PP-HDC: A Privacy-Preserving Inference Framework for Hyperdimensional Computing

Ruixuan Wang, Wengying Wen, Kyle Juretus, Xun Jiao
ECE Department, Villanova University
{rwang8, wwen, kyle.juretus, xun.jiao}@villanova.edu

***Abstract*—Recently, brain-inspired hyperdimensional computing (HDC), an emerging neuro-symbolic computing scheme that imitates human brain functions to process information using abstract and high-dimensional patterns, has seen increasing applications in multiple application domains and deployment in edge-cloud collaborative processing. However, sending sensitive data to the cloud for inference may face severe privacy threats. Unfortunately, HDC is particularly vulnerable to privacy threats due to its reversible nature. To address this challenge, we propose `PP-HDC`, a novel privacy-preserving inference framework for HDC. `PP-HDC` is designed to protect the privacy of both inference input and output. To preserve the privacy of inference input, we propose a novel hash-encoding approach in high-dimensional space by implementing a sliding-window-based transformation on the input hypervector (HV). By leveraging the unique mathematical properties of HDC, we are able to seamlessly perform training and inference on the hash-encoded HV with negligible overhead. For inference output privacy, we propose a multi-model inference approach to encrypt the inference results by leveraging the unique structure of HDC item memories and ensuring the inference result is only accessible to the owner with a proper key. We evaluate `PP-HDC` on three datasets and demonstrate that `PP-HDC` enhances privacy-preserving effects compared with state-of-the-art works while incurring minimal accuracy loss.**

I. Introduction

Hyperdimensional computing (HDC) is an emerging Non-von Neumann computing neuro-symbolic scheme inspired by the intricate functionality of the brain. HDC has gained significant traction in diverse domains and has shown remarkable advantages over deep neural networks (DNNs). Notably, by offering smaller model sizes, less computation cost, and one-shot learning capabilities, HDC becomes a promising alternative to DNNs [7]. Recently, HDC has been implemented in edge-cloud collaborative processing scenarios, e.g., machine learning as a service (MLaaS), including cloud computing and federated learning due to its notable processing efficiency [10], [19].

Despite the growing popularity of HDC, the privacy threat of HDC models is still under-investigated. This is particularly important in the context of edge-cloud collaborative processing. Transmitting sensitive data from edge users to the cloud for inference raises significant privacy concerns, since untrusted cloud hosts may expose the data to suspicious parties [8].

Unfortunately, HDC is indeed particularly vulnerable to privacy threats due to its reversible nature. Unlike non-linear operations introduced in DNNs, HDC hardly maintains any privacy since its fully reversible computation [12]. Prive-HD [12] attempted to preserve privacy by applying certain modifications on the data hypervectors (HVs), the data representation in HDC, through quantization and noise injection.

However, the effects of such strategies are considerably limited and the accuracy drops due to the modifications on the HDC model. To address these challenges, we propose **`PP-HDC`**, a novel privacy-preserving strategy to protect both the inference input and output of HDC. Our main contributions are as follows: (1) In **`PP-HDC`**, we propose a novel HV hash-encoding approach, designed to preserve the privacy of inference input. The proposed hash-encoding can be seamlessly integrated into the HDC pipeline since it preserves the distance properties between HVs, enabling direct training and inference on the hashed HVs. (2) In **`PP-HDC`**, we propose a novel multi-model inference approach for protecting the privacy of inference output, which is largely overlooked in existing research [12], [10]. Leveraging the unique item memory structure of HDC, **`PP-HDC`** ensures that only the user with the right key can access the inference results. (3) Furthermore, We perform extensive evaluations of **`PP-HDC`** on three datasets containing sensitive information. Compared to previous work [12] which introduces more than 10% accuracy degradation, **`PP-HDC`** introduces negligible accuracy loss at less than 1%. (4) We conduct a comprehensive security validation analysis of **`PP-HDC`** against two attacks, namely the brute force attack on recover inference input and inference output knowledge attack. The experimental results and analysis demonstrate that **`PP-HDC`** can provide effective privacy preservation in both inference input and output.

II. Related Work

The development of machine learning as a service (MLaaS) has witnessed remarkable growth in recent years, driven by the proliferation of edge devices and diverse learning tasks. Meanwhile, the requirement for privacy-preserving machine learning (PPML) solutions is also increasing due to privacy concerns in the ML community. One conventional privacy-preserving solution is differential privacy [5], [6], which can preserve privacy and train the ML model by introducing perturbation and noise to data. However, differential privacy is not extended to the inference phase [8], where the privacy of inference input and output demand to be protected. Homomorphic encryption (HE) algorithm [17] has emerged as a popular solution for preserving privacy during inference HE allows computing functions or operations on encrypted data without deciphering, and different operations on encrypted data require diverse implementations of HE [1], [2], [3]. CryptoNets [8] implements one of the first HE-based DNNs inference frameworks. However, the adoption of the HE approach comes with significant inefficiency and computational overhead [10].

979-8-3503-4859-0/DATE24/© 2024 EDAA

In the context of HDC, few studies have focused on data security and privacy concerns within HDC processing. Recently, SecureHD [10] implements an HDC cloud computing framework based on Multi-party Computation (MPC), which protects data security by encryption during data transmission. Prive-HD [12] proposes the first privacy-preserving approach for HDC, which provides differential privacy protection during the HDC training and inference process. Moreover, PRID [9] addresses the defense of the HDC model against model inversion attacks and provides privacy preservation approaches through noise injection and HV quantization.

III. Preliminaries

A. General HDC Arithmetic

Hypervectors (HVs) are the basic components of HDC which are high-dimensional and holographic vectors with i.i.d. elements [11]. HDC has two main arithmetic operations, bundling ($+$) and binding ($*$), as elaborated in Eq. (1). Different operations make HDC able to aggregate information and generate new representations. The bundling operation performs an element-wise addition on the two input HVs, while the binding operation multiplies two HVs element-wisely. For the calculation result, the bundling operation creates the third HV similar to the two input HV and the multiply operation generates the third HV different from the two input HVs [11].

$$\begin{aligned} \overrightarrow{HV_i} + \overrightarrow{HV_j} &= \langle hv_{i1} + hv_{j1}, \ldots, hv_{id} + hv_{jd} \rangle \\ \overrightarrow{HV_i} * \overrightarrow{HV_j} &= \langle hv_{i1} * hv_{j1}, \ldots, hv_{id} * hv_{jd} \rangle \end{aligned} \tag{1}$$

In this paper, we use the cosine similarity δ, provided as Eq. (2), to measure the similarity of information between HVs.

$$\delta(\overrightarrow{HV_i}, \overrightarrow{HV_j}) = \frac{\overrightarrow{HV_i} \cdot \overrightarrow{HV_j}}{||\overrightarrow{HV_i}|| \times ||\overrightarrow{HV_j}||} \tag{2}$$

B. General HDC Process

The implementation of HDC consists of the encoding process, training, and inference process. First of all, the HDC encoding process involves projecting a data sample into an HV the encoding function can be domain-specific. Here we adopt the ID-based encoding strategy similar to the previous work [10], where we utilize N Base HVs in item memories (IMs) that map N features spatially. After the encoding process, all the data samples in datasets are embedded into HVs during the process of HDC. However, the encoded HV has a natural drawback in privacy, which will be discussed in Sec. IV.

Secondly, the training process involves establishing associative memories (AMs) across the entire training set. For the training set containing C classes of samples, each training sample is encoded into an HV $\overrightarrow{HV^i}$ where i corresponds to the label of the sample. The HDC training process, represented by Eq. (3), aggregates (bundles) HVs with the same label together. The resulting AMs consists of C class HVs namely $\{\overrightarrow{HV_{L_1}}, \ldots, \overrightarrow{HV_{L_C}}\}$, representing C classes of samples.

$$AM = \{\overrightarrow{HV_{L_1}}, \ldots, \overrightarrow{HV_{L_C}}\} = \{\sum \overrightarrow{HV^1}, \ldots, \sum \overrightarrow{HV^C}\} \tag{3}$$

Last but not least, the inference process involves predicting the category of a query data sample in the testing set. Firstly, each query data Q is encoded into an HV named query HV $\overrightarrow{HV_Q}$ based on the same set of IMs and encoding strategy. Then we measure the cosine similarity (δ) between the query HV and all the C class HVs in AMs. As Eq. (4) indicates, the index of the highest similarity denotes the inference outcome x of the query HV $\overrightarrow{HV_Q}$.

$$x = argmax(\delta(\overrightarrow{HV_Q}, \overrightarrow{HV_{L_1}}), \ldots, \delta(\overrightarrow{HV_Q}, \overrightarrow{HV_{L_C}})) \tag{4}$$

IV. HDC Privacy Threat Model

In this section, we analyze the inherent privacy vulnerabilities of HDC, which is consistent with the previous work [12]. Within the context of HDC-based MLaaS, users initiate the process by encoding their raw inputs into HVs and transmitting them to the cloud for subsequent processing. However, the data privacy of users is not secured. Since the HDC encoding only relies on binding and bundling, which are fully linear and reversible, the information encoded in HVs can be easily decoded and reconstructed [12]. As a result, the reversible HVs allow potential adversaries to reconstruct the original information from the raw inputs.

V. **PP-HDC** Framework

An overall architecture of **PP-HDC** framework is illustrated in Fig. 1. Based on the aforementioned circumstances, we present a comprehensive assessment of the HDC privacy preservation task. Here **PP-HDC** provides privacy-preserving

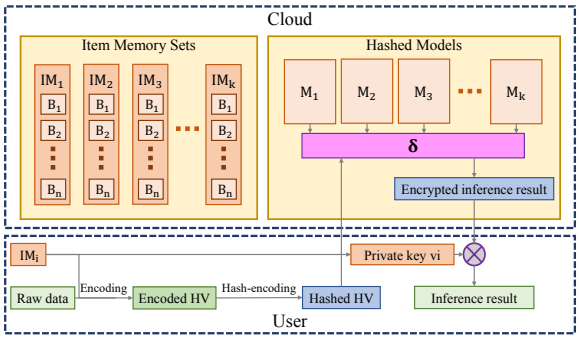


Fig. 1. Proposed **PP-HDC** framework. **PP-HDC** establishes a privacy-preserved HDC inference pipeline by guarding both inference input privacy and inference output privacy.

inference by protecting the privacy of two critical components, including the inference input transmitted from users to the cloud and the inference output transmitted from the cloud to users. In this paper, users represent edge devices with limited memory and computational resources, and rely on cloud computing service providers for model execution due to resource constraints, as used by existing works [10], [19]. We assume that the cloud server is an untrusted semi-honest server, which may follow the execution protocol but will attempt to learn as much as it can. Consistent with the existing work [8], we summarize four principles for the cloud computing circumstance as follows: (1) The cloud possesses and stores several pre-trained models. (2) The user processes encoded representations of sensitive data and securely transmits them to the cloud. (3) The cloud server can compute the inference result and provide services based on

the user's submission, without access to the original data or the inference outcome. (4) The user receives the inference result and decodes the true final inference outcome.

A. Privacy Preservation for Inference Input

In this section, we aim to protect the privacy of the encoded HV of the raw input data in the HDC-based model. Here we use notation $\overrightarrow{HV_P}$ to represent the original encoded HV. We propose a hash-encoding scheme for concealing the information in HVs. Notably, hash-encoding is post-processing on top of encoded HVs and is feasible for existing encoding approaches in HDC [7]. After hash-encoding, the original HV $\overrightarrow{HV_P}$ will be projected into a hashed HV $\overrightarrow{HV_C}$ with the same dimensions.

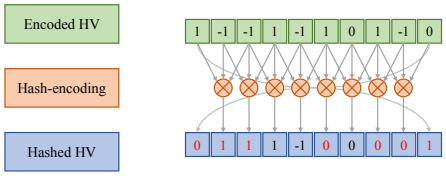


Fig. 2. An illustration of HV hash-encoding process with $s = 1$.

The proposed HV hash-encoding scheme includes two steps, as illustrated in Fig. 2. Initially, we swap the first and last elements in $\overrightarrow{HV_P}$ to the opposite position in the hashed HV $\overrightarrow{HV_C}$ to augment information obfuscation. Subsequently, we deploy hash-encoding on $\overrightarrow{HV_P}$ and project the $\overrightarrow{HV_P}$ from HV space to hashing space. As Eq. (5) indicates, starting from the second element, **PP−HDC** multiplies every $2 * s + 1$ element to derive the corresponding element in $\overrightarrow{HV_C}$, where s represents the number of neighbors associated with the related element.

$$hv_{C_i} = \prod_{i-s}^{i+s} hv_{P_i}, (s < i < D - s) \tag{5}$$

In **PP−HDC**, the key attribute of HV hash-encoding lies in its ability to preserve the distance properties between HVs, and simultaneously obfuscate the elements in the HVs to conceal the original information. Essentially, the HV hash-encoding approach is inspired by and has similar characteristics to the local sensitive hashing (LSH) approach [14], which is also widely utilized in privacy-preserving computation field [16], [15]. LSH contains a series of hash functions and maintains the distance relationship of data in both input space and hash-code space with high probability. However, directly deploying LSH or traditional hash function on HVs will introduce dimension inconsistency between hash input and output, which is not feasible to integrate the HV hash-encoding within the HDC training and inference process. Consequently, inspired by LSH, we develop HDC-specific hash-encoding, which retains the dimensionality and distance relationship and effectively conceals sensitive information encoded in hashed HV. Given the HDC algorithm highly depends on the distance relationship comparison, fits seamlessly within the HDC pipeline by maintaining the essential distance relationship between original HVs and hashed HVs. It is worth mentioning that while our proposed hash-encoding is effective, it may not be optimal. Although it is out of the scope of this paper, our future endeavors will optimize the hash-encoding algorithm by conducting a comprehensive analysis of the security level and accuracy.

The HV hash-encoding in **PP−HDC** is non-reversible, which means the original HVs cannot be fully recovered from the hashed HVs. The main reason is that some elements are digested during the hash-encoding process. For instance, as depicted in Fig. 2, a "0" element in the HV will digest the elements nearby no matter what the values are. Meanwhile, the selection of s can also lead to different hash-encoding effects and information digestion. Moreover, **PP−HDC** provides the capability to perform HDC training and inference directly on the hashed HVs, allowing the training of a hashed HDC model. Consequently, users can securely transmit the hashed HVs to the cloud or other users without the risk of privacy leakage.

B. Privacy Preservation for Inference Output

In the context of MLaaS, the unencrypted inference result of the user's data will be exposed to the cloud, which can pose a serious risk to user data privacy. However, prior research [10], [12] has largely overlooked the inference output privacy. To address this gap, we propose a multi-model-based HDC inference approach to protect the privacy of inference output on the cloud. The multi-model-based inference stems from the attribute of the HDC algorithm, where the base HVs should be the same during the encoding of class HVs and query HVs. In other words, HDC necessitates that all training and testing samples utilize the same IM for encoding. Any mismatch of IMs will lead to accuracy degradation to the level of random guessing. This phenomenon is caused by the quasi-orthogonality of randomly generated base HVs in IMs. We will further discuss the observation based on our experimental results in Sec. VI-D.

Inspired by this inherent property of HDC, the multi-model inference comprises the following process: We first generate k different sets of IMs $\{IM_1, \ldots, IM_k\}$ and train k hashed HDC models based on HV hash-encoding in **PP−HDC**. Each set of IMs is matched one-to-one with a hashed model. The k hashed models and the corresponding k sets of IMs are publicly accessible to all the users. Subsequently, the user can select one of the k IMs provided on the cloud to encode and hash-encode their input data and transmit the hashed HVs to the cloud for inference. When the user selects one of the k IMs, denoted as IM_i, a private key v_i is generated simultaneously to signify the index of the chosen IM. On the cloud side, the user's hashed HV is concurrently processed by all the k pre-trained hashed HDC models for inference. Consequently, the cloud provides the k-length encrypted inference result to the user instead of a single inference outcome. The cloud has no knowledge of which inference outcome corresponds to the user's true inference result, the privacy of the user's inference output remains secured. Upon receiving the encrypted inference result, the user can simply decode the true inference outcome through the private key.

VI. Experiment Results

A. Experimental Setup

In order to assess the privacy-preserving performance of **PP−HDC**, we conduct experiments on three real image datasets, **MNIST** [13], Breast MNIST (**BREAST**) and Pneumonia MNIST (**PNEU**)[18]. These three datasets encompass a range of tasks, from handwritten digit recognition to medical image classification, while all of these image datasets necessitate a privacy-preserving strategy to protect data privacy and inference privacy from leakage.

In our experiments, we implement **PP−HDC** using Python on a Raspberry Pi, to comprehensively evaluate the performance and overhead of **PP−HDC**. To quantitatively evaluate the privacy preservation achieved by **PP−HDC**, we utilize two metrics, the peak signal-to-noise ratio (PSNR) and mean squared error (MSE), similar to the approach in the previous work [12]. In our experiment, we compare the PSNR and the MSE values between the original image and reconstructed images generated from the hashed HV to assess the efficacy of **PP−HDC** privacy preservation. The PSNR metric indicates the quality of reconstruction results from the corresponding HV and the MSE metric quantifies the deviation between the original and the reconstructed images. Lower PSNR and higher MSE indicate stronger privacy preservation performance.

B. Hash-encoding Configuration

To determine the optimal hyperparameter in **PP−HDC**, we conduct a comprehensive analysis on the s in hash-encoding. Here we conduct an ablation experiment where we select different values of s in $\{1, 2, 3\}$. As shown in the left side of Fig. 3, the increase in the value of s leads to a quite minor influence on the PSNR of the reversed-engineered image, with a difference of approximately 1%. However, the accuracy of **PP−HDC** has a noticeable degradation due to the increase of s. On the right side of Fig. 3, when we select $s = 3$ instead of 1, the accuracy for **MNIST** declines by over 15% compared with the baseline ($s = 0$). The phenomenon is consistent with other datasets. Hence, we select $s = 1$ in **PP−HDC** for all subsequent experiments since $s = 1$ is enough to achieve satisfactory privacy preservation and maintain acceptable accuracy.

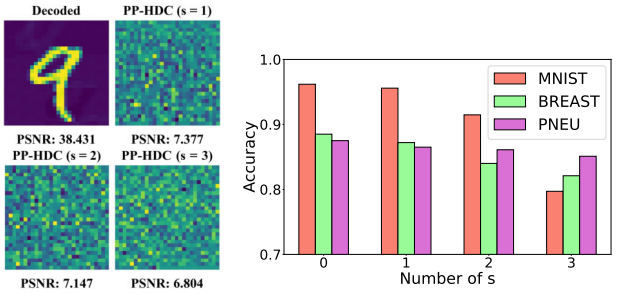


Fig. 3. Impact of different value of s in **PP−HDC**. **left**: Different values of s yield similar degrees of information obfuscation. **right**: Accuracy comparison of different hashed HDC models using varying values of s.

C. Evaluation of Privacy Preservation for Inference Input

The experimental results are denoted in Fig. 4, HV hash-encoding leads to a negligible decline in accuracy (0.5% on the **MNIST** dataset, 0.7% on the **BREAST** dataset and 1% on the **PNEU** dataset), which outperforms the Prive-HD [12] that introduces more than 15% accuracy drop [12]. Additionally, we also quantify the computational overhead of the HV hash-encoding by assessing the extra energy consumption on Raspberry Pi. As presented in Fig. 4, our results reveal that **PP−HDC** only incurs 2.1%, 4.2% and 4.9% extra energy overhead.

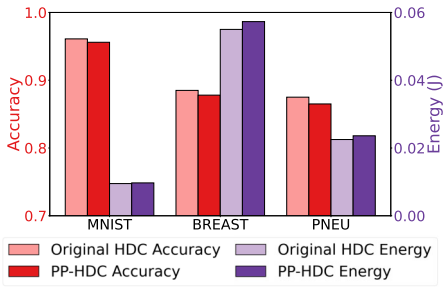
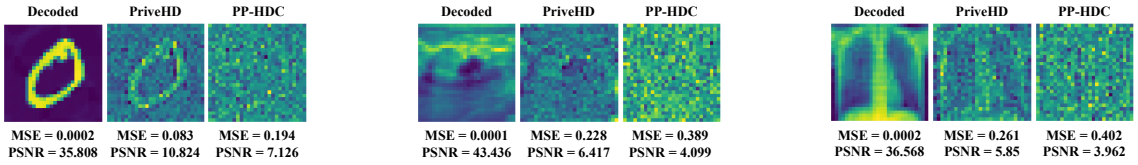


Fig. 4. Comparison of accuracy and energy efficiency between the original HDC model and **PP−HDC**. **PP−HDC** provides inference input privacy preservation with negligible accuracy drop and computational overhead.

Furthermore, we compare the information obfuscation of different approaches across three datasets, and the experimental results are presented in Fig. 5. Here we utilize the **MNIST** dataset as an example. The first column denotes that, without any privacy preservation, the reconstructed image will expose almost all the information. In the second column, we implement the 1-bit quantization and dimension masking approaches proposed in Prive-HD [12], yet the reconstructed image is sufficiently clear. However, as depicted in the third column, the reconstruction image from the HV hashed by **PP−HDC** exhibits minimal information disclosure. According to our observation, the PSNR of the reconstructed image from the hashed HV is only 7.126 dB, which achieves 1.5X lower PSNR than the PSNR of the reconstruction obtained from Prive-HD (10.824 dB). **PP−HDC** demonstrates superior information concealing compared to the previous approaches.

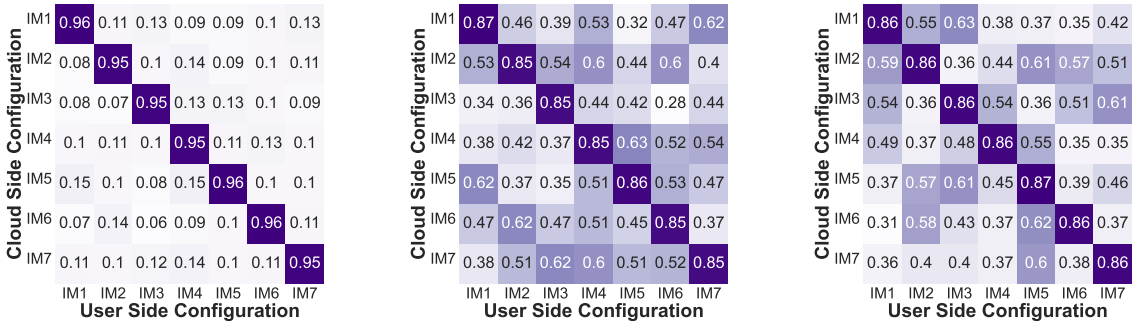
D. Evaluation of Privacy Preservation for Inference Output

Here we investigate the performance of privacy preservation on the inference output. We select $k = 7$ which means we pre-train seven different models based on seven different sets of item memories (IMs). Notably, the proposed multi-model inference approach demonstrates no accuracy loss since it does not disturb the inference process itself. We evaluate the query HVs encoded with seven IMs on seven different pre-trained models, as Fig. 6 illustrates, the inference results are presented as a $(7, 7)$ matrix. Each element indicates the accuracy of the HDC model corresponding to the IMs pair. For instance, the element (i, j) denotes the accuracy of the query HVs encoded with IM_j while the model is established with IM_i. One noteworthy observation is that when the selected IMs are mismatched, i.e., $i \neq j$, the inference accuracy is close to random guessing, where the class HVs and query HVs are orthogonal. On the contrary, if and only if $i = j$, the inference accuracy remains at the original level without any loss, as demonstrated by the diagonal elements in the inference matrix.



(a) **MNIST** (b) **BREAST** (c) **PNEU**

Fig. 5. The privacy preservation performance comparison of different approaches across three datasets. Each result contains three columns that illustrate the following: (1) The reconstructed image of the original HV using HDC decoding, (2) The reconstructed image of HV implemented with the Prive-HD approach, and (3) The reconstructed image of the **PP−HDC** hashed HV.



(a) **MNIST** (b) **BREAST** (c) **PNEU**

Fig. 6. Evaluation of multi-model inference in **PP−HDC**. Only the private key indicates the correct inference outcome in the encrypted inference result.

VII. Security Validation Analysis

In this section, we primarily focus on the information leakage hazard of **PP−HDC**, the adversary can retrieve information from the inference input (the hashed HV) and inference output (the encrypted inference result).

A. Threat Model

As mentioned in Sec. V, the cloud provider is an untrusted semi-honest server. Additionally, we consider some malicious users in the MLaaS environment are also honest-but-curious (HBC), which means they can eavesdrop and spy on other users. According to the threat model we considered, the adversary can be either the cloud or malicious users, and have access to all of the k sets of item memories (IMs) and the k pre-trained models. Furthermore, the adversary is well-informed about the proposed privacy preservation methodology in **PP−HDC**.

B. Information Recovery of the Original HV

Under this attack scenario, the adversary aims to retrieve the original HV from the hashed HV. To model this attack, the Z3 SMT solver [4] was utilized with a model that was given a 5-symbol hashed HV sub-vector and iteratively solved to determine all possible input patterns for the given hashed output. As shown in Fig. 7, each 5-symbol hashed HV block was fed with the seven inputs from the original HV.

The 5-symbol hashed sub-vector model was chosen as the influence of the middle three (a_2 through a_4) is completely captured by the output model. Thus, no other hashed output values would be altered by a change in a_2 through a_4. Then the z3 SMT solver model is iteratively shifted by one symbol over the entire hashed HV to get the number of possible solutions (a_0 through a_6) for each 5-symbol block of the hashed HV (e_0 through e_4). If a given symbol is fixed, meaning that the value is constant for all possible solutions, the constant value is propagated to the next window of analyzed symbols. Thus, a map of all known values is kept track of during the analysis and the results are validated against the original HV.

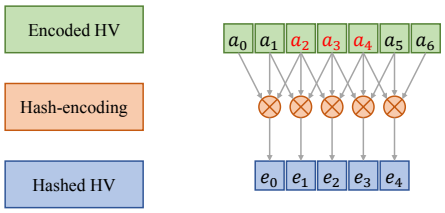


Fig. 7. z3 SMT model utilized to evaluate bits of interest a_2 through a_4. Any known symbols in a_0-a_6 are set before the model is run.

C. PP−HDC against Brute Force Attack

According to the **PP−HDC** configuration, the benchmark HVs have values in $\{-1, 0, 1\}$ and the average number of possible solutions per seven bits was found to be 207.28 with the number of fixed symbols being an average of 1745. The reason for the number of the known symbols of the original HV has to do with the information loss when multiplying by a "0" during the HV hash-encoding. A randomly generated HV with elements in $\{-1, 0, 1\}$ has an entropy of approximately 7.7 bits per 7-symbols of the original output. This equates to the search space for the adversary of $2^{7.7*10000/7} \approx 2^{11000}$. Notably, the search space is significantly lower than the theoretical search space of the original HV of 3^{10000}, but the number of unknown bits remains very large and is not susceptible to brute force attack. For example, even a supercomputer with 100 billion cores running at 100 GHz would require $6.7 * 10^{3281}$ years to search through the entire 2^{11000} search space, assuming one HV could be tried every CPU cycle. Additionally, even though

some of the elements are recoverable in the original HV, the orthogonality between the HV before and after hash-encoding means the information in those two HVs is not relevant.

D. *PP-HDC against Inference Output Knowledge Attack*

In **PP-HDC**, the primary information accessible to the adversary is the encrypted inference results themselves during the inference process. However, an untrusted cloud would also have access to the cosine similarities between the hashed HV and each of the k pre-trained models. We assume that an adversary is given the hashed HV input and the encrypted inference results consist of k inference outcomes from each of the pre-trained hashed models, similar to the situation depicted in Fig. 6. The adversary wins if the correct inference results can be produced with a greater than a $1/k$ probability. Consequently, we conduct an inference experiment based on **MNIST** dataset with $k = 7$, and the experimental results are denoted as Fig.8.

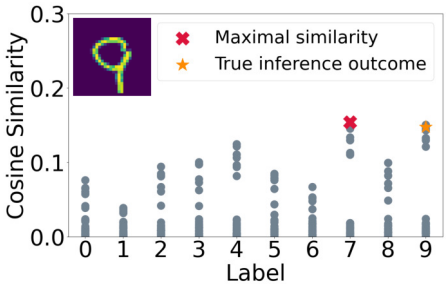


Fig. 8. An multi-model inference analysis based on **PP-HDC**, here we select an image "9" from **MNIST** dataset and the number of pre-trained models k = 7. Each node represents a one-time similarity measurement between the query HV and one class HV in the pre-trained model.

Based on our implementation, it is worth mentioning that **PP-HDC** provides a solid concealment on inference results by generating the encrypted inference results following a close to random distribution over label "0" to "9". The adversary cannot access the true prediction outcome by searching over the encrypted inference results. The experimental results are consistent with the scenarios aforementioned in Sec. VI-D, the cosine similarity is close to zero when the IMs are mismatched. Since the adversary cannot statistically access the true inference outcome, another way is to select the maximal cosine similarity as the possible label. However, as indicated in Fig.8, the maximal similarity achieves 0.154 and points to the label "7" as the inference outcome while the true inference outcome is "9" with a cosine similarity of 0.133. Therefore, given the class vectors alone, the adversary is not able to determine the correct model being used and is thus unaware of the inference result.

VIII. Conclusion

This paper focuses on the privacy-preserving inference of an emerging neuro-symbolic learning method named HDC. Specifically, we propose **PP-HDC**, a privacy-preserving inference framework for HDC that can preserve privacy on both inference input and output. By leveraging a novel HV hash-encoding approach, the hashed HVs preserve their dimensionality and original distance properties in high-dimensional space, which is essential to the subsequent processing. Our experimental results denote that the proposed HV hash-encoding in **PP-HDC** can largely obfuscate the information on the input data, outperforming existing state-of-the-art methods. Furthermore, we present a multi-model inference approach to prevent the cloud from accessing the inference outcome. Notably, **PP-HDC** accomplishes the inference input and output privacy preservation with minimal accuracy drop compared with existing work. Moreover, we conduct a comprehensive security validation analysis of **PP-HDC**. This paper aims to open up new directions and challenges for future privacy-preserving and secure HDC model designing.

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] Frederik Armknecht et al. A guide to fully homomorphic encryption. *IACR Cryptol. ePrint Arch.*, 2015, 2015.
[2] Joppe W Bos et al. Improved security for a ring-based fully homomorphic encryption scheme. In *IMA International Conference on Cryptography and Coding*. Springer, 2013.
[3] Jung Hee Cheon et al. A full rns variant of approximate homomorphic encryption. In *International Conference on Selected Areas in Cryptography*. Springer, 2018.
[4] Leonardo De Moura and Nikolaj Bjørner. Z3: An efficient smt solver. In *International conference on Tools and Algorithms for the Construction and Analysis of Systems*, pages 337–340. Springer, 2008.
[5] Cynthia Dwork. Differential privacy. In *International Colloquium on Automata, Languages, and Programming*. Springer, 2006.
[6] Cynthia Dwork et al. Calibrating noise to sensitivity in private data analysis. In *Theory of cryptography conference*. Springer, 2006.
[7] Lulu Ge and Keshab K Parhi. Classification using hyperdimensional computing: A review. *IEEE Circuits and Systems Magazine*, 20(2), 2020.
[8] Gilad-Bachrach et al. Cryptonets: Applying neural networks to encrypted data with high throughput and accuracy. In *International conference on machine learning*. PMLR, 2016.
[9] Alejandro Hernández-Cano et al. Prid: Model inversion privacy attacks in hyperdimensional learning systems. In *2021 58th ACM/IEEE Design Automation Conference (DAC)*, pages 553–558. IEEE, 2021.
[10] Mohsen Imani et al. A framework for collaborative learning in secure high-dimensional space. In *2019 IEEE 12th International Conference on Cloud Computing (CLOUD)*. IEEE, 2019.
[11] Pentti Kanerva. Hyperdimensional computing: An introduction to computing in distributed representation with high-dimensional random vectors. *Cognitive computation*, 1(2), 2009.
[12] Behnam Khaleghi, Mohsen Imani, and Tajana Rosing. Prive-hd: Privacy-preserved hyperdimensional computing. In *2020 57th ACM/IEEE DAC*. IEEE, 2020.
[13] Yann LeCun et al. Gradient-based learning applied to document recognition. *Proceedings of the IEEE*, 86(11), 1998.
[14] Paulevé et al. Locality sensitive hashing: A comparison of hash function types and querying mechanisms. *Pattern recognition letters*, 31(11):1348–1358, 2010.
[15] Lianyong Qi et al. A two-stage locality-sensitive hashing based approach for privacy-preserving mobile service recommendation in cross-platform edge environment. *Future Generation Computer Systems*, 88:636–643, 2018.
[16] Lianyong Qi et al. Privacy-aware cross-platform service recommendation based on enhanced locality-sensitive hashing. *IEEE Transactions on Network Science and Engineering*, 8(2):1145–1153, 2020.
[17] Ronald L Rivest et al. On data banks and privacy homomorphisms. *Foundations of secure computation*, 4(11), 1978.
[18] Jiancheng Yang et al. Medmnist classification decathlon: A lightweight automl benchmark for medical image analysis. In *2021 IEEE 18th ISBI*. IEEE, 2021.
[19] Quanling Zhao et al. Fedhd: federated learning with hyperdimensional computing. In *Proceedings of the 28th Annual International Conference on Mobile Computing And Networking*, pages 791–793, 2022.