DropHD: Technology/Algorithm Co-design for Reliable Energy-efficient NVM-based Hyperdimensional Computing under Voltage Scaling

Paul R. Genssler*, Mahta Mayahinia†, Simon Thomann‡, Mehdi B. Tahoori†, and Hussam Amrouch*‡
*Semiconductor Test and Reliability, University of Stuttgart, Stuttgart, Germany
†Department of Computer Science and Informatics, Karlsruhe Institute of Technology, Karlsruhe, Germany
‡Chair of AI Processor Design, TUM School of Computation, Information and Technology, Technical University of Munich; Munich Institute of Robotics and Machine Intelligence, Munich, Germany
Email: genssler@iti.uni-stuttgart.de; {mahta.mayahinia, mehdi.tahoori}@kit.edu; {simon.thomann, amrouch}@tum.de

***Abstract*—Brain-inspired hyperdimensional computing (HDC) offers much more efficient computing compared to other classical deep learning and related machine learning algorithms. Unlike classical CMOS, emerging non-volatile memories (NVMs) used in the realization of HDC are susceptible to failures under voltage scaling, which is essential for energy saving. Although HDC is inherently robust against errors, this is only possible when hypervectors with a large dimension (e.g., 10,000 bits) are being used, resulting in significant energy consumption. This work demonstrates, for the first time, that different NVM technologies exhibit different error characteristics under voltage scaling. In contrast to conventional CMOS-based SRAM, we demonstrate that the error behavior is data-dependent and not captured by simple bit flips in emerging NVMs. We employ our cross-layer framework that starts from the underlying technology all the way up to the algorithm to develop the novel HDC training approach DropHD. DropHD considerably shrinks the size of hypervectors (e.g., from 10,000 bits down to merely 3000 bits), while maintaining a high inference accuracy. The use of aggressive voltage scaling reduces energy consumption by 1.6 x. DropHD further reduces it to up to 9.5 x while fully recovering the induced accuracy drop, i.e., without a tradeoff.**

***Index Terms*—Hyperdimensional computing, Non-volatile memory, Content addressable memory (CAM), Voltage Scaling**

I. Introduction

Brain-inspired hyperdimensional computing (HDC) is a promising machine learning (ML) algorithm. Instead of neurons, HDC is based on large hyperdimensional vectors [1]. Each component in such a hypervector is independent, enabling massively parallel computations. Furthermore, operations on binary hypervectors are very lightweight, e.g., Hamming distance computations during the inference. The independence of the components of a hypervector makes HDC very robust against noise and errors from memory and operations [2–5]. Nevertheless, this robustness and also the inference accuracy depend on the size of the hypervectors. Their dimensionality is in the thousands. The resulting overhead is considerable because high-capacity off-chip storage is slow, has a limited bandwidth, and the repeated data transfers are energy inefficient. Emerging non-volatile memories (NVMs) are a promising group of technologies offering a low-power alternative to SRAM as on-chip memory. They consume little to no static power and have a smaller footprint.

Nevertheless, processing large hypervectors still consumes large dynamic power. Voltage scaling effectively reduces this dynamic power. Yet, reducing the voltage of NVMs shrinks their noise margins, which are already narrower than SRAM, making them more susceptible to noise and their higher variation. Consequently, voltage scaling also introduces errors in HDC's computations and thus reduces the inference accuracy. HDC's robustness against failures stems from the large dimensionality, which contradicts the goal of power reduction. With conventional SRAM, failures express themselves as random data-independent bit flips [4, 6] softening the impact of errors on binary hypervectors. During an HDC inference operation, these bit flips can balance each other out.

In this work, we reveal that voltage scaling, when applied to NVMs, introduces unprecedented error characteristics. Instead of the equiprobable data-independent bit flips, errors are orders of magnitude more probable if the bits of two hypervectors match impacting the computations with similar hypervectors. *Typically the query and correct class hypervector are the most similar ones and thus are impacted the most.* If the matches are evaluated as mismatches instead, then the Hamming distance is reduced increasing the likelihood of misclassification.

Our novel contributions within this paper are as follows:
1) We reveal unprecedented error characteristics for HDC computations. Instead of simple bit flips, traditionally exhibited by SRAMs, voltage scaling applied to NVMs leads to data-dependent and skewed error probability.
2) We propose DropHD, a NVM-aware approach to HDC model training. Starting with a larger dimensionality than desired, the least informative and most error-prone dimensions are dropped. With DropHD, we recover from the errors stemming from the technology and enable significant energy savings.
3) Our technology/algorithm co-design framework covers the full technology stack from a single NVM cell all the way to

This work was partially supported by Advantest as part of the Graduate School "Intelligent Methods for Test and Reliability" at the University of Stuttgart as well as by funding from the pilot program Core Informatics of the Helmholtz Association (HGF) at Karlsruhe Institute of Technology.

979-8-3503-4859-0/DATE24/© 2024 EDAA

the application. By linking these abstraction layers together, we demonstrate reliable yet energy-efficient NVM-based HDC, for the first time, under voltage scaling. Our framework currently supports spin-transfer torque magnetic RAM (STT-MRAM), resistive RAM (ReRAM), and ferroelectric FET (FeFET).
4) We show that the design space for voltage corners does not only include power and latency but also the error probability. Our proposed DropHD makes more Pareto-optimal voltage corners feasible for designers to chose from.

II. Background and Related Work

A. Hyperdimensional Computing

HDC as an ML classifier has been employed successfully for a wide range of tasks such as language recognition [2], image classification [7], gesture recognition [3], wafer map defect pattern classification [8], among others [9]. HDC is based on large vectors in the size of thousands [1]. The large dimensionality creates redundancy in the hypervectors, providing HDC a high degree of robustness to noise and errors [2]. Many different types of individual components of a hypervector have been explored [1] with real numbers, integers, and simple bits being the most common. These types determine the implementation of HDC's fundamental algebra to encode real-world values into hypervectors. During the training phase, for each class, the samples' hypervectors are bundled together into one single class hypervector representing all the samples. During inference, the similarities between an unknown query hypervector and all the class hypervectors are then computed. For binary hypervectors, the similarity metric is the Hamming distance. A common approach to improve the accuracy of an HDC model is retraining, repeatedly passing over the training dataset and adjusting the class hypervectors.

Retraining has been employed in previous work on voltage scaling with HDC. In [4], correct classifications are treated as incorrect if the similarity to the second-best class is too high. By increasing the margin between the classes during training, the model can tolerate more errors from voltage scaling during inference. In [6], integer hypervectors are employed. Since the integer components have a most significant bit, a masking technique was proposed to correct these most impactful errors. An adaptive bit-width scaling scheme is proposed in [10]. If a component can be represented by an 8-bit integer instead of 32-bit, then its width is reduced. This minimizes the impact of a random flip in the most significant bits during the similarity computation, but is challenging to implement at the software level. An SRAM-based ternary content addressable memory was explored in [5]. They explored the impact of voltage scaling on their analog in-memory circuit but did not propose any error mitigation. All of these works focused on voltage scaling in CMOS-based SRAM cells and data-independent bit flips. A reduction of the dimension was only indirectly explored in [10] through the reduction of the bit widths of individual components. A pruning approach for HDC was proposed in [11] but is limited in inference accuracy, increases area, and does not consider errors from the technology.

B. Similarity Check with Non-volatile Memories

STT-MRAM [12], ReRAM [13], and FeFET [14] are among the most prominent NVMs. Although all of these three NVM technologies feature a different storage mechanism, they follow the same idea to create a memory cell. Their state is reflected as a resistance value for STT-MRAM and ReRAM or as a current for FeFET. The high resistive state (HRS) and low resistive state (LRS) values of these properties are mapped to the binary '1' and '0' by convention. Two NVM devices form a CAM cell as shown in Fig. 1 (a). In addition to storing a single bit C, the CAM cell can perform in-memory bit-wise logic computations [15] or logical similarity operation with the query bit Q. Multiple CAM cells and periphery form a CAM structure (Fig. 1 (b)). Before performing an operation by applying Q (represented as a voltage) to the *search-line* of the CAM structure, the *match-line* is precharged. Applying Q then discharges the match-line at a rate dependent on the values of C and Q. In case they match, the discharge path is across the NVM device in the HRS state and is therefore slower. In case of a mismatch, the discharge path passes through the cell in LRS and is faster.

HDC and CAM structures share the concept of constant storage and a variable query. In previous work, the bits of binary class hypervectors were stored in the NVM devices of the CAM

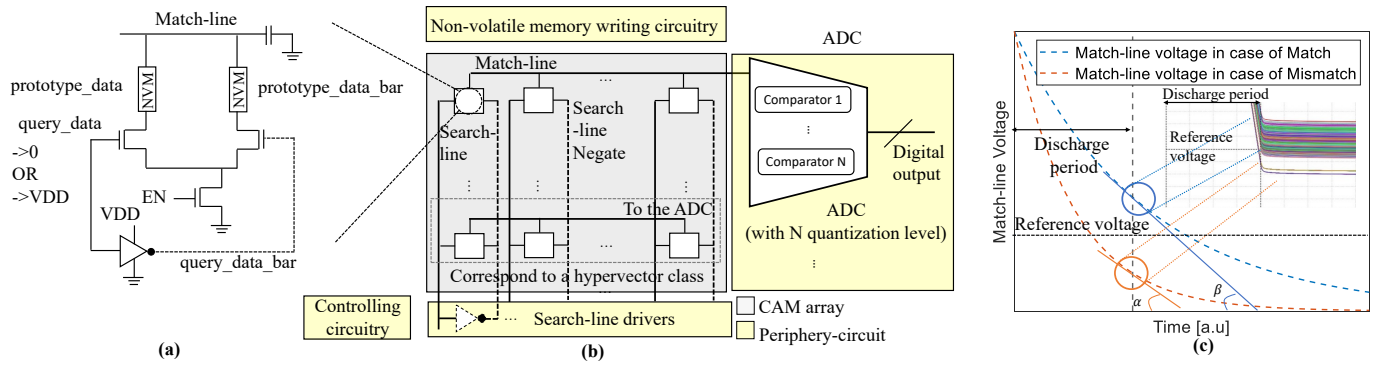

Figure 1. Transistor-level implementation of the NVM-based CAM cell in (a), which is integrated in the CAM structure in (b). The class hypervectors of the HDC model are stored in this CAM structure. During the inference, each CAM cell computes match or mismatch between its stored bit and the applied query bit. The result as a match-line voltage is shown in (c). Voltage scaling can increase the error probability for matches significantly.

structures. At the circuit level, crossbar arrays combine many CAM cells [3] or the match-line is connected to multiple CAM cells [5] enabling an energy-efficient HDC inference in an analog fashion. Moreover, both works report that the observed errors originating from the analog computation were compensated, to some degree, by the inherent robustness of HDC.

III. Voltage-Scaled NVM-CAM structure

In this work, the CAM stores the class hypervectors. An unknown query hypervisor is applied to the CAM structure and is compared to all the class hypervectors. The Hamming distance to each class corresponds to the state of the respective match-line. At the technology level, energy efficiency is increase by voltage scaling. Interestingly, reducing the voltage introduces unprecedented skewed error characteristics in the computation. At the algorithmic level, energy efficiency is increase by scaling the dimensionality of the HDC model. However, both scaling approaches make each level more susceptible to errors. Technology/algorithm co-design is required to mitigate these errors and achieve the highest energy savings. In this section, the error characteristics from the technology are explore and modeled. Section IV introduces a novel HDC training approach to scale the dimensionality.

A. Error Modeling for CAM Structure Under Voltage Scaling

The CAM structure is governed by two main voltage sources. First, array voltage V_{Array} is biasing the CAM array (denoted in gray in Fig. 1 (b)). More specifically, the precharge voltage of the match-line has the value of V_{Array}. Second, the voltage bias of the entire periphery circuit is the periphery voltage $V_{Periphery}$ (denoted in yellow in Fig. 1 (b)), the voltage level of query bits, as well as the enabling signals are also determined based on the $V_{Periphery}$.

To investigate the effect of voltage scaling on the overall functionality of the CAM structure, we sweep them in the viable range. Below this range, the voltage levels are below the threshold voltage and cannot guarantee the expected functionality. Above this range, the risk of transistor break down is high. Although V_{Array} and $V_{Periphery}$ are selected from a viable range, not all combinations result in a functional CAM structure for each NVM technology. The impact of the two voltage biases on the energy consumption and latency is as expected. Higher bias voltages increase energy consumption and decrease the latency. However, the effect of voltage scaling on P_{error} is non-linear and requires further investigation.

B. Effect of Voltage Scaling on the Error Probability

Due to the process variation in the NVM cells, the voltage of the match-line after the pre-defined discharge period is not consistent. Nevertheless, the distributions of the LRS and HRS in different NVM technologies follow the normal distribution. Hence, the two match-line state match and mismatch can each be modeled as a normal distribution. The distance between the means of the two distributions $\Delta\mu$ as well as their standard deviations (σ_{match} and $\sigma_{mismatch}$) determine P_{error}. The higher the distance $\Delta\mu$ and the narrower the distributions are (smaller σs), the lower the error probability. Scaling the two bias voltages in the CAM structure impacts the $\Delta\mu$ and σs in the same way. Either both are increased or both decreased. In other words, the distance between the distributions increase while their width also increases. Alternatively, the distributions become narrower but closer together. Hence, P_{error} has a non-linear dependency with voltage scaling, which is also demonstrated in Fig. 2.

C. Skewed Error Probability of CAM structure

The different states of the match-line for match and mismatch are expressed as different discharging rates after the operand Q has been applied to the search line. Fig. 1 (c) shows a discharge curve for both cases. In the case of a match, the match-line voltage as well as its *slope* β are larger than in case of a mismatch. The larger slope β reflects that the match-line voltage is more susceptible to small changes. For a CAM cell, this change is based on process variation in the underlying NVM devices, which is expressed as a change in the LRS and HRS values. Hence, the voltage of the match-line after the discharge period varies widely although the changes in the

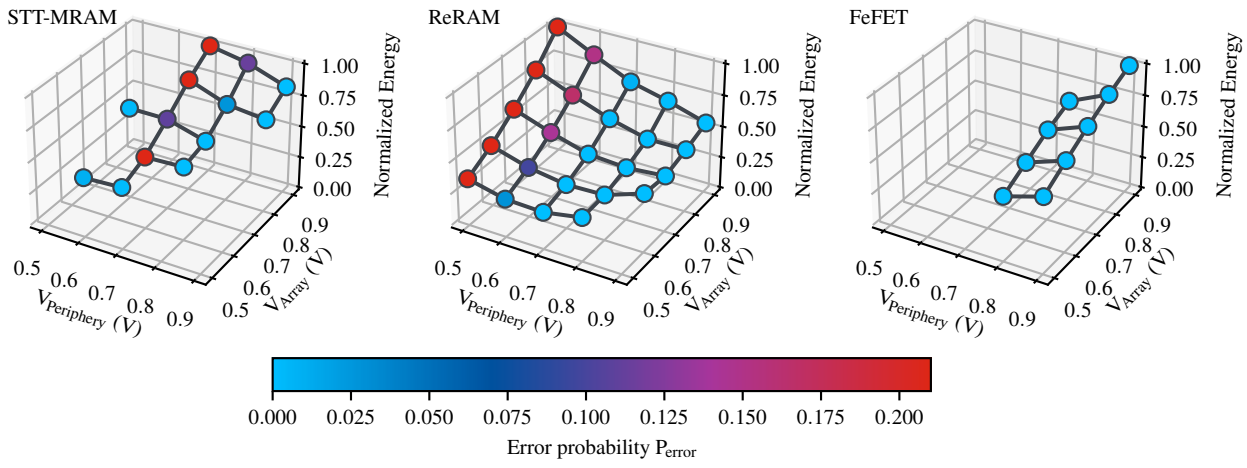

Figure 2. Average energy consumption and P_{error} for STT-MRAM, ReRAM, and FeFET at different voltage corners. The energy values are normalized for each technology and are not directly comparable between them. P_{error} depends on the combination of the voltages but is not proportional to the energy consumption.

Table I
DROPHD REMOVES DIMENSIONS EQUAL OR BELOW THE THRESHOLD E_{th}.

Class 1	0	1	0	1	0
Class 2	0	1	1	1	0
Class 3	0	0	1	1	1
Class 4	1	0	1	1	1
Popcount	1	2	3	4	2
Entropy E	1	2	1	0	2
$E_{th} = 0; E \geq E_{th}$	✓	✓	✓	X	✓
$E_{th} = 1; E \geq E_{th}$	X	✓	X	X	✓

Entropy E = *min*(popcount, number of classes - popcount)

NVM device is small. The mismatch case has a shallower slope α and is less impacted. In summary, the P_{error} for a match can be orders of magnitude higher than for a mismatch. This is experimentally demonstrated by the inset in Fig. 1 (c). On top of the match/mismatch dependency of P_{error}, it also depends on the *combination* of the class bit C (stored in the NVM) and the query bit Q. I.e., P_{error} of (C, Q) = (0,0) can be very different from (1,1) despite the fact that they are both matches. The skewed P_{error} are also observed for mismatches (i.e., (C, Q) = (0,1) and (1,0)). The inverter required to invert the *search-line* in the CAM structure (shown in Fig. 1 (b)), makes the discharging paths asymmetric, leading to the skewed data-dependent error characteristics.

FeFET is different from STT-MRAM and ReRAM. FeFET is a three-terminal device as it is based on a regular transistor [14]. Hence, FeFET has a very high resistance in the HRS state because it is a transistor in the cut-off region. Consequently, a FeFET-based a CAM cell does not discharge at all in the mismatch case, its slope β is close to zero. This eliminates the errors and makes FeFET more reliable compared to the other two technologies as demonstrated in Fig. 2.

IV. DROPHD: A NOVEL HDC TRAINING METHOD

The main source of errors from the hardware are matches between the stored and the query bit. As shown in Fig. 1 (c), errors with mismatching data are orders of magnitudes less likely. Hence, reducing the number of cases with matching data reduces the number of errors and consequently incorrect Hamming distance computations. This is a significant difference compared to typical bit flip error characteristics. With equally likely errors, regardless of the data of the class hypervectors, no optimization and technology/algorithm co-design is possible. Instead, changes to the retraining algorithm have been proposed [4] or errors masked [6]. None of these approaches have reduced the dimensionality.

We propose the new entropy-maximizing approach DropHD to tackle two challenges at once, the reduction of the dimensionality while increasing the robustness against increased errors from the hardware due to voltage scaling. To achieve both, the entropy of each dimension in the class hypervectors is analyzed as illustrated in Tab. I. In this work, the entropy of a single dimension d_i is defined as the popcount of the bits from all class hypervectors at d_i or the number of class subtracted by that popcount, whatever is lower. In other words, very high or low popcounts have less entropy. If the entropy is zero, i.e., all classes share the same bit value at d_i, then this dimension has no entropy and is removed. It contributes to all Hamming distances equally, regardless of the query hypervector, and thus does not impact the inference accuracy. The dimension is also removed from the hypervectors used for encoding. That is possible because each dimension is independent and does not influence others. By removing the no-entropy dimensions, the importance of the remaining ones is increased. It can be further maximized by removing dimensions with an entropy below or equal to the entropy threshold E_{th}. However, removing distinguishing dimensions ($E_{th} > 0$) impacts the inference accuracy because those dimensions could be the difference between two otherwise similar classes. This drop in inference accuracy can be compensated to a degree by retraining on the reduced hypervectors.

Removing low-entropy dimensions does not only reduce the dimensionality and thus energy consumption, it actually improves the accuracy while utilizing the CAM structure for inference. If the entropy of d_i is low, the likelihood is very high that the query hypervector has the same bit value as the classes. With ideal hardware, these matching values are not of concern. In contrast, these matching values are the main source of errors in the CAM structure. Hence, removing them reduces the number of errors. This creates an optimization problem with contradicting goals. On the one hand, removing dimensions shared by many classes reduces the number of errors and thus recovers some inference accuracy. On the other hand, removing too many distinguishing dimensions reduces the inference accuracy. To solve the optimization problem, both the P_{error} of the CAM structure from the technology level and the hypervectors from the algorithmic level, have to be considered in a holistic framework.

Table II
SIMULATION PARAMETERS FOR THE NVM TECHNOLOGIES.

Technology node for CMOS Nominal V_{DD} / temperature	GlobalFoundries 22FDX 0.8 V / 27 °C
FeFET model [14]	- Material = $Hf_{0.5}Zr_{0.5}O_2$ - Ferro layer thickness = 10 nm
MTJ model [12]	- RA = 7.5 Ω µm^2 - Nominal TMR = 150 %
ReRAM model: *JART* [13, 16]	- Filament radius = 45 nm - Disc region length = 0.6 nm

V. OUR CO-DESIGN METHODOLOGY AND RESULTS

The developed technology/algorithm co-design framework reveals unprecedented error characteristics and bridges the gap between the technology and algorithmic levels. The framework is introduced first before demonstrating the benefits of the proposed DropHD.

A. Technology/Algorithm Co-design Framework

We perform a detailed electrical-level simulation on the 1-bit CAM cell (Fig. 1 (a)) to extract its latency, energy, and P_{error} for all voltage corners ($V_{Periphery}$, V_{Array}) and data combinations. Monte-Carlo SPICE simulations are performed to obtain the

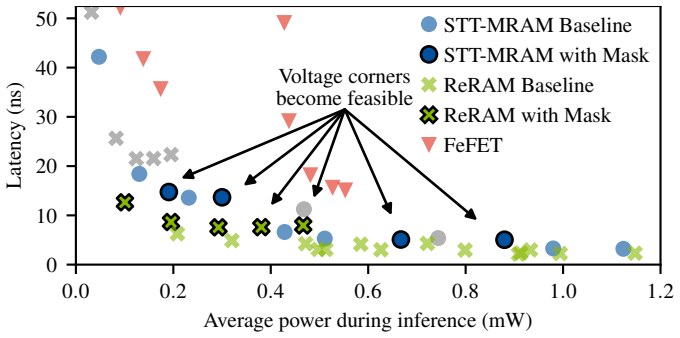

Figure 3. Voltage corners are infeasible if they cause an inference accuracy drop by more than 2 %. The proposed DropHD enables otherwise infeasible corners, including Pareto-optimal and low-power corners.

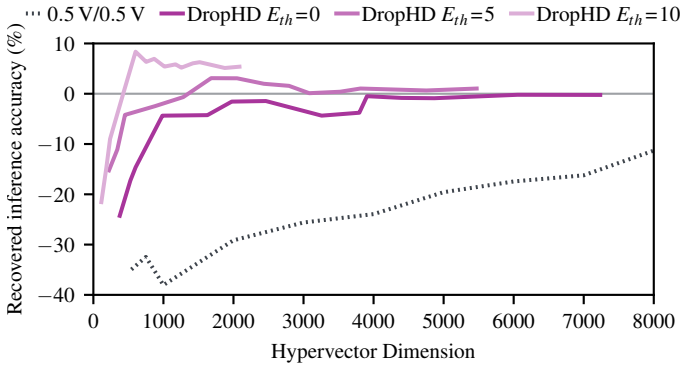

Figure 4. Capability of DropHD to recover from hardware-induced errors across different dimensionalities. The baseline is a retrained model executed with ideal error-free hardware. The same model with ReRAM's 0.5 V/0.5 V low-power corner suffers catastrophically loss in inference accuracy for the ISOLET dataset. DropHD is robust and even improves the inference accuracy at lower dimensions.

distributions of match-line states for match and mismatch. The results of the 1-bit CAM cell are then extrapolated to the entire size of the hypervector (d-bit). The extraction is repeated for each technology with the NVM models as described in Tab. II, which have been validated against measurements. Due to the additive nature of energy, the energy of the 1-bit CAM cell is accumulated accurately based on the bits of the query and class hypervectors, i.e., the data dependency of the energy consumption is taken into account.

At the algorithmic level, the samples from the dataset are encoded to d-dimensional hypervectors. A basic HDC model is created by averaging (bundling) the sample hypervector of each class. TorchHD [17] provides a basic HDC implementation and is extended to model the error characteristics and energy consumption. With the class hypervectors the associate memory, based on the CAM structure, is programmed. To emulate process variation, two error masks are randomly initialized based on the P_{error} of the voltage corner and the bits of the class hypervector. Each mask represents a combination of class and query bit to capture the data dependency. If the error bit is set in a mask, then the inference will be incorrect later for this specific combination stored and applied value. The masks are constant during retraining, but regenerated for testing to mimic the deployment of the model on a different chip. In addition to the explored technologies, an ideal error-free hardware model is included as a baseline.

For the retraining under the impact of errors, hypervectors with integer components are employed to improve the retraining efficiency. During the forward pass, i.e., the inference, the hypervectors are quantized to binary to fit the proposed hardware architecture. If a classification is incorrect, the integer class hypervector is adjusted slightly to make it more similar to the query. As proposed in [4], even correct classifications are treated as incorrect if the Hamming distance to the second-most similar class is less than 0.5 % of the dimensionality. During the 100 epochs of retraining, the best HDC model (based on the training accuracy) is stored and selected upon completion. Lastly, this model classifies the test dataset to report the inference accuracy. DropHD is applied to all hypervectors after the retraining step.

During retraining and inference, the energy consumption for each CAM cell is recorded for future analysis. The presented results are based on the ISOLET voice recognition dataset [9] and MNIST. All experiments are repeated ten times with different initial hypervectors to reduce the influence of randomness.

B. Enabling More Voltage Corners

Voltage scaling can increase P_{error} above 20 % as shown in Fig. 2. Some voltage corners become infeasible, in particular in ReRAM with a low $V_{Periphery}$. In Fig. 3, a voltage corner is marked as infeasible (gray) if the inference accuracy of a retrained HDC model for the ISOLET benchmark with a dimensionality of 3000 drops by more than 2 %. For ReRAM, the corners with the lowest power consumption are infeasible. With DropHD, four and five additional voltage corners become feasible for STT-MRAM and ReRAM, respectively. This includes some Pareto-optimal as well as low-power corners. With MNIST, a total of six corners become feasible. The results show that the proposed algorithmic-level technique DropHD increases the design space at the technology level.

C. Recovering Inference Accuracy

One principle of HDC is that an increase in the dimensionality also increases the robustness against errors. This is also demonstrated by Fig. 4, which compares ReRAM's voltage corner with the lowest power against an ideal error-free hardware model. Baseline models are trained for different dimensions and their accuracy forms the zero-line. The retrained HDC model cannot recover from the high P_{error} of the corner. Although inference accuracy steadily increases, it is more than 10 % below the achievable accuracy without voltage scaling. The retraining approach cannot compensate at the hardware level because the error-inducing cells are distributed differently from one manufactured chip to the next. In contrast, DropHD removes error-inducing dimensions at the algorithmic level and thus is independent of the hardware.

In Fig. 4, different entropy thresholds E_{th} are explored. DropHD with $E_{th} = 0$, but without retraining, provides an 8 % to 15 % improvement over retraining (not shown). Even more

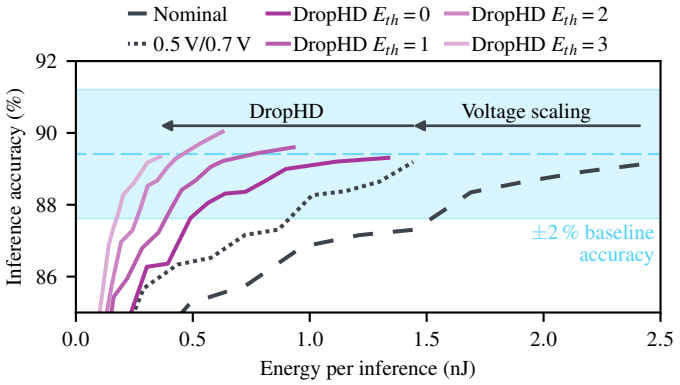

Figure 5. Voltage scaling to the low-energy corner of ReRAM (0.5 V/0.7 V) reduces the energy consumption by 1.67 x. With the MNIST dataset, shown here, DropHD reduces energy consumption further to a combined of 6.70 x, for ISOLET by 5.26 x.

accuracy is recovered by applying DropHD full and retraining the reduced model for 100 epochs. Without the redundant dimensions, altering the remaining ones has a higher impact and is more efficient than retraining with all dimensions. At a dimensionality of 4000, the inference accuracy has recovered to that of the ideal error-free hardware. If low-entropy dimensions are removed more aggressively, e.g., with E_{th} = 10, then the inference accuracy recovers faster and exceeds the baseline.

D. Voltage Scaling and DropHD for Energy Reduction

Not all voltage corners have a high P_{error}. For example, at $V_{Array} = 0.5$ V and $V_{Periphery} = 0.7$ V the energy per inference is the lowest with ReRAM. At the same time, the combined P_{error} is negligible at 1.4×10^{-8}. Hence, recovering the inference accuracy is only secondary and DropHD can aggressively reduce the dimensionality as demonstrated in Fig. 5. Through voltage scaling down from the nominal corner of 0.8 V, energy consumption per inference is reduced by 1.67 x. Due to the data dependency, for ISOLET the reduction is 1.61 x. A reduction of the dimensionality through DropHD directly reduces not only the energy consumption but also chip area, data movement, and other non-functional metrics. On average, 52.5 % of the 10,000 dimensions are removed with E_{th} = 0. One reason for such a high number of non-entropy dimensions could be due to the bundling of the training sample hypervectors into a class hypervector and the consequent binarization. The underlying integer HDC model accumulates the samples in a fine-grained manner and could utilize this information during inference. However, to utilize the hardware, all components are binarized resulting in a loss of information.

The data dependency is also reflected in Tab. III. While the impact on voltage scaling is consistent ±0.1 x, DropHD's selection of low-entropy dimensions dictates the energy savings. The results in Tab. III are averaged over different dimensionalities, all of which are within a ±2 % range of the target accuracy at d = 10,000. For MNIST, the dimensionality can be reduced by 6 x (E_{th} = 5), for ISOLET by 4 x (E_{th} = 3). This is reflected in the energy reduction of 8.6 x to 9.5 x and 5.2 x to 6.3 x.

Table III
ENERGY REDUCTION WITH VOLTAGE SCALING FROM THE NOMINAL VOLTAGE, WITH DROPHD, AND BOTH COMBINED AT ISO-ACCURACY.

	Technology Voltage Corner	ReRAM 0.5 V/0.7 V	STT-MRAM 0.5 V/0.6 V	FeFET 0.5 V/0.8 V
ISOLET	Voltage scaling	1.6 x	1.5 x	1.6 x
ISOLET	DropHD	3.3 x	3.9 x	3.3 x
ISOLET	Combined	5.7 x	6.3 x	5.2 x
MNIST	Voltage scaling	1.7 x	1.6 x	1.6 x
MNIST	DropHD	5.5 x	5.6 x	5.5 x
MNIST	Combined	9.5 x	9.5 x	8.6 x

VI. CONCLUSION

In this work, we have proposed DropHD, a novel approach to HDC model training, to achieve two otherwise contradictory goals. The dimensionality of the hypervectors can be reduced by up to 6 x while increasing the robustness against errors from the hardware. Our detailed framework connects the hardware with the algorithmic level to enable technology/algorithm co-design. With DropHD, we make more voltage corners feasible and recover more than 40 % inference accuracy compared to the retraining baseline. Energy can be reduced by 1.6 x with voltage scaling and up to 9.5 x if combined with DropHD.

REFERENCES

[1] D. Kleyko et al., "A survey on hyperdimensional computing aka vector symbolic architectures, part i: Models and data transformations," ACM Comput. Surv., 2022.

[2] A. Rahimi et al., "A Robust and Energy-Efficient Classifier Using Brain-Inspired Hyperdimensional Computing," in *Proceedings of the International Symposium on Low Power Electronics and Design*, 2016.

[3] G. Karunaratne et al., "In-memory hyperdimensional computing," Nature Electronics, vol. 3, 2020.

[4] D. Liang et al., "DependableHD: A hyperdimensional learning framework for edge-oriented voltage-scaled circuits," in *Proceedings of the 28th Asia and South Pacific Design Automation Conference*, 2023.

[5] S. Thomann et al., "Hw/sw co-design for reliable tcam-based in-memory brain-inspired hyperdimensional computing," IEEE TC, 2023.

[6] S. Zhang et al., "Energy-efficient brain-inspired hyperdimensional computing using voltage scaling," in *IEEE DATE*, 2022.

[7] D. Kleyko et al., "Holographic graph neuron: A bioinspired architecture for pattern processing," IEEE TNNLS, 2017.

[8] P. R. Genssler et al., "Brain-inspired computing for wafer map defect pattern classification," in *IEEE International Test Conference*, 2021.

[9] M. Imani et al., "Voicehd: Hyperdimensional computing for efficient speech recognition," in *2017 IEEE ICRC*, 2017.

[10] S. Zhang et al., "Scalehd: Robust brain-inspired hyperdimensional computing via adapative scaling," in *IEEE/ACM ICCAD*, 2022.

[11] R. A. G. Antonio et al., "Redundancy pruning for binary hyperdimensional computing architectures," in *IEEE ISCAS*, 2022.

[12] F. Bernard-Granger et al., "SPITT: A magnetic tunnel junction SPICE compact model for STT-MRAM," in *Proceedings of the MOS-AK Workshop of the Design, Automation & Test in Europe (DATE)*, 2015.

[13] C. Bengel et al., "Variability-aware modeling of filamentary oxide-based bipolar resistive switching cells using spice level compact models," IEEE TCAS-I: Regular Papers, 2020.

[14] S. Kumar et al., "Cross-layer reliability modeling of dual-port fefet: Device-algorithm interaction," IEEE TCAS-I: Regular Papers, 2023.

[15] M. Mayahinia et al., "Voltage tuning for reliable computation in emerging resistive memories," in *IEEE VTS*, 2022.

[16] S. Wiefels et al., "Hrs instability in oxide-based bipolar resistive switching cells," IEEE Transactions on Electron Devices, 2020.

[17] M. Heddes et al., "Torchhd: An open source python library to support research on hyperdimensional computing and vector symbolic architectures," Journal of Machine Learning Research, vol. 24, 2023.