Adaptive Confidence-Based Prototype Aggregation for Federated Prototype Learning on Non-IID Data

1st Yohannis Kifle Telila
Department of Electrical and Computer Engineering
Western University
ytelila@uwo.ca

2nd Anojan Yogenthiran
Department of Computer Science
Western University
ayogenth@uwo.ca

***Abstract*—Federated Learning (FL) is a distributed machine learning approach that enables multiple nodes to collaboratively train a shared model while keeping their data private. It offers several key advantages, including enhanced data privacy, security and scalability, by keeping data local and exchanging only model updates through the communication network. Despite its advantages, FL faces significant challenges with non-identical data distributions (non-IID), leading to biased updates, slower convergence, and reduced global model performance. Prototype Federated Learning is an approach proposed to address the challenges of non-IID data heterogeneity in federated learning by leveraging class prototypes. These prototypes, computed as the mean embeddings of feature representations for each class, are aggregated by the server to generate global prototypes. Clients then adjust their local feature embeddings to align with these global prototypes, improving consistency across clients and enhancing the generalization capability of each client model. However, existing prototype-based federated learning approaches treat all embeddings equally when computing prototypes, overlooking the varying quality and reliability of individual embeddings. A key limitation of existing prototype-based Federated Learning (FL) approaches is their assumption that all embeddings contribute equally when computing prototypes. This uniform treatment overlooks the varying quality and reliability of individual embeddings generated by clients. In real-world non-iid scenarios, clients often have heterogeneous data characterized by imbalances or insufficient representation of certain classes. Consequently, low-quality embeddings from clients with less representative data can disproportionately influence the computed prototypes, leading to suboptimal global aggregation. To address these limitations, we propose Adaptive Confidence-Based Prototype Aggregation (ConfFedProto), a novel approach that incorporates quality scores(confidence) to weigh feature embedding based on their reliability. By dynamically adjusting the influence of each embedding during prototype computation, our method enhances the robustness of prototypes and generalization of the local models across clients. Experimental results on benchmark datasets demonstrate notable improvements in model performance compared to prototype-based federated learning approaches, while also enhancing communication efficiency in federated learning settings. These findings underscore the potential of confidence-based prototype aggregation for addressing the inherent challenges of federated learning in heterogeneous environments.**

***Index Terms*—Federated Learning, Non-IID Data, Prototype Aggregation, Confidence Weighting, Machine Learning**

I. Introduction

Federated Learning (FL) is a decentralized approach to machine learning that enables collaborative model training across distributed datasets while maintaining data privacy [1]. Unlike traditional centralized methods, FL keeps data localized, reducing the need to transfer sensitive information to a central server. This approach addresses key concerns related to privacy and security, making it particularly suitable for domains such as healthcare, finance, and energy forecasting, where data confidentiality is critical.

In Federated Learning (FL), devices independently train models on their local datasets and share model updates with a central server, which aggregates them to build a global model. This decentralized approach preserves data privacy by ensuring that raw data never leaves the client devices. However, it presents significant challenges in real-world applications, particularly when data distributions across clients are non-independent and identically distributed (non-IID). Such data heterogeneity in client data can reduce the global model's performance [2]. To address these limitations, prototype-based approaches, including Federated Prototype Learning (FedProto) [3] and Prototypical Contrastive Federated Learning (FedProc) [4], have been proposed. These methods focus on learning and utilizing class-level prototypes, which are mean embeddings of feature representations for each class. Instead of exchanging raw model updates or gradients, clients compute prototypes from their local data and share them with the server. The server aggregates these prototypes to align class-level representations across clients, thereby reducing the impact of heterogeneity. This prototype-based federated learning method facilitates more effective knowledge sharing and enables the local models to better generalize across diverse and distributed client data.

This research project proposes ”Adaptive Confidence-Based Prototype Aggregation for Federated Prototype Learning on Non-IID Data” (ConfFedProto), a novel framework that addresses the limitations of existing methods. By integrating a confidence-weighted aggregation mechanism, ConfFedProto dynamically adjusts the influence of client prototypes based on their reliability, prioritizing contributions from clients with higher-quality data. This adaptive strategy improves global model accuracy, mitigates the adverse effects of client heterogeneity, and enhances the robustness of FL systems in

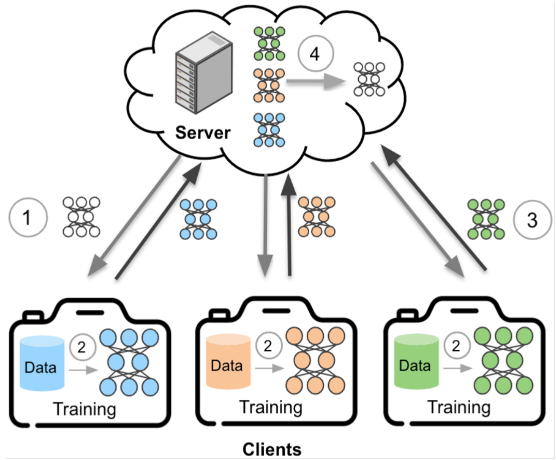

Fig. 1. Federated learning training process [5]

non-IID settings.

Key contributions of this work include:

- Proposed a confidence-based prototype aggregation framework that enhances the scalability and robustness of FL systems.
- Empirical validation demonstrating improved convergence and accuracy in non-IID environments.
- Comprehensive experimentation on benchmark datasets (MNIST, EMNIST, CIFAR-10), showing superior performance compared to state-of-the-art methods such as FedAvg [1], FedProto [3], and FedProc [4].

II. Background

This section outlines the foundational concepts that form the basis of this research, focusing on Federated Learning (FL) and Federated Prototype Learning.

A. Federated learning (FL)

Federated Learning (FL) is a decentralized machine learning paradigm aims to train a global model collaboratively across multiple clients (e.g., devices) without sharing their private data [6]. Each client trains a local model using its private dataset, and a central server aggregates the locally computed updates to form a global model. This iterative process enables collaborative learning while keeping sensitive data localized. The FL diagram shown in Fig. 1 illustrates a central server coordinating the training process while multiple clients perform local computations. The process begins with the server initializing the global model and sending it to selected clients. The clients train the model on their local datasets and share their updates with the server. The server aggregates these updates to refine the global model and sends the updated model back to the clients for the next round of training. This iterative process continues until convergence is achieved.

Mathematically, the goal of FL is to minimize a global objective function $F(w)$, which is an aggregation of local objective functions $F_i(w)$ across K clients. The global objective is defined as:

$$F(w) = \sum_{i=1}^{K} \frac{n_i}{n} F_i(w),$$

where w represents the global model parameters, $F_i(w)$ is the local objective function for client i, n_i is the number of data samples on client i, and $n = \sum_{i=1}^{K} n_i$ is the total number of data samples across all clients.

Aggregation methods in FL are crucial for combining updates from clients to train a global model effectively. The most common method, Federated Averaging (FedAvg) [6], computes a weighted average of client updates based on their data size, while variants like Federated Weighted Averaging (FedWeighted) [7] incorporate custom weights for factors such as reliability or data quality. Robust methods, such as Median Aggregation [8], address outliers, while Proximal Aggregation (FedProx) [9] mitigates issues arising from non-IID data. Advanced techniques, including Adaptive Federated Optimization (FedOpt) [10] and clustering-based approaches, further enhance FL's adaptability, robustness, and privacy. The choice of aggregation method depends on factors like data heterogeneity, client reliability, and privacy requirements of the application.

B. Non-IID Data Challenges in FL

Federated learning often results in non-IID data across clients, which poses significant challenges to model training and convergence. Non-IID data can be in various forms of heterogeneity, including feature, label, and task heterogeneity, each of which uniquely impacts FL systems. Feature heterogeneity occurs when the distribution of input features varies significantly across clients. For instance, in healthcare, different hospitals might use varying imaging equipment, leading to differences in the pixel distributions of medical images. This form of heterogeneity can result in local models optimizing for feature representations that do not generalize well to other clients, reducing the global model's performance.

Label heterogeneity, the focus of this work, arises when the label distributions differ across clients. In real-world scenarios, clients often have access to data from a limited subset of classes. For example, in a handwriting recognition task, some clients might predominantly have data for only a few digits (e.g., 0, 1, and 2), while others might have data for the remaining digits. Such imbalanced label distributions lead to biased updates from clients, slowing global model convergence and decreasing its generalization capability.

C. Federated Prototype Learning

Federated Prototype Learning (FPL) is an extension of Federated Learning (FL) that addresses challenges related to data heterogeneity and communication efficiency. Instead of relying on traditional aggregation of model updates or gradients, FPL uses prototypes, which are representative vectors capturing the key features or characteristics of data classes.

Prototypes are computed on each client as the average of the feature embeddings for all data points belonging to a specific class. Mathematically, for a class j on client i, the prototype $\mathbf{p}_{i,j}$ is computed as:

$$\mathbf{p}_{i,j} = \frac{1}{|D_{i,j}|} \sum_{\mathbf{x} \in D_{i,j}} f(\mathbf{x}; \theta_i),$$

Where $D_{i,j}$ represents the set of data points in class j on client i, $f(\mathbf{x}; \theta_i)$ is the feature embedding function (e.g., the output of an intermediate network layer), θ_i denotes the local model parameters on client i.

After computing the prototypes locally, each client transmits these prototypes to the central server instead of the full model updates. The server aggregates the prototypes from all participating clients to update the global prototypes.

$$\bar{P}_j = \frac{1}{|N_j|} \sum_{i \in N_j} \frac{|D_{i,j}|}{N_j} \mathbf{p}_{i,j} \tag{1}$$

This aggregation process ensures that the global prototypes represent the combined knowledge from all clients, even in the presence of non-IID data distributions.

III. Related Work

Statistical heterogeneity among clients, often referred to as the non-IID problem, represents one of the most significant challenges in federated learning (FL). Prototype learning has emerged as a promising technique to address data heterogeneity in FL. It has been shown to facilitate faster convergence while addressing challenges posed by data heterogeneity [11].

Tan et al. [12] introduced FedProto, which uses prototypes to represent key features of each data class in a client's local dataset. Instead of sharing local model updates, clients compute these prototypes and sends them to the central server, and align their local models with aggregated global prototypes. Mu et al. [13] proposed FedProc, which utilizes prototypes for contrastive federated learning. FedProc improves prototype alignment across clients and mitigates the impact of data heterogeneity. Their work offers valuable insights into prototype reliability and its role in mitigating challenges posed by data heterogeneity.

Li et al. [9] proposed FedProx, an extension of FedAvg that incorporates a proximal term in the optimization objective. This term improves the stability and accuracy of FL under system and data heterogeneity, particularly when clients have diverse computational capabilities or data distributions. However, FedProx focuses primarily on global model optimization and does not inherently address personalization for individual clients [9].

Moreover, recent advancements such as FedNTProto [14], have further enhanced prototype-based FL by introducing mechanisms to address both statistical and system heterogeneity. FedNTProto focuses on improving the robustness and efficiency of prototype aggregation in FL by introducing strategies that adapt to varying data distributions and client capabilities. By leveraging prototype-based representations, FedNTProto reduces the impact of data imbalances and non-IID distributions across clients, ensuring that aggregated prototypes better reflect the underlying data patterns of the entire federation.

A common limitation among existing prototype-based methods is the assumption of uniform reliability across all embeddings during aggregation. This assumption fails to account for variations in data quality and representation reliability among clients, especially those with insufficient or imbalanced datasets. Confidence-weighted approaches have been proposed to dynamically adjust the influence of individual embeddings based on their quality, enhancing the robustness and generalization of global prototypes. These methods aim to address the limitations of uniform prototype aggregation and optimize learning in heterogeneous environments.

IV. Methodology

This section provides a comprehensive methodology for implementing the proposed confFedProto framework. Each component is described in detail below, starting with an overview of the framework and followed by specific implementation steps, dataset preparation, and experimental setup.

A. Framework Overview

The proposed confFedProto framework introduces confidence based prototype learning into federated learning (FL). Fig. 3 illustrates the overall architecture of the confFedProto framework, highlighting the interaction between the client-side operations and the server-side global

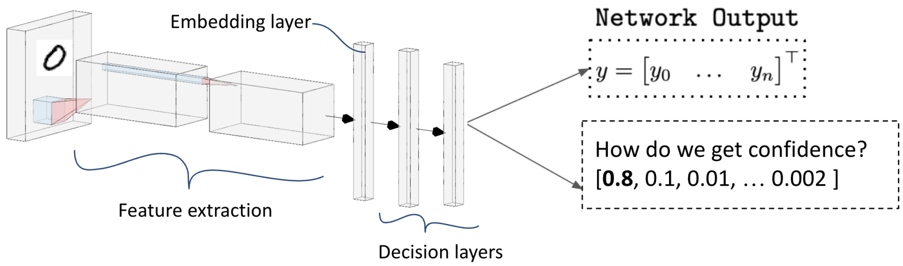

Fig. 2. Prototypes: The embedding layer extracts features that are used to calculate confidence scores for each class. The confidence score represents the probability assigned by the model to the input label and is used to compute the corresponding prototype for the class.

aggregation.

Each client trains their local data x_j through an embedding function $f_i(\omega; x_j)$, which encodes the input into a latent representation z_j. A decision function $g_i(v_i, z_j)$ takes the latent embedding z_j to produce an output $\hat{y}_j$. A supervised loss L_S ensures the predicted output aligns with the actual label y_j.

Feature exraction: The embedding function $f_i(\phi_i; x_j)$, parameterized by ϕ_i, extracts latent features z_j from the input x_j. These embeddings form the backbone of the ConfFedProto framework, as they are used to compute local prototypes for each class:

$$z_i = f_i(\phi_i; x),$$

where z_i represents the embedding vector of input x. These embeddings are used locally for prototype computation and globally for alignment with server-provided prototypes. The quality of these embeddings directly impacts the effectiveness of prototype-based aggregation and overall model generalization.

Decision Layers: In the context of the ConfFedProto framework, predictions for an input x are generated using the decision function $g_i(\nu_i; z_j)$, parameterized by ν_i. Here, $z_j = f_i(\omega_i; x_j)$ represents the embedding generated by the representation layer. The overall prediction function for the i-th client can be expressed as:

$$F_i(\phi_i, \nu_i) = g_i(\nu_i) \circ f_i(\phi_i),$$

where $f_i(\phi_i)$ is the embedding function, and $g_i(\nu_i)$ is the decision function. To simplify, the parameters (ϕ_i, ν_i) are collectively denoted as ω_i.

This hierarchical composition ensures that the decision layers map embeddings z_j to outputs $\hat{y}_j$ based on the parameters ν_i, allowing client-specific and global adjustments to prediction outputs during training.

Prototype: Each client maintains a set of local prototypes p_j^l. Local prototypes are computed as the:

$$p_{i,j}^{(l)} = \frac{1}{|D_{i,j}|} \sum_{(x,y) \in D_{i,j}} c(x) f_i(\phi_i, x), \tag{2}$$

where $D_{i,j}$ represents the set of latent features assigned to prototype p_j. $c(x)$ represents the probability assigned to a specific class within the model's output probability distribution. It reflects the model's confidence that the input x belongs to that particular class.

B. Local Model Update

A regularization loss L_R is used to align these local prototypes p_j with the global prototypes $P_j^{(g)}$, promoting consistency between client-specific and globally shared patterns. The local prototypes will then be sent to the server for aggregation.

$$\mathcal{L}(D_i, \omega_i) = \mathcal{L}_S(\mathcal{F}_i(\omega_i; x), y) + \lambda \cdot \mathcal{L}_R(f_i(\phi_i, x), P_j^{(g)}), \tag{3}$$

This loss function combines two components to optimize the model in the uFedProto framework. The supervised loss $\mathcal{L}_s$ ensures that the model learns to make accurate predictions based on the client's local data. Regularized loss $\mathcal{L}_R$ enforces alignment between the client's local embedding $(f_i(\phi_i, x))$ and the corresponding global prototype($P_j^{(g)}$).

$$\mathcal{L}_R = \sum_j d(f_i(\phi_i, x), P_j^{(g)}), \tag{4}$$

where d is a distance metrics, mean squared error(MSE).

C. Global update

On the server side, the global prototypes $P_j^{(g)}$ are computed by aggregating the local prototypes from all clients using the formula:

$$P_j^{(g)} = \frac{1}{m} \sum_{i=1}^{m} P_{i,j}^{(l)} \tag{5}$$

where m represents the number of clients, and $P_{i,j}^{(l)}$ is the set of local prototypes from client i. These prototypes are updated iteratively on each global round. During each global round, the server sends a set of global prototypes $\{P_j^{(g)}\}$ to all clients. Each client then performs a local update using

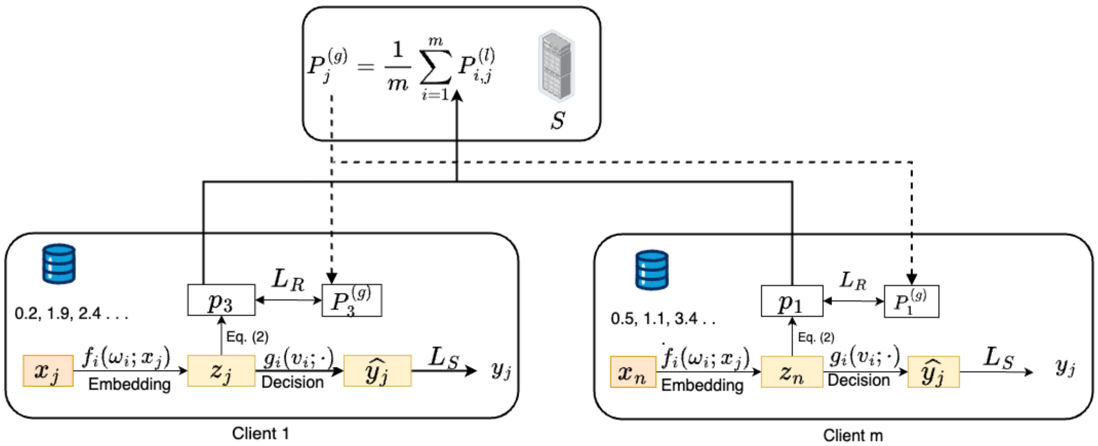

Fig. 3. An overview of confFedproto

Algorithm 1 confFedProto: Federated Prototype Learning

Input: Local datasets $\{D_i\}_{i=1}^m$, model parameters $\{\omega_i\}_{i=1}^m$, number of clients m

Server executes:

for each round $T = 1, 2, \dots$ **do**
  for each client $i \in \{1, \dots, m\}$ **in parallel do**
    Send global prototypes $\{P_j^{(g)}\}_{j=1}^k$ to client i.
    $\{P_{i,j}^{(l)}\}_{j=1}^k \leftarrow \text{LocalUpdate}(D_i, \{P_j^{(g)}\}_{j=1}^k, \omega_i)$
  end for
  Update global prototypes using equation(5)
end for

LocalUpdate$(D_i, \{P_j^{(g)}\}_{j=1}^k, \omega_i)$:

for each batch $(x_j, y_j) \in D_i$ **do**
  Compute feature representation: $z_j = f_i(\omega_i; x_j)$.
  Compute confidence $c(x)$ for each input.
  Compute loss $\mathcal{L}(D_i, \omega_i)$ using Eq. (3).
  Update model parameters ω_i using $\mathcal{L}(D_i, \omega_i)$.
  Update local prototype $\mathcal{C}_{i,p_j}^{(l)}$ using Eq. (2).
end for
return $\{p_{i,j}^{(l)}\}_{j=1}^k$

their dataset D_i, the received global prototypes, and their local model parameters ω_i. The local update process involves computing feature representations of data points, estimating the confidence scores $c(x)$ for each class, and adjusting local prototypes based on these representations and confidences. Clients also optimize their local models by minimizing a predefined loss function that ensures alignment with both their local data and the global prototype structure. After local updates, clients return their updated local prototypes to the server.

The server aggregates these local prototypes to update the global prototypes, which are used to guide the training process in subsequent rounds. The algorithm is particularly well-suited for non-iid (non-independent and identically distributed) data scenarios, where traditional federated learning methods may struggle to converge or achieve good accuracy.

D. Dataset

We implement a typical federated learning framework in which each client retains ownership of its local data and communicates with a central server to exchange model information. This setup enables collaborative training while maintaining data privacy, as raw data remains decentralized. For our experiments, we utilize three widely used benchmark datasets: MNIST [15], FEMNIST [16], and CIFAR-10 [17], each chosen to represent varying levels of complexity and heterogeneity in data distributions.

To ensure appropriate model performance for each dataset, we employ tailored architectures. We use a multi-layer Convolutional Neural Network (CNN) consisting of two convolutional layers followed by two fully connected layers, optimized for extracting spatial features from simpler image datasets. This combination of datasets and architectures provides a robust foundation for evaluating the performance and adaptability of the federated learning framework.

E. Data Preparation

The datasets were divided into partitions to simulate the federated learning environment, where each partition represents the local data of a single client. To introduce the non-iid nature of the data, label distributions across the partitions were manipulated using a Dirichlet distribution. This method controls the degree of label imbalance among clients, with the Dirichlet parameter α determining the level of skewness. Smaller α values create highly imbalanced partitions, where clients may only have data for a limited subset of labels, while larger α values result in more uniform distributions. This process was applied to all datasets, including MNIST, FEMNIST, and CIFAR-10.

Figure 4 presents the label distribution for the CIFAR-10 dataset under a non-iid setup. The figure clearly illustrates

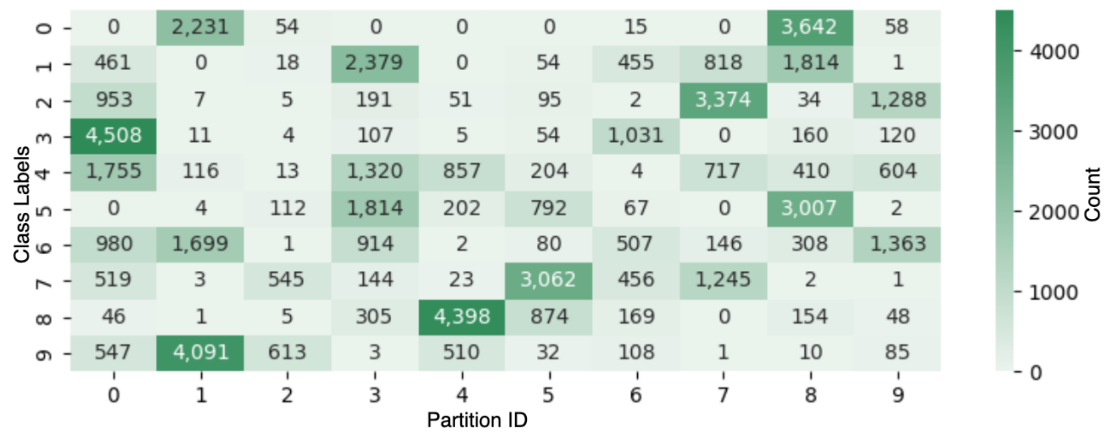

Fig. 4. CIFAR-10 Non-iid data distribution, $\alpha = 0.3$

the label non-iid distribution across clients, which reflects the variability in data heterogeneity often encountered in federated learning applications. This preparation method allows for a robust evaluation of the confFedProto framework's ability to address the challenges posed by non-iid client data.

V. Experiments and Results

This section presents the performance comparison of the proposed confFedProto framework against baseline methods, including FedAvg and FedProto, across three benchmark datasets (MNIST, CIFAR-10, and EMNIST) under varying levels of data heterogeneity. The experiments were conducted on a Tesla V100-SXM2-16GB GPU, with 16GB memory and CUDA version 12.4. The heterogeneity is controlled by the Dirichlet distribution parameter α, where smaller α values correspond to higher data imbalance across clients. The key results are summarized in Table I.

Dataset	Method	$\alpha = 0.1$	$\alpha = 0.3$	$\alpha = 0.5$
MNIST	FedAvg	88.34	95.29	97.16
	FedProto	98.11	98.70	98.67
	confFedproto	**98.43**	**98.79**	**98.91**
CIFAR10	FedAvg	47.18	62.07	63.65
	FedProto	56.60	73.57	78.59
	confFedproto	**57.29**	**75.02**	**79.71**
EMNIST	FedAvg	73.89	82.59	84.45
	FedProto	76.32	89.59	91.01
	confFedproto	**77.09**	**90.40**	**94.42**

Table I. Model comparison with different α values.

The results demonstrate that confFedProto consistently outperforms the baseline methods, showcasing its robustness and adaptability under varying levels of data heterogeneity. In scenarios with high data heterogeneity (small α), confFedProto showed significant performance improvements over FedAvg, particularly in challenging datasets like CIFAR-10 and EMNIST. For instance, at $\alpha = 0.1$, confFedProto achieved the highest accuracy across all datasets, demonstrating its ability to handle severe label imbalance effectively. As the data became less heterogeneous (larger α), the performance gap between confFedProto and the other methods narrowed, but confFedProto maintained superior accuracy, underscoring its consistent efficiency regardless of the degree of data imbalance.

Overall, the results highlight the strength of confFedProto in addressing key challenges in federated learning, particularly under non-IID data distributions. By integrating confidence-based prototype learning, the framework enhances global model performance while maintaining scalability and adaptability across diverse datasets and varying levels of data heterogeneity.

A. *Communication efficiency*

One of the key advantages of prototype-based federated learning, as demonstrated by FedProto and confFedProto, is their significant reduction in communication overhead compared to traditional methods like FedAvg.

Method	Model	Total Parameters
FedAvg	CIFAR 10	163690
	MNIST	44426
	EMNIST	47571
FedProto	CIFAR 10	1270
	MNIST	1270
	EMNIST	1270
confFedProto	CIFAR 10	1270
	MNIST	1270
	EMNIST	1270

Table II. Number of Parameters shared in FL.

Table II shows the total number of parameters shared during the federated learning process for each method. In FedAvg, the full model parameters are exchanged between the clients and the server during each communication round. This results in a large number of parameters being transferred, particularly for complex datasets like CIFAR-10, where the total number of parameters reaches 163,690. In contrast, both FedProto and confFedProto only exchange prototypes, significantly reducing the communication cost to a

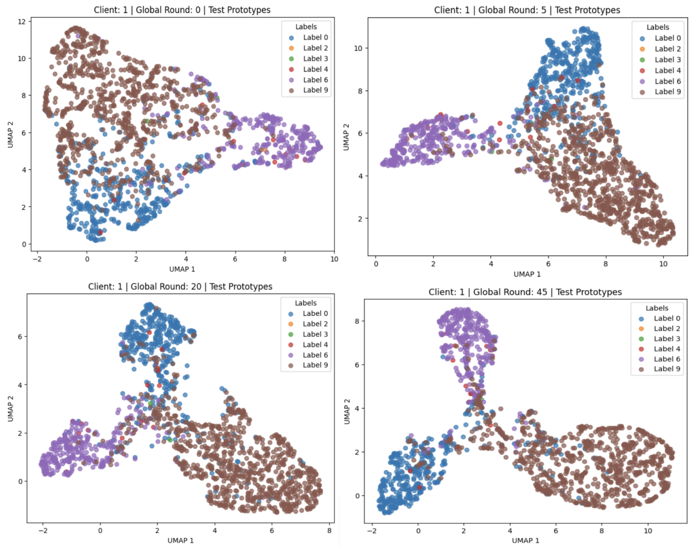

Fig. 5. Visualization of the test set embeddings across global rounds (0, 5, 20, and 45).

constant value of 1,270 parameters, irrespective of the dataset.

This reduction in communication cost not only makes confFedProto more scalable but also enhances its suitability for resource-constrained environments where bandwidth and computational resources are limited. The results clearly demonstrate that prototype-based federated learning methods are more communication-efficient, offering a practical advantage without compromising model performance, even under challenging conditions like high data heterogeneity.

B. Representation layer visualization

The visualizations in Figure 5 depict the evolution of the representation layer for the test set across different global rounds (0, 5, 20, and 45) in the confFedProto framework.

At the start of the training (Global Round 0), the embeddings for the test set are scattered, with significant overlap among different labels. This lack of clear clustering indicates that the model has not yet learned distinct representations for the various classes.

By Global Round 5, the embeddings start to form distinguishable clusters, with some labels showing partial separation. As training progresses, the clusters become increasingly distinct, with noticeable improvement by Global Round 20 and more refined separation at Global Round 45. The improved clustering highlights the efficacy of confFedProto in learning robust and distinct representations.

This result indicates the model's convergence towards a stable and accurate representation of the test data. The use of prototypes ensures that the embeddings for each label are consistently aligned across clients, facilitating robust and scalable learning in a federated setting.

VI. Discussion

In this section, we discuss the advantages of ConfFedProto from model inference and privacy preservation.

A. Addition of a new client

Similar to Fedoroto [3], the global model in ConfFedProto is not a trained model but a set of confidence-weighted class prototypes. When a new client joins the network, it can initialize its local model using the representation layers of a pre-trained model, such as ResNet18 on ImageNet, with randomly initialized decision layers. The client then recieves the global prototypes for the classes represented in its local dataset and fine-tunes its local model by minimizing the local objective function. This enables ConfFedProto to support new clients with new model architectures.

B. Privacy Preservation

ConfFedProto enhances privacy preservation by exchanging prototypes instead of raw model parameters between the server and clients. Prototypes are low-dimensional vectors generated by averaging feature embeddings, which inherently protects the privacy of local data. This process is irreversible, ensuring that attackers cannot reconstruct raw data from prototypes. Additionally, since ConfFedProto does not require the transfer of complete local models, the risk of exposing sensitive client information is minimized. Furthermore, ConfFedProto can be integrated with advanced privacy-preserving techniques, such as differential privacy [18] or secure multi-party computation [19], to further enhance the security and reliability of the system.

VII. Conclusion

This study introduces ConfFedProto, an adaptive confidence-based prototype aggregation framework designed to address challenges in Federated Prototype Learning

(FPL) on non-IID data. By incorporating confidence-weighted aggregation, ConfFedProto prioritizes high-quality contributions from clients, significantly enhancing the robustness and accuracy of the global model. Experimental evaluations on benchmark datasets, including MNIST, CIFAR-10, and EMNIST, demonstrate the superiority of ConfFedProto over existing methods like FedAvg and FedProto, particularly in scenarios with high data heterogeneity.
Key findings from this research include the ability of ConfFedProto to handle severe label imbalances, reduce communication overhead through prototype-based aggregation, and achieve consistent model performance across diverse datasets. Additionally, the visualization of representation layers confirms the effectiveness of the framework in learning distinct and robust embeddings.

The results underscore the potential of confidence-based prototype aggregation in improving the scalability, efficiency, and adaptability of federated learning systems. Future work may explore extending this approach to other domains with complex non-IID characteristics and integrating advanced optimization techniques to further enhance its performance.

VIII. Limitations and Future Work

While the proposed ConfFedProto framework demonstrates significant improvements in addressing label heterogeneity and non-IID data challenges in federated learning, several limitations remain that provide opportunities for future research.

While this work primarily focuses on label heterogeneity, real-world federated learning systems often encounter feature heterogeneity. Feature heterogeneity introduce additional layers of complexity that are not directly addressed by ConfFedProto. Expanding the framework to handle feature and label heterogeneity would significantly broaden its applicability in diverse federated learning environments.

References

[1] D. R. S. H. H. McMahan, E. Moore and B. A. Arcas, "Communication-efficient learning of deep networks from decentralized data," in *Proceedings of the 20th International Conference on Artificial Intelligence and Statistics (AISTATS)*, 2017.

[2] A. S. S. T. Li and V. S. Subramanian, "Fedprox: Federated optimization in heterogeneous networks," in *Proceedings of the 2nd MLSys Conference*, 2020.

[3] J. Z. Y. Tan and Y. Li, "Fedproto: Federated prototype learning across heterogeneous clients," in *NeurIPS Workshop*, 2021.

[4] R. Z. L. Mu and X. Chen, "Fedproc: Prototypical contrastive federated learning on non-iid data," in *Proceedings of the 31st ACM International Conference on Information and Knowledge Management (CIKM)*, 2022.

[5] S. AI, "Recent breakthroughs tackle challenges in federated learning," 2024, accessed: 2024-12-31. [Online]. Available: https://ai.sony/blog/Recent-Breakthroughs-Tackle-Challenges-in-Federated-Learning/

[6] H. B. McMahan, E. Moore, D. Ramage, S. Hampson, and B. A. y Arcas, "Communication-efficient learning of deep networks from decentralized data," 2023. [Online]. Available: https://arxiv.org/abs/1602.05629

[7] M. Hong, S.-K. Kang, and J.-H. Lee, "Weighted averaging federated learning based on example forgetting events in label imbalanced non-iid," *Applied Sciences*, vol. 12, no. 12, 2022. [Online]. Available: https://www.mdpi.com/2076-3417/12/12/5806

[8] K. Pillutla, S. M. Kakade, and Z. Harchaoui, "Robust aggregation for federated learning," *IEEE Transactions on Signal Processing*, vol. 70, p. 1142–1154, 2022. [Online]. Available: http://dx.doi.org/10.1109/TSP.2022.3153135

[9] T. Li, A. K. Sahu, M. Zaheer, M. Sanjabi, A. Talwalkar, and V. Smith, "Federated optimization in heterogeneous networks," 2020. [Online]. Available: https://arxiv.org/abs/1812.06127

[10] S. Reddi, Z. Charles, M. Zaheer, Z. Garrett, K. Rush, J. Konečný, S. Kumar, and H. B. McMahan, "Adaptive federated optimization," 2021. [Online]. Available: https://arxiv.org/abs/2003.00295

[11] Y. Qiao, S.-B. Park, S. M. Kang, and C. S. Hong, "Prototype helps federated learning: Towards faster convergence," 2023. [Online]. Available: https://arxiv.org/abs/2303.12296

[12] Y. Tan, G. Long, L. Liu, T. Zhou, Q. Lu, J. Jiang, and C. Zhang, "Fedproto: Federated prototype learning across heterogeneous clients," 2022. [Online]. Available: https://arxiv.org/abs/2105.00243

[13] X. Mu, Y. Shen, K. Cheng, X. Geng, J. Fu, T. Zhang, and Z. Zhang, "Fedproc: Prototypical contrastive federated learning on non-iid data," 2021. [Online]. Available: https://arxiv.org/abs/2109.12273

[14] T.-K. Tran, H.-P. Tran, T.-L. Le, and T.-H. Tran, "Fedntproto: A prototype-based approach for personalized federated learning," in *2024 International Conference on Multimedia Analysis and Pattern Recognition (MAPR)*, 2024, pp. 1–6.

[15] L. Deng, "The mnist database of handwritten digit images for machine learning research," *IEEE Signal Processing Magazine*, vol. 29, no. 6, pp. 141–142, 2012.

[16] S. Caldas, S. M. K. Duddu, P. Wu, T. Li, J. Konečný, H. B. McMahan, V. Smith, and A. Talwalkar, "Leaf: A benchmark for federated settings," 2019. [Online]. Available: https://arxiv.org/abs/1812.01097

[17] A. Krizhevsky, "Learning multiple layers of features from tiny images," Tech. Rep., 2009.

[18] Z. Ji, Z. C. Lipton, and C. Elkan, "Differential privacy and machine learning: a survey and review," 2014. [Online]. Available: https://arxiv.org/abs/1412.7584

[19] C. Zhao, S. Zhao, M. Zhao, Z. Chen, C.-Z. Gao, H. Li, and Y. an Tan, "Secure multi-party computation: Theory, practice and applications," *Information Sciences*, vol. 476, pp. 357–372, 2019. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S0020025518308338