All You Need is Unary: End-to-End Bit-Stream Processing in Hyperdimensional Computing

Mehran Shoushtari Moghadam*, Sercan Aygun*, Faeze S. Banitaba, and M. Hassan Najafi
School of Computing and Informatics, University of Louisiana at Lafayette, Lafayette, LA, USA
{m.moghadam, sercan.aygun, faeze.banitaba, najafi}@louisiana.edu
(*: Both authors contributed equally to this research.)

ABSTRACT

Hyperdimensional Computing (HDC) is a brain-inspired computing paradigm introduced to achieve energy efficiency with a lightweight and single-pass training model. Hypervectors (HVs) at the heart of the HDC systems play a fundamental role in elevating the accuracy and obtaining the desired performance. Image-based HV encoding requires two types of HVs: *Position* and *Level* HVs. State-of-the-art approaches utilize pseudo-random methods for generating these HVs, which might degrade system performance and cause higher power consumption due to poor randomness in HV generation. These conventional methods require iteratively calculating orthogonal *Positional* HVs for acceptable accuracy. This work proposes a fast, ultra-lightweight, and high-quality HV generator incorporating low-discrepancy random sequences and the emerging unary bit-stream processing. For the first time, we employ unary computing (UC) to generate *Level* HVs, demonstrating that there is no need for randomness in HDC systems. We generate *Position* HVs using a single-source quasi-random sequence with a recurrence property. Our proposed HV generation technique improves the overall HDC accuracy by up to 6.4% for the medical MNIST dataset while reducing the power consumption of HV generation by 98%.

CCS CONCEPTS

• **Hardware** → **Emerging technologies**; • **Mathematics of computing**; • **Computer systems organization** → Real-time systems; • **Computing methodologies** → *Cross-validation*;

KEYWORDS

Hyperdimensional computing, low-discrepancy sequences, low-power AI, random number generators, unary computing.

1 INTRODUCTION

Unary computing (UC) [23, 31, 32] has emerged as a compelling computational paradigm, drawing inspiration from human brain signals. The paradigm is well-known for offering streamlined hardware architectures. In contrast to traditional positional binary encoding, where significance is attributed to bit positions, UC represents data using cumulative counts distributed throughout a bit-stream with logic 1, while the remaining positions hold logic 0. This unconventional presentation of data significantly simplifies arithmetic operations while providing high robustness to error. Hyperdimensional computing (HDC) is another brain-inspired computational model representing scalars (and symbols) using long hypervectors ($\mathcal{HV}$s) reminiscent of bit-streams. For machine learning (ML) tasks such as classification, HDC encodes input data into long vectors to capture class information and construct learning models [17]. The encoding process involves various steps, including $\mathcal{HV}$ generation, shifting, multiplication, and addition of resulting $\mathcal{HV}$s. Each new data point contributes to the $\mathcal{HV}$ of the same class with no error optimization. The process is single-pass, meaning each input data is processed only once. While some state-of-the-art (SOTA) approaches adopt single-pass learning [15], epoch-based processing is also popular [7, 35].

In the existing literature, only a few studies explored unary bit-stream processing in classifier systems [4, 10, 22, 26]. This work employs UC in designing HDC systems to achieve the lightest possible classifier network. Conventionally, HDC systems employ correlation-aware bit flipping for data encoding. In this approach, similar numerical values are encoded with correlation, while distant values exhibit a larger margin of uncorrelation. Prior methods introduce randomness in bit changes when transitioning $\mathcal{HV}$ bits from one value to another. In this work, we advocate *unary* $\mathcal{HV}$s, free from randomness. Generating unary $\mathcal{HV}$s is straightforward and cost-efficient, offering a promising alternative to conventional random $\mathcal{HV}$s. We further introduce a novel encoding approach utilizing single-source quasi-randomness using low-discrepancy (LD) sequences [16] to generate *Position* $\mathcal{HV}$s. Unlike previous methods that employed a different random sequence for each *Positional* $\mathcal{HV}$, our approach uses a simple logic design to produce various $\mathcal{HV}$s while ensuring the necessary orthogonality for the position $\mathcal{HV}$s [30]. The key contributions of this work are as follows:

❶ Introducing novel encoding methods for generating *Position* and *Level* $\mathcal{HV}$s.
❷ Presenting a cost-efficient design for generating $\mathcal{HV}$s by exploiting quasi-random sequences.
❸ Reevaluating feature extraction-free, straightforward data processing for HDC by utilizing unary bit-stream processing.
❹ Assessing the Medical MNIST dataset for biomedical applications of HDC and analyzing various ML metrics derived from the confusion matrix.

ISLPED'24, August 5–7, 2024, Newport Beach, CA

ACM ISBN 979-8-4007-0688-2/24/08
https://doi.org/10.1145/3665314.3670834

2 BACKGROUND

HDC is a niche computing paradigm emerging as a promising tool in electronic design systems for tiny ML applications, especially for classification tasks. HDC system blocks are built with several levels of logic gates, such as XOR gates, counters, and shifters [1, 9]. Encoding, identified as the first and most crucial step in HDC systems [5, 12], is nearly *the only* stage where input data undergoes processing. Data processing typically occurs in a single pass in most HDC models. The encoding process transforms input data (e.g., scalars/symbols) into a distinct format (i.e., bit vectors), generating class information corresponding to each data label. An efficient encoding stage plays a pivotal role in enhancing the overall system performance, improving accuracy, and facilitating energy-efficient design [5, 9, 14]. In the final stage, a similarity calculation is necessary to identify similar classes and label the sample [29].

In a high-level classification, HDC systems can be divided into two categories: ① symbol-only systems and ② numerical-value systems [9, 11]. If the classification problem contains only symbols, such as language classification or text processing [3], the symbols are the only input values to process. For instance, in these cases, letters or positions are the critical symbol-like inputs converted to $\mathcal{HV}$s. For inputs such as pixel values in image classification problems, the HDC system treats the data as numerical values. Generally, the closer the numerical values, the more similar $\mathcal{HV}$s are in the HDC model.

This study focuses on a medical image processing system [33], where pixels and their positions are important for the HDC model. The encoding process begins by converting these data into suitable $\mathcal{HV}$s. The resulting $\mathcal{HV}$s are binary, comprising logic 1s and 0s. Ensuring correlation among the generated vectors is crucial, making the choice of the random source needed for generating $\mathcal{HV}$s pivotal. Particularly for $\mathcal{HV}$s requiring orthogonality, the level of randomness holds significant importance. Since symbols (here, pixel positions) lack numerical information, they must be equally treated, with an equal probability for both logic 1 and 0 within the vector. There should be no inherent similarity between the $\mathcal{HV}$s corresponding to different symbols. Each symbol must remain independent to ensure classifiers estimate it unbiasedly. Hence, for symbol-based problems, the midpoint of a probability range ($0 < Pr = \frac{1}{2} < 1$) is chosen for each $\mathcal{HV}$.

The SOTA methods commonly rely on pseudo-random sequences for the encoding stage [2, 13, 27, 34]. However, employing quasi-random sequences for $\mathcal{HV}$ generation could revolutionize the paradigm. In this study, we explore the use of quasi-random *Van der Corput (VDC) sequences*, as the basis for $\mathcal{HV}$ generation. In general, any VDC sequence in an arbitrary base $\mathcal{B}$ (VDC-$\mathcal{B}$) could be obtained by simply reversing the digits with respect to the radix point, which is a value in the $[0, 1]$ interval. For instance, the decimal value 107 in base 5 is represented by $(412)_5$. The corresponding VDC-5 value is found by $2 \times 5^{-1} + 1 \times 5^{-2} + 4 \times 5^{-3} = \frac{59}{125}$. Considering the high demand for low-cost generation of $\mathcal{HV}$s, we explore the special case of using powers-of-2 bases for the VDC sequences (VDC-2^n). This is as simple as designing a $log_2(D)$-bit counter, where D is the $\mathcal{HV}$ length. In this case, a simple hard-wiring scheme can easily generate any VDC-2^n sequence without adding any extra hardware component [21, 28].

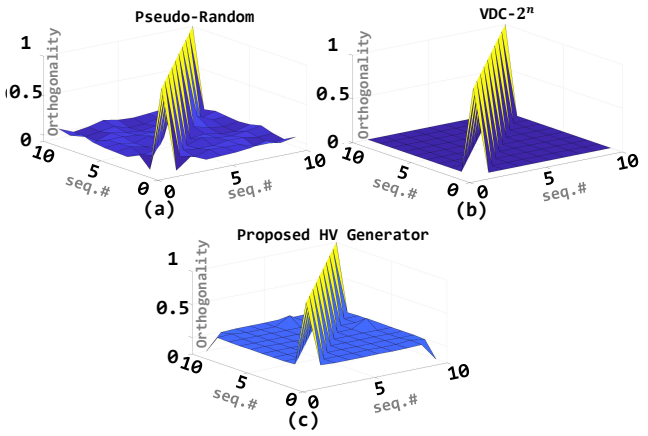

Figure 1: Similarity comparison of different $\mathcal{HV}$ generation sources. (a) Pseudo-Random, (b) VDC-2^n, and (c) Proposed $\mathcal{HV}$ generator. The first 10 $\mathcal{HV}$s are selected for each method (D = 1024).

$\mathcal{HV}$s must have an equal number of logic 1s and 0s. On the other hand, the performance of HDC models is highly dependent on the level of orthogonality between $\mathcal{HV}$s; The more orthogonal $\mathcal{HV}$s, the better the HDC performance. The conventional (Baseline) $\mathcal{HV}$ generation methods with pseudo-random sequences create low-quality $\mathcal{HV}$s due to the poor "randomness" of these sequences. Figure 1 demonstrates the inter-orthogonality between a sample of ten $\mathcal{HV}$s when utilizing different $\mathcal{HV}$ generator sources. The *cosine similarity* is used to measure the level of orthogonality [2]. For the pseudo-random method (Figure 1 (a)), the orthogonality is poor due to existing intrinsic randomness in $\mathcal{HV}$ generation. On the other hand, the one with VDC-2^n sequences performs perfectly, as there are no fluctuations in its orthogonality plot (Figure 1 (b)). As the symbol $\mathcal{HV}$s (or *Positional* $\mathcal{HV}$s) require high orthogonality, the VDC-2^n sequences may not perform well when the number of distinct symbols exceeds $log_2(D)$. To address this limitation, we propose a novel technique to generate independent $\mathcal{HV}$s by utilizing **only one** sequence generator (VDC-2), one T flip-flop (T-FF), and one XOR gate. Figure 1 (c) depicts the inter-orthogonality performance of the first ten $\mathcal{HV}$s utilizing the proposed single-random source $\mathcal{HV}$ generator.

3 PROPOSED METHOD

3.1 Design 1: Single-Source, Yet Sufficiently-Random Generator for *Position* $\mathcal{HV}$s

Our initial design proposal focuses on symbol-based $\mathcal{HV}$ generation and its corresponding encoding. Presently, the SOTA utilizes *any* random source, in most cases "pseudo"-random [18, 19]. However, relying on such random sources poses several risks. Firstly, there is the issue of randomness, which necessitates repetition. While a training trial with a particular randomness may yield satisfactory validation accuracy for a classification problem, another iteration could produce a better or worse result. Consequently, it is necessary to iterate multiple times to achieve the best accuracy. The number of needed iterations to guarantee high accuracy, particularly for the cases of using shorter $\mathcal{HV}$s, can be very high.

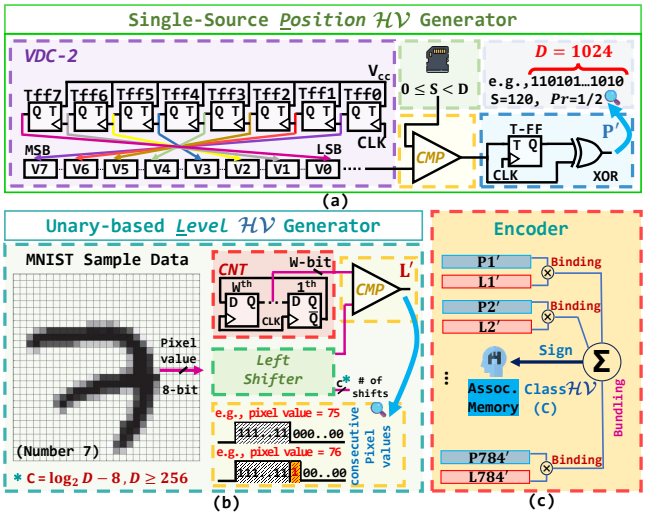

Figure 2: The overall design of proposed $\mathcal{HV}$ generators. (a) *Position* $\mathcal{HV}$ generator including 1×VDC-2 random sequence generator + 1×comparator + 1×T-FF + 1×XOR gate. (b) *Level* $\mathcal{HV}$ generator design including 1×up-counter + 1×left-shifter + 1×comparator. The output $\mathcal{HV}$s (L's) are correlated to each other in a unary format representation. (c) The conventional record-based encoding in the HDC model.

Another concern with pseudo-randomness in $\mathcal{HV}$ generation is the efficiency of hardware design. The concern extends beyond its relation to randomness and encompasses hardware design considerations. While it may be acceptable to utilize pre-determined random vectors for a limited dataset, thereby disregarding computational load, certain problems require dynamic vector generation. For instance, in cases where the size of input images (e.g., pixel positions) varies, additional position vectors are essential. Hence, an ideal HDC system requires an $\mathcal{HV}$ generator with: *strong orthogonality*, *lightweight hardware*, and *reduced iteration* to ensure efficient generation of $\mathcal{HV}$s. To satisfy these requirements, we use a quasi-random sequence generator to produce the VDC-2 sequence. Our primary objective is to achieve optimal randomness in a single iteration, entangled with the recurrent nature of the random sequence and an ultra-lightweight design. Distinguishing itself from the Baseline HDC (with pseudo-random sources, such as linear-feedback shift registers - LFSR), our method does not employ multiple random sequences to generate m different D-sized vectors and subsequently use them for $\mathcal{HV}$ generation. Instead, we generate only a single D-sized sequence and employ it to generate m different vectors.

Once a VDC-2 sequence is generated, we employ the proposed circuit structure of Figure 2 (a) to generate different symbol (*Position*) $\mathcal{HV}$s. This circuit comprises a T-FF and an XOR gate. The binary sequence elements are paired and compared with a scalar (S) value (in binary) within the range $[0, D]$. Each element from the VDC-2 generator (with size D) is compared with S, and the result is recorded as logic 1 if $S > D$ and logic 0 otherwise. The generated bit is then fed to a T-FF and XORed with itself. With this configuration, the resulting vector exhibits a $\frac{1}{2}$ probability (half logic 1s and half logic 0s) with quasi-random distribution. By repeating this operation for K different symbol $\mathcal{HV}$s, independent quasi-random

Figure 3: Unary *Level* $\mathcal{HV}$ compared to the conventional randomly bit-flipped *Level* $\mathcal{HV}$.

$\mathcal{HV}$s with a probability of $Pr = \frac{1}{2}$ are generated at a very low cost. At this juncture, we establish a design checkpoint to report the cost of the proposed $\mathcal{HV}$ generation design. The proposed design consumes 25% less power than the Baseline design for generating each D=1024 size $\mathcal{HV}$.

3.2 Design 2: Unary Computing for *Level* $\mathcal{HV}$s

Another key contribution of this work is to develop lightweight logic hardware for representing *Level* $\mathcal{HV}$s in the HDC system. For the first time in the literature, we represent *Level* $\mathcal{HV}$s not randomly but deterministically by unary generated $\mathcal{HV}$s. We argue there is no need for randomness in *Level* $\mathcal{HV}$s. Our proposed design for generating unary style *Level* $\mathcal{HV}$s includes a left shifter module, an up-counter (*CNT*), and a comparator (*CMP*). For $D \geq 256$, the shifter block shifts the pixel intensity value by $(log_2 D - 8)$-bits to generate the desired *Level* $\mathcal{HV}$ for the current pixel intensity value. The up-counter is a $log_2 D$-bit Johnson counter built with simple D-type flip flops. The structure of the proposed *Level* $\mathcal{HV}$ generator is depicted in Figure 2(b). The rest of the encoding process, including binding and bundling phases, remains the same as in the Baseline HDC [20] (Figure 2(c)).

A significant aspect of utilizing unary-style *Level* $\mathcal{HV}$s is in their inherent energy efficiency due to a single transition from 0 to 1 (or from 1 to 0) [24], as depicted in Figure 3. Since there is only one bit-level transition (≈ 0 activity factor), the associated switching power dissipation is negligible. This provides a significant improvement over the Baseline $\mathcal{HV}$ generation methods. The Baseline approach suffers from high switching activity due to leveraging a random bit-flipping process [6], which increases the overall switching (dynamic) power consumption of the system.

4 EXPERIMENTAL RESULTS

4.1 Hardware Efficiency

To evaluate the hardware efficiency of the proposed design, we implemented the design of Figure 2 in Verilog HDL and synthesized it using the Synopsys Design Compiler v2018.06 with the 45nm FreePDK gate library. Table 1 compares the hardware cost of the Baseline and the proposed *Position* $\mathcal{HV}$ generator. Since the Baseline *Position* $\mathcal{HV}$ generator utilizes LFSR as the random source, generating the needed independent and orthogonal $\mathcal{HV}$s significantly increases area, power, and energy consumption proportional to the number of distinct pixel positions inside the image. In other words, for any image as the model input, we require r×c distinct LFSRs, where r and c are the numbers of image rows and columns, respectively. On the other hand, incorporating the proposed *Position* $\mathcal{HV}$ generator does not require many distinct $\mathcal{HV}$ generators. Employing a single VDC-2 sequence as the random source would be sufficient to generate independent and orthogonal *Position* $\mathcal{HV}$s when integrating it with a T-FF and an XOR gate. Utilizing the

Table 1: Hardware Cost Comparison of Generating *Position* $\mathcal{HV}$s using the Baseline and the Proposed Method (D = 1024)

Design Approach	Baseline				Proposed			
	CPL (ns)	Area (μm^2)	Power (mW)	Energy (nJ)	CPL (ns)	Area (μm^2)	Power (mW)	Energy (nJ)
Per $\mathcal{HV}$ bit	0.380	246	0.797	3.03×10^{-4}	0.430	288	**0.597**	$\mathbf{2.57\times10^{-4}}$
Per entire $\mathcal{HV}$	0.380	246	0.797	0.310	0.430	288	**0.597**	**0.263**
Per Image	0.380	192864	624.8	243.0	0.430	**288**	**0.597**	**206.1**

The results are obtained by considering the MNIST dataset images as a reference. ‖ CPL: Critical Path Latency.

Table 2: Hardware Cost Comparison of Generating a Single *Level* $\mathcal{HV}$ using the Baseline and the Proposed Method (D = 1024)

Design Approach	CPL (ns)	Area (μm^2)	Power (mW)	Area×Delay ($\mu m^2 \times$ns)
Baseline	0.330	10587	49.621	3493.710
Proposed	**0.310**	**287**	**0.725**	**88.970**

Considering 8-bit gray-scale image pixels within the [0,255] interval.

proposed *Position* $\mathcal{HV}$ generator reduces the power consumption by 98% while improving energy efficiency by 15% compared to the Baseline method.

Similarly, we implemented the proposed unary-based *Level* $\mathcal{HV}$ generator. In contrast to the Baseline method, which requires flipping the bits in random positions of *Level* $\mathcal{HV}$s at each iteration, the proposed method is free from randomness. For the Baseline approach, we generate *Level* $\mathcal{HV}$s by flipping $\frac{D}{M} = \frac{1024}{256}$ number of bits at each iteration starting with the $\mathcal{HV}$ of full zeros (M is pixel intensity range or maximum value). Table 2 reports the corresponding hardware costs. As can be seen, the Baseline HDC with random bit-flipping consumes significantly higher area and power. More importantly, our proposed unary *Level* $\mathcal{HV}$ generator outperforms the Baseline design in terms of area-delay product.

4.2 Medical MNIST Performance

We evaluated the performance of the proposed $\mathcal{HV}$ generator on various datasets of medical MNIST (medMNIST) [33], including DermaMNIST, BloodMNIST, RetinaMNIST, and BreastMNIST. The primary goal of this analysis is to see how hardware simplification in our proposal impacts the accuracy of classification tasks, particularly those involving challenging biomedical datasets.

The medMNIST contains diverse medical datasets. We selected specific sub-datasets based on varying numbers of classes. Specifically, DermaMNIST comprises seven classes, BloodMNIST eight, RetinaMNIST five, and BreastMNIST two classes. Figure 4 assesses the performance of our proposal (which employs VDC-2-based single-source random *Position* $\mathcal{HV}$s and unary *Level* $\mathcal{HV}$s) and the Baseline design (with pseudo-random sources for *Position* and *Level* $\mathcal{HV}$s) across all datasets. We incorporate epoch-based training options, given the increased complexity of these datasets compared to conventional handwritten digit classification tasks [8].

Throughout each epoch, we process the entire training dataset and evaluate the accuracy of the validation set. We monitor training accuracy and perform bias-variance checks to ensure generalization and avoid overfitting. The best-performing model from the validation is tested based on heatmap confusion matrix metrics, including sensitivity, precision, specificity, F1-measure, balanced accuracy, and Fowlkes–Mallows Index (FMI). The equations of all these metrics are given in Table 3 (TP: True Positives, TN: True Negatives, FP: False Positives, and FN: False Negatives).

Table 3: Performance Metric Equations

Sensitivity	$\frac{TP}{(TP+FN)}$	**F1-Measure**	$\frac{2\times TP}{2\times TP+FP+FN}$
Precision	$\frac{TP}{TP+FP}$	**Balanced Acc.**	$\frac{Sensit.+Specif.}{2}$
Specificity	$\frac{TN}{(FP+TN)}$	**FMI**	$\sqrt{(Prec.\times Sensit.)}$

Examining the results, our method consistently outperforms in accuracy ($Acc : \frac{TP+TN}{TP+TN+FP+FN}$) as depicted in Figures 4 (a), (c), (e), and (g). When assessing the validation accuracy performance for the initial 30 epochs, the gradual ascent indicates faster improvement with our method compared to the Baseline design. Furthermore, the Baseline design needs to undergo more than one iteration. Hence, for the Baseline design with random $\mathcal{HV}$ generators, we present the best result among 10 trials. We adopted a learning rate (η)-based model update for incremental learning. During each sampling process, the model undergoes validation accuracy evaluation, considering the impact of the new training sample $\mathcal{HV}$ (h). If validation improves based on the new contribution to the class $\mathcal{HV}$ (C), then the training sample's effect on the learning model is incorporated, contributing to the class $\mathcal{HV}$ via accumulation (as illustrated in Figure 2 (c)). The formula for updating class $\mathcal{HV}$ in the event of validation accuracy improvement is $C_{new} = C_{old} + (\eta \times h)$ (otherwise, it is $C_{new} = C_{old} - (\eta \times h)$ [25]). Our experiments achieved optimal results around η=0.1 and η=0.2. We reported the results based on η=0.1.

Next, we conducted a more comprehensive model evaluation for each dataset using confusion matrices. Heatmap plots in Figure 4 visualize the improvement of each metric from 0 to 1. Generally, for each metric, our design consistently outperforms the Baseline design (Figures 4 (b), (d), (f), and (h)) when considering the performance of individual class labels. For example, for DermaMNIST, the sensitivity (a metric indicating the correctly predicted positive values) never drops to 0.1 with our approach, whereas the Baseline design reaches that level for some classes. The highest scores achieved in both architectures are for specificity (representing the proportion of correctly predicted negative cases). For precision (true positive accuracy, reflecting confidence score), our design outperforms for nearly every per-class label across datasets. Additionally, the proposed architecture exhibits superior performance for F1-measure, which considers both precision and recall. The second-best metric for both hardware architectures is balanced accuracy, offering a more insightful perspective for performance analysis considering imbalanced confusion matrices (i.e., unevenly distributed class labels). Lastly, FMI, a similarity calculation metric, consistently yields better scores with our HDC architecture, indicating higher *predicted-actual* class similarities. Thus, the new hardware design with VDC-2 sequences and unary processing facilitates end-to-end processing with a lightweight design and better

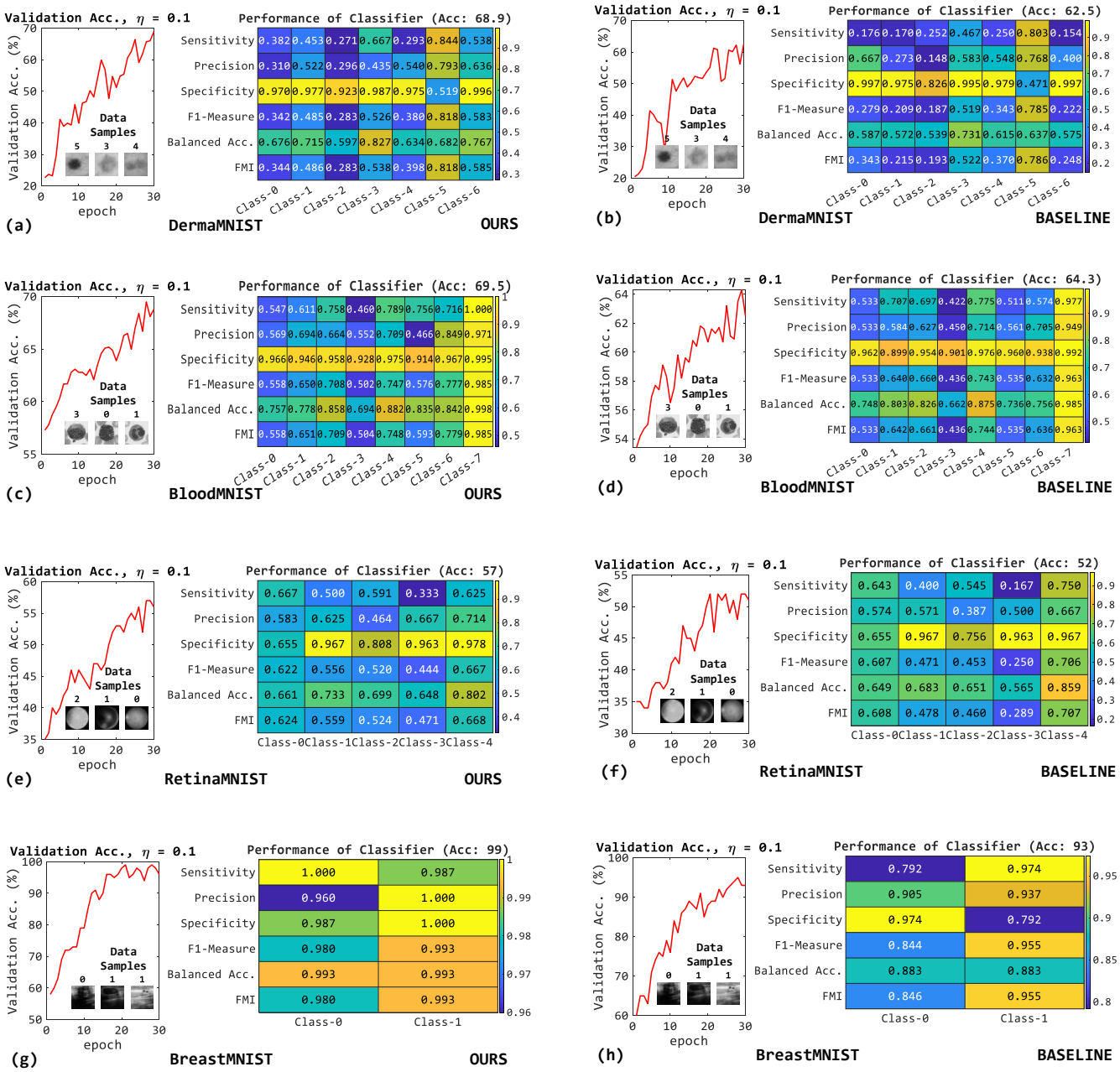

Figure 4: Performance evaluation of the proposed architecture on Medical MNIST datasets [33]. (a) Our approach in DermaMNIST, (b) Baseline HDC in DermaMNIST, (c) Our approach in BloodMNIST, (d) Baseline HDC in BloodMNIST, (e) Our approach in RetinaMNIST, (f) Baseline HDC in RetinaMNIST, (g) Our approach in BreastMNIST, and (h) Baseline HDC in BreastMNIST. D=1024 in all experiments.

ML performance, even for challenging medical datasets across various performance metrics.

5 CONCLUSION

Hypervector ($\mathcal{HV}$) generation is a crucial step in Hyperdimensional Computing (HDC) in terms of accuracy and hardware efficiency. The record-based encoding of HDC necessitates incorporating orthogonal $\mathcal{HV}$s for the *Positional* data types and utilizing correlated neighbor $\mathcal{HV}$s for the *Level* $\mathcal{HV}$s. The state-of-the-art (SOTA) methods employ pseudo-randomness for generating orthogonal *Positional* $\mathcal{HV}$s. The intrinsic nature of pseudo-randomness leads to the deterioration of the overall model performance and throughput of the HDC system. In this work, we apply ① Van der Corput (VDC) quasi-random sequence for generating *Position* $\mathcal{HV}$s and ② unary-based *Level* $\mathcal{HV}$ for the first time in the HDC literature. While the SOTA methods utilize distinct random sources

for generating *Position* $\mathcal{HV}$s, our proposal utilizes a single cost-efficient sequence generator. We avoid using a random bit-flipping scheme by employing unary-based *Level* $\mathcal{HV}$ generation. This makes the hardware implementation more convenient and efficient. Our evaluation results demonstrate significant improvements of 6.4%, 98%, and 15% in classification accuracy, power consumption, and energy efficiency, respectively.

6 ACKNOWLEDGMENTS

This work was supported in part by the National Science Foundation (NSF) under Grants 2339701, 2019511 and in part by generous gifts from Xilinx and Nvidia.

REFERENCES

[1] Hussam Amrouch, Mohsen Imani, Xun Jiao, Yiannis Aloimonos, Cornelia Fermuller, Dehao Yuan, Dongning Ma, Hamza E. Barkam, Paul R. Genssler, and Peter Sutor. 2022. Brain-Inspired Hyperdimensional Computing for Ultra-Efficient Edge AI. In *2022 CODES+ISSS*. https://doi.org/10.1109/CODES-ISSS55005.2022.00017

[2] Fatemeh Asgarinejad, Xiaofan Yu, Danlin Jiang, Justin Morris, Tajana Rosing, and Baris Aksanli. 2024. Enhanced Noise-Resilient Pressure Mat System Based on Hyperdimensional Computing. *Sensors* 24, 3 (2024). https://doi.org/10.3390/s24031014

[3] Alaaddin Goktug Ayar, Sercan Aygun, M. Hassan Najafi, and Martin Margala. 2024. Word2HyperVec: From Word Embeddings to Hypervectors for Hyperdimensional Computing. In *Proceedings of the Great Lakes Symposium on VLSI 2024 (GLSVLSI '24)*. Association for Computing Machinery, New York, NY, USA, 355–356. https://doi.org/10.1145/3649476.3658795

[4] Sercan Aygun, Mehran Shoushtari Moghadam, and M. Hassan Najafi. 2024. uHD: Unary Processing for Lightweight and Dynamic Hyperdimensional Computing. In *2024 Design, Automation & Test in Europe Conference & Exhibition (DATE)*. 1–6.

[5] Sercan Aygun, Mehran Shoushtari Moghadam, M. Hassan Najafi, and Mohsen Imani. 2023. Learning from Hypervectors: A Survey on Hypervector Encoding. arXiv:2308.00685 [cs.LG]

[6] Toygun Basaklar, Yigit Tuncel, Shruti Yadav Narayana, Suat Gumussoy, and Umit Y. Ogras. 2021. Hypervector Design for Efficient Hyperdimensional Computing on Edge Devices. arXiv:arXiv:2103.06709

[7] Sohum Datta, Ryan A. G. Antonio, Aldrin R. S. Ison, and Jan M. Rabaey. 2019. A Programmable Hyper-Dimensional Processor Architecture for Human-Centric IoT. *IEEE Journal on Emerging and Selected Topics in Circuits and Systems* 9, 3 (2019), 439–452. https://doi.org/10.1109/JETCAS.2019.2935464

[8] Li Deng. 2012. The mnist database of handwritten digit images for machine learning research. *IEEE Signal Processing Magazine* 29, 6 (2012), 141–142.

[9] Lulu Ge and Keshab K. Parhi. 2020. Classification using hyperdimensional computing: A review. *IEEE Circ. and Syst. Mag.* 20, 2 (2020), 30–47. https://doi.org/10.1109/mcas.2020.2988388

[10] Yilun Hao, Saransh Gupta, Justin Morris, Behnam Khaleghi, Baris Aksanli, and Tajana Rosing. 2021. Stochastic-HD: Leveraging stochastic computing on hyperdimensional computing. In *2021 IEEE ICCD*. 321–325. https://doi.org/10.1109/ICCD53106.2021.00058

[11] Arman Kazemi, Mohammad Mehdi Sharifi, Zhuowen Zou, Michael Niemier, X. Sharon Hu, and Mohsen Imani. 2021. MIMHD: Accurate and Efficient Hyperdimensional Inference Using Multi-Bit In-Memory Computing. In *2021 IEEE/ACM International Symposium on Low Power Electronics and Design (ISLPED)*. 1–6. https://doi.org/10.1109/ISLPED52811.2021.9502498

[12] Jiseung Kim, Hyunsei Lee, Mohsen Imani, and Yeseong Kim. 2024. Advancing Hyperdimensional Computing Based on Trainable Encoding and Adaptive Training for Efficient and Accurate Learning. *ACM Trans. Des. Autom. Electron. Syst.* (Jun 2024). https://doi.org/10.1145/3665891

[13] Kei Kitagawa, Kohei Tsuji, Koyo Sagehashi, Tomoaki Niiyama, and Satoshi Sunada. 2024. Optical hyperdimensional soft sensing: speckle-based touch interface and tactile sensor. *Opt. Express* 32, 3 (Jan 2024), 3209–3220. https://doi.org/10.1364/OE.513802

[14] Denis Kleyko, Dmitri A. Rachkovskij, Evgeny Osipov, and Abbas Rahimi. 2022. A Survey on Hyperdimensional Computing Aka Vector Symbolic Architectures, Part I: Models and Data Transformations. *ACM Comput. Surv.* 55, 6, Article 130 (dec 2022), 40 pages. https://doi.org/10.1145/3538531

[15] Dehua Liang, Jun Shiomi, Noriyuki Miura, and Hiromitsu Awano. 2024. StrideHD: A Binary Hyperdimensional Computing System Utilizing Window Striding for Image Classification. *IEEE Open Journal of Circuits and Systems* 5 (2024), 211–223. https://doi.org/10.1109/OJCAS.2024.3401028

[16] S. Liu and J. Han. 2018. Toward energy-efficient stochastic circuits using parallel sobol sequences. *IEEE TVLSI* 26, 7 (2018).

[17] Dongning Ma, Cong Hao, and Xun Jiao. 2024. Hyperdimensional computing vs. neural networks: Comparing architecture and learning process. In *2024 25th International Symposium on Quality Electronic Design (ISQED)*. IEEE, 1–5.

[18] Alisha Menon, Anirudh Natarajan, Laura I. Galindez Olascoaga, Youbin Kim, Braeden Benedict, and Jan M. Rabaey. 2022. On the Role of Hyperdimensional Computing for Behavioral Prioritization in Reactive Robot Navigation Tasks. In *2022 International Conference on Robotics and Automation (ICRA)*. 7335–7341. https://doi.org/10.1109/ICRA46639.2022.9811939

[19] Alisha Menon, Daniel Sun, Melvin Aristio, Harrison Liew, Kyoungtae Lee, and Jan M. Rabaey. 2021. A Highly Energy-Efficient Hyperdimensional Computing Processor for Wearable Multi-Modal Classification. In *BioCAS*. 1–4. https://doi.org/10.1109/BioCAS49922.2021.9645008

[20] Mehran Shoushtari Moghadam, Sercan Aygun, and M. Hassan Najafi. 2023. No-Multiplication Deterministic Hyperdimensional Encoding for Resource-Constrained Devices. *IEEE ESL* (2023). https://doi.org/10.1109/LES.2023.3298732

[21] Mehran Shoushtari Moghadam, Sercan Aygun, Mohsen Riahi Alam, and M. Hassan Najafi. 2024. P2LSG: Powers-of-2 Low-Discrepancy Sequence Generator for Stochastic Computing. (2024). In ASP-DAC 2024.

[22] Justin Morris, Yilun Hao, Saransh Gupta, Behnam Khaleghi, Baris Aksanli, and Tajana Rosing. 2022. Stochastic-HD: Leveraging stochastic computing on the hyper-dimensional computing pipeline. *Front. in Neurosc.* 16 (2022).

[23] M. Hassan Najafi, David. J. Lilja, Marc D. Riedel, and Kia Bazargan. 2018. Low-Cost Sorting Network Circuits Using Unary Processing. *IEEE Transactions on Very Large Scale Integration (VLSI) Systems* 26, 8 (2018), 1471–1480. https://doi.org/10.1109/TVLSI.2018.2822300

[24] M. Hassan Najafi, David. J. Lilja, Marc D. Riedel, and Kia Bazargan. 2018. Low-Cost Sorting Network Circuits Using Unary Processing. *IEEE Transactions on Very Large Scale Integration (VLSI) Systems* 26, 8 (2018), 1471–1480. https://doi.org/10.1109/TVLSI.2018.2822300

[25] Yujin Nam, Minxuan Zhou, Saransh Gupta, Gabrielle De Micheli, Rosario Cammarota, Chris Wilkerson, Daniele Micciancio, and Tajana Rosing. 2023. Efficient Machine Learning on Encrypted Data Using Hyperdimensional Computing. In *2023 IEEE/ACM International Symposium on Low Power Electronics and Design (ISLPED)*. 1–6. https://doi.org/10.1109/ISLPED58423.2023.10244262

[26] Prathyush Poduval, Zhuowen Zou, Hassan Najafi, Houman Homayoun, and Mohsen Imani. 2021. StocHD: Stochastic hyperdimensional system for efficient and robust learning from raw data. In *2021 DAC*. 1195–1200. https://doi.org/10.1109/DAC18074.2021.9586166

[27] Netanel Raviv. 2024. Linear Codes for Hyperdimensional Computing. arXiv:2403.03278 [cs.IT]

[28] Mehran Shoushtari Moghadam, Sercan Aygun, Mohsen Riahi Alam, Jonas I Schmidt, M. Hassan Najafi, and Nima Taherinejad. 2024. Accurate and Energy-Efficient Stochastic Computing with Van Der Corput Sequences. In *Proceedings of the 18th ACM International Symposium on Nanoscale Architectures (NANOARCH '23)*. Association for Computing Machinery, New York, NY, USA, Article 27, 6 pages. https://doi.org/10.1145/3611315.3633265

[29] Ruixuan Wang, Sabrina Hassan Moon, Xiaobo Sharon Hu, Xun Jiao, and Dayane Reis. 2024. A Computing-in-Memory-Based One-Class Hyperdimensional Computing Model for Outlier Detection. *IEEE Trans. Comput.* 73, 6 (2024), 1559–1574. https://doi.org/10.1109/TC.2024.3371782

[30] You Wang, Yefan Xu, Yu Gong, Ke Chen, and Weiqiang Liu. 2023. STT-MRAM Based Highly Orthogonal Hypervector Generator for Hyperdimensional Computing. In *2023 IEEE 23rd International Conference on Nanotechnology (NANO)*. 666–670. https://doi.org/10.1109/NANO58406.2023.10231281

[31] Di Wu, Jingjie Li, Zhewen Pan, Younghyun Kim, and Joshua San Miguel. 2022. UBrain: A Unary Brain Computer Interface. In *Proceedings of the 49th Annual International Symposium on Computer Architecture* (New York, New York) *(ISCA '22)*. Association for Computing Machinery, New York, NY, USA, 468–481. https://doi.org/10.1145/3470496.3527401

[32] Di Wu, Jingjie Li, Ruokai Yin, Hsuan Hsiao, Younghyun Kim, and Joshua San Miguel. 2020. UGEMM: Unary Computing Architecture for GEMM Applications. In *2020 ACM/IEEE 47th Annual International Symposium on Computer Architecture (ISCA)*. 377–390. https://doi.org/10.1109/ISCA45697.2020.00040

[33] Jiancheng Yang, Rui Shi, Donglai Wei, Zequan Liu, Lin Zhao, Bilian Ke, Hanspeter Pfister, and Bingbing Ni. 2023. MedMNIST v2 - A large-scale lightweight benchmark for 2D and 3D biomedical image classification. *Scientific Data* 10, 1 (19 Jan 2023), 41. https://doi.org/10.1038/s41597-022-01721-8

[34] Tianyang Yu, Bi Wu, Ke Chen, Gong Zhang, and Weiqiang Liu. 2024. Fully Learnable Hyperdimensional Computing Framework With Ultratiny Accelerator for Edge-Side Applications. *IEEE Trans. Comput.* 73, 2 (2024), 574–585. https://doi.org/10.1109/TC.2023.3337316

[35] Zhuowen Zou, Haleh Alimohamadi, Ali Zakeri, Farhad Imani, Yeseong Kim, M. Hassan Najafi, and Mohsen Imani. 2022. Memory-inspired spiking hyperdimensional network for robust online learning. *Scientific Reports* 12, 1 (10 May 2022), 7641. https://doi.org/10.1038/s41598-022-11073-3