Energy-Efficient Brain-Inspired Hyperdimensional Computing Using Voltage Scaling

Sizhe Zhang*, Ruixuan Wang*, Dongning Ma*, Jeff (Jun) Zhang‡, Xunzhao Yin§, Xun Jiao*
*Villanova University, ‡Harvard University, §Zhejiang University

***Abstract*—Recently, brain-inspired hyperdimensional computing (HDC) has demonstrated promising capability in a wide range of applications such as medical diagnosis, human activity recognition, and voice classification, etc. Despite the growing popularity of HDC, its memory-centric computing characteristics make the associative memory implementation under significant energy consumption due to the massive data storage and processing. In this paper, we present a systematic case study to leverage the application-level error resilience of HDC to reduce the energy consumption of HDC associative memory by using voltage scaling. Evaluation results on various applications show that our proposed approach can achieve 47.6% energy saving on associative memory with a $\leq$1% accuracy loss. We further explore two low-cost error masking methods: word masking and bit masking, to mitigate the impact of voltage scaling-induced errors. Experimental results show that the proposed word masking (bit masking) method can further enhance energy saving up to 62.3% (72.5%) with accuracy loss $\leq$1%.**

I. Introduction

Brain-inspired hyperdimensional computing (HDC) is an emerging computational paradigm that mimics the working mechanism of brain which computes with deep and abstract patterns of the neural activity instead of actual numbers. Recently, HDC has shown advantages over traditional machine learning (ML) methods, such as smaller model size, less computation cost, making it a promising alternative in low-cost computing platforms [5]. HDC has achieved promising results in a wide range of applications such as robotics [11], language classification [12], and voice classification [6], etc.

Conventional computing platform typically requires a near-perfect execution with negligible error rates (e.g., $< 10^{-15}$ [8]) to guarantee correctness. Such requirement, however, poses a high design cost (e.g., timing/voltage margin) at both device and architecture levels. Modern applications offer a new opportunity to relax this strict requirement due to their inherent error-tolerant characteristics, which can be leveraged by hardware designers to enable application-specific better-than-worse-case design [15], [14]. ML workloads, for example, are known to be more error-tolerant than conventional workloads due to their statistical nature [9]. Leveraging such error resilience, designers have shown improved energy efficiency of ML systems by using *voltage scaling* on hardware such as ASICs [15] and FPGAs [14].

Such exploration in traditional ML systems also motivates us to understand and explore the error resilience of HDC models, which can potentially allow safely pushing the design guardbands by voltage scaling for HDC systems. However, for digital circuits that run with the under-scaled supply voltage, timing errors may occur if the critical paths are exercised [9], [15]. These errors usually manifest as bit flips in the circuits which can lead to incorrect computations and therefore degrade application quality. To this end, it is interesting to study if HDC models are resilient to a certain extent of voltage-induced timing errors and how much energy benefit we can gain from voltage scaling.

To address these questions, we perform a systematic study in this paper to examine the impacts of voltage scaling on HDC models. Our contributions are summarized as follows:

- We perform extensive error injection experiments under a wide range of voltage levels on HDC models across different applications. We quantify the impact of errors on the accuracy of HDC models and the resulted energy saving by voltage scaling. Our results show that HDC models can allow up to 47.6% energy saving on associative memory with a negligible accuracy loss ($\leq 1\%$).
- We explore two low-cost error mitigation mechanisms by detecting and masking the corrupted words/bits, which can effectively improve the resilience of HDC by up to $10,000X$. Experimental results show that with word/bit-level error masking, HDC can allow up to 62.3%/72.5% energy saving with negligible accuracy loss ($\leq 1\%$). Energy saving comparison with different technology node SRAM are also included.
- We perform design space exploration to reveal the effects of voltage scaling on HDC systems with different model configurations. This provides further insights on optimizing and developing future error-tolerant HDC systems.

II. HDC Model Development

A. HDC Basics

Hypervectors (HV) are high-dimensional, holographic vectors with i.i.d. elements [10]. An HV $\vec{H}$ with d dimensions can be denoted as $\vec{H} = \langle h_1, h_2, \ldots, h_d \rangle$, where h_i refers to the elements inside the HV. HVs are the fundamental blocks in HDC that can represent information in different types, scales and layers of features for its high dimensionality.

HDC utilizes different HV operations as means of aggregating information. In HDC, addition, multiplication and permutation are the three basic operations that HVs can support. Additions and multiplication take two input HVs as operands and perform element-wise add or multiply operations on them. Permutation takes one HV as the input operand and perform cyclic rotation. All the three operations do not modify the dimension of the HVs.

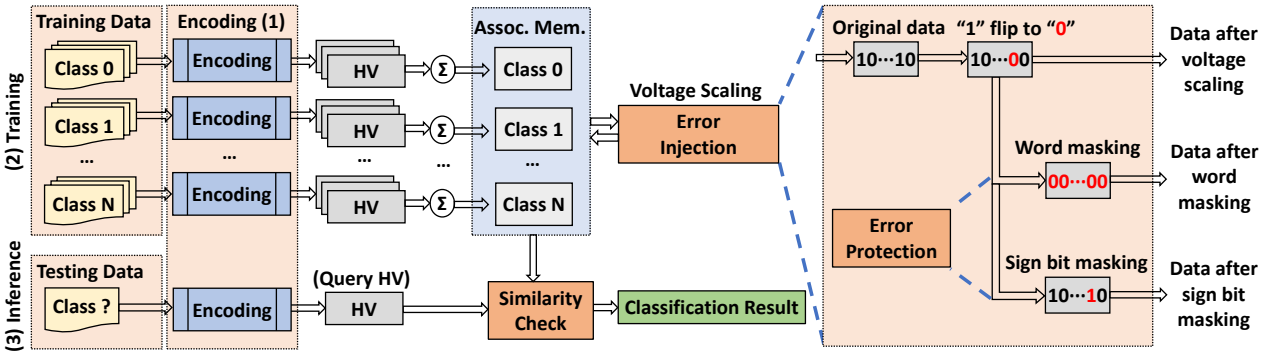

Fig. 1. Error injection and error protection for voltage-scaled HDC. HDC model has 3 phases: **Encoding**, **Training**, and **Inference**. Voltage scaling-induced errors are injected into the associative memory during the inference stage.

B. Developing HDC Model

There are three key phases in developing an HDC model for classification tasks: **Encoding**, **Training**, and **Inference**. The flow of HDC model is illustrated in Fig. 1.

Encoding is the fundamental process in developing an HDC model. It maps input features of a sample into an HV which belongs to the high-dimensional space and makes it available for HDC training and inference. This is done by applying combinations of HDC operations on the corresponding HVs in the item memory indexed by the feature value. Item memory stores base HVs representing different feature values. Assume in the classification task, each sample has input features $\vec{F}$ of m dimensions, thus there are m item memories $\mathcal{R}$ corresponding to each feature. In addition, the application-specific combination of HDC operations can be denoted as E determined by the application. Therefore, for each feature in the input sample, we can find its corresponding base HV in the item memory and then the encoded HV $\vec{H}$ is built by applying operation combination E: $\vec{H} = E(\mathcal{R}, \vec{F})$.

Training is the process of aggregating information of training samples from the same class. HDC performs training by summing up the encoded HVs sharing the same label into a class HV inside the associative memory. **Associative memory** stores the class HVs, each representing a class in the specific learning problem. Assume there are k classes in the classification problem and have encoded the HVs $\vec{H^l}$ for each training sample (l refers to the class label), training process to establish associative memory $\mathcal{A}$ is thus by adding up HVs having the same label l: $\mathcal{A} = \{\sum \vec{H^1}, \sum \vec{H^2}, \ldots, \sum \vec{H^k}\}$.

Inference is the process of using the learnt information to predict an unseen sample's class. In HDC, this is done by comparing the similarity of the unseen sample's HV with every class HV in the associative memory: $l = argmax(\{\delta(\vec{H_q}, \mathcal{A})\})$. HDC first encodes the input sample into its representing HV $\vec{H_q}$, referred to as the **query HV**, then checks the similarity between the query HV and each class HV inside the associative memory. The class of the highest similarity with the query HV is selected as the predicted label for the input sample.

III. Voltage Scaling on HDC Associative Memory

A. Voltage Scaling-induced Error Injection

While voltage scaling can lead to significant reduction in static and dynamic power dissipation in SRAM, it also compromises the memory data integrity that has prevented aggressive voltage scaling on SRAM cells. Generally, voltage scaling on SRAM can introduce several different types of failures, i.e., read failures, write failures, and access failures [1]. In HDC inference, memory reading is the dominant operation for memory access, so in this paper we are focusing on evaluating over the read failures. Typically, voltage scaling leads to random bit flips in SRAM bitcells with a certain probability [4].

Due to the statistical nature of SRAM cell failure, in our framework, when performing the similarity simulation during inference stage, we inject random bit flips at individual associative memory bitcells that store each class HV to emulate the voltage-induced errors before every inference as show in Fig. 1. For example, considering a class HV with 10,000 dimensions and each dimension is a 32-bit number, each bit position has a pre-determined probability (depending on the voltage level) to flip. The bit error rate (BER) is obtained from real chip characterization [4].

B. Low-cost Error Protection

To mitigate the impact of memory errors on HDC model performance, we explore two low-cost error masking techniques that can detect and mask errors. A simple Razor double-sampling based circuitry [3] can be employed for error (and its location) detection. Note that parity bits protection [7] can also detect memory fault but provides no information on the location of the affected bit. Upon the detection of a bit flip, our scheme can mask the error by setting the faulty bit(s) to 0. The detailed implementations of Razor detection circuitry can be found in [3], [13].

Two error masking granularities are explored: word-level masking and bit-level masking. As shown in Fig. 1, for word masking, upon detecting any bit flips, it masks the entire word to 0. While for bit-level masking, it only recovers the erroneous bit(s) within the word to the value of the sign bit. Combined with Razor circuitry, both methods can

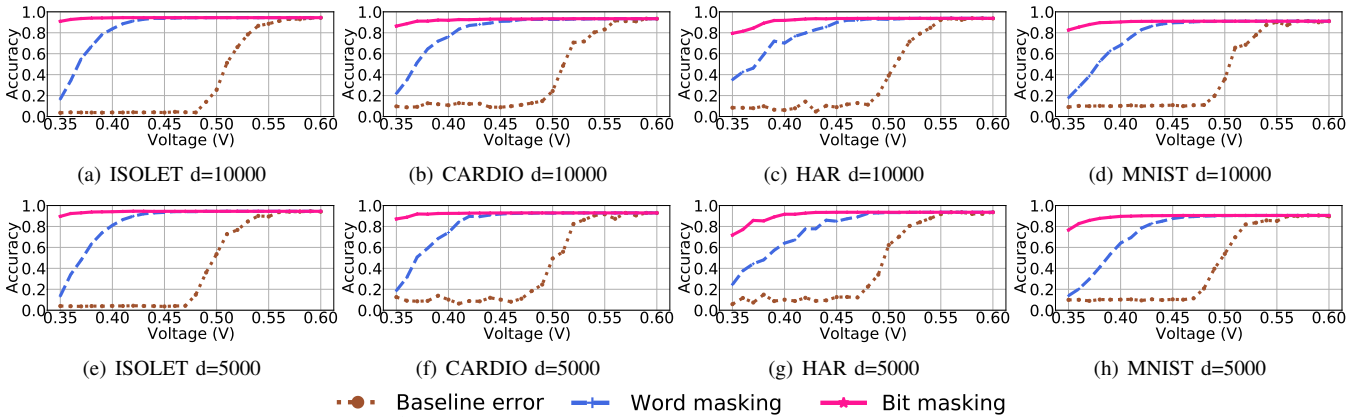

Fig. 2. Classification accuracy of HDC models with 10000/5000 dimensions. "Baseline error" is the basic error injection under voltage scaling, "Word masking" and "Bit masking" apply the word-level and bit-level error protection to the basic error injection.

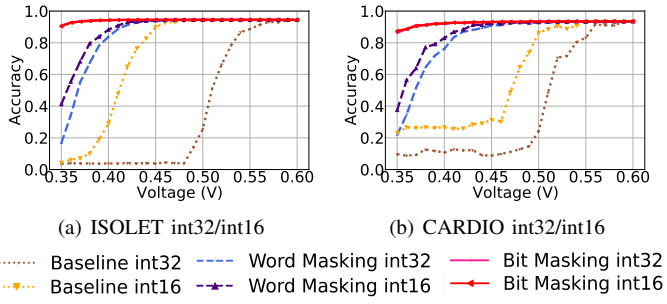

Fig. 3. Voltage Scaling of HDC model across different data-widths

be implemented with 0.3% silicon area and 12.8% power overheads on a single-port SRAM [13]. We incorporate these overheads in our energy analysis in Section IV.

IV. Experimental Results

A. Experimental Setup

We conduct our experiments based on the measurement of voltage-scaled on-chip SRAMs fabricated in FDX22 (22nm) technology [4], which has a nominal V_{DD} at 0.8V (with a BER at $< 10^{-13}$). The BER as a function of voltage scaling is calculated based on $BER = 2e{+}08 \ * \ e^{-61.69 * V_{dd}}$. We then perform SRAM voltage scaling and HDC inference on four datasets: **ISOLET**, **CARDIO**, **HAR**, and **MNIST**. We perform our simulation 10 times and get the average. We measure the original classification accuracy of HDC models with different configurations in terms of dimensions(d)(5000, 10000) and data-widths (int32, int16).

B. Effects of Voltage Scaling on HDC Accuracy

Fig. 2(a)-(d) shows the classification accuracy of HDC models with $D = 10000$ and Int_32 under different voltage levels across four datasets. Note that that all HDC models do not experience accuracy drop until 0.6V (BER $< 10^{-7}$) so we can treat accuracy at 0.6V as the error-free accuracy. This reveals the strong robustness of HDC models to hardware errors. The same phenomenon can also be observed for the models with $D = 5000$ as shown in Fig. 2(e)-(h). Thus, voltage levels above 0.6V can be considered as a "safe" region. When the supply voltage drops below 0.6V, the classification accuracy starts to degrade, as the "baseline error" curve indicates. For example, at voltage level 0.50V, the accuracy drops to around 40% for all the HDC models, i.e., nearly 50% accuracy drop within a 0.1V voltage interval. Thus, we define this interval as voltage "critical" region, which also can be observed for HDC models with $D = 5000$.

The two error masking schemes – word and bit masking, have shown significant accuracy improvement and are able to defer the arrival of the voltage critical region. Specifically, word masking is able to maintain a negligible accuracy loss until the voltage scales to around 0.45V. This is 0.15V lower than the "baseline error" curve that allows the errors to propagate into the HDC computation. Moreover, bit-level masking can further push the voltage critical region down to $< 0.4V$ (BER $\sim 10^{-3}$) with negligible accuracy loss. This error rate is approximately 10,000X of error rate at 0.6V. For certain dataset, e.g., ISOLET, the voltage can even scale down to 0.35V with an acceptable accuracy. Similar phenomenon is observed in Fig. 2(e)-(h) where $D = 5000$. This indicates that bit-level masking is able to improve the HDC error resilience by approximately 10,000X. Bit-level masking performs precisely error correction instead of the whole word,

We also evaluate the effects of data-width in HDC under voltage scaling, as shown in Fig. 3. Our quantization analysis shows that, for CARDIO and ISOLET datasets, 16 bits are sufficient to cover the data range and thus we use INT_16 for these two datasets. An interesting observation from Fig. 3 is that, under the baseline error injection, INT_16 shows better resilience than INT_32. The reason is that the more bits we have, the higher likely bit flips resulting in greater value deviation would happen. For example, a bit flip at most significant bit (MSB) would incur a 2^{31} magnitude change for a 32 bit representation. This makes wider data-widths more vulnerable to voltage scaling-induced errors. The observation also provides a guideline in designing resilient HDC models: we should avoid using "one size fits all" data-width to represent different HDC models but rather developing application-specific configurations by considering the dynamic data characteristics of the application.

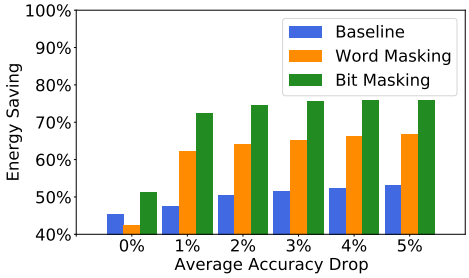

Fig. 4. Energy and accuracy trade-off with 0%, 1%, 2%, 3%, 4%, 5% accuracy loss across all applications and different settings.

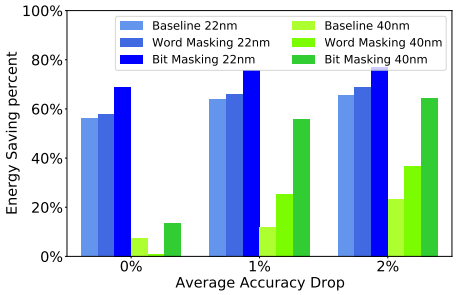

Fig. 5. Energy and accuracy trade-off with 0%, 1%, 2% accuracy loss under different technology nodes (ISOLET 16bit d=10000)

C. Energy Savings

To explore the benefit of voltage scaling, we calculate the average power saving percentage across all four different applications and dimensions. By simply down-scaling the voltage in the "safe" region and allowing the errors to propagate (i.e., the baseline solution), we can achieve up to 47.6% energy saving with negligible accuracy drop ($\leq 1\%$) using $P = CV^2f$. (Energy is proportional to power in this case). Further, word-level masking enables 62.3% energy saving at around 0.46V; bit-level masking can save energy up to 72.5% at around 0.39V with $\leq 1\%$ accuracy loss.

To explore the accuracy-energy trade-off, we relax our accuracy constraints as shown in Fig. 4. We can observe that as the accuracy constraint is increasingly relaxed, more energy saving can be achieved. For example, with 5% accuracy drop, we can achieve 53% energy saving compared to 47.6% energy saving at 1% accuracy drop for baseline voltage scaling. Since HDC itself owns strong robustness, we can find with 0% accuracy drop, word masking save less energy than the baseline because the masking protection hardware need extra 12.8% energy [13]. In addition, we can see that bit-level masking always better than work-level masking, which is always better than baseline case except 0% loss due to the inherent overhead of error protection. This confirms the effects of error protection mechanisms. Note that we incorporate the energy overhead of all the error protection mechanisms.

We further apply our methods to a 40nm SRAM [2] and present the accuracy-energy tradeoff in Fig. 5. Due to space limitation, we only show a specific HDC configuration and application but the trend holds for all the other configurations. Results show that regardless of the technology, voltage scaling, especially with error protection, can achieve significant energy saving. Note that the voltage scaling will not have any impact on the memory latency because it solely leverages the inherent HDC error tolerance to save energy.

V. Conclusion

This paper presents a systematic case study in exploring and characterizing the error resilience of HDC by performing extensive error injection experiments under aggressive voltage scaling. Experimental results show that HDC is inherently resilient to hardware errors, which can lead to 47.6% energy saving of the HDC associative memory, the key hardware component of HDC system. We further investigate two low-cost error mitigation mechanisms that can improve the error resilience of HDC models by up to 10,000X, translating to energy saving by up to 72.5%.

Acknowledgments. This work was partially supported by NSF grant #2028889. Any opinions, findings, and conclusions or recommendations expressed in this material are those of the authors and do not necessarily reflect the views of the National Science Foundation.

References

[1] Amit Agarwal et al. Process variation in embedded memories: failure analysis and variation aware architecture. *JSSC*, 2005.

[2] Daniele Bortolotti, Hossein Mamaghanian, Andrea Bartolini, Maryam Ashouei, Jan Stuijt, David Atienza, Pierre Vandergheynst, and Luca Benini. Approximate compressed sensing: ultra-low power biosignal processing via aggressive voltage scaling on a hybrid memory multi-core processor. In *2014 IEEE/ACM International Symposium on Low Power Electronics and Design (ISLPED)*, pages 45–50. IEEE, 2014.

[3] Shidhartha Das, David Roberts, Seokwoo Lee, Sanjay Pant, David Blaauw, Todd Austin, Krisztián Flautner, and Trevor Mudge. A self-tuning dvs processor using delay-error detection and correction. *IEEE Journal of Solid-State Circuits*, 41(4):792–804, 2006.

[4] Alfio Di Mauro, Francesco Conti, Pasquale Davide Schiavone, Davide Rossi, and Luca Benini. Pushing on-chip memories beyond reliability boundaries in micropower machine learning applications. In *2019 IEEE International Electron Devices Meeting*, pages 30–4. IEEE, 2019.

[5] Lulu Ge et al. Classification using hyperdimensional computing: A review. *IEEE Circuits and Systems Magazine*, 2020.

[6] Mohsen Imani, Deqian Kong, Abbas Rahimi, and Tajana Rosing. Voicehd: Hyperdimensional computing for efficient speech recognition. In *2017 IEEE International Conference on Rebooting Computing (ICRC)*, pages 1–8. IEEE, 2017.

[7] Shah M Jahinuzzaman, Jaspal Singh Shah, David J Rennie, and Manoj Sachdev. Design and analysis of a 5.3-pj 64-kb gated ground sram with multiword ecc. *JSSC*, 2009.

[8] JEDEC Standard JESD218. Solid-state drive (ssd) requirements and endurance test method. *Arlington, VA, JEDEC Solid State Technology Association*, 1:1–1, 2010.

[9] Xun Jiao, Mulong Luo, Jeng-Hau Lin, and Rajesh K Gupta. An assessment of vulnerability of hardware neural networks to dynamic voltage and temperature variations. In *2017 IEEE/ACM International Conference on Computer-Aided Design*, pages 945–950. IEEE, 2017.

[10] Pentti Kanerva. Hyperdimensional computing: An introduction to computing in distributed representation with high-dimensional random vectors. *Cognitive computation*, 1(2):139–159, 2009.

[11] Anton Mitrokhin, P Sutor, Cornelia Fermüller, and Yiannis Aloimonos. Learning sensorimotor control with neuromorphic sensors: Toward hyperdimensional active perception. *Science Robotics*, 4(30), 2019.

[12] Abbas Rahimi, Pentti Kanerva, and Jan M Rabaey. A robust and energy-efficient classifier using brain-inspired hyperdimensional computing. In *Proceedings of the 2016 International Symposium on Low Power Electronics and Design*, pages 64–69, 2016.

[13] Brandon Reagen et al. Minerva: Enabling low-power, highly-accurate deep neural network accelerators. In *ISCA*. IEEE, 2016.

[14] Mohammad Samragh et al. Customizing neural networks for efficient fpga implementation. In *FCCM*. IEEE, 2017.

[15] Jeff Zhang et al. Thundervolt: enabling aggressive voltage underscaling and timing error resilience for energy efficient deep learning accelerators. In *DAC*, 2018.