This MICCAI paper is the Open Access version, provided by the MICCAI Society. It is identical to the accepted version, except for the format and this watermark; the final published version is available on SpringerLink.

FedProIn: Mitigating Client Drift for Learnable Prototypes in Federated Medical Imaging

Harsh Kumar[1,3], Tarun Kumar Garg[2,3], and Vaanathi Sundaresan[1,4]

[1] Department of Computational and Data Sciences, Indian Institute of Science Bengaluru, Karnataka 560012, India
[2] Department of IISc Mathematics Initiative, Indian Institute of Science Bengaluru, Karnataka 560012, India
[3] Equally contributed to the work
[4] Corresponding author: vaanathi@iisc.ac.in

Abstract. Federated learning (FL) is severely hindered by statistical heterogeneity due to variations in scanners, acquisition protocols, and patient populations. Such non-IID data induces client drift during local optimization, leading to unstable convergence and suboptimal global models when parameter-based aggregation is applied. We propose a prototype-based, influence-aware federated learning framework (FedProIn) that uses multiple learnable class prototypes to capture shared semantic structures across heterogeneous clients. We introduce feature divergence loss and prototype contrastive loss to mitigate client drift by decomposing it into feature drift and prototype drift. In addition, we propose a normalized influence aggregation strategy that adaptively weights client prototypes according to their contribution to the global representation, reducing the impact of biased or low-quality updates. Experimental results on two publicly available medical datasets, HAM10000 and Matek-19, demonstrate that FedProIn achieves accuracies of (83.5% IID, 81.1% non-IID) on HAM10000 and (96.2% IID, 95.8% non-IID) on Matek-19, respectively, outperforming existing baselines in both conditions. Our code is available at https://github.com/harsh-kmr/FedProIn.

Keywords: Federated Learning · Prototype Learning · Learnable Prototypes

1 Introduction

Medical imaging research faces limitations posed by small datasets and shortage of resources [3]. Further, pooling data across centers is limited by privacy regulations like HIPAA [17] and GDPR [2]. Federated Learning (FL) allows local models to be trained on-site while sharing only updates with a central server. However, the performance of FL is often compromised by statistical heterogeneity in medical deployments, due to differences in scanner protocols, demographics, and disease prevalence [3]. This heterogeneity causes local models to diverge from the global model, leading to *client drift* and degraded FL performance. Heterogeneous distributions cause client models to map identical instances to

distinct feature space regions. Furthermore, local representations become biased toward client-specific characteristics, deviating from the global model.

Related work. Recent literature highlights two approaches. The first, constraint-based regularization, includes methods like FedProx [7], and MOON [6], which add penalties to limit disparity between local and global model parameters. Though effective, these methods struggle to align semantic representations in high-dimensional space. The second is prototype-based learning [14, 11, 20, 4, 18, 15], using prototypes to guide local training. Typically, methods aggregate these prototypes as mean feature vectors, summarizing each class by a single global prototype via weighted averaging of local prototypes. Most approaches (e.g., FedProto [14], FedProc [11]) assume each class is represented by a single prototype, which is inadequate in medical FL due to data heterogeneity. Some methods use multiple prototypes (e.g., FedPLVM [18], FPL [4]), but clustering adds complexity. Methods like FedPCL [15] use multiple encoders for distinct prototypes, increasing costs. FedTGP [20] introduces trainable global prototypes optimized on servers, but still relies on unimodal client representations, using mean feature vectors. These prototypes often sacrifice inter-class separability by not optimizing decision boundaries. Learnable prototypes (LPs) maximize class separation by maximizing the margin between prototypes, indicating stronger separability. Convolutional Prototype Learning (CPL) [19] showed that optimizing learnable prototypes and a feature extractor from raw data creates intra-class compact and inter-class separable representations, versus using a softmax layer. However, this remains unexplored in FL setup.

Contributions. In this work, we propose **Fed**erated Learning with **Pro**totype aggregation with **In**fluence (**FedProIn**) to maximize class separation. Firstly, we adopt the LP formulation that treats prototypes as optimizable model parameters, unlike standard approaches that rely on mean vectors. Secondly, we explicitly tackle client drift by decomposing it into feature drift and prototype drift, addressing them separately using feature divergence loss and prototype contrastive loss. Additionally, we also introduce a Normalized Influence Aggregation (NIA) mechanism that efficiently aggregates multi-prototype models by quantifying the adaptive relevance of prototypes in each client. We evaluate our approach against existing methods on public long-tailed medical imaging datasets.

2 Methodology

Our objective is to train a global model that maps an input image $x \in \mathcal{X}$ to a label $y \in \{1, \ldots, C\}$, where C denotes the number of classes. Unlike standard classification approaches that map inputs directly to logits, we adopt a prototype-based learning framework where the global model is decomposed into two distinct components: a feature encoder f_θ parameterized by θ, which maps input images to a d-dimensional embedding space, and a set of LPs $\mathcal{P} \in \mathbb{R}^{C \times M \times d}$, where M

denotes the number of prototypes per class. The global model parameters are denoted as $w = \{\theta, \mathcal{P}\}$.

We consider a setup with K clients, where each client $i \in \{1, \ldots, K\}$ owns a private local dataset $\mathcal{D}_i = \{(x_{i,j}, y_{i,j})\}_{j=1}^{n_i}$. Here, $n_i = |\mathcal{D}_i|$ is the size of the local dataset, and $n = \sum_{i=1}^{K} n_i$ denotes the total number of samples across all clients. The goal is to minimize the global empirical risk, which is formulated as

$$\min_{w} \ \mathcal{L}(w) = \sum_{i=1}^{K} \frac{n_i}{n} \ell(w, \mathcal{D}_i), \tag{1}$$

where $\ell(w, \mathcal{D}_i)$ is the local objective at the client i over the client's dataset $\mathcal{D}_i$.

2.1 Federated Learning with Prototype aggregation with Influence (FedProIn)

In FedProIn, we consider prototypes as model parameters [19], instead of mean feature vectors, optimized jointly with the feature extractor via backpropagation. Fig. 1 illustrates the overall framework of FedProIn. At the start of the communication round t, the server broadcasts the global state w^{t-1} to the participating clients. Clients perform local training and return the updated parameters w_i^t alongside an *influence matrix* $\mathcal{I}_i$, which guides the server's aggregation.

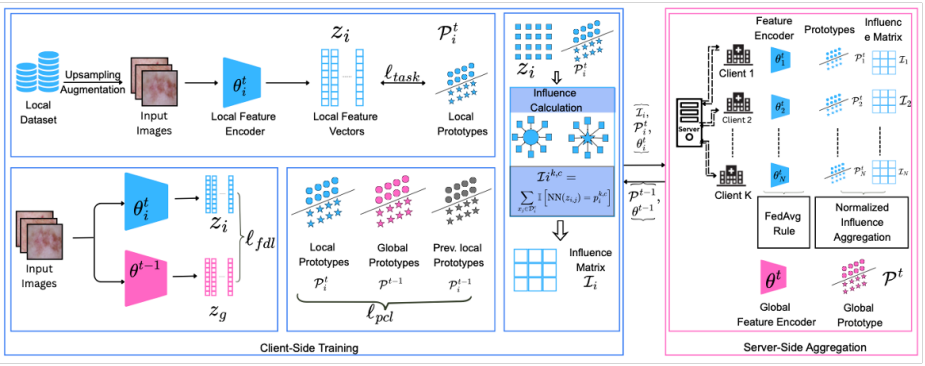

Fig. 1: **Overview of FedProIn framework.** In client-side training (blue), clients extract features using local encoder θ_i^t and update local prototypes $\mathcal{P}_i^t$. *Feature Divergence Loss (FDL)* aligns local and frozen global features, while *Prototype Contrastive Loss (PCL)* pulls local prototypes toward global prototypes and pushes away from previous ones (grey). Clients construct an influence matrix $\mathcal{I}_i$ tracking prototype usage. In server-side aggregation (pink), encoders are aggregated via FedAvg and prototypes via *Normalized Influence Aggregation (NIA)* to produce global encoder θ^t and prototypes $\mathcal{P}^t$.

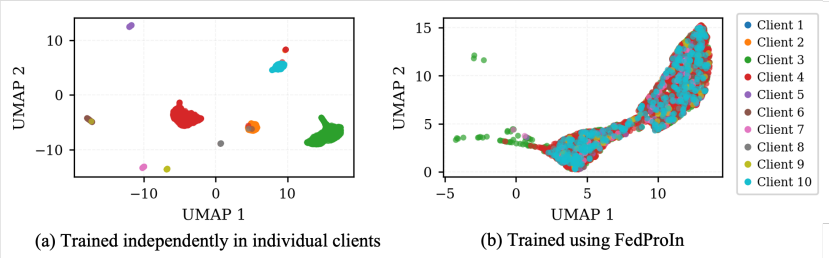

Fig. 2: **Comparison of feature drift between individual client training and training with FedProIn.** UMAP projections shown for a sample class when (a) training across individual clients independently without FL, and (b) training across all clients under FedProIn.

Client-side local training. The local training process addresses client drift through: (i) *feature drift*, where client models map images to different regions in feature space due to data heterogeneity, and (ii) *prototype drift*, where client prototypes are pulled towards local data characteristics, causing prototypes to diverge from global representation. These lead to semantic misalignment, degrading global model robustness. Hence, we introduce two regularization terms.

Feature divergence loss (FDL). To address feature drift (shown in Fig. 2), we employ a mean square loss between the global and local feature vectors. Each client maintains a copy of the global feature extractor (as shown in Fig. 1), parameterized by θ^{t-1}, received from the server at the start of round t. This frozen encoder is used to compute reference global feature representations and is not involved in local training. Given $z_{i,j} = f_{\theta_i^t}(x_{i,j})$ and $z_{g,j} = f_{\theta^{(t-1)}}(x_{i,j})$, we align the local features toward the global features by minimizing:

$$\ell_{FDL} = \frac{1}{n_i} \sum_{j=1}^{n_i} \|z_{i,j} - z_{g,j}\|_2^2. \tag{2}$$

Prototype contrastive loss (PCL). To regulate prototype drift, we utilize a contrastive loss since global prototypes likely capture more generalized representation of the overall data distribution compared to local prototypes that might be skewed due to shift in their data distribution. Let $\mathcal{P}^{(t-1)}$ denote the current global prototypes and $\mathcal{P}_i^{(t-1)}$ denote the client's prototypes from the previous round. We aim to simultaneously pull the updated local prototypes $\mathcal{P}_i^t$ closer to the global prototypes while pushing them away from their previous states $\mathcal{P}_i^{(t-1)}$. To achieve this, we compute distances $d_{\text{glob}} = \|\mathcal{P}^{(t-1)} - \mathcal{P}_i^t\|_2$ and $d_{\text{prev}} = \|\mathcal{P}_i^{(t-1)} - \mathcal{P}_i^t\|_2$, and minimize:

$$\ell_{PCL} = \max(0, \Delta + m), \quad \text{where} \quad \Delta = \frac{d_{\text{glob}} - d_{\text{prev}}}{d_{\text{glob}} + d_{\text{prev}} + \epsilon} \tag{3}$$

where m is a margin hyperparameter (so that $\mathcal{P}_i^t$ must be closer to $\mathcal{P}^{(t-1)}$ than to $\mathcal{P}_i^{(t-1)}$ by at least m) and ϵ is a stability constant in the denominator.

The Update Challenge and Class Imbalance. Optimizing multiple prototypes creates a structural *update challenge* in contrastive loss. For any sample, the loss function interacts with the nearest true-class prototype (positive) and competing-class prototype (negative), with gradients flowing to these two prototypes while others receive zero gradients. In long-tailed medical datasets, prototypes for rare disease subtypes may rarely be selected as nearest neighbors, causing them to stagnate and fail to represent their semantic subregions. To address this, we use class-balanced sampling and adaptive augmentation [13], where clients oversample minority classes and apply augmentation to generate diverse views for rare subtypes.

The total local objective for client i combines the primary classification loss ℓ_{Task} with FDL and PCL terms as follows:

$$\ell_{\text{Total}} = \ell_{Task} + \lambda_{FDL}\ell_{FDL} + \lambda_{PCL}\ell_{PCL}, \tag{4}$$

where λ_{FDL} and λ_{PCL} are weighting coefficients of ℓ_{FDL} and ℓ_{PCL}, respectively.

Server-side Aggregation. At the end of round t, the server receives updated model parameters $\{\theta_i^t, \mathcal{P}_i^t\}$ and influence matrix $\mathcal{I}_i$ from each client $i \in \mathcal{S}_t$, where $\mathcal{S}_t$ represents participating clients. The server coordinates global update by aggregating encoder parameters and prototypes via distinct protocols, as described below.

Encoder parameter aggregation. We employ the standard FedAvg protocol [10] to aggregate the feature encoder. The global encoder is computed as a weighted average based on local dataset sizes and $N_t = \sum_{i\in\mathcal{S}_t} n_i$.

$$\theta^t = \sum_{i\in\mathcal{S}_t} \frac{n_i}{N_t}\theta_i^t, \tag{5}$$

Normalized influence aggregation (NIA) of prototypes. Unlike the encoder, FedAvg/ FedProto aggregation cannot be directly applied to prototypes due to limitations with multiple prototypes per class: (1) *Heterogeneous prototype utilization:* within a class, prototypes are not utilized uniformly; some are more frequently activated than others. Hence, FedAvg assigns prototype weights proportional to the dataset size n_i, amplifying noise from less utilized prototypes. (2) *Sparse Prototype Activation:* Some prototypes remain unutilized by specific clients when representing rare disease subtypes are absent from local distribution. FedAvg would assign them the full n_i weight, suppressing information from clients that utilize these prototypes, potentially losing rare but important case representations.

To address these limitations, we propose NIA mechanism. During the local training phase at client i, we explicitly track the utility of each prototype by

constructing an influence matrix $\mathcal{I}_i$. For the k-th prototype of class c, denoted as $p_i^{k,c}$, the corresponding influence score $\mathcal{I}_i^{k,c}$ is computed as:

$$\mathcal{I}_i^{k,c} = \sum_{x_j \in \mathcal{D}_i^c} \mathbb{I}\left[\mathrm{NN}(z_{i,j}) = p_i^{k,c}\right], \tag{6}$$

where $\mathcal{D}_i^c$ is the set of training samples from class c at client i, $z_{i,j}$ is the feature embedding of sample x_j, $\mathrm{NN}(\cdot)$ returns the nearest prototype (in Euclidean distance), and $\mathbb{I}[\cdot]$ is the indicator function. Intuitively, $\mathcal{I}_i^{k,c}$ counts how many times prototype $p_i^{k,c}$ served as the closest match for local samples during training.

The server then aggregates the prototypes as follows. For each class c and prototype index k, the global prototype is computed as:

$$p_{global}^{k,c,t} = \frac{\sum_{i \in \mathcal{S}_t} \mathcal{I}_i^{k,c} \cdot p_i^{k,c}}{\sum_{i \in \mathcal{S}_t} \mathcal{I}_i^{k,c} + \epsilon}. \tag{7}$$

This formulation weights prototypes proportionally to their usage during local training, automatically down-weighting noisy updates from underutilized prototypes (addressing limitation 1) while preserving specialized prototypes relevant to a subset of clients (addressing limitation 2). Crucially, because normalization is computed per prototype index across clients (rather than across all class prototypes), only clients activating prototype $\{k, c\}$ contribute to its update. Thus, a rare prototype activated by only one or two clients is aggregated solely from those updates, preventing it from being diluted by higher-frequency prototypes.

3 Experimental setup

Dataset details. We evaluate FedProIn on two public long-tailed medical imaging datasets: (i) HAM10000 [16] contains 10,015 dermatoscopic images of pigmented skin lesions collected from multiple sources with 7 classes, and (2) Matek-19 [8] consists of over 18,000 annotated peripheral blood cell images collected from 100 patients diagnosed with acute myeloid leukemia. All images are resized to 128×128 pixels and normalized using ImageNet statistics [1] prior to training. For both datasets, the training:validation:test split is 70%:10%:20% respectively. *Federated Simulation:* We simulate FL setup with $K = 10$ clients under two partitioning schemes: (1) *IID*, and (2) *Non-IID*, using a Dirichlet distribution with $\alpha = 0.5$ to model statistical heterogeneity.

Implementation. All experiments use a ResNet-18 encoder, trained via Adam [5] (learning rate 1×10^{-4}, batch size 128) for 50 rounds with 50% client participation. Results are averaged over three runs with $E = 5$ local epochs. we empirically employ Minimum Classification Error Loss (MCEL) with $\lambda_{FDL}, \lambda_{PCL} = 1.0$, and Margin-based Classification Loss (MCL) [19] with $\lambda_{FDL} = 0.1, \lambda_{PCL} = 0.01$. We tuned hyperparameters via a sequential search on a 1,000-sample subset, sweeping one parameter at a time. The implementation is done using PyTorch [12] (2.6.0). Experiments are conducted on an Intel i9-14900k CPU, 64 GB RAM, and one NVIDIA RTX A5000 GPU (with 24 GB memory).

Evaluation Metrics. We evaluate classification performance using: Accuracy (Acc) in %, Weighted F1-Score (W-F1) in %, and Matthews Correlation Coefficient (MCC) [9]. MCC computes the Pearson correlation coefficient between the observed and predicted classifications. MCC yields a value between -1 and +1, offering a balanced metric even when classes are of very different sizes.

3.1 Experiments and results

Comparison with state-of-the-art methods. We initially determine the individual client-trained (SOLO) performance without FL. We compare FedProIn with (1) FL baselines: FedAvg [10], FedProx [7], MOON [6], (2) prototype-based methods: FedProto [14], FedProc [11], FedTGP [20], and FedPLVM [18], and (3) methods that alleviate the update problem: BalanceFL [13]. Since FedProIn maintains a global model, we generally exclude personalized methods that do not aggregate a model at the server. However, we make exceptions for FedProto (standard prototype-based method) and FedTGP (introduces a trainable global prototype). SOLO is trained for 300 local epochs, and for personalized methods (FedProto, FedTGP) that lack a global model, local predictions are aggregated for evaluation. Table 1 reports the comparative analysis on both datasets. We evaluated FedProIn across losses (MCL, MCEL) and prototype counts $M \in \{1, 2, 4, 8, 16\}$, reporting the best validation F1 configuration (as specified in Table 1). Optimal M varies with dataset and loss function. In both IID and non-IID settings, FedProIn outperforms all baselines on both the Matek-19 and HAM10000 datasets across all metrics. Especially under non-IID conditions, FedProIn shows robustness against data heterogeneity. Especially, the existing prototype methods suffer severe performance drops across all metrics from IID to non-IID configurations. For instance, they show a decrease of 6-7% in accuracy on HAM10000 and 1-1.6% on Matek-19, while FedProIn restricts these relative losses to merely 2.4% and 0.4%, respectively. This consistent generalization is directly driven by our dual regularization (FDL and PCL), which mitigates the client drift. Conversely, personalized methods like FedProto suffer significant performance drops because their lack of a globally shared representation restricts generalization across highly skewed local data distributions. FedProIn adds $M \times C \times 512$ parameters ($\approx$0.5 MB, M=2, C=7) and a tiny $M \times C$ influence matrix alongside the model weights per round, adding negligible communication overhead.

Ablation Study. Fig. 3 shows the ablation study on HAM10000 under non-IID conditions with MCL. Although different from Table 1, we selected this configuration since it is most challenging and yielded the lowest validation F1 across all settings. Removing FDL causes a sharper drop in W-F1 than removing PCL, indicate a stronger association that feature drift is the dominant failure mode under heterogeneity. However, both components are nonetheless necessary, as the full model significantly ($p - value < 0.05$ using Mann–Whitney U test on results across prototype counts as independent samples, given our experimental design) outperforms any single-component variant. The degenerate case (no FDL, no PCL, no NIA) reduces to FedAvg and performs worst, while adding

Table 1: Performance comparison with the state-of-the-art on HAM10000 and Matek-19 datasets under IID and non-IID (Dirichlet $\alpha = 0.5$) settings. For FedProIn, the optimal configurations yielding these results are: HAM10000 (IID: MCL, M=2; non-IID: MCEL, M=2) and Matek-19 (MCL, M=1 for both).

Method	IID			Dirichlet (non-IID)		
	Acc (%)	W-F1 (%)	MCC	Acc (%)	W-F1 (%)	MCC
HAM10000						
SOLO	75.1 ± .3	73.3 ± .1	.491 ± .002	61.0 ± .1	63.5 ± .1	.319 ± .002
FedAvg	81.7 ± .2	80.8 ± .3	.633 ± .005	78.1 ± 1.8	73.4 ± 2.6	.530 ± .049
FedProx	81.8 ± .2	80.9 ± .3	.636 ± .004	80.2 ± .8	77.9 ± .9	.593 ± .019
MOON	81.5 ± 0	80.7 ± .1	.629 ± .001	77.8 ± .8	73.8 ± 1.1	.522 ± .021
BalanceFL	83.0 ± .7	82.1 ± .7	.658 ± .014	79.2 ± .4	75.2 ± .6	.564 ± .012
FedProto	73.6 ± .2	70.9 ± .3	.444 ± .005	59.8 ± .1	62.0 ± .1	.296 ± .005
FedTGP	62.6 ± .1	65.8 ± .1	.383 ± .001	57.0 ± .1	61.2 ± .1	.320 ± .001
FedProc	82.6 ± .5	81.9 ± .4	.654 ± .008	76.4 ± .5	71.4 ± .8	.483 ± .014
FedPLVM	83.2 ± .3	82.6 ± .4	.666 ± .007	76.2 ± .9	71.3 ± 2.1	.487 ± .027
FedProIn	**83.5 ± .5**	**82.8 ± .7**	**.670 ± .013**	**81.1 ± .3**	**79.3 ± .3**	**.613 ± .004**
Matek-19						
SOLO	91.2 ± .3	90.4 ± .3	.874 ± .004	77.3 ± .8	77.6 ± 1.0	.688 ± .011
FedAvg	95.2 ± .1	94.7 ± .1	.931 ± .001	94.8 ± .3	94.1 ± .3	.925 ± .004
FedProx	95.1 ± .1	94.7 ± .1	.930 ± .001	94.6 ± .1	93.9 ± .1	.922 ± .002
MOON	95.4 ± .2	94.8 ± .3	.934 ± .003	95.3 ± .2	94.6 ± .2	.932 ± .002
BalanceFL	95.7 ± .3	95.2 ± .3	.938 ± .004	95.6 ± .1	94.9 ± .1	.937 ± .001
FedProto	91.8 ± .5	90.8 ± .4	.882 ± .007	84.6 ± 1.6	83.3 ± 1.5	.777 ± .023
FedTGP	71.2 ± .1	76.9 ± .1	.619 ± .001	59.9 ± .1	67.1 ± .1	.490 ± .001
FedProc	94.9 ± .2	93.8 ± .2	.927 ± .003	93.3 ± 2.2	92.4 ± 2.3	.904 ± .030
FedPLVM	95.7 ± .2	95.2 ± .2	.939 ± .003	94.7 ± .1	94.4 ± .1	.927 ± .001
FedProIn	**96.2 ± .1**	**95.8 ± .1**	**.946 ± .001**	**95.8 ± .1**	**95.1 ± .1**	**.940 ± .001**

NIA alone already yields noticeable gains, validating influence-aware prototype aggregation as an effective standalone improvement.

4 Conclusion

In this paper, we introduce **FedProIn**, a multiple Learnable Prototype based framework to tackle client drift by decomposing it into feature drift and prototype drift. We mitigate the feature and prototype drift by introducing Feature divergence loss and prototype contrastive loss. We also introduce a novel aggregation method for learnable prototypes, which prioritizes prototypes based on their local utility. Experimental results on the HAM10000 and Matek-19 datasets demonstrate that FedProIn outperforms existing federated learning methods. Future work will explore adapting FedProIn for personalized FL by eliminating the dependency on a global model to enhance local adaptability. We also aim to extend the framework to heterogeneous FL settings, enabling collaboration across clients with diverse model architectures.

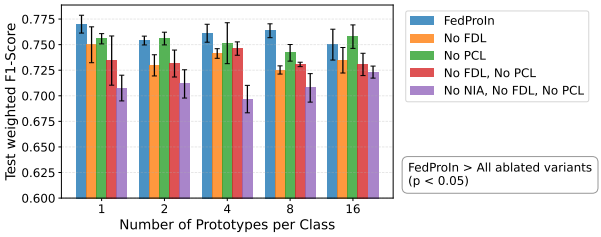

Fig. 3: **Ablation results.** Bar plots showing weighted F1-Score with varying prototype counts (M) on HAM10000. Aggregated across all runs, the full FedProIn method demonstrates statistically significant improvements over all ablated variants (Mann–Whitney U test; vs. No FDL $p < 0.001$, vs. No PCL $p < 0.05$, vs. No FDL & No PCL $p < 0.001$, vs. No NIA, FDL & PCL $p < 0.001$).

Acknowledgments. This work was supported by DBT/Wellcome Trust India Alliance Fellowship [IA/E/22/1/506763], Council of Scientific & Industrial Research (CSIR) ASPIRE program [25WS(013)/2023-24/EMR-II/ASPIRE], Start-up Research Grant [SRG/2023/001406] from the Science and Engineering Research Board and Pratiksha Trust, Bangalore, India [FG/PTCH-23-1004].

Disclosure of Interests. The authors have no competing interests to declare that are relevant to the content of this article.

References

1. Deng, J., Dong, W., Socher, R., Li, L.J., Li, K., Fei-Fei, L.: Imagenet: A large-scale hierarchical image database. In: 2009 IEEE conference on computer vision and pattern recognition. pp. 248–255. Ieee (2009)
2. GDPR Info: General data protection regulation (gdpr). https://gdpr-info.eu/ (2026), last accessed: 2026-01-25
3. Guan, H., Yap, P.T., Bozoki, A., Liu, M.: Federated learning for medical image analysis: A survey. Pattern recognition **151**, 110424 (2024)
4. Huang, W., Ye, M., Shi, Z., Li, H., Du, B.: Rethinking federated learning with domain shift: A prototype view. In: 2023 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR). pp. 16312–16322. IEEE (2023)
5. Kingma, D.P., Ba, J.: Adam: A method for stochastic optimization. arXiv preprint arXiv:1412.6980 (2014)
6. Li, Q., He, B., Song, D.: Model-contrastive federated learning. In: Proceedings of the IEEE/CVF conference on computer vision and pattern recognition. pp. 10713–10722 (2021)
7. Li, T., Sahu, A.K., Zaheer, M., Sanjabi, M., Talwalkar, A., Smith, V.: Federated optimization in heterogeneous networks. Proceedings of Machine learning and systems **2**, 429–450 (2020)
8. Matek, C., Schwarz, S., Spiekermann, K., Marr, C.: Human-level recognition of blast cells in acute myeloid leukaemia with convolutional neural networks. Nature Machine Intelligence **1**(11), 538–544 (2019)

9. Matthews, B.W.: Comparison of the predicted and observed secondary structure of t4 phage lysozyme. Biochimica et Biophysica Acta (BBA)-Protein Structure **405**(2), 442–451 (1975)
10. McMahan, B., Moore, E., Ramage, D., Hampson, S., y Arcas, B.A.: Communication-efficient learning of deep networks from decentralized data. In: Artificial intelligence and statistics. pp. 1273–1282. Pmlr (2017)
11. Mu, X., Shen, Y., Cheng, K., Geng, X., Fu, J., Zhang, T., Zhang, Z.: Fedproc: Prototypical contrastive federated learning on non-iid data. Future Generation Computer Systems **143**, 93–104 (2023)
12. Paszke, A., Gross, S., Massa, F., Lerer, A., Bradbury, J., Chanan, G., Killeen, T., Lin, Z., Gimelshein, N., Antiga, L., et al.: Pytorch: An imperative style, high-performance deep learning library. Advances in neural information processing systems **32** (2019)
13. Shuai, X., Shen, Y., Jiang, S., Zhao, Z., Yan, Z., Xing, G.: Balancefl: Addressing class imbalance in long-tail federated learning. In: 2022 21st ACM/IEEE International Conference on Information Processing in Sensor Networks (IPSN). pp. 271–284. IEEE (2022)
14. Tan, Y., Long, G., Liu, L., Zhou, T., Lu, Q., Jiang, J., Zhang, C.: Fedproto: Federated prototype learning across heterogeneous clients. In: Proceedings of the AAAI conference on artificial intelligence. vol. 36, pp. 8432–8440 (2022)
15. Tan, Y., Long, G., Ma, J., Liu, L., Zhou, T., Jiang, J.: Federated learning from pre-trained models: A contrastive learning approach. Advances in neural information processing systems **35**, 19332–19344 (2022)
16. Tschandl, P., Rosendahl, C., Kittler, H.: The ham10000 dataset, a large collection of multi-source dermatoscopic images of common pigmented skin lesions. Scientific data **5**(1), 180161 (2018)
17. U.S. Department of Health and Human Services: Hipaa home. https://www.hhs.gov/hipaa/index.html (2025), last accessed: 2025-01-12
18. Wang, L., Bian, J., Zhang, L., Chen, C., Xu, J.: Taming cross-domain representation variance in federated prototype learning with heterogeneous data domains. Advances in Neural Information Processing Systems **37**, 88348–88372 (2024)
19. Yang, H.M., Zhang, X.Y., Yin, F., Liu, C.L.: Robust classification with convolutional prototype learning. In: Proceedings of the IEEE conference on computer vision and pattern recognition. pp. 3474–3482 (2018)
20. Zhang, J., Liu, Y., Hua, Y., Cao, J.: Fedtgp: Trainable global prototypes with adaptive-margin-enhanced contrastive learning for data and model heterogeneity in federated learning. In: Proceedings of the AAAI conference on artificial intelligence. vol. 38, pp. 16768–16776 (2024)