Energy-Efficient Hyperdimensional Computing via PATH-Based Digital In-Memory Crossbar Architectures

Pranav Sinha[1], Muhammad Rashed[2], Nathaniel D Bastian[3], Alvaro Velasquez[4], Sumit Jha[5], Kris Campbell[6] and Sunny Raj[1]

[1]Oakland University {pranavsinha,raj}@oakland.edu, [2]University of Texas at Arlington [muhammad.rashed@uta.edu], [3]United States Military Academy at West Point [nathaniel.bastian@westpoint.edu], [4]University of Colorado Boulder [alvaro.velasquez@colorado.edu], [5]University of Florida [sumit.jha@ufl.edu], [6]Boise State University [kriscampbell@boisestate.edu]

ECI 2026

Abstract

Hyperdimensional Computing (HDC) represents data as high-dimensional binary hypervectors ($D \approx 10,000$) using bit-level primitives: binding (XOR), bundling (majority), and similarity search (Hamming distance). While HDC offers noise robustness and lightweight inference, its high dimensionality magnifies the energy cost of data movement in von Neumann systems. We present an in-memory computing framework that maps HDC operations onto memristive crossbar arrays using PATH-based digital logic for XOR-intensive binding stages and current-accumulation for majority-based bundling. Building upon a foundational flow-based computing paradigm, PATH-based logic synthesis frameworks, and Self-Directed Channel (SDC) memristor device, we demonstrate sub-100 nanojoule inference energy on binarized MNIST and Fashion-MNIST datasets in our computational experiments.

PATH-based IMC

In-memory computing addresses the energy and latency overhead of von Neumann architectures by performing computation directly within memory arrays, thereby minimizing data movement.

PATH-based computing [4] is a digital in-memory computing paradigm that evaluates Boolean logic using pre-programmed conductive paths in memristive crossbar arrays.

- Logic functions are written once by configuring the crossbar devices and evaluated using read-only operations.
- Large circuits can be decomposed into smaller subcircuits, enabling scalable and modular hardware realization.

Hyperdimensional Computing

Hyperdimensional Computing (HDC) is a brain-inspired computing paradigm that represents data using high-dimensional binary hypervectors (typically $D \approx 10,000$) [1]. Information is encoded and manipulated using simple, bit-level operations that are robust to noise and hardware variability.

- HDC relies on three core primitives: binding to associate vectors, bundling to aggregate information, and similarity search for classification.
- For image classification, pixel values are bound with their spatial coordinates and bundled across local patches to form an image hypervector, which is then compared against class prototypes during inference [2].

Challenge: While these operations are computationally lightweight, the high dimensionality makes data movement expensive on conventional architectures, motivating in-memory computing approaches for energy-efficient HDC execution.

Binding and Bundling

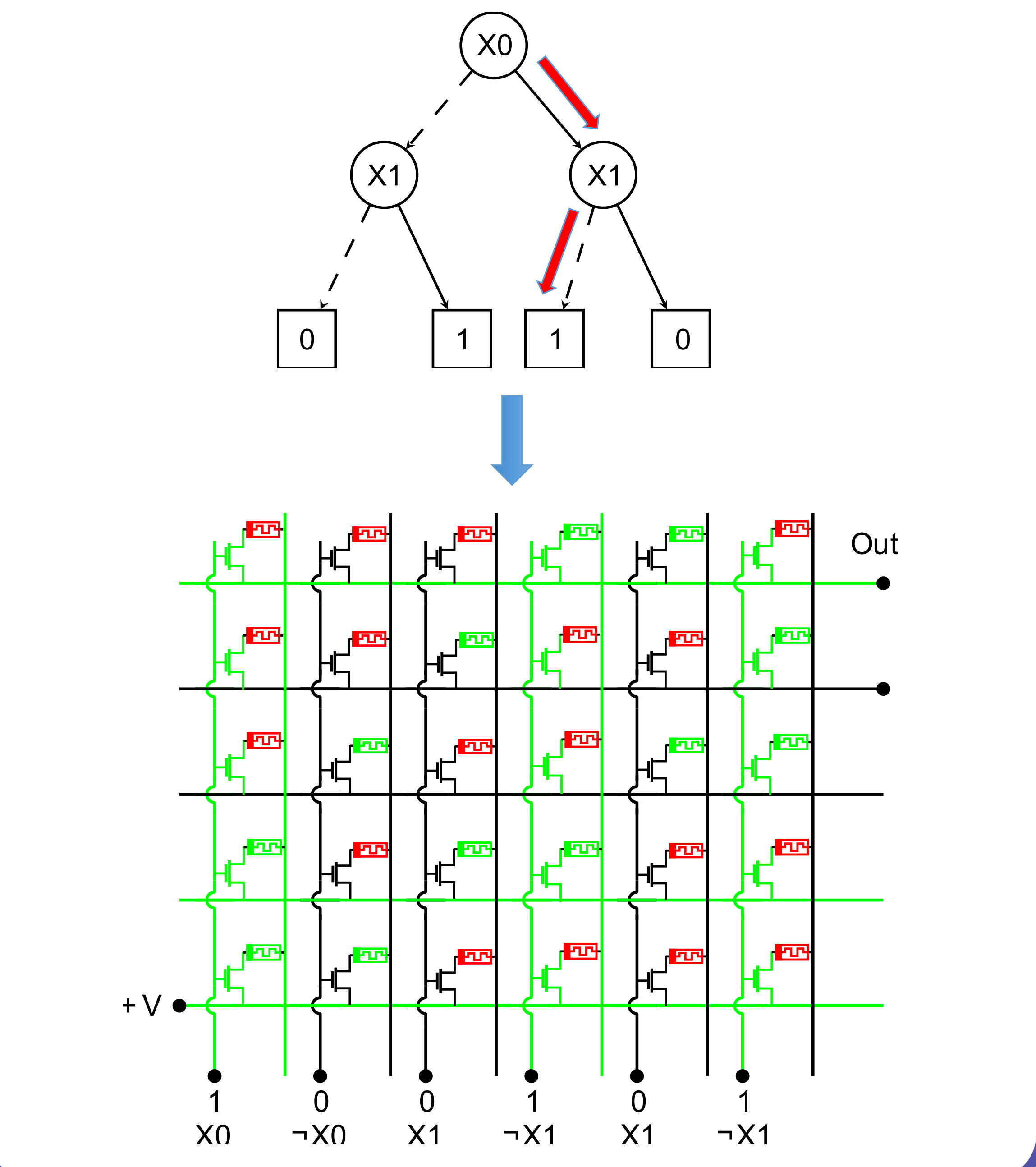
Binding ($\otimes$) associates symbols by combining their hypervectors using bitwise XOR, producing a result that is dissimilar to both inputs. Enables coordinate–value associations.

- Binding is implemented using PATH-style XOR logic mapped onto memristive crossbar arrays.
- A single-bit XOR is realized as a 5×6 crossbar tile.

Bundling ($\oplus$) aggregates multiple hypervectors into a single representation using a component-wise majority operation.

- Bundling leverages current-accumulation techniques from flow-based computing [3].
- Column currents represent the sum of '1's across the bundled vectors.
- ADC/comparator thresholds the accumulated current against $N/2$ to determine the majority value.

XOR using Crossbar

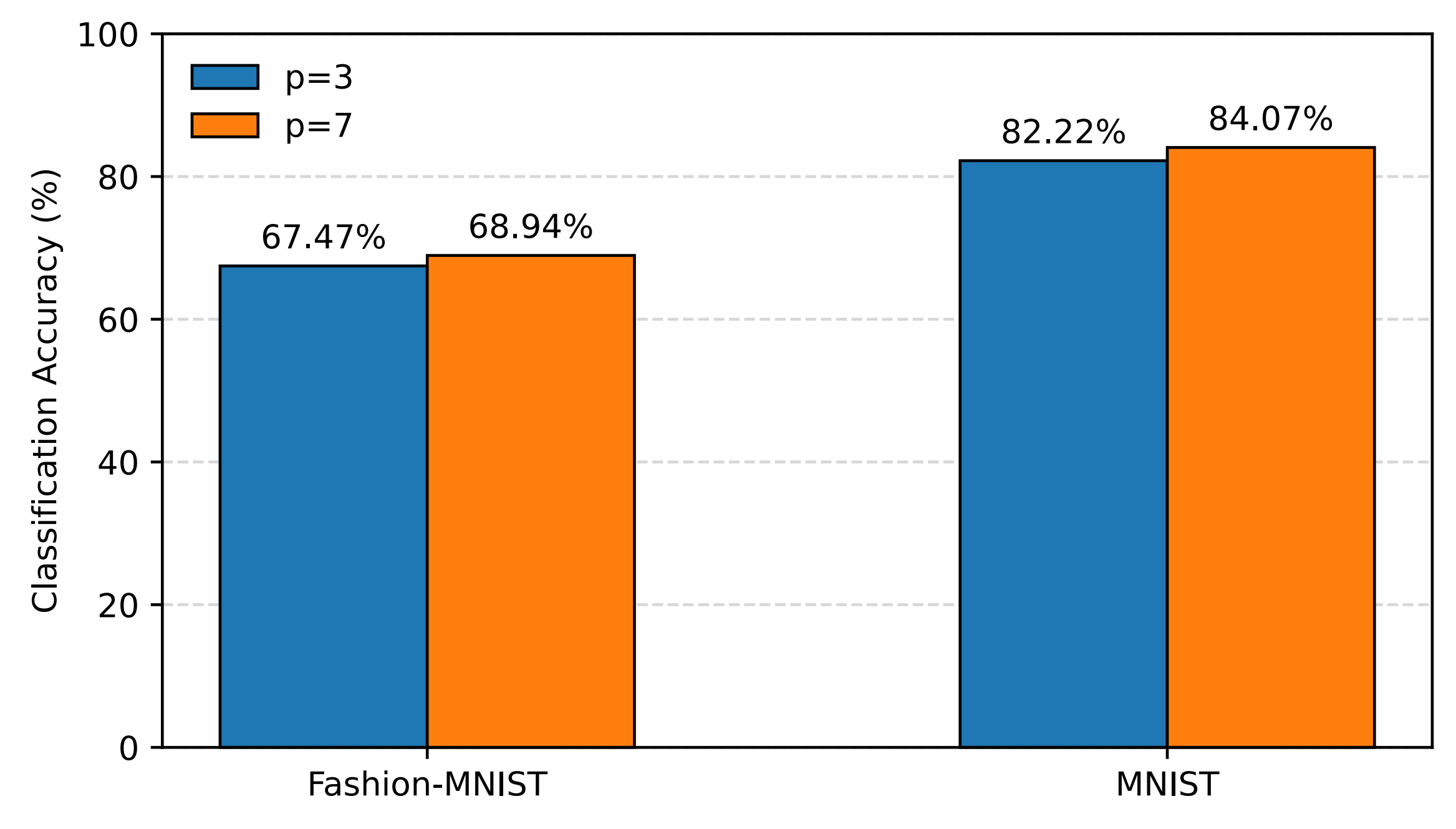


Inferencing

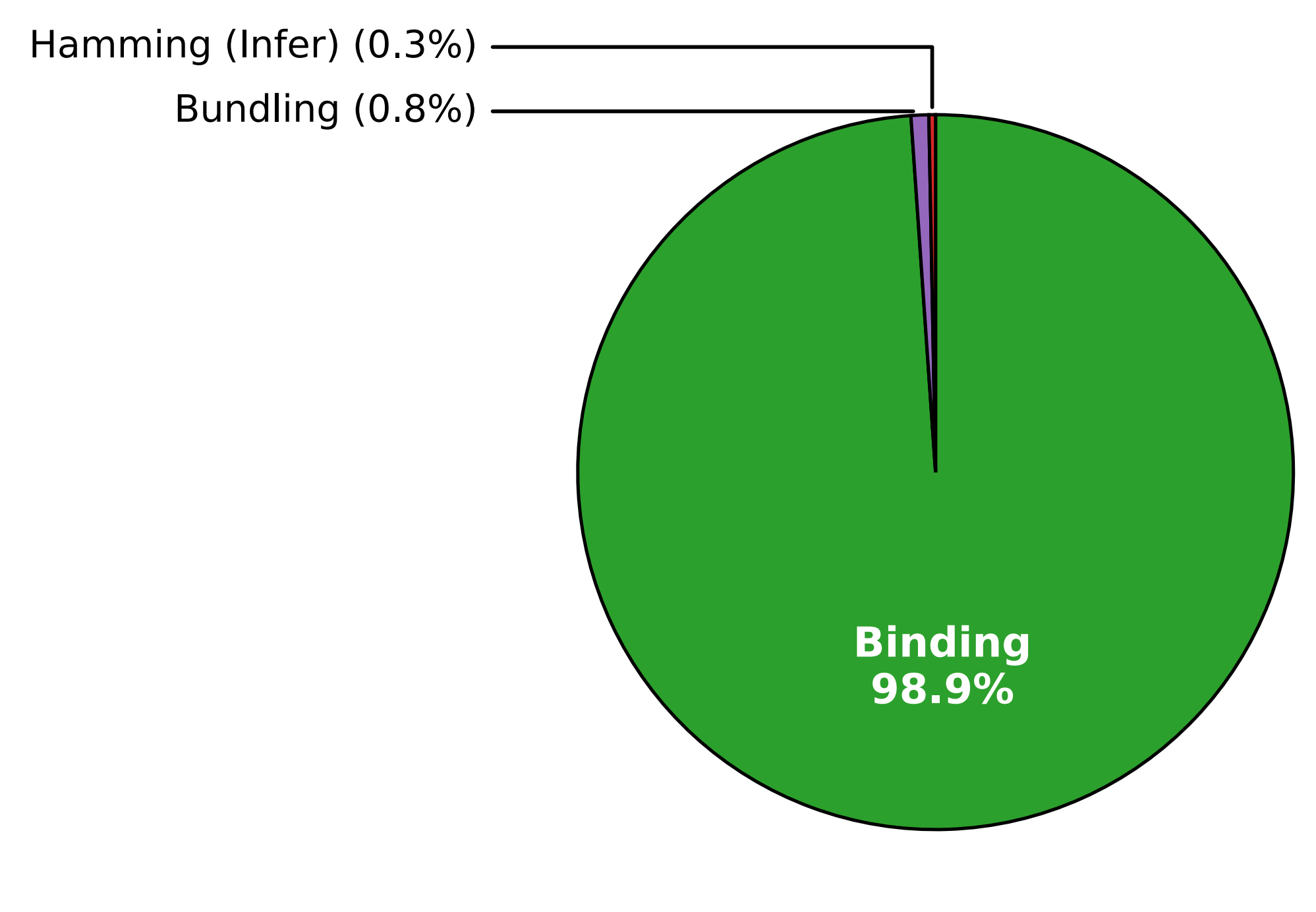
During inference, a query is encoded into an ImageHV using the same binding and bundling operations as training. Classification is performed by computing the **Hamming distance** between the query ImageHV and c class prototype hypervectors using XOR and current accumulation. The class with the minimum Hamming distance is selected as the predicted label. This similarity-based inference is highly robust to noise and hardware variability.

Experimental Results

- Evaluated on binarized MNIST and Fashion-MNIST with hypervector dimension $D = 10{,}000$.
- Increasing patch size from $p = 3$ to $p = 7$ consistently improves accuracy, 84.07% for MNIST.
- A representative MNIST query ($p = 3$) consumes a total of 94.88 nJ.
- Binding dominates energy consumption at approximately 99%, while bundling and inference together contribute less than 2% of total energy.
- Reveals an encode-heavy, search-light behavior favoring encode-once, query-many workloads.



a) Accuracy across Datasets and Patch Sizes



b) Energy Breakdown per Query

References

[1] Pentti Kanerva. Hyperdimensional Computing: An Introduction to Computing in Distributed Representation with High-Dimensional Random Vectors. *Cognitive Computation*, 1(2):139–159, June 2009.

[2] Laura Smets, Werner Van Leekwijck, Ing Jyh Tsang, and Steven Latré. An encoding framework for binarized images using hyperdimensional computing. *Front. Big Data*, 7:1371518, June 2024.

[3] Sunny Raj, Dwaipayan Chakraborty, and Sumit Kumar Jha. In-memory flow-based stochastic computing on memristor crossbars using bit-vector stochastic streams. In *2017 IEEE 17th International Conference on Nanotechnology (IEEE-NANO)*, 2017.

[4] Sven Thijssen, Sumit Kumar Jha, and Rickard Ewetz. PATH. In *Proceedings of the 59th ACM/IEEE Design Automation Conference*. ACM, July 2022.