FedHDC-KDS: A Hyperdimensional Computing Framework for Efficient Personalized Federated Learning

Daoyong Chen
FEECS, Ningbo University
Ningbo, China
chendaoyong0128@163.com

Haiming Chen
FEECS, Ningbo University
Ningbo, China
chenhaiming@nbu.edu.cn

Ruijin Zhuang
FEECS, Ningbo University
Ningbo, China
17359899601@163.com

***Abstract*—Federated Learning (FL) enables collaborative model training without sharing raw data, but its performance is severely hindered by statistical heterogeneity, which causes client drift and inconsistent local updates. Personalized Federated Learning (PFL) addresses this issue by tailoring models to individual clients, yet many existing methods introduce substantial computation, communication, or storage overhead. This paper proposes FedHDC-KDS, a lightweight PFL framework that leverages Hyperdimensional Computing (HDC) for efficient model representation. FedHDC-KDS incorporates two complementary personalization components: (i)a similarity-aware server aggregation that produces personalized teacher models via weighted averaging of top-k similar client models, and (ii) a gradient-free client-side adaptation scheme that combines error-correction updates with knowledge distillation for fast on-device learning. Experiments on standard FL benchmarks show that FedHDC-KDS achieves competitive personalized accuracy under heterogeneous settings while reducing model storage by 29× compared with DNN-based PFL baselines.**

***Index Terms*—Federated Learning, Hyperdimensional Computing, Personalized Federated Learning.**

I. Introduction

The rapid proliferation of edge devices has resulted in vast amounts of data being generated and stored locally at the network edge. This paradigm fundamentally challenges traditional centralized machine learning, where collecting all data at a central server is increasingly impractical due to privacy risks, security concerns, and prohibitive communication costs [1], [2]. Federated Learning (FL) has emerged as a compelling solution, enabling collaborative model training across distributed clients without sharing sensitive raw data [3]. By exchanging only model updates with a central server, FL mitigates privacy risks and reduces communication overhead, aligning with modern data protection regulations.

However, a fundamental hurdle for FL in real-world edge environments is statistical heterogeneity. Client data is often non-independent and identically distributed (non-IID), which inherently causes "client drift", a phenomenon where local models optimize towards divergent local optima [4]. This leads to unstable convergence and severely degrades the performance of a single global model, as seen in the classical FedAvg algorithm.

To mitigate this challenge, Personalized Federated Learning (PFL) aims to learn customized models that better adapt to each client's unique and highly heterogeneous local data distribution [5], [6]. Although various PFL approaches—such as model regularization, parameter decoupling, and meta-learning—have demonstrated strong potential, they predominantly rely on deep neural networks (DNNs). This architectural dependence leads to computationally and communication-intensive training pipelines and substantial storage demands, ultimately failing to resolve the inherent trade-off between performance and efficiency when deployed on resource-constrained edge devices [7].

HDC offers a paradigm shift, providing an energy-efficient, brain-inspired computational model [8]. HDC represents information as high-dimensional random vectors (hypervectors) and performs learning through simple, parallelizable algebraic operations. This approach has demonstrated robust performance and interpretability, while offering significantly higher computational and memory efficiency than DNNs [9], [10].However, existing HDC-based FL methods often struggle with statistical heterogeneity, leading to suboptimal personalization.

Motivated by these insights and the pressing need for truly lightweight yet highly effective PFL, we propose FedHDC-KDS, a novel personalized federated learning framework that seamlessly integrates the extreme efficiency of HDC with powerful personalized learning strategies. The main contributions of this paper are summarized as follows:

- We introduce FedHDC-KDS, a unified framework that seamlessly integrates HDC with PFL, achieving lightweight personalization with substantially reduced computation and communication overhead.
- We design a similarity-aware server-side aggregation strategy and a gradient-free client adaptation mechanism that combines error-correction updates with knowledge distillation, enabling expressive and robust personalized models under severe non-IID conditions.
- Through extensive experiments, we show that FedHDC-KDS matches or outperforms DNN-based PFL baselines while being much more efficient, reducing model storage from 2.32 MB to 78.13 KB (29× smaller).

II. Related Work And Background

A. Personalized Federated Learning (PFL)

The objective of PFL is to mitigate statistical heterogeneity by producing client-specific models [6]. Existing approaches can be broadly categorized according to how they balance global knowledge sharing and local adaptation. Heterogeneity-aware global methods such as FedProx [11] introduce proximal constraints to stabilize optimization; however, they still produce a single shared global model and therefore provide limited personalization.

To improve client specificity, parameter decoupling methods, including FedPer [12] and FedRep [13], separate global representations from client-specific layers. This design enhances personalization while maintaining moderate communication overhead. Meta-learning-based approaches, such as Per-FedAvg [5], instead learn a model initialization that supports rapid local adaptation. Despite their effectiveness, these methods typically rely on backpropagation within deep neural networks (DNNs), which introduces substantial computational and communication overhead and limits deployment on resource-constrained edge devices.

Another line of research employs knowledge distillation to achieve personalization without fully sharing model parameters. Methods such as FedDF [14], FedMD [15], and DS-FL [16] transfer knowledge from client models into shared or client-specific models through soft predictions. These approaches decouple personalization from model architectures and reduce communication overhead by exchanging logits rather than full model parameters. However, distillation-based PFL commonly requires auxiliary public data or synthetic data generation, both of which are difficult to obtain in privacy-sensitive or resource-constrained environments.

B. Hyperdimensional Computing (HDC)

Hyperdimensional Computing (HDC), also known as Vector Symbolic Architectures (VSA), represents data using high-dimensional hypervectors (HVs) [8], [17]. Due to the concentration of measure in high-dimensional spaces, randomly generated HVs are nearly orthogonal, which provides strong robustness to noise. HDC relies on a small set of algebraic operators—*Bundling* ($+$), *Binding* ($\times$), and *Permutation* (ρ) [9]. Bundling aggregates multiple hypervectors into a single representation, Binding performs element-wise interactions to encode associations, and Permutation applies cyclic shifts to capture structural or sequential information.

The standard HDC pipeline begins with an encoding stage that maps inputs from the original feature space into HVs. Common encoders include linear random projection and Random Fourier Features. The encoded HVs are then aggregated through bundling to construct the associative memory (AM), which stores one prototype hypervector for each class.

During inference, a query HV is compared with the stored prototypes using similarity metrics such as Hamming distance or cosine similarity. The AM can be refined iteratively using perceptron-style update rules [18]. Specifically, hypervectors from misclassified samples are subtracted from the hypervector of the incorrect class and added to that of the correct class. This procedure improves the robustness of the AM without requiring gradient-based optimization.

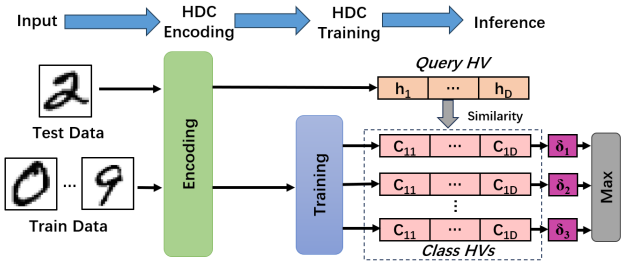


Fig. 1: Overview of the HDC training and inference flow

C. HDC and Federated Learning

Recent studies indicate that the computational and memory efficiency of HDC transfers naturally to Federated Learning (FL) [19]–[22]. In contrast to conventional FL methods that communicate high-precision neural network parameters, HDC-based FL exchanges compact hypervectors, which substantially reduces communication overhead. Server-side aggregation is implemented using the native HDC bundling operator ($+$), which further simplifies the overall training pipeline.

FL-HDC [19] reduces communication cost by transmitting bipolar HVs. RE-FHDC [20] further decreases overhead by partitioning HVs during training and synchronization. FHDnn [21] integrates a CNN feature extractor with an HDC classifier to exploit hybrid representations, and FedHD [22] demonstrates the practicality of this paradigm on real-world edge devices and networks.

Despite these advances, existing HDC-FL methods generally assume a single global model, which leads to performance degradation under severe statistical heterogeneity. Because HVs serve as compact summaries of data distributions, they provide a natural foundation for designing personalized or distribution-aware FL mechanisms. This limitation of prior work motivates the framework introduced in the next section.

III. FedHDC-KDS Framework

A. System Overview

In this section, we introduce FedHDC-KDS, a personalized federated learning framework that integrates the efficiency of hyperdimensional computing (HDC) with client-specific model aggregation and gradient-free knowledge distillation. Each round of communication consists of five stages, summarized in Fig. 2, and detailed below.

1) **Local Training.** Each client i maintains a pair of coupled HDC models: a local *student* model S_i and a local *teacher* model T_i. The student updates its class prototypes using prototype-based HDC encoding on local data, enabling rapid and lightweight adaptation to client-specific distributions. In parallel, the teacher model

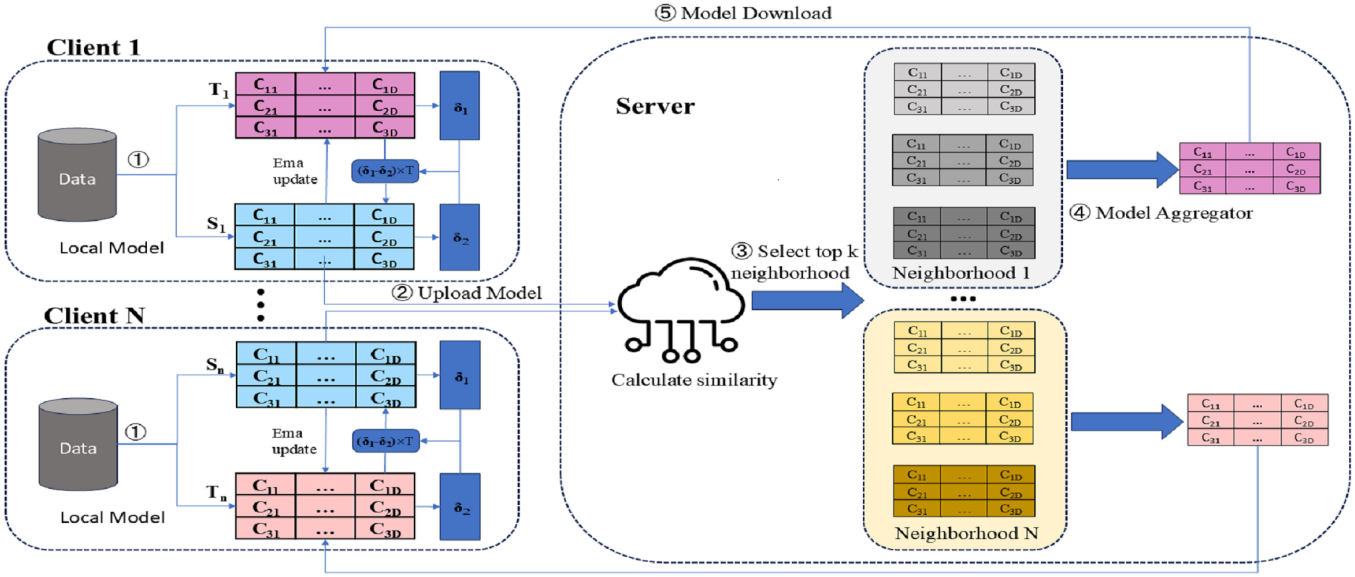


Fig. 2: Overall Workflow of FedHDC-KDS

incorporates external knowledge and provides additional hypervector-level guidance that stabilizes and refines the student's updates, preventing drift from aggregated peer representations while preserving local specialization.

2) **Model Upload.** After local training, each client transmits its student model S_i to the server. Because HDC representations are fixed-dimensional and noise-tolerant, this communication remains lightweight and does not reveal raw data.
3) **Similarity-Based Client Neighborhood Construction.** The server characterizes inter-client relationships by comparing update vectors ΔS_i computed from consecutive student model changes. Cosine similarity identifies the top-k most relevant peers for each client, forming a personalized neighborhood $\mathcal{N}_i$ and avoiding the degradation associated with naive global averaging.
4) **Personalized Aggregation.** For each client i, the server aggregates student model from its neighborhood $\mathcal{N}_i$ using similarity-based weights, producing a personalized teacher model $T_i^{(t+1)}$. This selectively transfers knowledge from statistically compatible clients while suppressing noise from mismatched distributions.
5) **Model Download.** The server sends the personalized teacher $T_i^{(t+1)}$ back to the client i, which then guides the next round of local HDC updates through similarity-based distillation.

Through this iterative coupling of local hypervector adaptation and similarity-aware cross-client distillation, FedHDC-KDS achieves efficient and highly personalized learning while maintaining the lightweight computation and communication advantages inherent to HDC.

Algorithm 1 Training Strategy of FedHDC-KDS

Input: total rounds R, clients N, classes C, neighborhood size k
Output: personalized models $S_1, \ldots, S_N$

for $t = 1$ to R **do**
  for $i = 1$ to N **in parallel do**
    //Client$_i$ **does:**
    $T_i \leftarrow download_{\text{Server}\rightarrow\text{Client}_i}$;
    for sample, label$_{real}$ in local data **do**
      $Q \leftarrow$ encode(sample);
      label$_{pred} \leftarrow$ AssociativeSearch(Q, S_i);
      if label$_{pred} \neq$ label$_{real}$ **then**
        $S_i \leftarrow$AM_Correction$(S_i, Q$,label$_{real}$,label$_{pred})$;
      end if
      $S_i \leftarrow$ SimilarityDistill(S_i, T_i, Q);
    end for
    $T_i \leftarrow$ EMA_Update(T_i, S_i);
    upload$_{\text{Client}_i\rightarrow\text{Server}}(S_i)$;
  end for
  //Server does:
  $gather_{\text{Client}_i\rightarrow\text{Server}(S_i)}$;
  Compute update vectors ΔS_i for all clients;
  $\mathcal{N}_k(i) \leftarrow$ Top-k neighbors based on similarity(ΔS_i);
  for $i = 1$ to N **do**
    $T_i \leftarrow$ SimilarityAwareAggregation$(\{S_j\}_{j\in\mathcal{N}_k(i)})$;
  end for
  broadcast$_{\text{Server}\rightarrow\text{Client}_i}(T_i)$;
end for

B. Server-Side: Similarity-Aware Aggregation

In conventional federated learning, the server aggregates all client models into a single global model and redistributes it for the next round of local training. However, under heterogeneous and non-IID data distributions, such a global model fails to represent the diverse local semantics of individual clients, leading to degraded personalization and slow convergence.

To address these limitations, we introduce a similarity-aware aggregation mechanism tailored for hyperdimensional feder-

ated learning. Instead of relying on static model parameters, the server infers inter-client relationships from their update dynamics.

For client i at communication round t, let $S_i^{(t)}$ denote its hypervector model, and define the update vector

$$\Delta S_i^{(t)} = S_i^{(t)} - S_i^{(t-1)}, \tag{1}$$

which captures how client i incrementally adjusts its hypervectors based on local data. Since clients have heterogeneous data distributions, their update vectors generally diverge, making the similarity between $\Delta S_i^{(t)}$ a natural measure of the similarity of clients' underlying data distributions.

The pairwise cosine similarity between update vectors is computed to form a similarity matrix A, with entries defined as

$$A_{i,j} = \frac{\Delta S_i^{(t)} \cdot \Delta S_j^{(t)}}{\|\Delta S_i^{(t)}\| \, \|\Delta S_j^{(t)}\|}. \tag{2}$$

Each row of A quantifies how closely client i's update aligns with all other clients and serves as the basis for similarity-weighted aggregation in the personalized federated learning framework.

Each client i then forms a soft neighborhood $\mathcal{N}_k(i)$ consisting of its top-k most similar peers according to the similarity matrix A. The personalized teacher model is aggregated via attention-weighted averaging:

$$T_i^{(t+1)} = \sum_{j \in \mathcal{N}_k(i)} \alpha_{i,j} \, S_j^{(t)}, \tag{3}$$

where $\alpha_{i,j}$ are normalized attention coefficients that assign greater weight to peers with more aligned update behaviors. The resulting $T_i^{(t+1)}$ provides client i with a personalized teacher model for the next round of client-side updates.

This mechanism ensures that the aggregation process prioritizes statistically compatible clients while suppressing noise from less relevant peers.

C. Client-Side Gradient-Free Hyperdimensional Learning

After initialization, each client encodes its local dataset into high-dimensional hypervectors using the standard HDC encoding process, where raw samples are transformed into representative hypervectors via item memories and basic compositional operations. These encoded hypervectors are then aggregated into class-specific prototype vectors, forming a local associative-memory classifier. We denote the resulting prototype sets as $S_1, S_2, \ldots, S_N$, where S_i is the student model maintained on client i. Building on these locally constructed prototypes, FedHDC-KDS enhances personalization through three client-side optimization mechanisms described below.

1) HDC Associative-Memory Update: To improve robustness under non-IID data, we adopt a similarity-weighted refinement rule inspired by OnlineHD [18]. Each sample is encoded into a query hypervector $\mathbf{Q}$, and its prediction is obtained by cosine similarity:

$$\hat{y} = \arg\max_c \cos(\mathbf{Q}, S_{ic}). \tag{4}$$

When a misclassification occurs ($\hat{y} \neq y$), the prototypes of the true and predicted classes are updated using similarity-scaled corrections:

$$S_{iy} \leftarrow S_{iy} + lr\,(1 - \Delta_y)\,\mathbf{Q}, \tag{5}$$

$$S_{i\hat{y}} \leftarrow S_{i\hat{y}} - lr\,(1 - \Delta_{\hat{y}})\,\mathbf{Q}, \tag{6}$$

where Δ_y and $\Delta_{\hat{y}}$ represent the cosine similarity between $\mathbf{Q}$ and the prototypes of the true and predicted classes, and $lr \in [0, 1)$ is the learning rate.

The similarity-dependent weighting $(1 - \Delta)$ reduces prototype fuzziness and enhances class separability.

2) Similarity-Based Hyperdimensional Distillation: After the AM correction step, each client further refines its student model by aligning it with the personalized teacher hypervectors received from the server. Rather than using probabilistic targets, FedHDC-KDS aligns the student's similarity responses to its personalized teacher. For $\mathbf{Q}$, the student and teacher produce similarity vectors:

$$\mathbf{s}_i^{\text{stu}} = S_i\mathbf{Q}, \qquad \mathbf{s}_i^{\text{tea}} = T_i\mathbf{Q}, \tag{7}$$

where each similarity vector has length C, with each element representing the similarity of the query to a particular class. The distillation signal is defined as the difference between the teacher and student similarity vectors:

$$\mathbf{e}_i = \mathbf{s}_i^{\text{tea}} - \mathbf{s}_i^{\text{stu}}. \tag{8}$$

This residual is incorporated into the student's hypervectors via an binding operation, yielding a gradient-free hyperdimensional alignment rule:

$$S_i \leftarrow S_i + \eta\,(\mathbf{e}_i \times T_i), \tag{9}$$

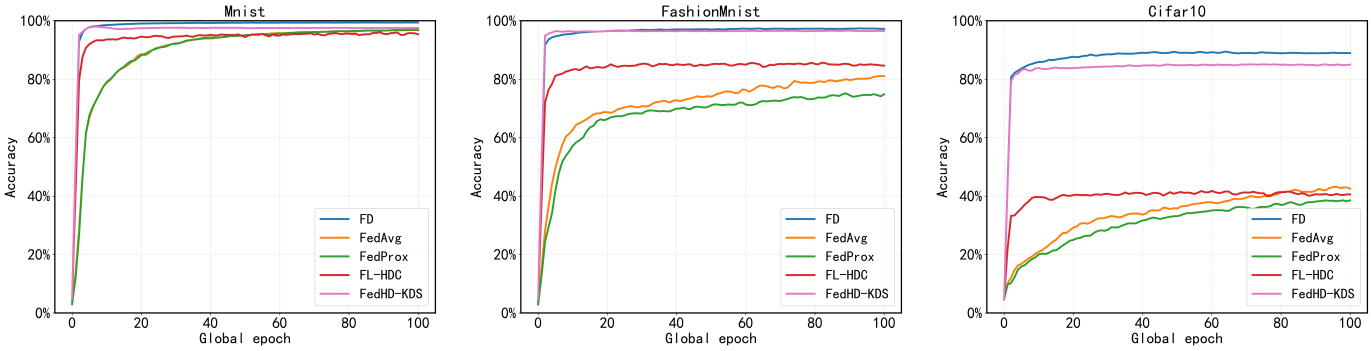
where η controls the strength of teacher guidance. This update binds the teacher's similarity information into the student hypervectors, ensuring the student captures task-relevant patterns distilled from the teacher. It also enables stable adaptation under non-IID data without backpropagation or extra communication.

3) Momentum-Based Teacher Smoothing: After similarity-based hyperdimensional distillation, each client smooths its teacher hypervectors via an exponential moving average (EMA) of the student hypervectors:

$$T_i \leftarrow \beta\,T_i + (1 - \beta)\,S_i, \tag{10}$$

where $\beta \in [0, 1)$ controls the degree of temporal smoothing. This momentum-based update enables the teacher to accumulate stable, high-dimensional representations while filtering out fluctuations from noisy local updates.

By evolving gradually over time, the teacher acts as a personalized anchor for client i, providing consistent supervisory signals for distillation and reducing variance during federated aggregation. It complements the fast, margin-based updates of the student, jointly enabling robust and efficient adaptation under non-IID data distributions.



(a) (b) (c)

Fig. 3: Accuracy comparison of all algorithms on (a) MNIST, (b) FashionMNIST, (c) CIFAR-10 datasets.

	FedAvg	FedProx	FedDistill	FL-HDC	FedHD-KDS
MNIST	96.92%	96.93%	99.37%	95.32%	97.51%
Fashion MNIST	80.72%	75.56%	97.29%	85.12%	96.54%
CIFAR-10	42.43%	38.35%	88.98%	40.56%	84.95%
Size of trained model	2.32 MB	2.32 MB	2.32 MB	39.06KB	78.13KB

TABLE I: Comparison of final accuracy and model size for all algorithms across different datasets.

IV. Experiments And Results

A. Experiment Setup

Datasets and Data Partitioning: We evaluate FedHD-KDS on three widely used image classification benchmarks: MNIST, FashionMNIST, and CIFAR-10. To simulate realistic cross-client heterogeneity, we partition each dataset among $K = 20$ clients using a Dirichlet distribution with concentration parameter $\alpha = 0.1$, which produces highly skewed non-IID label distributions. Each client uses 75% of its local data for training and the remaining 25% for testing.

Baselines and Evaluation: We compare FedHD-KDS with four representative FL methods: (1) FedAvg, (2) FedProx, (3) FedDistill, and (4) FL-HDC. This selection allows us to examine both classical neural-network FL baselines and alternative HDC-based paradigms under identical FL conditions. For global FL baselines (FedAvg, FedProx), we report the accuracy of the aggregated global model. For personalized methods (FedDistill, FL-HDC, and FedHD-KDS), we report the average personalized test accuracy across all clients.

Model Architectures: To ensure a fair comparison, we employ the same convolutional neural network (CNN) for all non-HDC baselines. The CNN contains two 5×5 convolutional layers (with 32 and 64 channels, ReLU activations, and 2×2 max pooling), followed by a fully connected layer with 512 units and a final linear classifier. For MNIST and FashionMNIST, the CNN uses a single-channel input, while CIFAR-10 uses a three-channel input.

In contrast, FL-HDC and our FedHD-KDS framework employ HDC models rather than neural networks. For both methods use hypervectors of dimensionality $D = 1000$, our preliminary experiments indicated that performance saturates at this level, making higher dimensions (e.g., $D = 10,000$) computationally redundant without yielding accuracy gains. Separating CNN-based and HDC-based models ensures that architectural differences do not confound our analysis of the underlying learning paradigms.

B. Performance Results

We first compare the accuracy of FedHD-KDS with representative baseline methods. As shown in Table I, the proposed approach demonstrates consistently strong performance across MNIST, FashionMNIST, and CIFAR-10, despite relying on lightweight hyperdimensional representations. On MNIST, FedHD-KDS attains an accuracy of 97.51%, outperforming FL-HDC and approaching the performance of CNN-based FL methods. On FashionMNIST, FedHD-KDS reaches 96.54%, yielding an improvement of more than 11% over FL-HDC and achieving accuracy comparable to that of FedDistill (97.29%).

The advantage becomes more pronounced on CIFAR-10, where FedHD-KDS achieves 84.95%, corresponding to a substantial improvement of +42.52% over FL-HDC. Although FedDistill attains a slightly higher accuracy of 88.98%, it relies on a multi-megabyte CNN model, whereas FedHD-KDS remains competitive with a model that is nearly 29$\times$ smaller. Notably, the larger performance gap between FedHD-KDS and FD observed in Fig. 3 on CIFAR-10, compared with MNIST and FashionMNIST, is likely related to the higher visual complexity and inter-class similarity of CIFAR-10. These characteristics tend to favor the feature-rich CNN representations used by FD, while the proposed hyperdimensional distillation mechanism maintains competitive robustness under more challenging visual conditions. Furthermore, under heterogeneous client distributions, which are typically more pronounced in complex datasets such as CIFAR-10, the personalized knowledge transfer strategy in FedHD-KDS contributes to stable performance gains over conventional HDC-based baselines.

In terms of efficiency, FedHD-KDS eliminates the need for backpropagation and repeated gradient-based updates in each communication round. Instead, both student and teacher hypervectors are updated through lightweight algebraic operations in high-dimensional space, which can be executed in parallel and are well suited for memory- and compute-constrained devices. As illustrated in Fig. 3, FedHD-KDS also converges significantly faster than CNN-based FL methods, indicating that comparable accuracy can be achieved with substantially fewer client–server interactions. This reduction directly translates into lower communication requirements.

Moreover, the transmitted teacher model is only 78.13 KB, which is more than 29× smaller than the 2.32 MB models used by CNN-based baselines. This compact representation further reduces communication overhead and facilitates deployment in resource-limited federated environments.

Overall, the results indicate that FedHD-KDS achieves a strong balance among accuracy, efficiency, and communication cost. The method consistently outperforms existing HDC-based FL approaches, approaches the performance of CNN-based methods, and maintains clear advantages in convergence speed and model compactness.

V. Conclusion

This work introduces FedHDC-KDS, a lightweight and communication-efficient framework for personalized federated learning built upon hyperdimensional computing. By leveraging HDC's compact representations and efficient computing paradigm, the proposed method substantially reduces model storage and communication overhead. In addition, the gradient-free update rules, combined with similarity-aware teacher aggregation, enable effective semantic knowledge exchange across clients, ensuring robustness even under highly non-IID data distributions.

Experiments on standard benchmark datasets demonstrate that FedHDC-KDS strikes an optimal balance between performance and resource efficiency. It achieves accuracy comparable to DNN-based baselines on lightweight tasks while significantly outperforming state-of-the-art HDC methods on complex benchmarks, all while delivering a **29×** reduction in model size and faster convergence. Future work will explore advanced HDC encoding schemes to further bridge the accuracy gap on complex vision tasks without compromising efficiency.

References

[1] B. McMahan, E. Moore, D. Ramage, S. Hampson, and B. A. y Arcas, "Communication-efficient learning of deep networks from decentralized data," in *Artificial intelligence and statistics*, pp. 1273–1282, PMLR, 2017.

[2] P. Kairouz, H. B. McMahan, B. Avent, A. Bellet, M. Bennis, A. N. Bhagoji, K. Bonawitz, Z. Charles, G. Cormode, R. Cummings, *et al.*, "Advances and open problems in federated learning," *Foundations and trends® in machine learning*, vol. 14, no. 1–2, pp. 1–210, 2021.

[3] Q. Yang, Y. Liu, T. Chen, and Y. Tong, "Federated machine learning: Concept and applications," *ACM Transactions on Intelligent Systems and Technology (TIST)*, vol. 10, no. 2, pp. 1–19, 2019.

[4] S. P. Karimireddy, S. Kale, M. Mohri, S. Reddi, S. Stich, and A. T. Suresh, "Scaffold: Stochastic controlled averaging for federated learning," in *International conference on machine learning*, pp. 5132–5143, PMLR, 2020.

[5] A. Fallah, A. Mokhtari, and A. Ozdaglar, "Personalized federated learning with theoretical guarantees: A model-agnostic meta-learning approach," *Advances in neural information processing systems*, vol. 33, pp. 3557–3568, 2020.

[6] A. Z. Tan, H. Yu, L. Cui, and Q. Yang, "Towards personalized federated learning," *IEEE transactions on neural networks and learning systems*, vol. 34, no. 12, pp. 9587–9603, 2022.

[7] K. Pillutla, K. Malik, A.-R. Mohamed, M. Rabbat, M. Sanjabi, and L. Xiao, "Federated learning with partial model personalization," in *International Conference on Machine Learning*, pp. 17716–17758, PMLR, 2022.

[8] P. Kanerva, "Hyperdimensional computing: An introduction to computing in distributed representation with high-dimensional random vectors," *Cognitive computation*, vol. 1, no. 2, pp. 139–159, 2009.

[9] D. Kleyko, D. A. Rachkovskij, E. Osipov, and A. Rahimi, "A survey on hyperdimensional computing aka vector symbolic architectures, part i: Models and data transformations," *ACM Computing Surveys*, vol. 55, no. 6, pp. 1–40, 2022.

[10] M. Imani, J. Morris, S. Bosch, H. Shu, G. De Micheli, and T. Rosing, "Adapthd: Adaptive efficient training for brain-inspired hyperdimensional computing," in *2019 IEEE Biomedical Circuits and Systems Conference (BioCAS)*, pp. 1–4, IEEE, 2019.

[11] T. Li, A. K. Sahu, M. Zaheer, M. Sanjabi, A. Talwalkar, and V. Smith, "Federated optimization in heterogeneous networks," *Proceedings of Machine learning and systems*, vol. 2, pp. 429–450, 2020.

[12] M. G. Arivazhagan, V. Aggarwal, A. K. Singh, and S. Choudhary, "Federated learning with personalization layers," *arXiv preprint arXiv:1912.00818*, 2019.

[13] M. A. Husnoo, A. Anwar, N. Hosseinzadeh, S. N. Islam, A. N. Mahmood, and R. Doss, "Fedrep: Towards horizontal federated load forecasting for retail energy providers," in *2022 IEEE PES 14th Asia-Pacific Power and Energy Engineering Conference (APPEEC)*, pp. 1–6, IEEE, 2022.

[14] T. Lin, L. Kong, S. U. Stich, and M. Jaggi, "Ensemble distillation for robust model fusion in federated learning," *Advances in neural information processing systems*, vol. 33, pp. 2351–2363, 2020.

[15] D. Li and J. Wang, "Fedmd: Heterogenous federated learning via model distillation," *arXiv preprint arXiv:1910.03581*, 2019.

[16] S. Itahara, T. Nishio, Y. Koda, M. Morikura, and K. Yamamoto, "Distillation-based semi-supervised federated learning for communication-efficient collaborative training with non-iid private data," *IEEE Transactions on Mobile Computing*, vol. 22, no. 1, pp. 191–205, 2021.

[17] R. W. Gayler, "Vector symbolic architectures answer jackendoff's challenges for cognitive neuroscience," *arXiv preprint cs/0412059*, 2004.

[18] A. Hernández-Cano, N. Matsumoto, E. Ping, and M. Imani, "Onlinehd: Robust, efficient, and single-pass online learning using hyperdimensional system," in *2021 Design, Automation & Test in Europe Conference & Exhibition (DATE)*, pp. 56–61, IEEE, 2021.

[19] C.-Y. Hsieh, Y.-C. Chuang, and A.-Y. A. Wu, "Fl-hdc: Hyperdimensional computing design for the application of federated learning," in *2021 IEEE 3rd International Conference on Artificial Intelligence Circuits and Systems (AICAS)*, pp. 1–5, IEEE, 2021.

[20] N. Zeulin, O. Galinina, N. Himayat, and S. Andreev, "Resource-efficient federated hyperdimensional computing," *arXiv preprint arXiv:2306.01339*, 2023.

[21] R. Chandrasekaran, K. Ergun, J. Lee, D. Nanjunda, J. Kang, and T. Rosing, "Fhdnn: Communication efficient and robust federated learning for aiot networks," in *Proceedings of the 59th ACM/IEEE Design Automation Conference*, pp. 37–42, 2022.

[22] Q. Zhao, K. Lee, J. Liu, M. Huzaifa, X. Yu, and T. Rosing, "Fedhd: federated learning with hyperdimensional computing," in *Proceedings of the 28th annual international conference on mobile computing and networking*, pp. 791–793, 2022.