DistriHD: A Memory Efficient Distributed Binary Hyperdimensional Computing Architecture for Image Classification

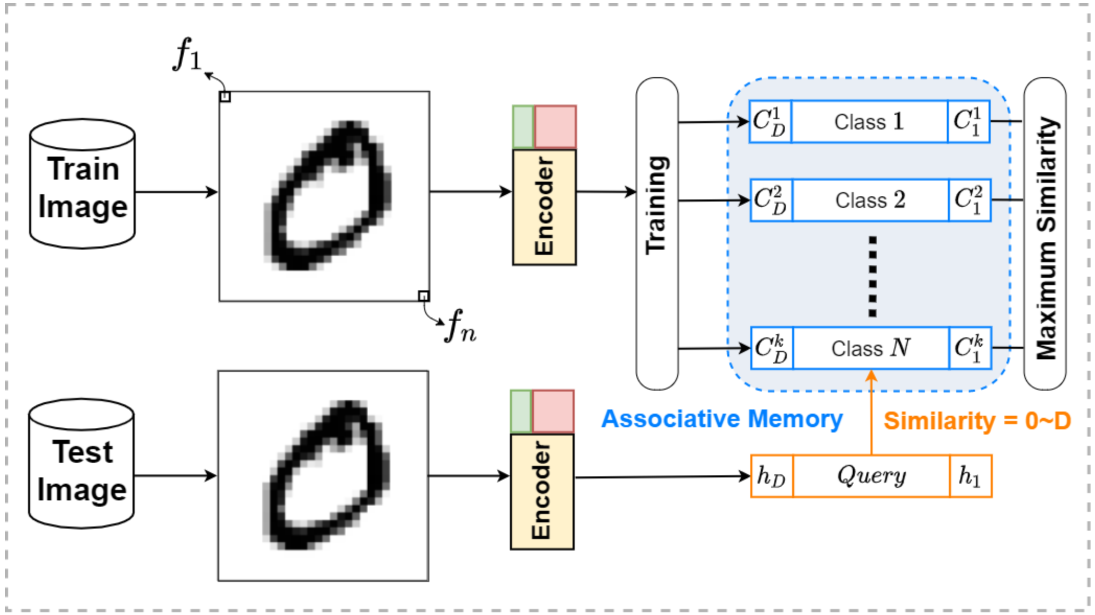
Dehua Liang	Jun Shiomi	Noriyuki Miura	Hiromitsu Awano
Osaka University	Osaka University	Osaka University	Kyoto University



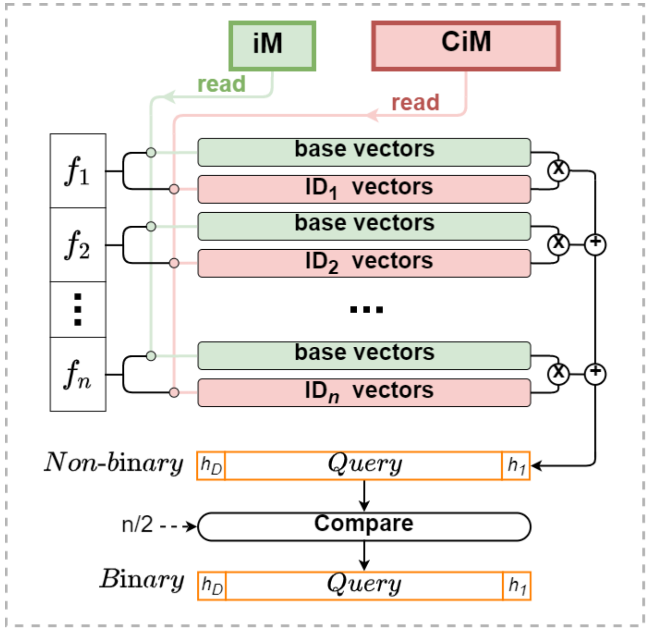

Outline

Background of Hyperdimensional Computing

- Demand for a more processing efficient model.
- Hyperdimensional (HD) Computing is a promising alternative method.
    - ✓Fast learning process.
    - ✓High robustness.
    - ✓Hardware friendly and light-weight.



Overview of General HD



General Encoding

Challenges and Contributions

- **Challenges in Previous Work:**
  - Use non-binarized hypervectors → High computational cost & memory requirement.
  - Retrain the model with iterative learning method → Tens of training iterations.
  - Compress the model base on dimension-wise sparsity → Feature-wise sparsity.
- **Main Contributions in *DistriHD*:**
  - **High memory efficiency**

    Support **binary** hypervectors and eliminate the costly CiM & iM.
  - **Fast Training**

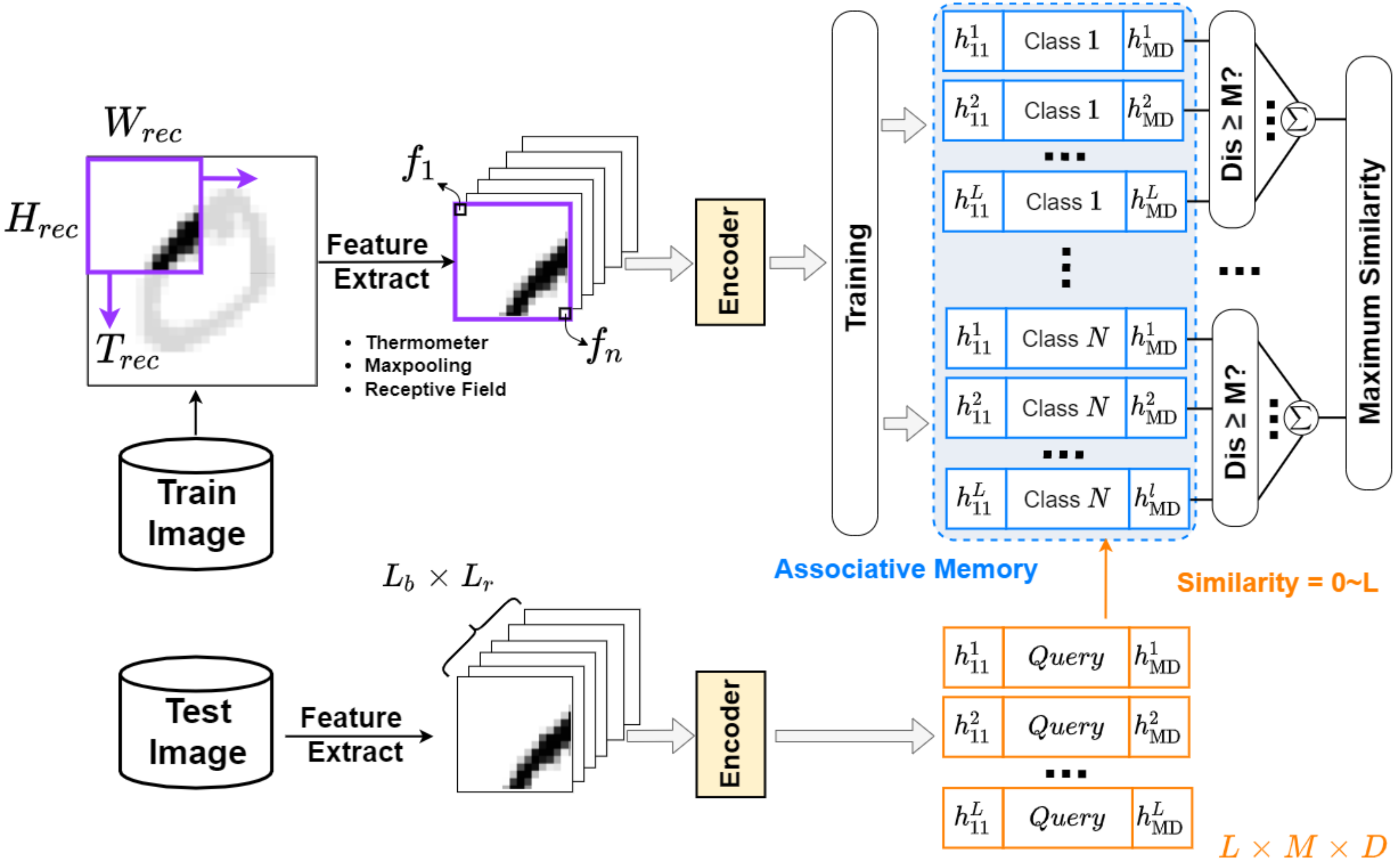
    Training process can be accomplished in **single-pass** way, while the baseline work [1] requires tens of iterations to retrain the model.
  - **Hardware friendly**

    27.6x reduction in inference memory cost without hurting the accuracy.

    9.9x and **28.8x** reduction in area and power, respectively.

[1] M. Imani et al., "A Binary Learning Framework for Hyperdimensional Computing," in DATE, 2019.

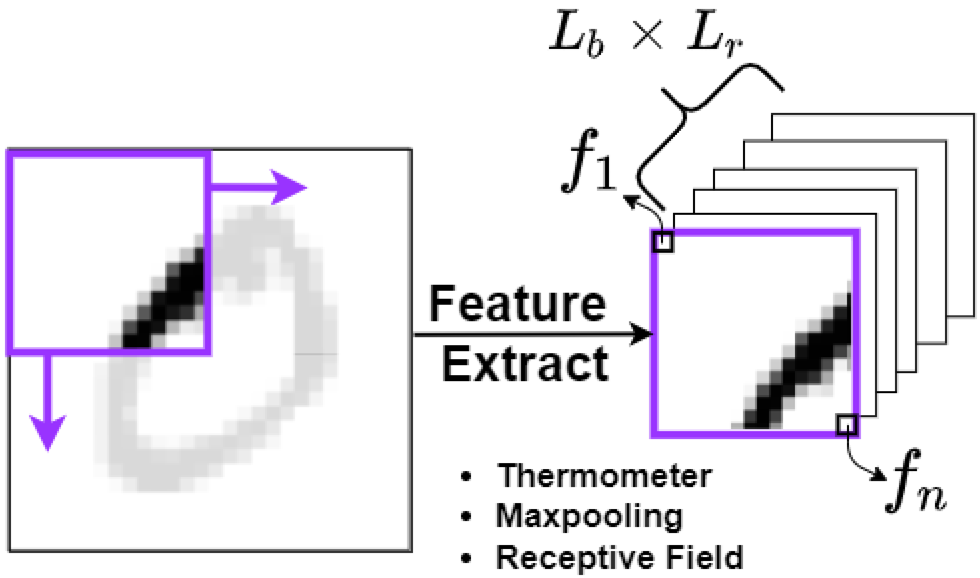
Proposed Method — Overview of *DistriHD* Architecture



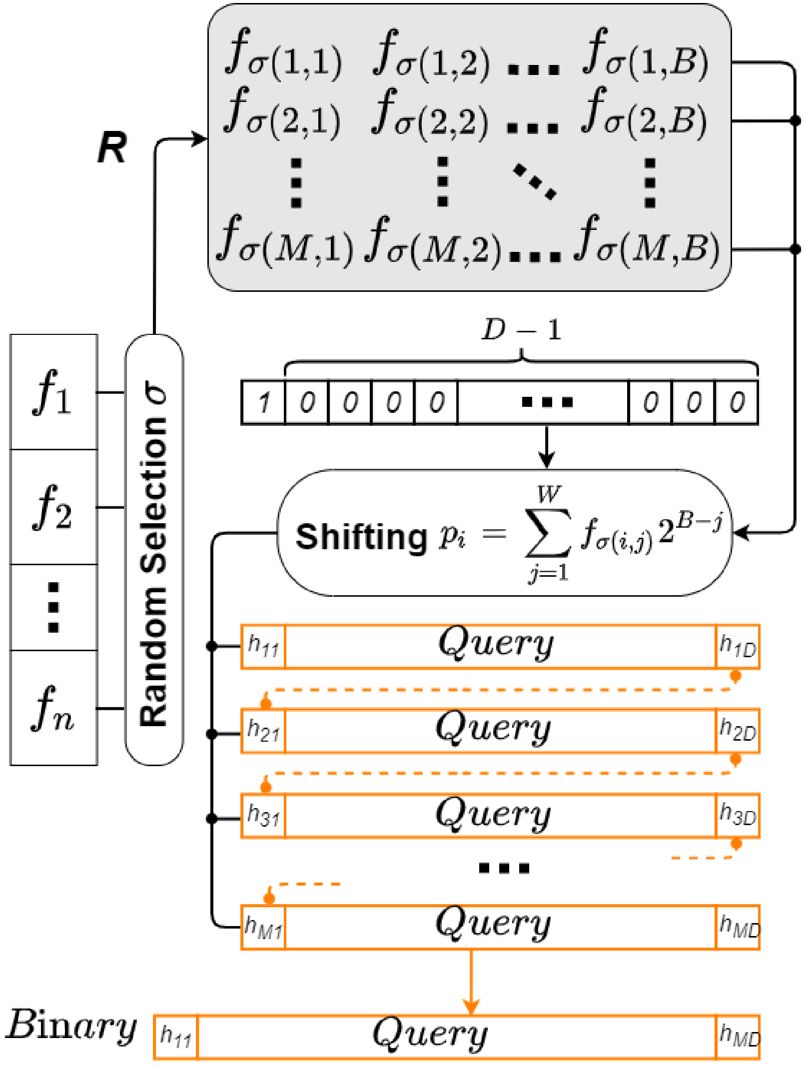
1. **Feature Extraction**
2. **Proposed Encoding**
3. **Training & Inference**
4. **Optimized Model Sparsification**

DistriHD Architecture(1/4) — Feature Extraction

- **Max-pooling**
- **Receptive Field**
  - Generate L_r distributed input patterns with non-binary elements.
- **Thermometer Binarization**
  - $L_r \rightarrow L_b \times L_r$ distributed input patterns with binary elements.



DistriHD Architecture(2/4) — Proposed Encoding



- Randomly select $M \times B$ elements from the binary input patterns $[f_1, f_2, \ldots, f_n]$ and construct the metric R
- For each row of R as an integer p_i in binary format, getting M binary vectors with D dimensions.
- Connect these M vectors to a single binary hypervector

  $H = [h_{11}, \ldots, h_{1D}, h_{21}, \ldots, h_{MD}]$

DistriHD Architecture(2/4) — Proposed Encoding

✓Eliminate the costly CiM and iM blocks.

- In traditional encoding of HD computing, according to the feature index and value, read the index hypervector and base hypervector from the pre-stored CiM and iM block.
- Example of Inference Memory Occupations in work [1].

	MNIST	ISOLET	UCIHAR	FACE
CiM	94.92%	91.41%	92.73%	94.70%
iM	3.87%	4.74%	5.29%	4.98%
AM	1.21%	3.85%	1.98%	0.31%

Expensive!

[1] M. Imani et al., "A Binary Learning Framework for Hyperdimensional Computing," in DATE, 2019.

DistriHD Architecture(3/4) — Training & Inference

- **Single-pass Training:** The distributed hypervectors from the same class are accumulated in the distributed class hypervectors.

- **Inference:** Check the similarity of the *L* distributed hypervectors.

Mathematically, the similarity metric in the traditional HD [1] is Hamming distance δ :

$$\sum_{l=0}^{L} \delta(H^l, C^l)$$

Similarity metric in the *DistriHD* is as follow:

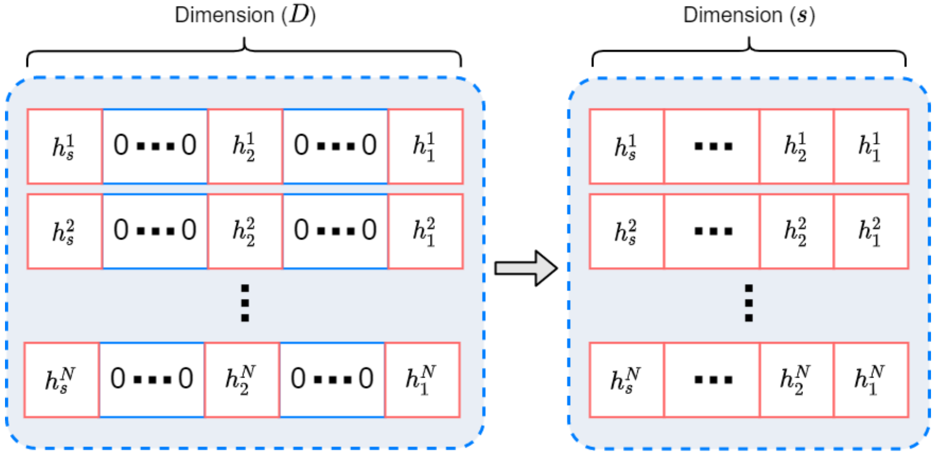
$$\sum_{l=0}^{L} \mathrm{sgn}[\, \delta\left(H^l \& C^l, C^l\right) - M]$$

[1] M. Imani et al., "A Binary Learning Framework for Hyperdimensional Computing," in DATE, 2019.

DistriHD Architecture(4/4) — Optimized Model Sparsification

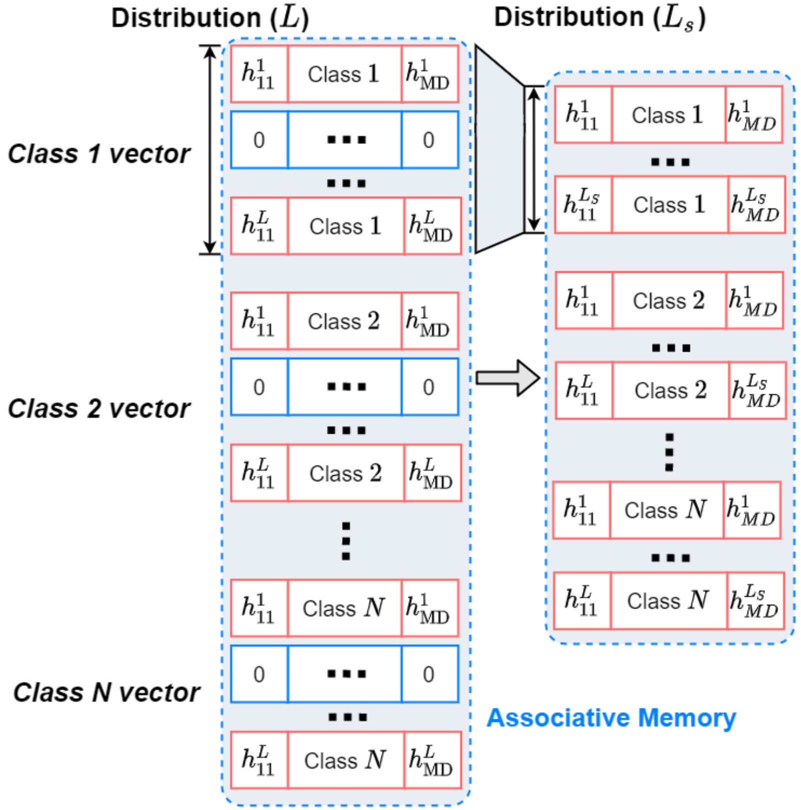
- Traditional HD

Dimension-wise Sparsification[1]



- *DistriHD*

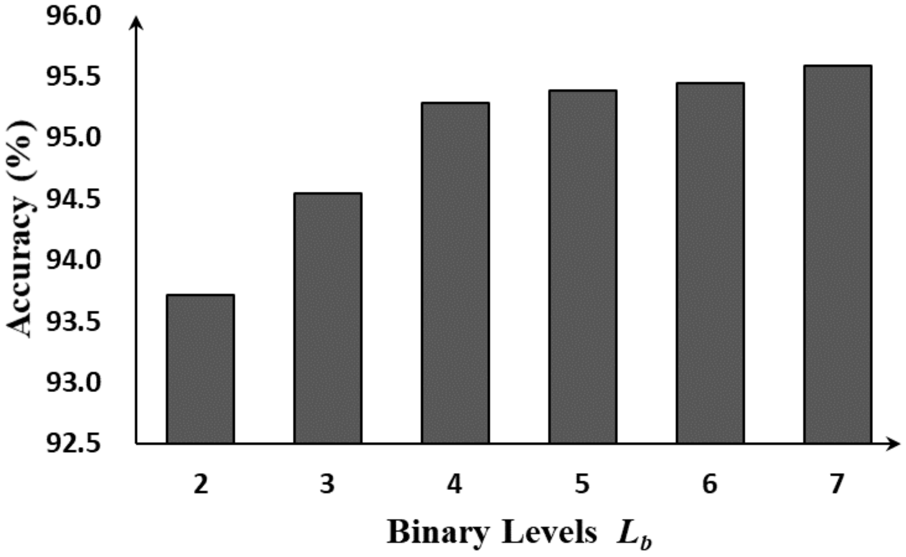
Additional Feature-wise Sparsification

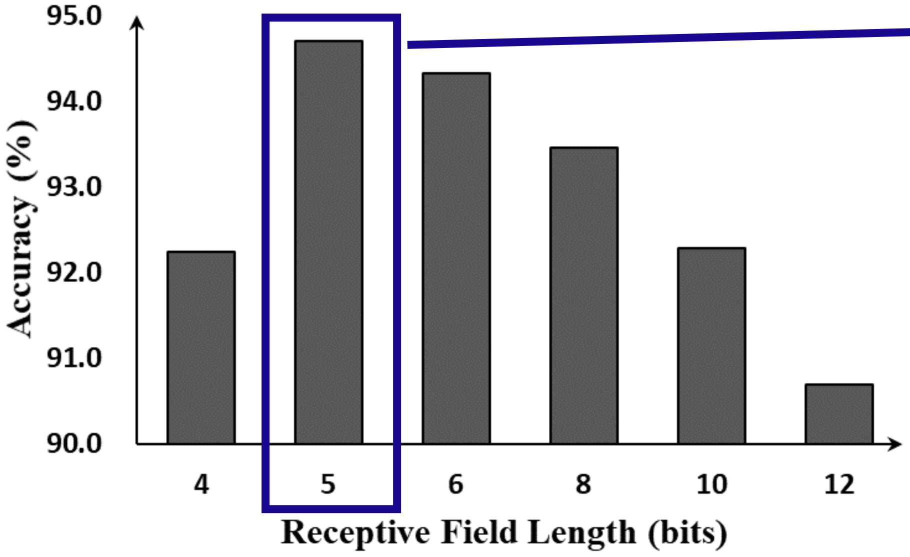


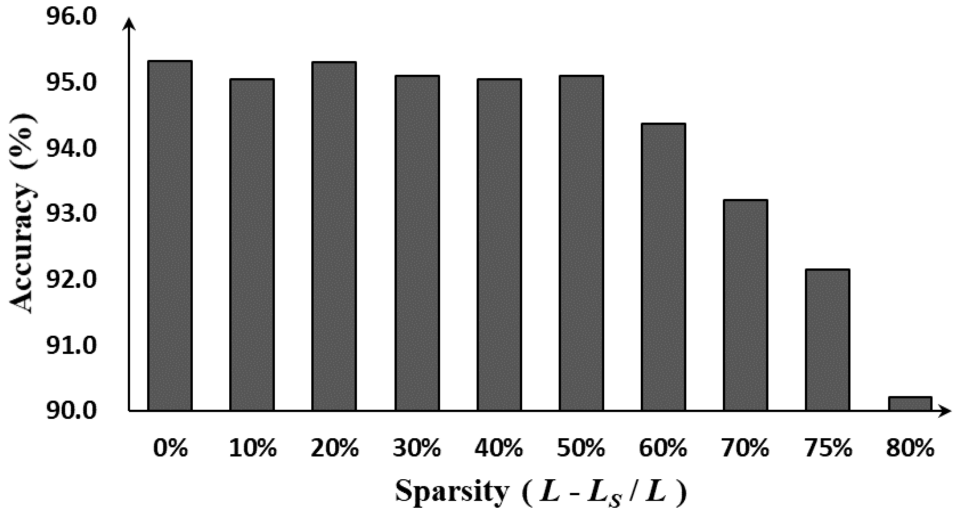
[1] M. Imani et al., "SparseHD: Algorithm-Hardware Co-Optimization for Efficient High-Dimensional Computing," in FCCM, 2019.

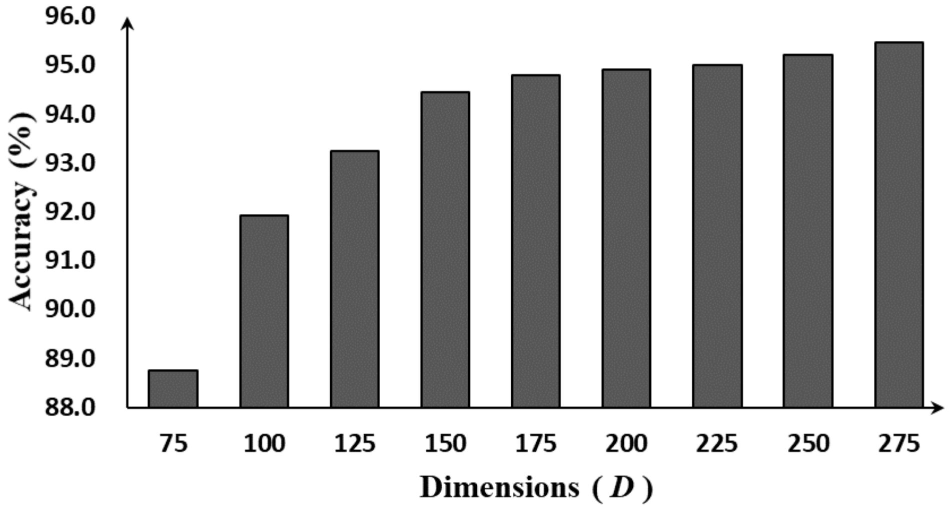
Experimental Results(1/3) —
Impact of Parameters in *DistriHD*

Parameter Tuning









Experimental Results(2/3) — Memory Cost Reduction

- The memory cost reduction mainly comes from the elimination of CiM and iM.

1. including iM:

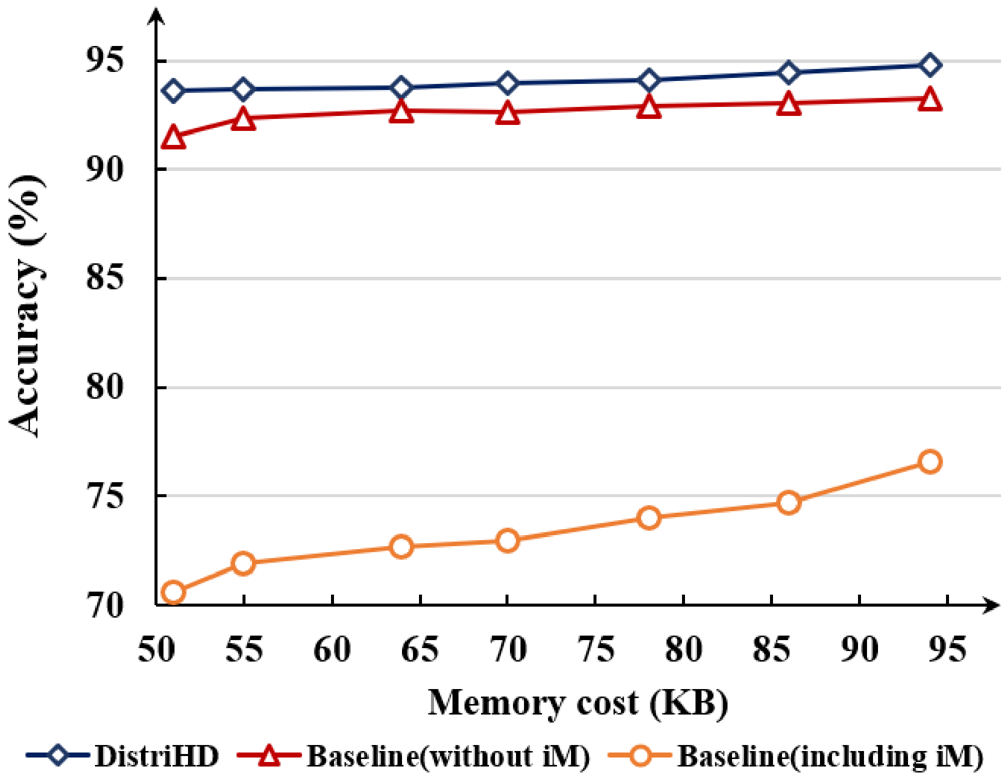
(Baseline[1] Memory Cost = CiM+iM+AM)

Achieve **27.6x** reduction in inference memory cost without hurting the accuracy.

2. without iM:

(Baseline Memory Cost = AM)

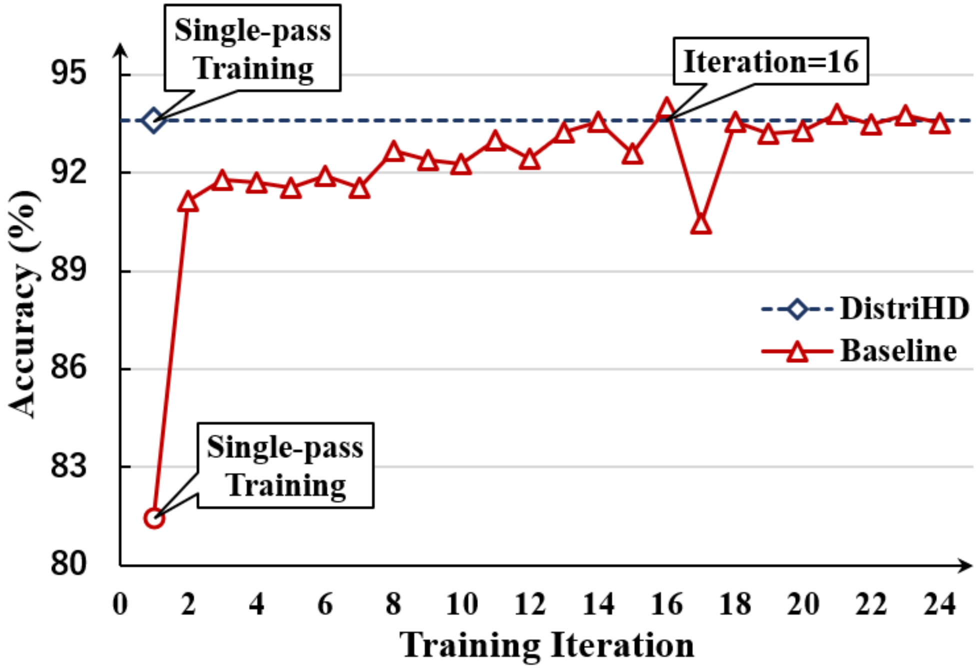
Achieve a similar accuracy.



[1] M. Imani et al., "A Binary Learning Framework for Hyperdimensional Computing," in DATE, 2019.

Experimental Results(3/3) — Training Iteration & Hardware Cost

- Efficient **Single-Pass** Training:
- (*DistriHD* vs 50*Baseline[1])



- Hardware Comparison:
  - **9.9x** and **28.8x** reduction in area and power, respectively.

[1] M. Imani et al., "A Binary Learning Framework for Hyperdimensional Computing," in DATE, 2019.

Conclusion

- Utilize **binary** hypervectors in both training and inference phase.
- Successfully eliminate the costly CiM and iM in the encoding procedure, resulting in **27.6x inference memory reduction** without hurting the accuracy.
- Training process can be accomplished in **single-pass** way.
- **9.9x** and **28.8x** reduction in **area** and **power**, respectively.