr(s=j)} \end{aligned} \quad (2)$$

Where D_{KL} is the KL divergence [33]. 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, whose values are the true/false positive/negative rates of the detector. The memory capacity is simplified as

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

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

V. Evaluation

A. Experimental Setup

The proposed framework has been executed with a software framework and a hardware accelerator. Our software framework is implemented using Pytorch and supports HDC encoding and classification. We study the effectiveness of our technique over a randomly generated 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.

We evaluate our framework with a FeFET model of 3nm and 10nm thickness. Subsequently, we study the effect of bit-precision, dimension of hypervectors, and noise quality. Finally, we compare the effect of adding noise during training and inference. 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 update of the model. Our simulator is connected to PyTorch for easy programmability and maximum efficiency.

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 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. Figure 4 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.

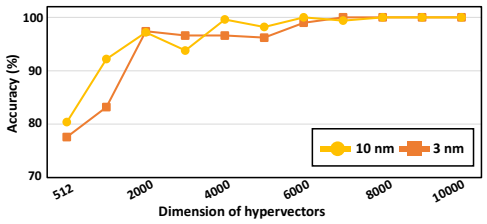

Fig. 4. Performance of HDGIM modeling a 10nm and 3nm thick FeFET.

C. 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%. Figure 5 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.

D. 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. Figure 6(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 Figure 6b) at lower dimensions. The highlighted zones show that for the same area dependent

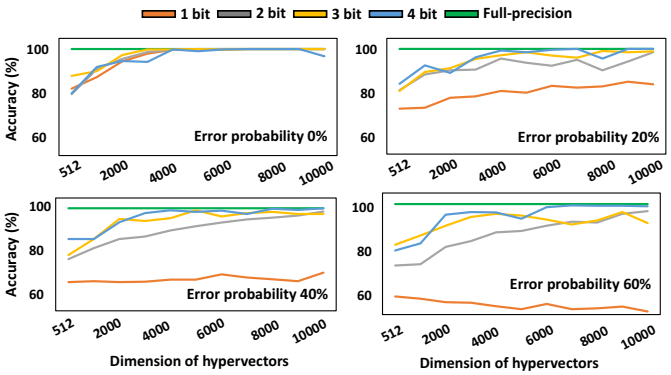

Fig. 5. Exploration of multi-bit HDGIM for different noise probabilities.

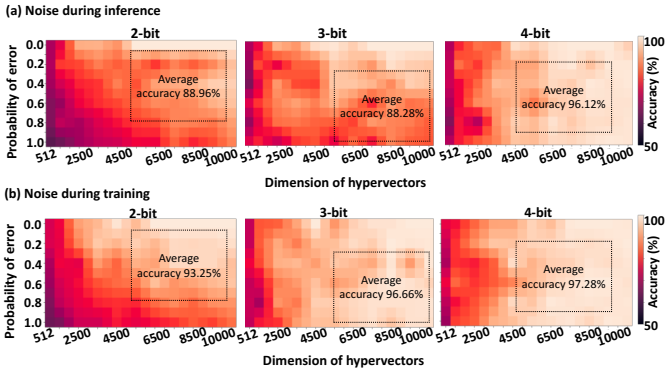

Fig. 6. Heatmap of accuracies for (a) noise introduced during inference and (b) training for 2, 3, and 4 bits.

on 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.

VI. Conclusions

In this paper, we revisit genome sequence matching and propose a framework able to perform FeFET-based Computing in Memory. Our framework is designed to optimally operation with FeFET non-idealities and in-memory architecture. Our results indicate that our approach provides equivalent performance to the full-precision model on a highly scaled FeFET-based CiM. Our results demonstrate that highly scaled FeFET realizing 3-bit and even 4-bit can withstand any noise given high dimensionality during inference.

Acknowledgment

This work was supported in part by National Science Foundation #2127780, Semiconductor Research Corporation (SRC) GRC AI Hardware and Hardware Security, Office of Naval Research grants #N00014-21-1-2225 and #N00014-22-1-2067, the Air Force Office of Scientific Research under award #FA9550-22-1-0253, and a generous gift from Cisco and Xilinx. This research was also supported in part by Advantest as part of the Graduate School "Intelligent Methods for Test and Reliability" (GS-IMTR) at the University of Stuttgart.

References

[1] H. Li, "Minimap2: pairwise alignment for nucleotide sequences," *Bioinformatics*, 2018.
[2] D. E. Wood *et al.*, "Improved metagenomic analysis with kraken 2," *Genome biology*, 2019.
[3] S. Deng *et al.*, "A comprehensive model for ferroelectric fet capturing the key behaviors: Scalability, variation, stochasticity, and accumulation," in *IEEE Symposium on VLSI Technology*, 2020.
[4] S. Beyer *et al.*, "Fefet: A versatile cmos compatible device with game-changing potential," in *IEEE IMW*, 2020.
[5] K. Ni *et al.*, "Ferroelectric ternary content-addressable memory for one-shot learning," *Nature Electronics*, 2019.
[6] P. R. Genssler *et al.*, "On the reliability of fefet on-chip memory," *IEEE TC*, 2022.
[7] Z. Jiang *et al.*, "Asymmetric double-gate ferroelectric fet to decouple the tradeoff between thickness scaling and memory window," in *2022 IEEE Symposium on VLSI Technology and Circuits*, 2022.
[8] S. Chatterjee *et al.*, "Comprehensive variability analysis in dual-port fefet for reliable multi-level-cell storage," *IEEE TED*, 2022.
[9] N. Yuvaraj *et al.*, "Analysis of protein-ligand interactions of sars-cov-2 against selective drug using deep neural networks," *Big Data Mining and Analytics*, 2021.
[10] C. Jain *et al.*, "Accelerating sequence alignment to graphs," in *IEEE IPDPS*, 2019.
[11] P. Kanerva, "Hyperdimensional computing: An introduction to computing in distributed representation with high-dimensional random vectors," *Cognitive Computation*, 2009.
[12] A. Hernandez-Cane *et al.*, "Onlinehd: Robust, efficient, and single-pass online learning using hyperdimensional system," in *DATE*, pp. 56–61, IEEE, 2021.
[13] Y. Kim, M. Imani, N. Moshiri, and T. Rosing, "Geniehd: Efficient dna pattern matching accelerator using hyperdimensional computing," in *DATE*, IEEE, 2020.
[14] P. Poduval *et al.*, "Cognitive correlative encoding for genome sequence matching in hyperdimensional system," in *DAC*, IEEE, 2021.
[15] Z. Zou *et al.*, "Biohd: an efficient genome sequence search platform using hyperdimensional memorization," in *ISCA*, 2022.
[16] Z. Zou *et al.*, "Eventhd: Robust and efficient hyperdimensional learning with neuromorphic sensor," *Frontiers in Neuroscience*, vol. 16, 2022.
[17] Z. Zou *et al.*, "Memory-inspired spiking hyperdimensional network for robust online learning," *Scientific reports*, vol. 12, no. 1, pp. 1–13, 2022.
[18] Z. Zou *et al.*, "Scalable edge-based hyperdimensional learning system with brain-like neural adaptation," in *Proceedings of the International Conference for High Performance Computing, Networking, Storage and Analysis*, pp. 1–15, 2021.
[19] M. Imani *et al.*, "Exploring hyperdimensional associative memory," in *IEEE HPCA*, pp. 445–456, IEEE, 2017.
[20] V. Joshi *et al.*, "Accurate deep neural network inference using computational phase-change memory," *Nature communications*, 2020.
[21] C. Alkan *et al.*, "Personalized copy number and segmental duplication maps using next-generation sequencing," *Nat Genet*, 2009.
[22] H. Xin *et al.*, "Accelerating read mapping with fasthash," *BMC Genomics*, 2013.
[23] X. Chen *et al.*, "Design and optimization of fefet-based crossbars for binary convolution neural networks," in *IEEE DATE*, 2018.
[24] C.-K. Liu *et al.*, "Cosime: Fefet based associative memory for in-memory cosine similarity search," *ACM/IEEE ICCAD*, 2022.
[25] X. Yin *et al.*, "Fecam: A universal compact digital and analog content addressable memory using ferroelectric," *IEEE TED*, 2020.
[26] A. F. Laguna *et al.*, "Seed-and-vote based in-memory accelerator for dna read mapping," in *ACM/IEEE ICCAD*, 2020.
[27] A. Kazemi *et al.*, "Fefet multi-bit content-addressable memories for in-memory nearest neighbor search," *IEEE TC*, 2021.
[28] S. Angizi *et al.*, "Pim-assembler: A processing-in-memory platform for genome assembly," in *DAC*, IEEE, 2020.
[29] S. Gupta *et al.*, "Rapid: A reram processing in-memory architecture for dna sequence alignment," in *ISLPED*, IEEE, 2019.
[30] X. Yin *et al.*, "An ultra-compact single fefet binary and multi-bit associative search engine," *arXiv preprint arXiv:2203.07948*, 2022.
[31] K. Ni *et al.*, "On the channel percolation in ferroelectric fet towards proper analog states engineering," in *IEEE IEDM*, 2021.
[32] E. P. Frady *et al.*, "A theory of sequence indexing and working memory in recurrent neural networks," *Neural Computation*, 2018.
[33] S. Kullback *et al.*, "On information and sufficiency," *The annals of mathematical statistics*, 1951.