Robust Hyperdimensional Computing Against Cyber Attacks and Hardware Errors: A Survey

Dongning Ma
dma2@villanova.edu
Villanova University
Villanova, PA, USA

Sizhe Zhang
szhang6@villanova.edu
Villanova University
Villanova, PA, USA

Xun Jiao
xjiao@villanova.edu
Villanova University
Villanova, PA, USA

ABSTRACT

Hyperdimensional Computing (HDC), also known as Vector Symbolic Architecture (VSA), is an emerging AI algorithm inspired by the way the human brain functions. Compared with deep neural networks (DNNs), HDC possesses several advantages such as smaller model size, less computation cost, and one/few-shot learning, making it a promising alternative computing paradigm. With the increasing deployment of AI in safety-critical systems such as healthcare and robotics, it is not only important to strive for high accuracy, but also to ensure its robustness under even highly uncertain and adversarial environments. However, recent studies show that HDC, just like DNNs, is vulnerable to both cyber attacks (e.g., adversarial attacks) and hardware errors (e.g., memory failures). While a growing body of research has been studying the robustness of HDC, there is a lack of systematic review of research efforts on this increasingly-important topic. To the best of our knowledge, this paper presents the first survey dedicated to review the research efforts made to the robustness of HDC against cyber attacks and hardware errors. While the performance and accuracy of HDC as an AI method still expects future theoretical advancement, this survey paper aims to shed light and call for community efforts on robustness research of HDC.

ACM Reference Format:
Dongning Ma, Sizhe Zhang, and Xun Jiao. 2023. Robust Hyperdimensional Computing Against Cyber Attacks and Hardware Errors: A Survey. In *28th Asia and South Pacific Design Automation Conference (ASPDAC '23), January 16–19, 2023, Tokyo, Japan.* ACM, New York, NY, USA, 8 pages. https://doi.org/10.1145/3566097.3568355

1 INTRODUCTION

Hyperdimensional computing (HDC), also known as Vector Symbolic Architecture (VSA), was first proposed as a computation scheme more than a decade ago [14]. For learning and recognition tasks, HDC is often referred to as a cross-over originating from the intersection between symbolic and connectionist artificial intelligence [18]. HDC is inspired by human brain recognition mechanisms and leverages the abstract patterns formed by high-dimensional vectors. Recently, HDC has shown promising results in performance and efficiency in various application domains, including natural language processing (NLP) [20, 30], robotics [26], anomaly detection [34, 35], and bio-informatics [3, 17, 22].

Despite the advantages, HDC also faces increasing robustness challenges from uncertain and adversarial environments, including cyber-attacks and hardware errors. For example, by perturbing just a few pixels in the input image, HDC can be fooled and misclassified digit even if the perturbation is almost imperceptible to humans [21]. On the other hand, HDC parameters can be reverse-engineered to perform IP stealing attacks [6]. These security and privacy loopholes urge a systematic analysis of the impact of different cyber attacks as well as the defense mechanisms to enhance the robustness against such attacks.

On the other hand, the versatility of HDC has led to its hardware implementation in different platforms such as FPGA, ASIC, and emerging memory. However, with increasing technology scaling, hardware becomes more prone to errors caused by factors such as microelectronic variations, soft errors, and inherent non-idealities [37, 42]. The hardware errors typically manifest as incorrect computation results at the application level, which can cause an accuracy drop of HDC models. Recent research efforts aim to characterize the impact of hardware errors on HDC accuracy, as well as enhancing HDC robustness to such errors.

In this paper, we present, to the best of our knowledge, the first survey of techniques for characterizing and improving the robustness of HDC, particularly against cyber attacks and hardware errors. For cyber attacks, we review diverse attack schemes, including adversarial attacks, data poisoning attacks, privacy attacks, and model stealing attacks. As to hardware errors, we review research efforts on characterizing and enhancing the robustness of HDC against hardware errors. The objective of this survey is to shed light and call for community efforts on this important research direction for the emerging HDC paradigm.

2 BACKGROUND

2.1 HDC Preliminaries

As the brain's neural circuits have a massive number of neurons and synapses, it is suggested that large circuits are fundamental and essential for neural processing. Based on the understanding that the brain computes with patterns of neural activity that are not readily associated with numbers, HDC was founded on the mathematical properties of high-dimensional spaces, which show remarkable agreement with behaviors controlled by the brain [14, 32]. Instead of computing with numbers, HDC computes with high-dimensional vectors (e.g., 10,000 Dimension), which are referred to as hypervectors (HV): $\vec{H} = \langle h_1, h_2, \ldots, h_d \rangle$.

ASPDAC '23, January 16–19, 2023, Tokyo, Japan
© 2023 Copyright held by the owner/author(s).
ACM ISBN 978-1-4503-9783-4/23/01.
https://doi.org/10.1145/3566097.3568355

In HDC, HVs are the fundamental blocks to represent "symbols" reflecting real-world features such as pixel values, signal levels, or language characters in the high-dimensional space. HVs are holographic and (pseudo)random with independent and identically distributed (i.i.d.) components. That is, every piece of information contained in the HV is equally distributed over all its elements in a full holistic representation of the vector so that no element inside the HV is more responsible for storing one piece of information over another. The formation of HVs naturally pertains to arithmetic operations such as (element-wise) addition, (element-wise) multiplication, and permutation (vector rotation), which can form an algebra over the vector space.

2.2 HDC Model Development

HDC model development is generally composed of three phases: training, retraining, and inference. All three phases feature a fundamental process of encoding.

Encoding uses item memory to project real-world feature vectors into their high-dimensional space representations using a set of HD operations Γ. Item memory $\mathbf{R}$ consists of k seed HVs, each representing a possible value of the discrete features. Let $\vec{F}$ be the real-word feature vector of a certain sample, the process of encoding $\vec{F}$ into its representing sample HV $\vec{S}$ can be noted as $\vec{S} = \Gamma(\mathbf{R}(\vec{F})) = \Gamma(\mathbf{R}(\{f_1, f_2, \ldots, f_m\}))$. HDC uses the value of discrete features to index the corresponding seed HVs and then applies HD operations to aggregate the seed HVs into one sample HV $\vec{S}$.

Training is the process of establishing the associative memory using the training set. Associative memory $\mathbf{A}$ is the trainable part of an HDC model. It consists of l class HVs, each representing a class in a classification task. For each training sample, HDC adds its sample HV to the corresponding class HV, i.e., $\vec{A_l} = \sum \vec{S_l}$. This process is to aggregate the information from sample HVs together into the AM.

Inference is the process of using unseen data from the inference set to evaluate the trained model's performance. First, the unseen sample with an unknown class is encoded into its sample HV, specifically referred to as query HV $\vec{Q_?}$. The HDC model then calculates the distance (δ) between this query HV and each class HV in the associative memory to obtain the distance metrics. The class with the smallest distance $p = argmin(\delta(\vec{Q_?}, \mathbf{A}))$ marks the prediction of the unseen sample.

3 ROBUSTNESS AGAINST CYBER ATTACKS

In this section, we discuss the robustness of HDC against cyber attacks, including adversarial attacks, data poisoning, privacy attack, and IP stealing.

3.1 Adversarial Attacks

In adversarial attacks, attackers attempt to generate adversarial samples with minimal perturbation to induce erroneous behaviors of the model. Adversarial attacks can be carried out under different attack assumptions: black-box, gray-box, and white-box, and target different applications from image classification to voice recognition.

3.1.1 Threat model.

We present the threat model for HDC adversarial attacks in Fig. 1. Using MNIST handwritten digit classification task as an example, assume we have an HDC model $G(x, \theta_g)$, for general users with benign input t to the model, the output label will be $y = G(t, \theta_g)$. However, for malicious users, the objective is to apply perturbations to generate adversarial inputs t' so that the model prediction $y' = G(t', \theta_g)$ is different from the benign input, i.e., $y != y'$. Meanwhile, to ensure that the differences between the benign image and the generated perturbed image are not noticeable, the attacker needs to minimize the perturbation, i.e., $min\ ||t - t'||$.

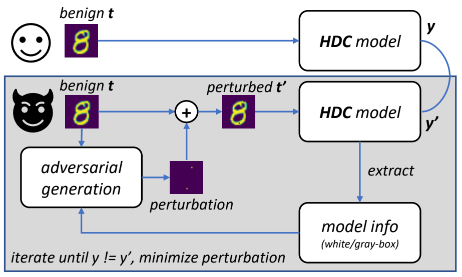

Figure 1: The threat model of adversarial attack on HDC.

Based on how the HDC model is accessible to the attacker or how much information the attacker can extract, we can classify the adversarial attacks into several scenarios: white-box, gray-box and black-box. In white-box scenario, the attacker has full access into the victim HDC model and is able to extract any information during the inference. However, such scenario may be less frequent in practical implementations compared with gray-box and black-box scenarios. More realistic scenarios are gray-box and black-box scenarios. Under gray-box scenario, attacker does not have full access to the HDC model such as the encoding scheme and item memories but can obtain some information such as the soft-labels and/or the associative memories. Under black-box scenario, the attacker is only able to input samples and observe the output labels of the model and no internal information of the mode can be obtained.

3.1.2 Robustness Study under Gray-box Scenarios.

Yang and Ren [39] explore the vulnerabilities of HDC model of attacks from perturbed images. They assume a gray-box scenario where attacker can send input queries to the model and obtain the Hamming distance between the image's query hypervector and each class hypervector as the similarity metrics. Inspired by the genetic algorithm, they propose **GA-CGC-PA** (Genetic Algorithm with Critical Gene Crossover and Perturbation Adjustment), which is modified from genetic algorithm with the objective of reducing the amount of perturbations. The two major contributions of **GA-CGC-PA** are the **critical gene cross-over** and the **perturbation adjustment**. Specifically, compared with standard cross-over algorithms that modify every pixel in the input images, the critical gene cross-over proposed can selectively cross the most important genes (pixels) of the parent images. For perturbation adjustment, each perturbed pixel is iterated to check if there is further space possible to reduce perturbation. The pixel is restored to its original value first then gradually changed towards the value during adversarial

Table 1: An overview of surveyed literature related to HDC robustness

Cyber Attacks	Adversarial Attacks	[4, 21, 23, 31, 39]
	Data Poisoning Attacks	[33]
	Privacy Attacks	[9, 16]
	IP Stealing Attacks	[6]
Hardware Errors	Hardware Error Models	[19, 24, 27, 37, 38, 40–42]
	Impact of Hardware Errors	[11, 13, 19, 24, 27, 37, 38, 42]
	Enhancement of HDC Robustness	[40, 41]

generation. For each pixel, the process can stop when the image can successfully mislead the HDC model.

The experiments are based on HDC models with dimensionality of 10000 under MNIST handwritten digits dataset. Two schemes of dealing with HV elements that summed up to 0 during encoding: random majority rule (RMR) which randomly assigns the value to -1 or 1, and fixed majority rule (FMR) which deterministically assigns either -1 or 1 per configuration. The performance is characterized by an amount of perturbation and the query count, which refers to the number of queries to generate the adversarial images. It is observed that different digits (classes) may have different sensitivities against perturbation, for example, digit "0" and "8" have the lowest accuracy, which means they are easier to attack, while digit "2" and "4" have the least number of query count under RMR. For FMR, the attack successful rate is 0.78 however, the median number of modified pixels is around 100.

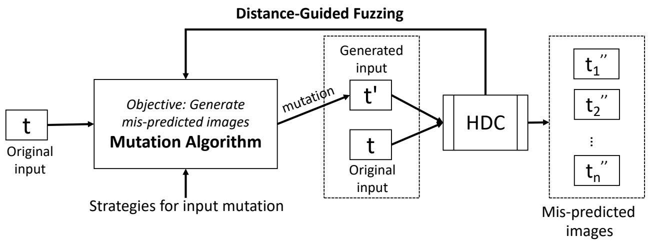

Figure 2: Overview of HDTest [21].

HDTest [21] employs fuzz testing, a classical method from the software testing community to automatically generate adversarial samples. Fuzz testing mutates inputs with the objective of generating faults or exceptions [7]. One major advantage of **HDTest** is that it does not require labeling of the data but relies on differential testing to automatically recognize the adversarial behaviors of the model under test, hence reducing the manual efforts of labeling and comparing the output. An overview of **HDTest** is illustrated in Fig. 2. **HDTest** first uses the original input image t and applies different types of mutations without necessarily knowing the label to generate new input t'. To observe the differential behaviors, both of the two inputs are sent to the HDC classifier for the test. If there is a discrepancy between the label of the original input t and the generated input t', **HDTest** can declare a successful generation of an adversarial input. The generation and differential testing can repeat for multiple times until success or reach the maximum allowed iterations as regulated by the user.

HDTest leverages different mutation algorithms, including pixel-wise or row and column-wise Gaussian or random mutation, as well as vertical or horizontal shift on the original image. Both L1 and L2 distances are used as the metrics to analyze the difference between the mutated and original images. Experimental results report that row and column-wise mutation strategies are inferior to pixel-wise Gaussian and random mutations. On the other hand, **HDTest** also reveals that distance metrics such as L1 and L2 are not suitable for all the mutation strategies. For example, although the "shift" mutation strategy has a very high distance between the mutated and original image, semantically the image is not "perturbed" but shifted in the positions. Therefore, to evaluate more mutation strategies other than pixel-wise perturbations, more pertinent metrics are necessary. To further elucidate that **HDTest** can enhance the robustness against adversarial attacks, an adversarial retraining is also evaluated. **HDTest** uses (part of) the generated adversarial samples (which has a 100% attack successful rate on the victim model) to retrain the victim model. Then **HDTest** uses the rest of the generated adversarial samples as unknown adversarial samples to attack the victim model again and observe the new successful rate. Experimental results show that using such an adversarial retraining scheme, **HDTest** is able to reduce the attack successful rate by more than 20%.

In addition to image classification applications, adversarial attacks have also been investigated on audio applications. Chen and Li propose a semi-black box attack on HDC models for voice recognition [4]. They attack the **VoiceHD** model, which is developed to classify voices pronouncing the English alphabet letters [12]. A differential evolution (DE) algorithm is proposed to attack on the amplitude of bins in the data and optimize the positions of the bins. DE algorithm can work under both non-targeted attack mode and targeted attack mode where the difference is on the objective function defined.

DE algorithm has four phases: initialization, mutation, crossover, and selection. Initialization can feature random population (vector) generation, and the mutation randomly chooses two vectors in the population to generate the third mutation vector based on the difference between them. During crossover, new vectors are generated by combining mutation vectors with predetermined vectors in some possibility distribution. During selection, the population that can yield a lower objective function is selected. Two measures are proposed to minimize the changes to the original data: 1). to attack as few bins as possible and 2). to attack more bins but keep the changes of each bin small. Experimental results show that the proposed DE algorithm can attack **VoiceHD** on the ISOLET dataset [5] with up to 85.7% successful rate under the non-targeted mode. Similar to adversarial attacks on image classification, some classes, such as the letter "w" are higher likely to be attacked than others.

3.1.3 *Robustness Study under Black-box Scenarios.*

Furthermore, under black-box scenarios, the attacker may have no access to the internal components of the deployed HDC model but only observe the output labels of an input. **HDXplore** assumes such a scenario and proposes a black-box attack also based on differential behaviors. Similar to **HDTest**, **HDXplore** incorporates a process of perturbing the benign samples to obtain adversarial samples. Different perturbation algorithms, including skew, noise, brightness, and elastic transform are evaluated. Two retraining schemes are proposed to increase the HDC model robustness: static retraining and dynamic retraining. During static retraining, a fixed set of adversarial images is used to iteratively retrain the HDC model, while during dynamic retraining, new adversarial images are generated and used after each retraining epoch. Experimental results show that for static retraining, the accuracy of HDC model slightly increases from about 81% first but drops below 70% after a certain number of epochs. For dynamic retraining, the accuracy can increase further to up to 90%.

Recently, natural language processing (NLP) applications using HDC are becoming increasingly popular [1, 20, 30]. Therefore, adversarial attacks on HDC models for NLP tasks start to be investigated, e.g., Moraliyage et al., target at the n-gram based HDC model and perform adversarial attacks using widely-used text adversarial frameworks: **DeepWordBug**, **PWWS** and **TextFooler** [23]. Each sample is regarded as a vector of words and the attacker modifies text input with small perturbations to misclassify the inputs. The attack can focus on character-level or word-level substitutions based on attack methods. Experiments are based on the European Language Classification dataset which is to classify languages of text and the Reuters Newsire dataset which is to classify topics of text [25]. Results show that for **PWWS** and **TextFooler**, the adversarial attack can cause about 15.5% and 7% accuracy degradation respectively on HDC models with gram-size of 3 for European Language classification, while for **DeepWordBug**, the accuracy degradation is much smaller. A similar trend can be observed for Reuters Newswire dataset, which indicates that HDC NLP models are more vulnerable to word-level attacks compared with character-level attacks. The paper also compares HDC with other machine learning models such as convolution neural networks and recurrent neural networks. It shows that HDC model is more robust to character-level attacks than neural networks yet less robust against word-level attacks.

3.2 Data Poisoning Attacks

The training of HDC model is also vulnerable to another type of cyber attack: the data poisoning attack. **PoisonHD** first explores such attack on HDC model [33] under white-box scenario. Instead of attacking the inference phase of HDC by perturbing input data, poisoning attack focuses on adding extra and modified samples to the training dataset. The objective is to inject false information in the training dataset so that the trained model will have corrupted performance. An overview of **PoisonHD** can be found in Fig. 3. **PoisonHD** embraces a confidence-based label-flipping method specifically leveraging the HDC models mechanisms for binary classification tasks. In **PoisonHD**, the confidence of each sample is defined as the absolute difference between the cosine similarity between the query HV and the positive class HV and the cosine similarity between the query HV and the negative class HV. **PoisonHD** ranks all the encoded training data by confidence in a descending order and then flips the labels of the top-N highest confidence samples to maximize the influence of the attack to the model. **PoisonHD** also provides the data sanitizing against such type of label-flipping data poisoning attack, which is also known as "oracle defense". Such defense relies on an extra verified dataset which is not available to the attacker. The verified dataset is used to build a new exclusive associative memory to verify the encoded HVs of samples of the original training dataset to identify if the label is flipped. The experiments are based on three binary classification datasets: Dog-Fish, MNIST 1-7 and Breast Cancer. Results show that flipping 15% of the data can cause up to 30% accuracy drop while using the oracle defense can reduce the impact of data poisoning that the accuracy drop is mitigated to less than 3%.

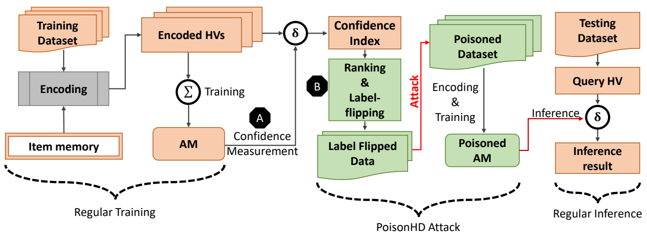

Figure 3: Overview of PoisonHD [33]

3.3 Privacy Attacks

As HDC is increasingly applied to applications with sensitive input data, it is also of great importance to examine how privacy attacks effective on other machine learning algorithms can transfer to HDC models. **PriveHD** targets at privacy breach issue that by observing the encoded HVs, the attacker can disclose information of the input data [16]. Specifically for HDC, because the dimension of HVs is usually very large (10000 and above), the base vectors are almost orthogonal. Therefore, each feature incorporated in the encoding can be retrieved through trials by multiplying the encoded HV and the base vectors. To defend such breach, **PriveHD** proposes a differentially private training scheme for HDC via dimensionality pruning and quantization. The dimensionality pruning is intuitive as when dimensions are reduced, the orthogonality between base vectors is impaired, subsequently lowering the effectiveness of the feature retrieval. To maximize the effect of pruning while minimizing the impact on classification accuracy, **PriveHD** identifies the importance of each dimension in the HV based on the proximity of the value to 0. Quantization can also enhance the differential privacy meanwhile causing huge performance degradation. To mitigate such impact, **PriveHD** chooses to only quantize the encoding HVs, which shows around 5% less accuracy drop compared with previous quantization works.

Prid also investigates the privacy attack that can reconstruct inputs based on encoded HVs [9]. Instead of using an analytical-based method like **PriveHD**, **Prid** uses a learning-based method that formulates the reconstruction as a linear regression problem. **Prid** configures a neural network to perform this task with the input being the base HVs and the output being the encoded HV. Two methods are proposed: feature replacement and dimension replacement to add masked information to ensure the reconstruction is a good estimation of train data that has been used for model generation. Two schemes are also proposed to enhance robustness against such privacy attacks: iterative intelligent noise injecting and iterative model quantization. Unimportant dimensions are identified and injected with noise to reduce the sensitivity of the information while at the same time keeping the impact on model performance minimal. Quantization can also reduce the risk of reverting the encoded HV to its original space, as already illustrated in **PriveHD**. Retraining is further applied on both robustness enhancement schemes to regain accuracy. Results show that **Prid** can reduce the information leakage to up to 81% while causing only around 2.1% quality loss on classification accuracy under 2000 dimensionality.

3.4 IP Stealing Attacks

Training of machine learning models is not an easy task as it requires enormous engineering and computational effort. Similar to the data used in training and inference, the trained model itself is also a valuable intellectual property, particularly when the model is trained using a proprietary dataset. Due to the straightforward implementation, HDC models are more vulnerable to model stealing attacks that the entire HDC model can be reverse-engineered even under a gray-box scenario where the attacker only has limited access to the model [6]. **HDLock** carries out the model stealing attack analytically by divide-and-conquer the value and feature HVs due to the orthogonality in the item memories similarly as introduced in [16]. To defend such attack, HDLock modifies the encoding module by adding a permutation to the base vectors when encoding. Since the permutation values require much less memory space and can be stored in the secure memory, the encoding is now more robust against dive-and-conquer attacks. Experimental results with HDC models implemented on a Xilinx Zynq UltraScale+ FPGA show that the complexity for stealing attack grows by 10 orders of magnitude with only 21% of overhead in encoding.

4 ROBUSTNESS AGAINST HARDWARE ERRORS

In this section, we discuss the robustness of HDC against hardware errors. We first present the hardware error models used in recent studies on various hardware platforms, and then we discuss the impact of such errors on the performance/accuracy of HDC models. Finally, we review the techniques used to enhance the robustness of HDC against hardware errors.

4.1 Hardware Error Models

Errors in hardware can be caused by microelectronic process variations, extreme temperatures, voltages, wear-out effects [28], and radiation [2]. Recently, HDC has been implemented in hardware platforms, including FPGA, GPU, and emerging memory, which are all susceptible to hardware faults/errors. As an example, if the supply voltage of the memory is reduced from its normal value, unstable memory cells cause bit flips. Flipping bits affect the value of the model and influence the accuracy of the system. However, if the model itself is robust to these errors, the memory can run at a lower voltage, thus saving energy while the model still performs as intended. This is referred to as voltage scaling, which is an approach that is widely used for the design of low-power systems. A number of existing HDC studies discuss how voltage scaling can improve memory energy efficiency by taking advantage of HDC hardware robustness. [13, 41]. Therefore, we introduce several common error models in order to evaluate the hardware robustness of HDC.

Random Bit Flip is arguably the most widely-used hardware error model. Under this error model, a randomly-chosen bit in a parameter will be flipped. Bit flips can occur in both computations and memory. As HDC is a memory-centric computing scheme, most studies focus on bit flips occurring in the associative memory, which stores the class HVs [40–42]. During inference, this affects the similarity comparison, which leads to an accuracy drop. An example of bit flips occurring in HVs is illustrated in Fig. 4 illustrates how random bit flip affects HV.

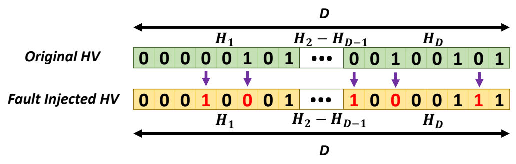

Figure 4: Fault injection (Random Bit-Flip) on Hypervector

Stuck-at-fault is a type of error that occurs in a digital circuit that is generally caused by a manufacturing defect. The assumption is that there is one gate/transistor in the digital circuit that is stuck, so it will always remain in a "0" or "1" state. Several studies explore HDC hardware robustness against stuck-at-fault [19, 27, 37, 38].

Communications noise can happen in wired and wireless communication, caused by factors such as transmission power levels, distance, propagation loss scenarios, and the number of interfering devices. For example, HyDREA [24] uses NS-3 [8]- a network simulator to model the communication noises and explore how such error impact HDC-based federated learning.

4.2 Impact of Hardware Errors on HDC Accuracy

Zhang et al. [42] presents a study on the impact of memory errors (under random bit flip assumption) on HDC model accuracy in three application domains: speech recognition, human activity recognition, and medical diagnosis. The paper examines the HDC robustness under different HDC dimensions and data types (e.g., bit widths). It is shown that HDC models with less bit-width representing parameters generally have higher better robustness to hardware errors. For example, binary HDC models exhibit the highest robustness: the accuracy does not drop until the error rates increase to 10^{-2}. Imani et al. [13] examine how bits errors in associative memory affect Hamming distance calculation. This study shows that HDC is able to achieve 97.8% classification accuracy with up to 1000-bit errors, which is equivalent to a 10% error rate in the 10000-dimension HDC system.

As a memory-centric computing paradigm, HDC is suitable to run on the processing-in-memory (PIM) platform with emerging memory devices. Unfortunately, emerging memory devices have inherent non-idealities that can lead to errors [19, 27, 37, 38]. A CNFETs and RRAM-based PIM system for HDC have been built [19, 27, 37, 38] and evaluated for its robustness against memory errors. After a certain number of write cycles, the nonvolatile memory cells will become stuck at either '1' or '0' [19, 27]. In RRAMs, the number of write cycles before the cell becomes stuck is known as the RRAM endurance constraint. HDC exhibited nearly no accuracy loss when the RRAM endurance constraint was reduced to 1000. An approximate accumulator has been designed to further enhance HDC robustness against stuck-at-fault [37, 38]. Hsu et al. examine the robustness of a NAND flash-based HDC implementation for genome sequencing [11]. NAND Flash is susceptible to current variation noise and current shift. It is shown that HDC maintains acceptable accuracy even when the variation in the current exceeds 0.1 (σ/μ) and the current shift exceeds 0.5 (δ/μ).

HyDREA [24] examines the impact of communication noise on the HDC-based federated learning model. To simulate wireless noise and model the communication between devices, the authors use the popular network simulator NS-3 [8]. A comparison was conducted between **HyDREA** and other lightweight machine learning algorithms such as linear regression, multilayer perception, perception, and support vector classification. **HyDREA** shows that HDC is 48X more robust than certain traditional ML algorithms under a compromised signal-to-noise ratio (SNR).

4.3 Enhancement of HDC Robustness to Hardware Errors

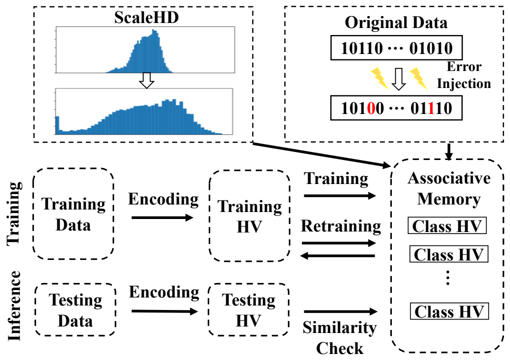

Figure 5: Overview of ScaleHD [40].

While HDC exhibits inherent robustness to hardware errors to a certain degree, an enhancement of robustness is always desirable to tolerate even higher error rates or enable efficient design by exploring the efficiency-robustness trade-off. Zhang et al. enhance the robustness of HDC by applying error masking techniques [41]. They deploy low-cost error masking schemes at two levels, bit-level masking and word-level masking, to detect and mask errors, as shown in Tab. 2. The error masking circuits are implemented with the Razor double sampling-based circuitry with a 0.3% silicon area and 12.8% power [29]. The simulation result shows that both masking methods can effectively enhance the robustness of HDC up to 10,000X. The paper further exploits enhanced robustness to enable energy-efficient design using voltage scaling. On a 22nm SRAM, up to 72.5% energy can be saved with 1% accuracy loss.

While error-masking circuitry can enhance robustness, it incurs extra hardware overhead. A software-only approach, **ScaleHD**, enhance HDC robustness without incurring extra hardware overhead, as shown in Fig. 5 [40]. **ScaleHD** is inspired by a simple observation: a larger value will have less relative error impact compared to a smaller value when the same bit position flips. Thus, **ScaleHD** uses an adaptive numeric scaling technique to enhance HDC robustness by scaling its class HV value. **ScaleHD** explore scaling at three levels: Global-**ScaleHD**, Class-**ScaleHD**, and (Class + Clip)-**ScaleHD**. Global-**ScaleHD** determines the ratio between the extreme value in the class HVs and the maximum value that can be represented by the available bits. Global-**ScaleHD** scale each of the class HV using this ratio. Similarly, Class-**ScaleHD** performs a similar operation but at an individual class level, which grants more space for scaling. (Class + Clip)-**ScaleHD** sacrifices a minor accuracy by clipping the extreme values in the class HVs to further increase the scaling ratio. The experimental results show that **ScaleHD** is capable of enhancing robustness by up to 10000X. Leveraging such robustness enhancement, **ScaleHD** can enable 70% energy saving with less than 1% accuracy loss.

In addition to bit-level enhancement, Hersche et al. present a HDC architecture that can detect, localize, and isolate faults in phase-change memory (PCM), and then replace new memory to recover these faults [10]. In the first step, it analyzes the stand deviation of the Hamming distance between associative memory and encoded samples in order to detect memory faults. In the subsequent step, the memory is divided into partition blocks, and hamming distances are calculated between these partition blocks and encoded testing samples. Consequently, it can locate fault partition blocks by clustering the relative sum of the distances of each block and analyzing the data. In the end, new blocks will be added to replace the fault partitions. By using old non-fault blocks, HDC will be able to train new blocks unsupervised. Using their HDC architecture, accuracy recovers from 16.02% to 95.05% when the fault rate is 48.5%. Furthermore, when the fault rate is 22%, 37%, and 42%, accuracy is able to fully recover to 96.86%.

5 FUTURE DIRECTIONS

In this section, we present potential future directions and opportunities for HDC robustness research.

5.1 Robustness against Cyber Attacks

Most of the existing defense schemes against cyber attacks are based on adversarial retraining. A significant drawback is the notably associated accuracy degradation. A potential future direction of defending against adversarial attacks with less impact is to introduce the defense network, which is used in the neural network domain for adversarial defense [36]. Specifically, a smaller model can be added along with the original model and is dedicated to identifying if the input is adversarial or benign. As there is no retraining or fine-tuning over the HDC model, the negative impact of the defense mechanism is trivial. For data poisoning attacks, more poisoning algorithms are necessary to evaluate, particularly for

Table 2: Recovery Schemes for Protecting HDC Models from Hardware Errors

	Sign-bit Masking [41]		Word Masking [41]		**ScaleHD** [40]	
error-free	10011010	00001001	10011010	00001001	10011010	00001001
after error injection	10110010	00101000	10110010	00101000	$\underline{10000000}$	$\underline{00001011}$ (after **ScaleHD**)
after masking	10$\underline{1}$1$\underline{1}$010	00$\underline{0}$0100$\underline{0}$	$\underline{00000000}$	$\underline{00000000}$	10101000	00101010 (after error injection)

tasks that are beyond binary classifications as used in **PoisonHD**. In the meanwhile, "oracle defense" requires significant overhead when the number of classes scale and the performance of data sanitizing can be sub-par if there is not enough verified data to train the verified AM. Therefore, novel methods to defend data poisoning attacks are necessary to reduce the overall defense overhead.

For a privacy model stealing attack, a more practical scenario assumes a complete black-box scenario [15]. Since the training data can be proprietary, the attacker may not know what datasets are used to train the HDC model, thus not able to use methods such as partial data. Under these scenarios, analytical methods of reverse engineering are almost impossible. A potential direction is to utilize techniques from neural network domains such as gradient estimation and generative models [15]. However, since the theoretical foundations of HDC are mathematically different than connectionist models, such methods require more in-depth analysis.

5.2 Robustness against Hardware Errors

Most existing studies focus on errors that occurred in associative memory of HDC models, while errors can occur in other parts of HDC models as well, such as item memory, computing parts, etc. Thus, one future direction would be to include all possible hardware components in HDC models to ensure a comprehensive robustness evaluation. With this, it is also possible to analyze the sensitivity of different parts of HDC to hardware errors and their impact on HDC accuracy. Another important consideration in HDC system design is to balance the trade-off between robustness, performance, and efficiency. While robustness can always be enhanced by adding redundancy, this will inevitably increase the cost of the model. Therefore, future efforts regarding designing robust-while-efficient HDC systems are pertinent.

6 CONCLUDING REMARKS

As HDC is increasingly applied in safety-critical application domains, ensuring its robustness against uncertain and adversarial environments is becoming more pertinent. In this paper, we review the recent research efforts on HDC robustness against cyber attacks and hardware errors. For cyber attacks, we review recent literature in examining various types of cyber attacks, including adversarial attacks, data poisoning attacks, privacy attacks, and model stealing attacks, and how to defend against these attacks. For hardware robustness, we review recent literature in characterizing the impact of hardware errors on HDC accuracy and enhancing HDC robustness to hardware errors. This paper aims to provide an insightful overview of the related literature and endeavors to encourage future research efforts on this important topic.

Acknowledgments. This work was partially supported by NSF grant #2202310. 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] Pedro Alonso, Kumar Shridhar, Denis Kleyko, Evgeny Osipov, and Marcus Liwicki. Hyperembed: Tradeoffs between resources and performance in nlp tasks with hyperdimensional computing enabled embedding of n-gram statistics. In *2021 International Joint Conference on Neural Networks (IJCNN)*, pages 1–9. IEEE, 2021.

[2] Pablo R Bodmann, George Papadimitriou, Rubens L Rech Junior, Dimitris Gizopoulos, and Paolo Rech. Soft error effects on arm microprocessors: Early estimations versus chip measurements. *IEEE Transactions on Computers*, 71(10):2358–2369, 2021.

[3] Alessio Burrello, Kaspar Schindler, Luca Benini, and Abbas Rahimi. One-shot learning for ieeg seizure detection using end-to-end binary operations: Local binary patterns with hyperdimensional computing. In *2018 IEEE Biomedical Circuits and Systems Conference (BioCAS)*, pages 1–4. IEEE, 2018.

[4] Wencheng Chen and Hongyu Li. Adversarial attacks on voice recognition based on hyper dimensional computing. *Journal of Signal Processing Systems*, 93(7):709–718, 2021.

[5] Ron Cole, Yeshwant Muthusamy, and Mark Fanty. The isolet spoken letter database, 1990.

[6] Shijin Duan, Shaolei Ren, and Xiaolin Xu. Hdlock: Exploiting privileged encoding to protect hyperdimensional computing models against ip stealing. *arXiv preprint arXiv:2203.09681*, 2022.

[7] Patrice Godefroid, Michael Y Levin, David A Molnar, et al. Automated whitebox fuzz testing. In *NDSS*, volume 8, pages 151–166, 2008.

[8] Thomas R Henderson, Mathieu Lacage, George F Riley, Craig Dowell, and Joseph Kopena. Network simulations with the ns-3 simulator. *SIGCOMM demonstration*, 14(14):527, 2008.

[9] Alejandro Hernández-Cano, Rosario Cammarota, and Mohsen Imani. Prid: Model inversion privacy attacks in hyperdimensional learning systems. In *2021 58th ACM/IEEE Design Automation Conference (DAC)*, pages 553–558. IEEE, 2021.

[10] Michael Hersche, Sara Sangalli, Luca Benini, and Abbas Rahimi. Evolvable hyperdimensional computing: unsupervised regeneration of associative memory to recover faulty components. In *2020 2nd IEEE International Conference on Artificial Intelligence Circuits and Systems (AICAS)*, pages 281–285. IEEE, 2020.

[11] Po-Kai Hsu and Shimeng Yu. In-memory 3d nand flash hyperdimensional computing engine for energy-efficient sars-cov-2 genome sequencing. In *2022 IEEE International Memory Workshop (IMW)*, pages 1–4. IEEE, 2022.

[12] 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.

[13] Mohsen Imani, Abbas Rahimi, Deqian Kong, Tajana Rosing, and Jan M Rabaey. Exploring hyperdimensional associative memory. In *2017 IEEE International Symposium on High Performance Computer Architecture (HPCA)*, pages 445–456. IEEE, 2017.

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

[15] Sanjay Kariyappa, Atul Prakash, and Moinuddin K Qureshi. Maze: Data-free model stealing attack using zeroth-order gradient estimation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 13814–13823, 2021.

[16] Behnam Khaleghi, Mohsen Imani, and Tajana Rosing. Prive-hd: Privacy-preserved hyperdimensional computing. In *2020 57th ACM/IEEE Design Automation Conference (DAC)*, pages 1–6. IEEE, 2020.

[17] Yeseong Kim, Mohsen Imani, Niema Moshiri, and Tajana Rosing. Geniehd: Efficient dna pattern matching accelerator using hyperdimensional computing. In *2020 Design, Automation & Test in Europe Conference & Exhibition (DATE)*, pages 115–120. IEEE, 2020.

[18] Denis Kleyko, Dmitri A Rachkovskij, Evgeny Osipov, and Abbas Rahimi. A survey on hyperdimensional computing aka vector symbolic architectures, part i: Models and data transformations. *ACM Computing Surveys (CSUR)*, 2021.

[19] Haitong Li, Tony F Wu, Abbas Rahimi, Kai-Shin Li, Miles Rusch, Chang-Hsien Lin, Juo-Luen Hsu, Mohamed M Sabry, S Burc Eryilmaz, Joon Sohn, et al. Hyperdimensional computing with 3d vrram in-memory kernels: Device-architecture co-design for energy-efficient, error-resilient language recognition. In *2016 IEEE International Electron Devices Meeting (IEDM)*, pages 16–1. IEEE, 2016.

[20] Fangxin Liu, Haomin Li, Xiaokang Yang, and Li Jiang. L3e-hd: A framework enabling efficient ensemble in high-dimensional space for language tasks. In *Proceedings of the 45th International ACM SIGIR Conference on Research and Development in Information Retrieval*, pages 1844–1848, 2022.

[21] Dongning Ma, Jianmin Guo, Yu Jiang, and Xun Jiao. Hdtest: Differential fuzz testing of brain-inspired hyperdimensional computing. In *2021 58th ACM/IEEE Design Automation Conference (DAC)*, pages 391–396. IEEE, 2021.

[22] Dongning Ma, Rahul Thapa, and Xun Jiao. Molehd: Drug discovery using brain-inspired hyperdimensional computing. *IEEE International Conference on Bioinformatics and Biomedicine (BIBM)*, 2022.

[23] Harsha Moraliyage, Sachin Kahawala, Daswin De Silva, and Damminda Alahakoon. Evaluating the adversarial robustness of text classifiers in hyperdimensional computing. In *2022 15th International Conference on Human System Interaction (HSI)*, pages 1–8. IEEE, 2022.

[24] Justin Morris, Kazim Ergun, Behnam Khaleghi, Mohsen Imani, Baris Aksanli, and Tajana Rosing. Hydrea: Utilizing hyperdimensional computing for a more robust and efficient machine learning system. *ACM Transactions on Embedded Computing Systems (TECS)*, 2022.

[25] Fateme Rasti Najafabadi, Abbas Rahimi, Pentti Kanerva, and Jan M Rabaey. Hyperdimensional computing for text classification. In *Design, automation test in Europe conference exhibition (DATE), University Booth*, pages 1–1, 2016.

[26] Peer Neubert, Stefan Schubert, and Peter Protzel. An introduction to hyperdimensional computing for robotics. *KI-Künstliche Intelligenz*, 33(4):319–330, 2019.

[27] Abbas Rahimi, Tony F Wu, Haitong Li, Jan M Rabaey, H-S Philip Wong, Max M Shulaker, and Subhasish Mitra. Hyperdimensional computing nanosystem. *arXiv preprint arXiv:1811.09557*, 2018.

[28] Brian Randell, Pete Lee, and Philip C. Treleaven. Reliability issues in computing system design. *ACM Computing Surveys (CSUR)*, 10(2):123–165, 1978.

[29] Brandon Reagen, Paul Whatmough, Robert Adolf, Saketh Rama, Hyunkwang Lee, Sae Kyu Lee, José Miguel Hernández-Lobato, Gu-Yeon Wei, and David Brooks. Minerva: Enabling low-power, highly-accurate deep neural network accelerators. In *2016 ACM/IEEE 43rd Annual International Symposium on Computer Architecture (ISCA)*, pages 267–278. IEEE, 2016.

[30] Rahul Thapa, Bikal Lamichhane, Dongning Ma, and Xun Jiao. Spamhd: Memory-efficient text spam detection using brain-inspired hyperdimensional computing. In *2021 IEEE Computer Society Annual Symposium on VLSI (ISVLSI)*, pages 84–89. IEEE, 2021.

[31] Rahul Thapa, Dongning Ma, and Xun Jiao. Hdxplore: Automated blackbox testing of brain-inspired hyperdimensional computing. In *2021 IEEE Computer Society Annual Symposium on VLSI (ISVLSI)*, pages 90–95. IEEE, 2021.

[32] Anthony Thomas, Sanjoy Dasgupta, and Tajana Rosing. Theoretical foundations of hyperdimensional computing. *Journal of Artificial Intelligence Research*, 72:215–249, 2021.

[33] Ruixuan Wang and Xun Jiao. Poisonhd: poison attack on brain-inspired hyperdimensional computing. In *2022 Design, Automation & Test in Europe Conference & Exhibition (DATE)*, pages 298–303. IEEE, 2022.

[34] Ruixuan Wang, Xun Jiao, and X Sharon Hu. Odhd: one-class brain-inspired hyperdimensional computing for outlier detection. In *Proceedings of the 59th ACM/IEEE Design Automation Conference*, pages 43–48, 2022.

[35] Ruixuan Wang, Fanxin Kong, Hasshi Sudler, and Xun Jiao. Brief industry paper: Hdad: Hyperdimensional computing-based anomaly detection for automotive sensor attacks. In *2021 IEEE 27th Real-Time and Embedded Technology and Applications Symposium (RTAS)*, pages 461–464. IEEE, 2021.

[36] Xingbin Wang, Rui Hou, Boyan Zhao, Fengkai Yuan, Jun Zhang, Dan Meng, and Xuehai Qian. Dnnguard: An elastic heterogeneous dnn accelerator architecture against adversarial attacks. In *Proceedings of the Twenty-Fifth International Conference on Architectural Support for Programming Languages and Operating Systems*, pages 19–34, 2020.

[37] Tony F Wu, Haitong Li, Ping-Chen Huang, Abbas Rahimi, Gage Hills, Bryce Hodson, William Hwang, Jan M Rabaey, H-S Philip Wong, Max M Shulaker, et al. Hyperdimensional computing exploiting carbon nanotube fets, resistive ram, and their monolithic 3d integration. *IEEE Journal of Solid-State Circuits*, 53(11):3183–3196, 2018.

[38] Tony F Wu, Haitong Li, Ping-Chen Huang, Abbas Rahimi, Jan M Rabaey, H-S Philip Wong, Max M Shulaker, and Subhasish Mitra. Brain-inspired computing exploiting carbon nanotube fets and resistive ram: Hyperdimensional computing case study. In *2018 IEEE International Solid-State Circuits Conference-(ISSCC)*, pages 492–494. IEEE, 2018.

[39] Fangfang Yang and Shaolei Ren. On the vulnerability of hyperdimensional computing-based classifiers to adversarial attacks. In *International Conference on Network and System Security*, pages 371–387. Springer, 2020.

[40] Sizhe Zhang, Mohsen Imani, and Xun Jiao. Scalehd: Robust brain-inspired hyperdimensional computing via adapative scaling. In *2022 IEEE/ACM International Conference on Computer-Aided Design (ICCAD).* IEEE, 2022.

[41] Sizhe Zhang, Ruixuan Wang, Dongning Ma, Jeff Jun Zhang, Xunzhao Yin, and Xun Jiao. Energy-efficient brain-inspired hyperdimensional computing using voltage scaling. In *2022 Design, Automation & Test in Europe Conference & Exhibition (DATE)*, pages 52–55. IEEE, 2022.

[42] Sizhe Zhang, Ruixuan Wang, Jeff Jun Zhang, Abbas Rahimi, and Xun Jiao. Assessing robustness of hyperdimensional computing against errors in associative memory. In *2021 IEEE 32nd International Conference on Application-specific Systems, Architectures and Processors (ASAP)*, pages 211–217. IEEE, 2021.