FedHD: Federated Learning with Hyperdimensional Computing

Quanling Zhao, Kai Lee, Jeffrey Liu, Muhammad Huzaifa, Xiaofan Yu, Tajana Rosing
Department of Computer Science and Engineering, University of California San Diego
La Jolla, CA 92093, USA
{quzhao,kal030,jsliu,mhuzaifa,x1yu,tajana}@ucsd.edu

ABSTRACT

Federated Learning (FL) is a widely adopted distributed learning paradigm for to its privacy-preserving and collaborative nature. In FL, each client trains and sends a local model to the central cloud for aggregation. However, FL systems using neural network (NN) models are expensive to deploy on constrained edge devices regarding computation and communication. In this demo, we present FedHD, a FL system using Hyperdimensional Computing (HDC). In contrast to NN, HDC is a brain-inspired and lightweight computing paradigm using high-dimensional vectors and associative memory. Our measurements indicate that FedHD is 3.2×, 3.2×, 5× better on performance, energy and communication efficiency respectively compared to NN-based FL systems whilst maintaining similar accuracy to the state of the art. Our code is available on GitHub[1].

ACM Reference Format:
Quanling Zhao, Kai Lee, Jeffrey Liu, Muhammad Huzaifa, Xiaofan Yu, Tajana Rosing. 2022. FedHD: Federated Learning with Hyperdimensional Computing. In *Proceedings of ACM Conference (Conference'17)*. ACM, New York, NY, USA, 3 pages. https://doi.org/10.1145/nnnnnnn.nnnnnnn

1 INTRODUCTION

Federated Learning (FL) is a distributed paradigm which trains models collaboratively without sharing data. FL boasts widespread popularity in many applications such as healthcare [5], smart cities [12], and self-driving vehicles [2]. Traditional FL systems adopt neural network (NN) based models, which are expensive to compute and communicate. Table 1 shows that NN models take 350 seconds per round of training on average with a model size of 8MB. Consequently, resource limitations present a major challenge when deploying FL on edge devices [18]. Unreliable wireless channels may also add noise during transmission and degrade model accuracy [3].

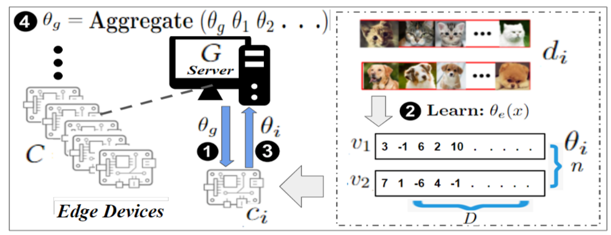

Figure 1: FedHD workflow in 4 stages. (1) Server sends global hypervectors to each client. (2) Clients train local class hypervectors using encoded local data. (3) Models are sent back to the server. (4) Server aggregates received class hypervectors.

Hyperdimensional computing (HDC) is a lightweight computing paradigm that encodes data into hypervectors (high dimensional vectors > 1000 bit) [13]. Learning is performed through simple arithmetic operations (addition, multiplication, nearest neighbor search), reducing power and memory usage [17] [9] [20]. HDC is also robust against noise due to its high dimensionality. Previous works on HDC primarily focus on non-FL settings [19] [8] [17]. In this demo, we present FedHD, an implementation of FL using HDC for low-power devices. FedHD is lightweight in computation and communication, and robust against unreliable communication.

Dataset	Time/Round (Second)	Energy/Round (Joule)	Model Size (Megabyte)
MNIST [15]	**73**/276	**350**/1325	**1.99**/9.95
FMNIST [21]	**71**/395	**341**/1896	**1.90**/9.91
HAR [4]	**68**/121	**326**/581	**1.99**/1.67
CIFAR10 [14]	**388**/591	**1862**/2837	**1.90**/12.3

Table 1: Measurement results: HDC/Baseline NN

2 METHOD AND IMPLEMENTATION

Figure 1 shows the FedHD workflow. Each class is represented as a class hypervector that encodes generic class features. The system consists of a server G and a set of edge devices $C = \{c_1, c_2 ...\}$. The hypervector dimensionality and the number of classes are D, n, respectively. Each client c_i

Conference'17, July 2017, Washington, DC, USA

ACM ISBN 978-x-xxxx-xxxx-x/YY/MM…$15.00
https://doi.org/10.1145/nnnnnnn.nnnnnnn

[1] https://github.com/QuanlingZhao/FedHD

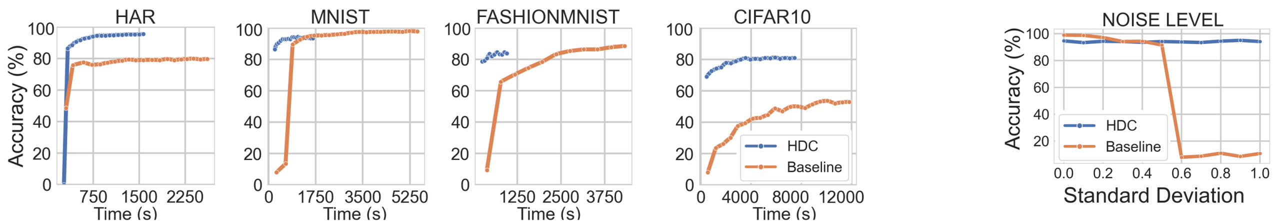

Figure 2: Left: Measured accuracy over wall-clock time on various datasets. Right: Accuracy under various noise levels.

holds a θ_i, a HD classifier defined by a set of class hypervectors $\{v_1 ... v_n\}$ and a similarity checker. d_i denotes local training samples. Likewise, $G = (\theta_g, d_g)$ denotes the servers. Both the server and the clients have access to an HDC encoder, θ_e.

2.1 HDC Learner

Learning in HDC consists of encoding raw data into hypervectors, adding hypervectors that belong to the same class together, and performing a few rounds of retraining if needed. Classification is done by comparing a query hypervector to the stored class hypervectors.

HDC Encoding: We use a random binary projection encoder which performs well on many datasets [6]. For any $(x, y) \in d_i$, a random projection encoder $\theta_e(x) \longrightarrow \{0, 1\}^D$ computes sample hypervector H by $\theta_e(x) = \beta(\mathbf{E}x)$, where $\mathbf{E} \in M^{D \times |x|}$ is a randomly generated binary matrix and β is an element-wise sign function.

HDC Training: Initial class hypervectors are generated by summing the sample hypervectors from the same class: $v_j = \sum_{(x,y) \in d_i | y=j} \theta_e(x)$. For all subsequent rounds, local class hypervectors are re-trained with local samples as shown below. The process iterates over all n classes and across multiple rounds, such that the class hypervectors gradually converge to a global optimal.

$$\forall (x, y) \in d_i | \theta_i(\theta_e(x)) \neq y \begin{cases} v_j - \theta_e(x) & \forall j \neq y \\ \textbf{Negative reinforce incorrect } \boldsymbol{v} & \\ v_j + \theta_e(x) & j = y \\ \textbf{Reinforce correct } \boldsymbol{v} & \end{cases}$$

HDC Classification: Each class in the HDC model is represented as a class hypervector $\{v_1 ... v_n\}$. Classification is done by checking the cosine similarity between the encoded sample and each class hypervector, then choosing the class with the greatest similarity: $\arg\max_{j=1}^{n} \cos(H, v_j) = \frac{H \cdot v_j}{\|H\| \|v_j\|}$.

2.2 FedHD

HDC Aggregation: In FL, locally trained models are exchanged each round. After the server collects all locally trained HDC models, a global HDC model θ_g can be aggregated: $\theta_g = \frac{\theta'_g + \sum_i^{|C|} \theta_i}{|C|+1}$. The global model from the previous round θ'_g is also included in aggregation process as a stabilizing factor to prevent an abrupt change in class hypervectors, which prevents catastrophic model failure.

FedHD Implementation: We implemented FedHD using FedML [10], an open-source FL framework that allows us to add and deploy HDC components on IoT devices.

3 DEMONSTRATION

Our demo uses Raspberry Pis [1] and Kubernetes cluster as clients with D=10000 and 500 local samples per client. A desktop is used as the server for model aggregation. A state of the art NN with FedAvg FL algorithm [16] is used as a baseline. The experimental setup is shown in Table 2.

Dataset	Client #	Method	Baseline
MNIST [15]	30	HDC	CNN w/ 2HL[2]
FMNIST [21]	30	HDC	CNN w/ 2HL[2]
HAR [4]	30	HDC	2FCL[3]
CIFAR10 [14]	7	SimCLR [7]+HDC	ResNet-18 [11]

Table 2: Experimental Setup. For complex image datasets, HDC requires a feature extractor trained by SimCLR [7].

Accuracy & Efficiency: Experimental results are shown in Fig. 2 and Table 1. FedHD achieves comparable or higher accuracy across all datasets while being 3.2x faster than the baseline. While the size of NN models grow dramatically with task complexity, the HDC model maintains communication efficiency by scaling linearly with the number of classes.

Robustness: We evaluate FedHD's robustness by directly applying additive Gaussian noise to both the HDC model [6] and the NN baseline while increasing noise standard deviation σ. Fig. 2b shows model performance of our method compared with the baseline on MNIST with varying levels of noise. When $\sigma > 0.5$, FedHD's accuracy is unaffected whereas the baseline suffers from catastrophic failure.

4 CONCLUSION

In this demo, we proposed FedHD, an efficient and robust FL system using HDC. Our results address two bottlenecks in current FL systems by greatly reducing computation and communication overhead and bolstering the robustness to remain nearly unaffected against unreliable communication.

5 ACKNOWLEDGEMENTS

This work was supported by National Science Foundation under Grants #2003279, #1826967, #2100237, #2112167, #1911095, and #2112665. The work was advised and completed as part of the Early Research Scholars Program at UC San Diego.

[2]HL: Hidden Layer.
[3]FCL: Fully Connected Layer.

REFERENCES

[1] Raspberry Pi 4B. https://www.raspberrypi.com/products/raspberry-pi-4-model-b/, 2020. [Online].

[2] Irfan Ahmad and Karunakar Pothuganti. Design & implementation of real time autonomous car by using image processing & iot. In *2020 Third International Conference on Smart Systems and Inventive Technology (ICSSIT)*, pages 107–113. IEEE, 2020.

[3] Fan Ang, Li Chen, Nan Zhao, Yunfei Chen, Weidong Wang, and F Richard Yu. Robust federated learning with noisy communication. *IEEE Transactions on Communications*, 68(6):3452–3464, 2020.

[4] Davide Anguita, Alessandro Ghio, Luca Oneto, Xavier Parra Perez, and Jorge Luis Reyes Ortiz. A public domain dataset for human activity recognition using smartphones. In *Proceedings of the 21th international European symposium on artificial neural networks, computational intelligence and machine learning*, pages 437–442, 2013.

[5] Sándor Beniczky, Philippa Karoly, Ewan Nurse, Philippe Ryvlin, and Mark Cook. Machine learning and wearable devices of the future. *Epilepsia*, 62:S116–S124, 2021.

[6] Rishikanth Chandrasekaran, Kazim Ergun, Jihyun Lee, Dhanush Nanjunda, Jaeyoung Kang, and Tajana Rosing. Fhdnn: Communication efficient and robust federated learning for aiot networks. In *Proceedings of the 59th Annual Design Automation Conference 2022*, 2022.

[7] Ting Chen, Simon Kornblith, Mohammad Norouzi, and Geoffrey Hinton. A simple framework for contrastive learning of visual representations. In *International conference on machine learning*, pages 1597–1607. PMLR, 2020.

[8] Arpan Dutta, Saransh Gupta, Behnam Khaleghi, Rishikanth Chandrasekaran, Weihong Xu, and Tajana Rosing. Hdnn-pim: Efficient in memory design of hyperdimensional computing with feature extraction. In *Proceedings of the Great Lakes Symposium on VLSI 2022*, pages 281–286, 2022.

[9] Eman Hassan, Yasmin Halawani, Baker Mohammad, and Hani Saleh. Hyper-dimensional computing challenges and opportunities for ai applications. *IEEE Access*, 2021.

[10] Chaoyang He, Songze Li, Jinhyun So, Xiao Zeng, Mi Zhang, Hongyi Wang, Xiaoyang Wang, Praneeth Vepakomma, Abhishek Singh, Hang Qiu, et al. Fedml: A research library and benchmark for federated machine learning. *arXiv preprint arXiv:2007.13518*, 2020.

[11] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 770–778, 2016.

[12] Nabaa Ali Jasim, Haider TH, and Salim AL Rikabi. Design and implementation of smart city applications based on the internet of things. *International Journal of Interactive Mobile Technologies*, 15(13), 2021.

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

[14] Alex Krizhevsky, Geoffrey Hinton, et al. Learning multiple layers of features from tiny images. 2009.

[15] Yann LeCun, Léon Bottou, Yoshua Bengio, and Patrick Haffner. Gradient-based learning applied to document recognition. *Proceedings of the IEEE*, 86(11):2278–2324, 1998.

[16] Brendan McMahan, Eider Moore, Daniel Ramage, Seth Hampson, and Blaise Aguera y Arcas. Communication-efficient learning of deep networks from decentralized data. In *Artificial intelligence and statistics*, pages 1273–1282. PMLR, 2017.

[17] Justin Morris, Kazim Ergun, Behnam Khaleghi, Mohsen Imani, Baris Aksanli, and Tajana Rosing. Hydrea: Towards more robust and efficient machine learning systems with hyperdimensional computing. In *2021 Design, Automation & Test in Europe Conference & Exhibition (DATE)*, pages 723–728. IEEE, 2021.

[18] Solmaz Niknam, Harpreet S Dhillon, and Jeffrey H Reed. Federated learning for wireless communications: Motivation, opportunities, and challenges. *IEEE Communications Magazine*, 58(6):46–51, 2020.

[19] Abbas Rahimi, Pentti Kanerva, and Jan M Rabaey. A robust and energy-efficient classifier using brain-inspired hyperdimensional computing. In *Proceedings of the 2016 international symposium on low power electronics and design*, pages 64–69, 2016.

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

[21] Han Xiao, Kashif Rasul, and Roland Vollgraf. Fashion-mnist: a novel image dataset for benchmarking machine learning algorithms. *arXiv preprint arXiv:1708.07747*, 2017.