HDCircuit: Brain-inspired HyperDimensional Computing for Circuit Recognition

Paul R. Genssler*, Lilas Alrahis†, Ozgur Sinanoglu†, and Hussam Amrouch*‡
*Semiconductor Test and Reliability, University of Stuttgart, Stuttgart, Germany; †New York University Abu Dhabi, UAE
‡Chair of AI Processor Design, Technical University of Munich; TUM School of Computation, Information and Technology; Munich Institute of Robotics and Machine Intelligence, Munich, Germany
Email: genssler@iti.uni-stuttgart.de; lma387@nyu.edu; ozgursin@nyu.edu; amrouch@tum.de

***Abstract*—Circuits possess a non-Euclidean representation, necessitating the encoding of their data structure (e.g., gate-level netlists) into fixed formats like vectors. This work is the first to propose brain-inspired hyperdimensional computing (HDC) for optimized circuit encoding. HDC does not require extensive training to encode a gate-level netlist into a *hypervector* and simplifies the similarity check between circuits from graph-based to the similarity between their hypervectors. We introduce a versatile HDC-based encoding method for circuit encoding. We demonstrate its effectiveness with the application of circuit recognition using ITC-99 and ISCAS-85 benchmarks. We maintain a 98.2 % accuracy, even when the designs are obfuscated using logic locking.**

***Index Terms*—Hyperdimensional computing, Circuit encoding, Circuit recognition, Hardware intellectual property**

I. Introduction

Brain-inspired hyperdimensional computing (HDC) is a promising machine learning (ML) model with successful applications in various domains, such as bio-signal processing [1], pattern and language recognition [2, 3], and others [4, 5]. By encoding data as high-dimensional *hypervectors* [6], HDC ensures that similar data samples are mapped closely in the *hyperspace*. This enables applications that measure similarity between samples, such as classification [1–3].

HDC has recently demonstrated remarkable performance in solving graph learning problems, utilizing its math-based algebra to encode graphs into large-dimensional hypervectors [7, 8]. Hypervectors represent nodes and edges, which are combined to form a comprehensive hypervector representing the entire graph. These hypervectors can then be used in further calculations to solve the intended end task, such as graph classification. While HDC has been used to analyze graphs of molecules and in the bioinformatics domain [7], *we are the first to explore its application for hardware circuits*.

Hardware circuits inherently possess a non-Euclidean representation. When applying ML to circuit-related tasks, such as circuit recognition/classification, it becomes necessary to encode these circuits into standardized formats, such as matrices or vectors, which can be subsequently utilized by ML models. Recent advancements have introduced various circuit encoding methods but these come with certain limitations. In an early attempt [9], the *sensitive circuit encoding* was identified as a problem as their method depends on the chosen technology library, limiting their model's generalizability. Recently, graph neural networks (GNNs) have been employed, enabling a diverse range of tasks [10–14]. Yet, these models often require the availability of balanced and *large datasets for training*.

This work was partially supported by Advantest as part of the Graduate School "Intelligent Methods for Test and Reliability" at the University of Stuttgart.

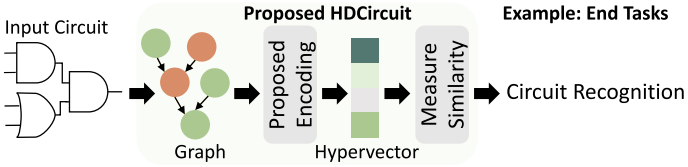

Figure 1. The proposed HDC encoding enables circuit recognition.

To address the above shortcomings, we propose the use of HDC for circuit encoding due to its ability to encode complex graphs without extensive training. Unlike GNNs, HDC performs the encoding in a single pass, as illustrated in Fig. 1. While HDC has been applied to graphs before, we demonstrate that conventional HDC encoding [7] fails when applied to circuits, posing unique challenges revealed in this work for the first time. Circuits contain nodes with different attributes that require careful consideration. Unlike previous work utilizing HDC for graph analysis in bioinformatics [7, 8], we incorporate information from the logic gates (nodes) into the graph representation itself. Neglecting that results in indistinguishable hypervectors.

We investigate, for the first time, how to effectively encode gate-level netlists as hypervectors using the HDC algebra.

Our novel contributions of this work are as follows:

1) We present a novel netlist-to-hypervector encoding. Our encoding method captures each gate's functionality and connectivity in the gate-level netlist.
2) We evaluate the effectiveness of our encoding in solving the *Circuit Recognition* task. We apply logic locking to conceal the functionality of various benchmark circuits. Then, we create an HDC model solely from the original circuits, i.e., *one training sample per class*. Given a locked circuit, the model recognizes the underlying original circuit with 98 % accuracy.

II. HDC for Circuit Recognition

A. Gate-Level Netlist to Hypervector Encoding

Gate-to-Hypervector: One gate type hypervector $\vec{G}_{TYPE}$ is randomly generated for each type of gate. Such a hypervector inherently includes the logical function ("AND") as well as the number of inputs (e.g., three for "AND3"). The number of

979-8-3503-4859-0/DATE24/© 2024 EDAA

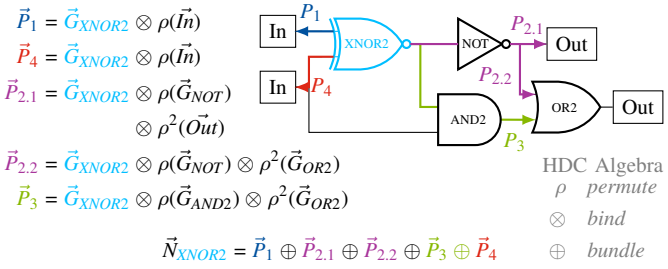

Figure 2. The proposed encoding scheme maps a gate-level netlist into hyperspace by creating a node hypervector $\vec{N}$ for each gate. These are then bundled into $\vec{C}$. The example shows the encoding of the highlighted XNOR2 gate.

outputs is also inherently included by the function of the gate, an AND has only one output while a full adder has two. The primary inputs and outputs of the circuit are treated as gates of type INPUT and OUTPUT, respectively.

Circuit-to-Hypervector: Our node-based method encodes individual gates and their neighborhoods. When encoding a node, a depth-first search is employed. Starting from the to-be-encoded node, all paths in the undirected circuit graph are encoded as n-grams with $n = d$ and the gate type hypervectors $\vec{G}_{TYPE}$. The path length is limited by the hyperparameter d, which is 2 for the example in Fig. 2. All n-grams are bundled together to form the node hypervector $\vec{N}$. These encoded nodes $\vec{N}$ are then bundled into a single circuit hypervector $\vec{C}$. Every node contributes equally to the circuit hypervector, making the encoding robust against changes to the inputs.

B. Creating the HDC Circuit Recognition Model

The encoded gate-level netlists create the model for circuit recognition. Hence, *a single sample* (i.e., the unlocked original netlist) is sufficient. The encoding itself does not require a training in the traditional sense. Hyperparameters, such as n-gram length or dimension, are tuned by minimizing the internal similarity between the circuit hypervectors. During inference, we compare the unknown locked circuit against each circuit hypervector. Locked versions of circuit A are assumed to have similar hypervectors to the original whereas hypervectors from circuit B are orthogonal.

III. Experimental Results and Evaluation

The hyperparameters n-gram/depth d are explored for 1, 2, 4, 6, 8 and hypervector dimensions of 512, 1024, 2048 with real-value components. GraphHD [7] is reimplemented. Logic locking is implemented using in-house Perl scripts.

Dataset Generation: We select ten base combinational ISCAS-85 and seven ITC-99 benchmarks to apply random XOR/XNOR logic locking with key sizes 32, 64, 128, 256, and 512 bits. Each circuit is locked 20 times with applicable key sizes (b18 only with 32 bits) resulting in a full dataset of 1217 benchmarks. The netlists are synthesized using the Nangate 45 nm Open Cell Library.

Recognition of Logic-locked Circuits: The HDC model is based on the original unlocked circuits. We examine whether this model can recognize the functionality of the locked circuits. In this work, an n-gram of $d = 8$ and a dimension of 2048

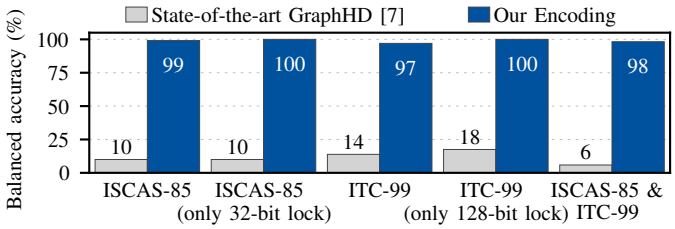

Figure 3. Our method encodes the original circuit and infers given a locked version. The previous GraphHD [7] encoding only reaches guessing accuracy.

are selected. Fig. 3 shows that the conventional GraphHD [7], even with 50k dimensions, does not exceed the guessing accuracy of 10 % for the ten classes in the ISCAS-85 dataset. GraphHD employs the PageRank algorithm to select the nodes' hypervectors, which is sensitive to changes in the graph's structure, leading to different hypervector-to-gate assignments in logic-locked circuits. In contrast, our encoding outperforms GraphHD achieving an accuracy of up to 99 % for the ISCAS-85 dataset. The key size directly impacts the number of inputs and gates in a circuit. Large key sizes result in more obfuscation, making the task of recognizing circuit types more challenging. If only one key size is used to lock the circuits, the balanced accuracy improves from 99 % (all key sizes) to 100 % for the ISCAS-85 dataset.

IV. Conclusion

This is the first work to demonstrate the effectiveness of brain-inspired HDC in circuit encoding and learning. We proposed an novel graph encoding method to recognize circuits. By demonstrating its capabilities, we are opening a new direction for circuit-level research.

References

[1] A. Burrello et al., "One-shot learning for ieeg seizure detection using end-to-end binary operations: Local binary patterns with hyperdimensional computing," in *BioCAS*, 2018, pp. 1–4.

[2] P. R. Genssler et al., "Brain-inspired computing for wafer map defect pattern classification," in *ITC*, 2021.

[3] F. R. Najafabadi et al., "Hyperdimensional computing for text classification," in *DATE*, 2016, pp. 1–1.

[4] P. R. Genssler et al., "Brain-inspired computing for circuit reliability characterization," IEEE Transactions on Computers, 2022.

[5] P. R. Genssler et al., "Modeling and predicting transistor aging under workload dependency using machine learning," IEEE TCAS-I, 2023.

[6] D. Kleyko et al., "A survey on hyperdimensional computing aka vector symbolic architectures, part ii: Applications, cognitive models, and challenges," ACM Computing Surveys, 2023.

[7] I. Nunes et al., "GraphHD: Efficient graph classification using hyperdimensional computing," in *DATE*, IEEE, 2022.

[8] P. Poduval et al., "Graphd: Graph-based hyperdimensional memorization for brain-like cognitive learning," Frontiers in Neuroscience, 2022.

[9] Y.-Y. Dai et al., "Circuit recognition with deep learning," in *HOST*, 2017, pp. 162–162.

[10] L. Alrahis et al., "GNN-RE: Graph neural networks for reverse engineering of gate-level netlists," TCAD, 2021.

[11] D. S. Lopera et al., "A survey of graph neural networks for electronic design automation," in *ACM/IEEE MLCAD*, 2021.

[12] L. Alrahis et al., "Graph neural networks: A powerful and versatile tool for advancing design, reliability, and security of ICs," in *ASP-DAC'23*.

[13] T. Bücher et al., "AppGNN: Approximation-aware functional reverse engineering using graph neural networks," in *ICCAD*, 2022.

[14] R. Yasaei et al., "GNN4IP: Graph neural network for hardware intellectual property piracy detection," in *DAC*, 2021.