Class-wise Balancing Data Replay for Federated Class-Incremental Learning

Zhuang Qi[1], Ying-Peng Tang[2], Lei Meng[1,*], Han Yu[2], Xiaoxiao Li[3], Xiangxu Meng[1]
[1]School of Software, Shandong University, China
[2]College of Computing and Data Science, Nanyang Technological University, Singapore
[3]Department of Electrical and Computer Engineering, The University of British Columbia, Canada
z_qi@mail.sdu.edu.cn, yingpeng.tang@ntu.edu.sg, lmeng@sud.edu.cn,
han.yu@ntu.edu.sg, xiaoxiao.li@ece.ubc.ca, mxx@sdu.edu.cn

Abstract

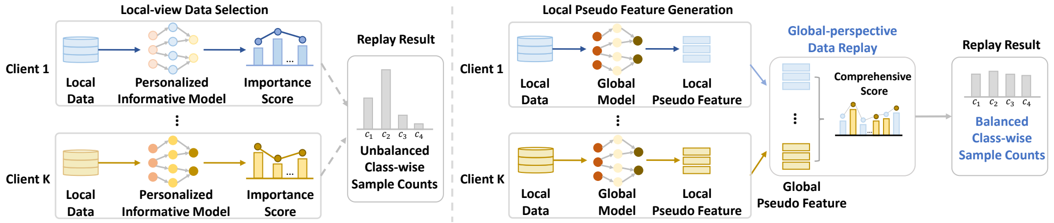
Federated Class Incremental Learning (FCIL) aims to collaboratively process continuously increasing incoming tasks across multiple clients. Among various approaches, data replay has become a promising solution, which can alleviate forgetting by reintroducing representative samples from previous tasks. However, their performance is typically limited by class imbalance, both within the replay buffer due to limited global awareness and between replayed and newly arrived classes. To address this issue, we propose a class-wise balancing data replay method for FCIL (FedCBDR), which employs a global coordination mechanism for class-level memory construction and reweights the learning objective to alleviate the aforementioned imbalances. Specifically, FedCBDR has two key components: 1) the global-perspective data replay module reconstructs global representations of prior task in a privacy-preserving manner, which then guides a class-aware and importance-sensitive sampling strategy to achieve balanced replay; 2) Subsequently, to handle class imbalance across tasks, the task-aware temperature scaling module adaptively adjusts the temperature of logits at both class and instance levels based on task dynamics, which reduces the model's overconfidence in majority classes while enhancing its sensitivity to minority classes. Experimental results verified that FedCBDR achieves balanced class-wise sampling under heterogeneous data distributions and improves generalization under task imbalance between earlier and recent tasks, yielding a 2%-15% Top-1 accuracy improvement over six state-of-the-art methods.

1 Introduction

Federated learning (FL) is a distributed machine learning paradigm that enables collaborative training of a shared global model across multiple data sources [1, 2, 3, 4]. It periodically performs parameter-level interaction between clients and the server instead of gathering clients' data, which can enhance data privacy while leveraging the diversity of distributed data sources to build a more generalized global model [5, 6, 7, 8]. Building upon this foundation, Federated Class-Incremental Learning (FCIL) extends FL by introducing dynamic data streams where clients sequentially encounter different task classes under non-independent and identically distributed data [9, 10, 11, 12]. However, this amplifies the inherent complexities of FL, as the global model must integrate heterogeneous and evolving knowledge from clients while mitigating catastrophic forgetting, despite having no or only limited access to historical data [13, 14, 15].

To address the challenge of catastrophic forgetting in FCIL, data replay has emerged as a promising strategy for retaining knowledge from previous tasks. Existing replay-based methods can be broadly

Preprint. Under review.



(a) Traditional Data Replay Strategy

(b) Global-perspective Data Replay Strategy

Figure 1: Motivation of the FedCBDR. Traditional data replay strategies typically focus on local information and, due to the lack of global awareness, often result in imbalanced class distributions during replay. FedCBDR aims to explore global information in a privacy-preserving manner and leverage it for sampling, which can alleviate the class imbalance problem.

categorized into two types: generative-based replay and exemplar-based replay. The former leverages generative models to synthesize representative samples from historical tasks [14, 16, 17]. Its core idea is to learn the data distribution of previous tasks and internalize knowledge in the form of model parameters, enabling the indirect reconstruction of prior knowledge through sample generation when needed [11, 18]. However, they often overlook the computational cost of training generative models and are inherently constrained by the quality and fidelity of the synthesized data [14, 19, 20]. In contrast, exemplar-based replay methods directly store real samples from previous tasks, avoiding the complexity of generative processes while leveraging high-quality raw data to ensure robust retention of prior knowledge [9, 19, 21, 22]. These methods rely on a limited set of historical samples to maintain the decision boundaries of previously learned task classes. However, due to the lack of a global perspective on data distribution across clients, these methods are prone to class-level imbalance in replayed samples, which undermines the model's ability to retain prior knowledge [21, 22].

To address these issues, this paper proposes a class-wise balancing data replay method for FCIL, termed FedCBDR, which incorporates the global signal to regulate class-balanced memory construction, aiming to achieve distribution-aware replay and mitigate the challenges posed by non-IID client data, as illustrated in Figure 1. Specifically, FedCBDR comprises two primary modules: 1) the global-perspective data replay (GDR) module reconstructs a privacy-preserving pseudo global representation of historical tasks by leveraging feature space decomposition, which enables effective cross-client knowledge integration while preserving essential attributes information. Furthermore, it introduces a principled importance-driven selection mechanism that enables class-balanced replay, guided by a globally-informed understanding of data distribution; 2) the task-aware temperature scaling (TTS) module introduces a multi-level dynamic confidence calibration strategy that combines task-level temperature adjustment with instance-level weighting. By modulating the sharpness of the softmax distribution, it balances the predictive confidence between majority and minority classes, enhancing the model's robustness to class imbalance between historical and current task samples.

Extensive experiments were conducted on three datasets with different levels of heterogeneity, including performance comparisons, ablation studies, in-depth analysis, and case studies. The results demonstrate that FedCBDR effectively balances the number of replayed samples across classes and alleviates the long-tail problem. Compared to six state-of-the-art existing methods, FedCBDR achieves a 2%-15% Top-1 accuracy improvement.

2 Related Work

2.1 Exemplar-based Replay Methods

In FCIL, exemplar-based replay methods aim to mitigate catastrophic forgetting by storing and replaying a subset of samples from previous tasks. They typically maintain a small exemplar buffer on each client, which is used during training alongside new task data to preserve knowledge of previously learned classes [9, 21, 22, 23]. For example, GLFC alleviates forgetting in FCIL by leveraging local exemplar buffers for rehearsal, while introducing class-aware gradient compensation and prototype-guided global coordination to jointly address local and global forgetting [9]. Moreover, Re-Fed introduces a Personalized Informative Model to strategically identify and replay task-relevant local samples, enhancing the efficiency of buffer usage and further reducing forgetting in heterogeneous client environments [21]. However, the lack of global insight in local sample selection often results in class imbalance, while the long-tailed distribution between replayed and current data is frequently overlooked, ultimately degrading the effectiveness of data replay [21, 22].

2.2 Generative-based Replay Methods

Generative replay methods aim to reconstruct the samples of past tasks through techniques such as generative modeling [11, 14, 18, 24, 25], which enables the model to revisit historical knowledge to mitigate catastrophic forgetting. Following this line of thought, TARGET generates pseudo features through a globally pre-trained encoder and performs knowledge distillation by aligning the current model's predictions with those of a frozen global model [18]; LANDER utilizes pre-trained semantic text embeddings as anchors to synthesize meaningful pseudo samples, and distills knowledge by aligning the model's predictions with class prototypes derived from textual descriptions [11]. However, these methods are typically limited by the high computational cost of training generative models and the suboptimal performance caused by low-fidelity pseudo samples [11, 18].

2.3 Knowledge Distillation-based Methods

Knowledge distillation-based methods generally follow two paradigms. The first focus om aligning the output predictions of the current model with those of previous models, which aims to preserve task-specific decision boundaries [26, 27, 28, 29, 30]. The second estimates the importance of model parameters for previously learned tasks and performs regularization to prevent forgetting [31, 32]. Both approaches avoid storing raw data but are prone to knowledge degradation over time, especially as the number of tasks increases [26, 31].

3 Preliminaries

We consider a federated class-incremental learning (FCIL) setting, where a central server aims to collaboratively train a global model with the assistance of K distributed clients. Each client k receives a sequence of classification tasks $\{\mathcal{D}_k^{(1)}, \mathcal{D}_k^{(2)}, \dots, \mathcal{D}_k^{(t)}\}$, where each task introduces a disjoint set of new classes. Upon the arrival of task t, the global model parameters θ_t are optimized to minimize the average loss over the union of all samples seen so far, i.e., $\mathbb{D}^t = \bigcup_{s=1}^{t} \bigcup_{k=1}^{K} \mathcal{D}_k^{(s)}$, by solving $\min_\theta \frac{1}{|\mathbb{D}^t|} \sum_{s=1}^{t} \sum_{k=1}^{K} \sum_{i=1}^{N_k^{(s)}} \mathcal{L}(f_k(x_{k,i}^{(s)}; \theta), y_{k,i}^{(s)})$.

In replay-based methods, each client maintains a memory buffer with a fixed budget of M samples. When task t arrives, the client selects up to N representative samples from each of the previous tasks $\{1, \dots, t-1\}$, subject to the total memory constraint. The resulting memory set is denoted by $\mathcal{B}_k^{(t-1)} = \bigcup_{s=1}^{t-1} \{(x_{k,i}^{(s)}, y_{k,i}^{(s)})\}_{i=1}^{N}$, where N is the number of samples stored per task and $\mathcal{B}_k^{(t-1)}$ satisfies $|\mathcal{B}_k^{(t-1)}| \leq M$. The local training set on client k then becomes $\mathcal{D}_{k,\mathrm{train}}^{(t)} = \mathcal{D}_k^{(t)} \cup \mathcal{B}_k^{(t-1)}$, combining current and replayed samples. Based on these local datasets, the server updates the global model by minimizing the aggregated loss: $\min_\theta \sum_{k=1}^{K} \sum_{(x,y) \in \mathcal{D}_{k,\mathrm{train}}^{(t)}} \mathcal{L}(f_k(x; \theta), y)$.

4 Class-wise Balancing Data Replay for Federated Class-Incremental Learning

This section presents an effective active data selection method for FCIL, which aims to explore global data distribution to balance class-wise sampling. Moreover, it leverages temperature scaling to adjust the logits, which can alleviate the imbalance between samples from previously learned and newly introduced tasks. Figure 2 illustrates the framework of the proposed method `FedCBDR`.

4.1 Global-perspective Data Replay (GDR)

Due to privacy constraints, traditional data replay strategies typically rely on local data distributions. However, the absence of global information often leads to class imbalance in the replay buffer. To address this, the GDR module aggregates local informative features into a global pseudo feature set, enabling exploration of the global distribution without exposing raw data.

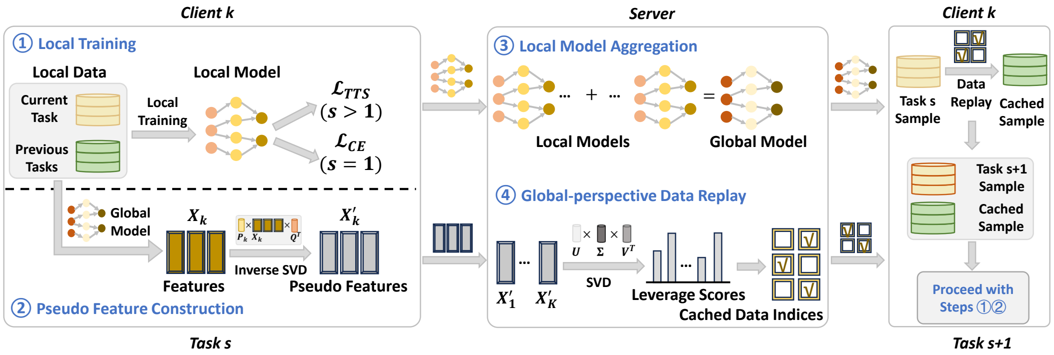


Figure 2: Illustration of the FedCBDR framework. It first trains local models using samples from current and previous tasks. After a fixed number of communication rounds, each client extracts local sample features using the global model and applies inverse singular value decomposition (ISVD) to obtain pseudo features. The server then aggregates both local models and pseudo features, performs SVD on the features, and selects representative samples based on leverage scores. The corresponding sample indices are sent back to the clients for balanced replay.

Inspired by Singular Value Decomposition (SVD) [33, 34], we first generate a set of random orthogonal matrices: a client-specific matrix $P_k^{(i)} \in \mathbb{R}^{|\mathcal{D}_k^{(i)}| \times |\mathcal{D}_k^{(i)}|}$ for each client k and task i, and a globally shared matrix $Q^{(i)} \in \mathbb{R}^{d \times d}$, where d denotes the dimension of the feature. Each client encrypts its local feature matrix $X_k^{(i)} = M_g(\mathcal{D}_k^{(i)})$ via Inverse Singular Value Decomposition (ISVD):

$$X_k^{(i)'} = P_k^{(i)} X_k^{(i)} Q^{(i)}, \tag{1}$$

and uploads the encrypted matrix $X_k^{(i)'}$ to the server, where $M_g(\cdot)$ is the feature extractor of the global model. The server then aggregates all encrypted matrices into a global matrix $X^{(i)'}$:

$$X^{(i)'} = \text{concat}\{X_k^{(i)'} \mid k = 1, \ldots, K\}, \tag{2}$$

and performs SVD as follows:

$$X^{(i)'} = U^{(i)'} \Sigma^{(i)'} V^{(i)'\top}, \tag{3}$$

where $U^{(i)'} \in \mathbb{R}^{n \times n}$, $\Sigma^{(i)'} \in \mathbb{R}^{n \times d}$, and $V^{(i)'} \in \mathbb{R}^{d \times d}$, with n denoting the total number of samples from all clients. Next, the server extracts a submatrix of the left singular vectors for each client k by:

$$U_k^{(i)} = \mathcal{I}_k(U^{(i)'}) \in \mathbb{R}^{|\mathcal{D}_k^{(i)}| \times n}, \tag{4}$$

where $\mathcal{I}_k(\cdot)$ denotes a row selection function that returns the indices corresponding to client k's samples. To quantify the importance of local samples within the global latent space, client k computes a leverage score [35] for j-th sample of task i as:

$$\tau_k^{i,j} = \|e_{i,j}^\top U_k^{(i)}\|_2^2, \tag{5}$$

where $e_{i,j}$ denotes the j-th standard basis vector in task i. Notably, a higher leverage score indicates that the sample has a larger projection in the low-dimensional latent space, suggesting that it contributes more significantly to the global structure and is more representative. Moreover, clients send their leverage scores to the server, which aggregates them into a global vector $\tau^i = \text{concat}\{\tau_k^{i,j} | k = 1, ..., K; j = 1, ..., n_k^i\}$ and normalizes it to obtain a sampling distribution:

$$p_k^{i,j} = \frac{\tau_k^{i,j}}{\sum_{j'=1}^{n_k^i} \tau_k^{i,j'}}. \tag{6}$$

Subsequently, we perform i.i.d. sampling based on the distribution $\mathbf{p} = \{p_k^{i,j} | k = 1, ..., K; j = 1, ..., n_k^i\}$. Once a sample x is selected, its sampling weight is adjusted to $\frac{1}{\sqrt{n_s \cdot p_x}} e_x$, where n_s denotes the number of selected samples and p_x is the original sampling probability of x, e_x is the standard basis vector of x. This adjustment ensures unbiased estimation during aggregation. Following the sampling procedure, the server communicates the selected sample indices to their respective clients, where the corresponding data points are subsequently marked for further use.

4.2 Task-aware Temperature Scaling (TTS)

Due to limited replay budgets, samples from previous tasks are often much fewer than those from the current task, leading to class imbalance and poor retention of past knowledge. To mitigate this, the TTS module dynamically adjusts sample temperature and weight based on task order, enhancing the contribution of tail-class samples during optimization.

Specifically, we use a lower temperature to sharpen logits for samples from earlier tasks. Furthermore, to further amplify the optimization effect of tail-class samples during training, we also leverage a re-weighted cross-entropy loss, i.e.,

$$\mathcal{L}_{\text{TTS}} = \frac{1}{N_{\text{old}}} \sum_{i=1}^{N_{\text{old}}} \omega_{\text{old}} \cdot \text{CE}\left(y_i, \text{Softmax}\left(\text{Concat}\left(\frac{z_i^{\text{old}}}{\tau_{\text{old}}}, \frac{z_i^{\text{new}}}{\tau_{\text{new}}}\right)\right)\right) + \frac{1}{N_{\text{new}}} \sum_{j=1}^{N_{\text{new}}} \omega_{\text{new}} \cdot \text{CE}\left(y_j, \text{Softmax}\left(\text{Concat}\left(\frac{z_j^{\text{old}}}{\tau_{\text{old}}}, \frac{z_j^{\text{new}}}{\tau_{\text{new}}}\right)\right)\right) \quad (7)$$

where N_{old} and N_{new} denote the number of samples from the previous and newly arrived task, respectively; y_i and y_j are the ground-truth labels; z_i^{old} and z_i^{new} denote the logits corresponding to old classes and new classes, respectively; τ_{old} and τ_{new} are the temperature scaling factors for previous and newly arrived task samples; ω_{old} and ω_{new} are the corresponding sample weights; $\text{CE}(\cdot)$ denotes the cross-entropy loss function; and $\text{Softmax}(z/\tau)$ is the temperature-scaled softmax function used to adjust the sharpness of the output distribution.

4.3 Training Strategy

The training strategy consists of two stages to progressively address the evolving challenges in federated class-incremental learning. Algorithm 1 presents the pipeline of the `FedCBDR`.

Stage 1: Initial Task Optimization. In the first task, client k learns from local data using the standard cross-entropy loss, i.e.,

$$\min_{\theta_k} \frac{1}{N} \textstyle\sum_{i=1}^{N} \text{CE}(y_i, \text{Softmax}(f_{\theta_k}(x_i))), \quad (8)$$

Stage 2: Class-Incremental Optimization. As new tasks arrive and class imbalance emerges between previous and current tasks in client k, we employ $\mathcal{L}_{TTS}$ to mitigate the imbalance, i.e.,

$$\min_{\theta_k} \frac{1}{N_{\text{old}}} \sum_{i=1}^{N_{\text{old}}} \omega_{\text{old}} \cdot \text{CE}\left(y_i, \text{Softmax}\left(\text{Concat}\left(\frac{f_{\theta_k}^{\text{old}}(x_i)}{\tau_{\text{old}}}, \frac{f_{\theta_k}^{\text{new}}(x_i)}{\tau_{\text{new}}}\right)\right)\right) + \frac{1}{N_{\text{new}}} \sum_{j=1}^{N_{\text{new}}} \omega_{\text{new}} \cdot \text{CE}\left(y_j, \text{Softmax}\left(\text{Concat}\left(\frac{f_{\theta_k}^{\text{old}}(x_j)}{\tau_{\text{old}}}, \frac{f_{\theta_k}^{\text{new}}(x_j)}{\tau_{\text{new}}}\right)\right)\right) \quad (9)$$

where x_i is the input sample, y_i is the corresponding ground-truth, $f_{\theta_k}^{\text{old}}(x)$ and $f_{\theta_k}^{\text{new}}(x)$ represent the outputs of the model corresponding to old and new classes, respectively. $\text{Softmax}(\cdot)$ converts the logits into a probability distribution.

5 Experiments

5.1 Experiment Settings

Datasets. Following existing studies [18, 21], we conducted all experiments on three commonly used datasets, including CIFAR10 [36], CIFAR100 [36] and TinyImageNet [37] to validate the effectiveness of the `FedCBDR`. We simulate heterogeneous data distributions across clients using the Dirichlet distribution with parameters $\beta = \{0.1, 0.5, 1.0\}$, where smaller values of β correspond to higher level of data heterogeneity. The statistical details are presented in the Table 5.

Evaluation Metric. Following prior studies [18, 11], we adopt Top-1 Accuracy as the evaluation metric, defined as Accuracy $= N_{\text{correct}}/N_{\text{total}}$, where N_{correct} and N_{total} denote the number of correct predictions and the total number of samples, respectively.

Implementation Details. In the experiments, the number of clients is fixed at $K = 5$, with each client running local epochs $E = 2$ per round, using a batch size $B = 128$. For all datasets, we adopt ResNet-18 as the backbone, with the classifier's output dimension dynamically updated as tasks progress and conduct $T = 100$ communication rounds per task. The SGD optimizer is employed with a learning rate of 0.01 and a weight decay of 1×10^{-5}. The number of stored samples per task

Table 1: Performance comparison between `FedCBDR` and baselines across three datasets under varying levels of heterogeneity (β). CIFAR10 is divided into 3 tasks, CIFAR100 into 5 tasks, and TinyImageNet into 10 tasks. All methods were executed under three different random seeds, and both the mean and standard deviation of the results are reported. The best results are **bolded**.

Method	**CIFAR10**		**CIFAR100**			**TinyImageNet**		
	β=0.5	β=1.0	β=0.1	β=0.5	β=1.0	β=0.1	β=0.5	β=1.0
Finetune	38.71$_{\pm3.7}$	40.49$_{\pm3.0}$	15.17$_{\pm2.2}$	16.75$_{\pm2.6}$	17.15$_{\pm1.3}$	6.06$_{\pm0.9}$	6.00$_{\pm0.8}$	6.40$_{\pm0.5}$
FedEWC	39.93$_{\pm1.1}$	42.70$_{\pm2.5}$	18.30$_{\pm2.4}$	20.70$_{\pm5.3}$	21.22$_{\pm3.4}$	6.30$_{\pm0.8}$	6.94$_{\pm0.7}$	7.36$_{\pm0.6}$
FedLwF	56.03$_{\pm1.6}$	58.29$_{\pm3.6}$	33.97$_{\pm2.6}$	37.09$_{\pm3.1}$	41.91$_{\pm2.5}$	11.81$_{\pm0.9}$	11.47$_{\pm1.0}$	14.87$_{\pm1.2}$
TARGET	44.17$_{\pm4.4}$	54.49$_{\pm4.5}$	30.15$_{\pm3.6}$	33.47$_{\pm4.3}$	35.25$_{\pm2.0}$	10.71$_{\pm1.4}$	10.18$_{\pm0.9}$	12.49$_{\pm1.1}$
LANDER	53.90$_{\pm3.2}$	60.79$_{\pm1.4}$	44.07$_{\pm3.3}$	47.63$_{\pm3.7}$	**52.77**$_{\pm1.4}$	13.80$_{\pm0.8}$	15.02$_{\pm1.9}$	16.36$_{\pm1.0}$
Re-Fed	53.46$_{\pm3.5}$	60.73$_{\pm4.3}$	32.67$_{\pm3.7}$	38.42$_{\pm2.9}$	45.28$_{\pm2.6}$	15.73$_{\pm1.7}$	15.93$_{\pm1.3}$	16.05$_{\pm1.1}$
`FedCBDR`	**64.11**$_{\pm1.2}$	**65.20**$_{\pm1.9}$	**46.40**$_{\pm1.6}$	**49.76**$_{\pm2.7}$	52.06$_{\pm1.5}$	**18.37**$_{\pm1.1}$	**18.86**$_{\pm0.9}$	**18.78**$_{\pm0.9}$

Table 2: Performance comparison between `FedCBDR` and baselines across three datasets under varying levels of heterogeneity (β). CIFAR10 is divided into 5 tasks, CIFAR100 into 10 tasks, and TinyImageNet into 20 tasks. All methods were executed under three different random seeds, and both the mean and standard deviation of the results are reported. The best results are **bolded**.

Method	**CIFAR10**		**CIFAR100**			**TinyImageNet**		
	β=0.5	β=1.0	β=0.1	β=0.5	β=1.0	β=0.1	β=0.5	β=1.0
Finetune	19.78$_{\pm2.3}$	23.34$_{\pm2.8}$	7.22$_{\pm1.1}$	9.39$_{\pm0.7}$	9.64$_{\pm0.5}$	3.40$_{\pm0.4}$	3.73$_{\pm0.5}$	3.95$_{\pm0.3}$
FedEWC	20.11$_{\pm2.7}$	28.97$_{\pm2.3}$	8.08$_{\pm0.3}$	11.69$_{\pm0.7}$	12.19$_{\pm1.7}$	3.50$_{\pm0.3}$	4.58$_{\pm0.4}$	5.08$_{\pm0.9}$
FedLwF	38.76$_{\pm2.3}$	52.95$_{\pm3.1}$	18.73$_{\pm1.1}$	25.30$_{\pm0.6}$	28.21$_{\pm1.0}$	3.67$_{\pm0.4}$	6.61$_{\pm0.6}$	10.22$_{\pm1.3}$
TARGET	35.27$_{\pm1.7}$	48.28$_{\pm1.2}$	13.61$_{\pm0.8}$	21.09$_{\pm0.4}$	24.22$_{\pm1.1}$	5.32$_{\pm0.6}$	5.39$_{\pm0.6}$	5.72$_{\pm0.5}$
LANDER	40.22$_{\pm2.4}$	58.07$_{\pm3.4}$	27.79$_{\pm1.9}$	33.51$_{\pm2.3}$	37.42$_{\pm1.8}$	8.89$_{\pm0.6}$	8.57$_{\pm0.8}$	10.45$_{\pm0.6}$
Re-Fed	54.94$_{\pm3.1}$	58.19$_{\pm2.5}$	29.33$_{\pm1.3}$	39.54$_{\pm1.3}$	40.96$_{\pm1.1}$	9.36$_{\pm0.9}$	11.44$_{\pm0.7}$	12.27$_{\pm1.1}$
`FedCBDR`	**61.18**$_{\pm1.3}$	**65.42**$_{\pm1.8}$	**45.11**$_{\pm1.2}$	**46.51**$_{\pm1.6}$	**47.79**$_{\pm1.4}$	**12.58**$_{\pm0.4}$	**14.47**$_{\pm0.7}$	**15.69**$_{\pm0.6}$

varies by dataset and split setting: for CIFAR10, 450 samples are stored under 3-task splits and 300 under 5-task splits; for CIFAR100, 1,000 samples are used for 5-task splits and 500 for 10-task splits; for TinyImageNet, 2,000 samples are stored for 10-task splits and 1,000 for 20-task splits. For the temperature and weighted parameters, we select $\tau_{old} \in \{0.8, 0.9\}$ and $w_{old} \in \{1.1, 1.2, 1.3, 1.4\}$ for previous tasks, while $\tau_{new} \in \{1.1, 1.2\}$ and $w_{new} \in \{0.7, 0.8, 0.9\}$ are used for newly arrived tasks. Moreover, the hyperparameters of baselines are tuned based on their original papers for fair comparison. And training on each client is performed using an NVIDIA RTX 3090 GPU (24 GB).

5.2 Performance Comparison

To evaluate the effectiveness of the proposed `FedCBDR`, we compare it with six representative baseline methods: Finetune [11], FedEWC [31], FedLwF [26], TARGET [18], LANDER [11], and Re-Fed [21]. As reported in Table 1 and Table 2, the results can be summarized as follows:

- `FedCBDR` achieves the highest Top-1 accuracy in most cases across the three datasets under varying levels of heterogeneity and task splits. The only suboptimal result occurs on CIFAR100 with 5 tasks and $\beta = 1.0$, where `FedCBDR` (52.06%) performs slightly worse than LANDER (52.77%). This demonstrates the adaptability and robustness of the proposed `FedCBDR` across complex settings.

- Despite LANDER attains the best performance on CIFAR100 under the 5-task and $\beta = 1.0$ setting, it demands the generation of more than 10,000 samples per task, and the overhead of training its data generator surpasses that of the federated model, raising concerns about its scalability.

- Knowledge distillation-based methods like FedLwF perform well on simpler tasks (CIFAR10) by using pretrained knowledge to guide local models. However, their performance drops on more complex or heterogeneous tasks due to limited adaptability to local variations.

Table 3: Ablation results under different levels of data heterogeneity and task splitting settings. "3/5/10" denotes CIFAR10 with 3 tasks, CIFAR100 with 5 tasks, and TinyImageNet with 10 tasks; "5/10/20" represents 5, 10, and 20 tasks respectively.

Task Splitting	Method	CIFAR10		CIFAR100			TinyImageNet		
		β=0.5	β=1.0	β=0.1	β=0.5	β=1.0	β=0.1	β=0.5	β=1.0
3/5/10	Finetune	$38.71_{\pm 3.7}$	$40.49_{\pm 3.0}$	$15.17_{\pm 2.2}$	$16.75_{\pm 2.6}$	$17.15_{\pm 1.3}$	$6.06_{\pm 0.9}$	$6.00_{\pm 0.8}$	$6.40_{\pm 0.5}$
	+GDR	$62.13_{\pm 2.1}$	$63.81_{\pm 1.9}$	$45.28_{\pm 1.5}$	$47.66_{\pm 0.9}$	$51.47_{\pm 1.7}$	$17.24_{\pm 0.6}$	$17.89_{\pm 0.5}$	$18.04_{\pm 0.4}$
	+TTS	$41.34_{\pm 2.3}$	$42.55_{\pm 2.2}$	$17.32_{\pm 0.5}$	$17.14_{\pm 0.4}$	$19.32_{\pm 0.5}$	$6.67_{\pm 0.2}$	$6.92_{\pm 0.3}$	$7.27_{\pm 0.4}$
	+GDR+TTS	$\mathbf{64.11}_{\pm 1.2}$	$\mathbf{65.20}_{\pm 1.9}$	$\mathbf{46.40}_{\pm 1.6}$	$\mathbf{49.76}_{\pm 2.7}$	$\mathbf{52.06}_{\pm 1.5}$	$\mathbf{18.37}_{\pm 1.1}$	$\mathbf{18.86}_{\pm 0.9}$	$\mathbf{18.78}_{\pm 0.9}$
5/10/20	Finetune	$19.78_{\pm 2.3}$	$23.34_{\pm 2.8}$	$7.22_{\pm 1.1}$	$9.39_{\pm 0.7}$	$9.64_{\pm 0.5}$	$3.40_{\pm 0.4}$	$3.73_{\pm 0.5}$	$3.95_{\pm 0.3}$
	+GDR	$59.34_{\pm 3.1}$	$63.20_{\pm 2.6}$	$44.04_{\pm 1.3}$	$46.33_{\pm 0.5}$	$46.50_{\pm 0.8}$	$11.44_{\pm 0.3}$	$13.85_{\pm 0.5}$	$14.51_{\pm 0.6}$
	+TTS	$22.43_{\pm 2.4}$	$25.81_{\pm 2.1}$	$8.31_{\pm 0.2}$	$10.21_{\pm 0.3}$	$10.33_{\pm 0.4}$	$3.78_{\pm 0.5}$	$4.04_{\pm 0.4}$	$4.16_{\pm 0.3}$
	+GDR+TTS	$\mathbf{61.18}_{\pm 1.3}$	$\mathbf{65.42}_{\pm 1.8}$	$\mathbf{45.11}_{\pm 1.2}$	$\mathbf{46.51}_{\pm 1.6}$	$\mathbf{47.79}_{\pm 1.4}$	$\mathbf{12.58}_{\pm 0.4}$	$\mathbf{14.47}_{\pm 0.7}$	$\mathbf{15.69}_{\pm 0.6}$

- Given an equal memory budget, class-balanced sampling (FedCBDR) consistently achieves superior performance compared to class-imbalanced strategy (Re-Fed), as it ensures more equitable representation across categories and effectively mitigates class-level forgetting in FCIL scenarios.

5.3 Ablation Study

In this section, we conducted an ablation study to investigate the contributions of key modules, including the Global-perspective Active Data Replay (GDR) module and the Task-aware Temperature Scaling (TTS) module. Table 3 presents the results, which can be summarized as follows:

- Incorporating the GDR module substantially improves performance across all cases, particularly under high data heterogeneity ($\beta = 0.1$), demonstrating its effectiveness in alleviating catastrophic forgetting even with a limited number of replay samples in federated class-incremental learning.
- Using the TTS module alone leads to consistent improvements over Finetune, highlighting its effectiveness in addressing intra-client class imbalance through temperature scaling. This contribution to better generalization is particularly evident under the more challenging "5/10/20" task splitting scenario.
- The integration of both modules results in the best overall performance, consistently achieving the highest Top-1 accuracy across various datasets and heterogeneity levels. This stems from their complementary strengths: the GDR module mitigates inter-task forgetting, while the TTS module alleviates both intra- and inter-client class imbalance.

5.4 Performance Evaluation of FedCBDR under Incremental Tasks

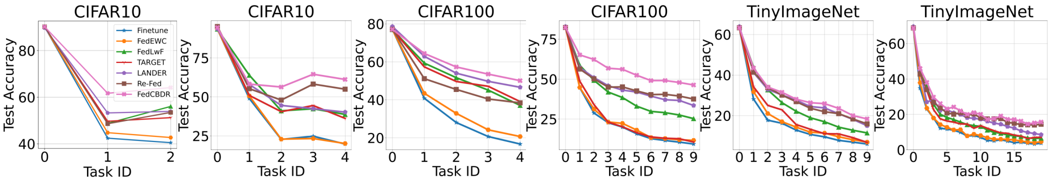


Figure 3: Performance comparison of all methods across varying task splits on CIFAR10 (3/5 tasks), CIFAR100 (5/10 tasks), and TinyImageNet (10/20 tasks) with $\beta = 0.5$.

This section investigates the performance of FedCBDR and the baselines in incremental cases on three datasets. Figure 3 presents the average accuracy of all methods on both current and previous tasks. Notably, FedCBDR consistently outperforms other baseline methods across all task splits, with its accuracy curves remaining higher throughout the incremental process. Furthermore, FedCBDR exhibits a slower performance degradation as the number of tasks increases, indicating stronger resistance to catastrophic forgetting. In addition, it maintains significantly higher accuracy on later tasks, especially in challenging settings such as CIFAR100 and TinyImageNet with 10 tasks, highlighting its ability to balance knowledge retention and adaptation to new classes.

5.5 Quantitative Analysis of Replay Buffer Size on Test Accuracy

Table 4: Comparison of model performance with varying memory size M across datasets.

Methods	CIFAR10			CIFAR100			TinyImageNet	
	M=150	M=300	M=450	M=500	M=1000	M=1500	M=2000	M=2500
LANDER (10240)	52.90			47.05			14.77	
Re-Fed	47.23	53.47	54.66	33.89	38.42	47.84	15.89	16.78
FedCBDR	51.99	59.02	63.81	40.12	49.66	55.94	18.33	19.41

In this section, we evaluate the performance of Re-Fed and FedCBDR under different buffer size M settings, and additionally include LANDER, which generates 10,240 synthetic samples for each task. As shown in Table 4, FedCBDR exhibits more significant performance advantages over Re-Fed under limited memory settings, and even surpasses LANDER, which relies on a large-scale generative replay buffer. Furthermore, as the buffer size increases, FedCBDR demonstrates more stable and significant performance improvements. This indicates that the method can effectively leverage larger replay buffers for continuous optimization. However, Re-Fed exhibits noticeable performance fluctuations under small and medium buffer settings. In particular, its accuracy is significantly lower than that of FedCBDR on CIFAR100 with $M = 500$ and TinyImageNet with $M = 2000$, indicating its limited ability to mitigate inter-class interference and retain knowledge from previous tasks. These findings validate that, under the same buffer budget, a balanced sampling distribution is more effective than an imbalanced one in alleviating forgetting and improving overall model performance.

5.6 Sensitivity Analysis of FedCBDR on Temperature and Weighted Hyperparameters

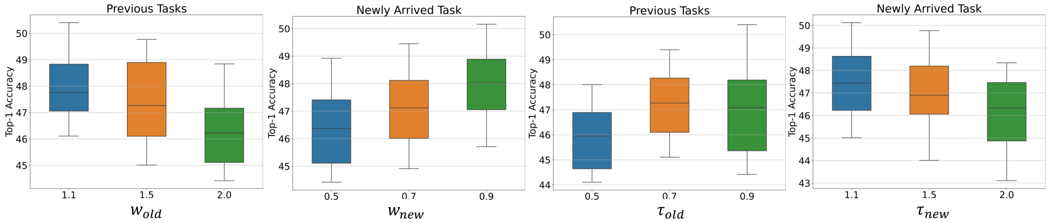


Figure 4: Performance of FedCBDR on CIFAR100 ($\beta = 0.5$, 5-task split) under varying temperature ($\tau_{old} \in \{0.5, 0.7, 0.9\}$, $\tau_{new} \in \{1.1, 1.5, 2.0\}$) and weighted ($w_{old} \in \{1.1, 1.5, 2.0\}$, $w_{new} \in \{0.5, 0.7, 0.9\}$) settings.

Figure 4 gives a sensitivity analysis of FedCBDR with respect to temperature and sample weighting hyperparameters. Overall, temperature scaling and sample re-weighting help mitigate class imbalance, but model performance varies considerably with different hyperparameter settings. The model achieves better overall performance when $\omega_{\text{old}} = 1.1$, $\omega_{\text{new}} = 0.9$, $\tau_{\text{old}} = 0.9$, and $\tau_{\text{new}} = 1.1$. This is because slightly higher weight and temperature for previous-task samples help retain old knowledge, while lower weight and higher temperature for newly arrived samples reduce overfitting and improve adaptation. However, inappropriate hyperparameter choices may harm performance. For instance, a large τ_{new} (e.g., 2.0) leads to overly smooth predictions, reducing discrimination among newly arrived classes. These results emphasize the need for proper tuning to ensure balanced learning.

5.7 Comparison of Per-Class Sample Distributions in the Replay Buffer

To evaluate the effectiveness of FedCBDR in balancing class-wise sampling, Figure 5 illustrates the per-class sample distributions in the replay buffer between FedCBDR and Re-Fed across different task stages. Overall, across different task stages, FedCBDR (orange bars) exhibits a per-class sample distribution that is consistently closer to the average level (red line), whereas Re-Fed shows noticeable skewness and fluctuations. This indicates that FedCBDR is more effective in achieving balanced class-wise sampling in the replay buffer. In addition, FedCBDR ensures that no class is overlooked during sampling, while Re-Fed may fail to retain certain classes in the replay buffer—for example, class 79 is missing in Task 4 under Re-Fed. This highlights the robustness of FedCBDR in maintaining class coverage throughout incremental learning.

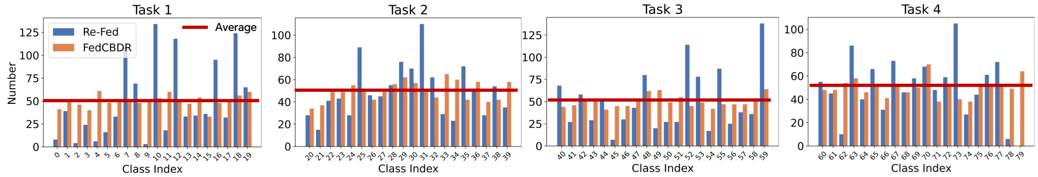
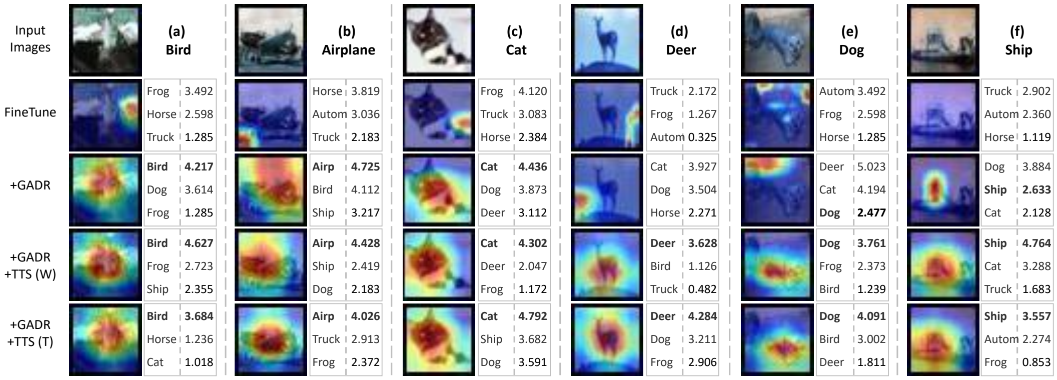


Figure 5: Comparison of per-class sample distributions in the replay buffer between FedCBDR and Re-Fed on the CIFAR100 dataset, under a heterogeneity level of $\beta = 0.5$ and a 5-task split case.

5.8 Visualization of Model Attention and Temperature-aware Logits Adjustment



Input Images
(a) Bird
(b) Airplane
(c) Cat
(d) Deer
(e) Dog
(f) Ship
FineTune
Frog 3.492 Horse 2.598 Truck 1.285
Horse 3.819 Autom 3.036 Truck 2.183
Frog 4.120 Truck 3.083 Horse 2.384
Truck 2.172 Frog 1.267 Autom 0.325
Autom 3.492 Frog 2.598 Horse 1.285
Truck 2.902 Autom 2.360 Horse 1.119
+GADR
Bird 4.217 Dog 3.614 Frog 1.285
Airp 4.725 Bird 4.112 Ship 3.217
Cat 4.436 Dog 3.873 Deer 3.112
Cat 3.927 Dog 3.504 Horse 2.271
Deer 5.023 Cat 4.194 Dog 2.477
Dog 3.884 Ship 2.633 Cat 2.128
+GADR +TTS (W)
Bird 4.627 Frog 2.723 Ship 2.355
Airp 4.428 Ship 2.419 Dog 2.183
Cat 4.302 Deer 2.047 Frog 1.172
Deer 3.628 Bird 1.126 Truck 0.482
Dog 3.761 Frog 2.373 Bird 1.239
Ship 4.764 Cat 3.288 Truck 1.683
+GADR +TTS (T)
Bird 3.684 Horse 1.236 Cat 1.018
Airp 4.026 Truck 2.913 Frog 2.372
Cat 4.792 Ship 3.682 Dog 3.591
Deer 4.284 Dog 3.211 Frog 2.906
Dog 4.091 Bird 3.002 Deer 1.811
Ship 3.557 Autom 2.274 Frog 0.853

Figure 6: Case studies of model attention and the effect of temperature-aware logits adjustment on CIFAR10 ($\beta = 0.5$, 3-task split).

This section presents case studies comparing prediction confidence and attention focus using Grad-CAM [38, 39] visualizations. As shown in Figure 6(a-c), in the absence of data replay, the model struggles to correctly classify samples from previous tasks and fails to attend to the relevant target regions. The incorporation of data replay in FedCBDR alleviates this issue by correcting predictions and guiding attention back to semantically important areas. Despite partially mitigating forgetting, data replay alone may still lead to misclassification or low-confidence predictions for tail classes with limited samples. The integration of temperature scaling (T) and sample re-weighting (W) in the TTS module enables the model to better distinguish confusing classes through temperature adjustment, improving tail class accuracy and enhancing prediction stability, as depicted in Figure 6(d-f). These findings demonstrate the crucial role of the collaboration between both modules in mitigating knowledge forgetting during incremental learning.

6 Conclusions and Future Work

To address the challenge of inter-class imbalance in replay-based federated class-incremental learning, we propose FedCBDR that combines class-balanced sampling with loss adjustment to better exploit the global data distribution and enhance the contribution of tail-class samples to model optimization. Specifically, it uses SVD to decouple and reconstruct local data, aggregates local information in a privacy-preserving manner, and explores i.i.d. sampling within the aggregated distribution. In addition, it applies task-aware temperature scaling and sample re-weighting to mitigate the long-tail problem. Experimental results show that FedCBDR effectively reduces inter-class sampling imbalance and significantly improves final performance.

Despite the impressive performance of FedCBDR, there remain several directions worth exploring to address its current limitations. Specifically, we plan to investigate lightweight sampling strategies to reduce feature transmission costs in FedCBDR, and to develop more robust post-sampling balancing methods that mitigate class imbalance with less sensitivity to hyperparameters. Moreover, designing sampling strategies for globally imbalanced distributions remains an open problem.

References

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

[2] Ming Hu, Yue Cao, Anran Li, Zhiming Li, Chengwei Liu, Tianlin Li, Mingsong Chen, and Yang Liu. Fedmut: Generalized federated learning via stochastic mutation. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 38, pages 12528–12537, 2024.

[3] Haozhao Wang, Haoran Xu, Yichen Li, Yuan Xu, Ruixuan Li, and Tianwei Zhang. Fedcda: Federated learning with cross-rounds divergence-aware aggregation. In *The Twelfth International Conference on Learning Representations, ICLR 2024, Vienna, Austria, May 7-11, 2024*.

[4] Tao Fan, Hanlin Gu, Xuemei Cao, Chee Seng Chan, Qian Chen, Yiqiang Chen, Yihui Feng, Yang Gu, Jiaxiang Geng, Bing Luo, et al. Ten challenging problems in federated foundation models. *IEEE Transactions on Knowledge and Data Engineering*, 2025.

[5] Zhuang Qi, Lei Meng, Zitan Chen, Han Hu, Hui Lin, and Xiangxu Meng. Cross-silo prototypical calibration for federated learning with non-iid data. In *Proceedings of the 31st ACM International Conference on Multimedia*, pages 3099–3107, 2023.

[6] Lele Fu, Sheng Huang, Yuecheng Li, Chuan Chen, Chuanfu Zhang, and Zibin Zheng. Learn the global prompt in the low-rank tensor space for heterogeneous federated learning. *Neural Networks*, 187:107319, 2025.

[7] Ming Hu, Peiheng Zhou, Zhihao Yue, Zhiwei Ling, Yihao Huang, Anran Li, Yang Liu, Xiang Lian, and Mingsong Chen. Fedcross: Towards accurate federated learning via multi-model cross-aggregation. In *IEEE International Conference on Data Engineering (ICDE)*, pages 2137–2150. IEEE, 2024.

[8] Peter Kairouz, H Brendan McMahan, Brendan Avent, Aurélien Bellet, Mehdi Bennis, Arjun Nitin Bhagoji, Kallista Bonawitz, Zachary Charles, Graham Cormode, Rachel Cummings, et al. Advances and open problems in federated learning. *Foundations and trends® in machine learning*, 14(1–2):1–210, 2021.

[9] Jiahua Dong, Lixu Wang, Zhen Fang, Gan Sun, Shichao Xu, Xiao Wang, and Qi Zhu. Federated class-incremental learning. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 10164–10173, 2022.

[10] Feng Wu, Alysa Ziying Tan, Siwei Feng, Han Yu, Tao Deng, Libang Zhao, and Yuanlu Chen. Federated class-incremental learning via weighted aggregation and distillation. *IEEE Internet of Things Journal*, 2025.

[11] Minh-Tuan Tran, Trung Le, Xuan-May Le, Mehrtash Harandi, and Dinh Phung. Text-enhanced data-free approach for federated class-incremental learning. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 23870–23880, 2024.

[12] Xin Yang, Hao Yu, Xin Gao, Hao Wang, Junbo Zhang, and Tianrui Li. Federated continual learning via knowledge fusion: A survey. *IEEE Transactions on Knowledge and Data Engineering*, 36(8):3832–3850, 2024.

[13] Yanyan Lu, Lei Yang, Hao-Rui Chen, Jiannong Cao, Wanyu Lin, and Saiqin Long. Federated class-incremental learning with dynamic feature extractor fusion. *IEEE Transactions on Mobile Computing*, 2024.

[14] Yuanlu Chen, Alysa Ziying Tan, Siwei Feng, Han Yu, Tao Deng, Libang Zhao, and Feng Wu. General federated class-incremental learning with lightweight generative replay. *IEEE Internet of Things Journal*, 2024.

[15] Xin Gao, Xin Yang, Hao Yu, Yan Kang, and Tianrui Li. Fedprok: Trustworthy federated class-incremental learning via prototypical feature knowledge transfer. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 4205–4214, 2024.

[16] Sara Babakniya, Zalan Fabian, Chaoyang He, Mahdi Soltanolkotabi, and Salman Avestimehr. A data-free approach to mitigate catastrophic forgetting in federated class incremental learning for vision tasks. *Advances in Neural Information Processing Systems*, 36:66408–66425, 2023.

[17] Naibo Wang, Yuchen Deng, Wenjie Feng, Jianwei Yin, and See-Kiong Ng. Data-free federated class incremental learning with diffusion-based generative memory. *arXiv preprint arXiv:2405.17457*, 2024.

[18] Jie Zhang, Chen Chen, Weiming Zhuang, and Lingjuan Lyu. Target: Federated class-continual learning via exemplar-free distillation. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 4782–4793, 2023.

[19] Jinglin Liang, Jin Zhong, Hanlin Gu, Zhongqi Lu, Xingxing Tang, Gang Dai, Shuangping Huang, Lixin Fan, and Qiang Yang. Diffusion-driven data replay: A novel approach to combat forgetting in federated class continual learning. In *European Conference on Computer Vision*, pages 303–319. Springer, 2024.

[20] Min Kyoon Yoo and Yu Rang Park. Federated class incremental learning: A pseudo feature based approach without exemplars. In *Proceedings of the Asian Conference on Computer Vision*, pages 488–498, 2024.

[21] Yichen Li, Qunwei Li, Haozhao Wang, Ruixuan Li, Wenliang Zhong, and Guannan Zhang. Towards efficient replay in federated incremental learning. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 12820–12829, 2024.

[22] Yichen Li, Haozhao Wang, Yining Qi, Wei Liu, and Ruixuan Li. Re-fed+: A better replay strategy for federated incremental learning. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 2025.

[23] Sylvestre-Alvise Rebuffi, Alexander Kolesnikov, Georg Sperl, and Christoph H Lampert. icarl: Incremental classifier and representation learning. In *Proceedings of the IEEE conference on Computer Vision and Pattern Recognition*, pages 2001–2010, 2017.

[24] Thinh Nguyen, Khoa D Doan, Binh T Nguyen, Danh Le-Phuoc, and Kok-Seng Wong. Overcoming catastrophic forgetting in federated class-incremental learning via federated global twin generator. *arXiv preprint arXiv:2407.11078*, 2024.

[25] Daiqing Qi, Handong Zhao, and Sheng Li. Better generative replay for continual federated learning. *arXiv preprint arXiv:2302.13001*, 2023.

[26] Zhizhong Li and Derek Hoiem. Learning without forgetting. *IEEE transactions on pattern analysis and machine intelligence*, 40(12):2935–2947, 2017.

[27] Alysa Ziying Tan, Siwei Feng, and Han Yu. Fl-clip: Bridging plasticity and stability in pre-trained federated class-incremental learning models. In *2024 IEEE International Conference on Multimedia and Expo (ICME)*, pages 1–6. IEEE, 2024.

[28] Athanasios Psaltis, Christos Chatzikonstantinou, Charalampos Z Patrikakis, and Petros Daras. Fedrcil: Federated knowledge distillation for representation based contrastive incremental learning. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 3463–3472, 2023.

[29] Jiao Chen, Jiayi He, Jianhua Tang, Weihua Li, and Zihang Yin. Knowledge efficient federated continual learning for industrial edge systems. *IEEE Transactions on Network Science and Engineering*, 2025.

[30] Zhengyi Zhong, Weidong Bao, Ji Wang, Jianguo Chen, Lingjuan Lyu, and Wei Yang Bryan Lim. Sacfl: Self-adaptive federated continual learning for resource-constrained end devices. *arXiv preprint arXiv:2505.00365*, 2025.

[31] James Kirkpatrick, Razvan Pascanu, Neil Rabinowitz, Joel Veness, Guillaume Desjardins, Andrei A Rusu, Kieran Milan, John Quan, Tiago Ramalho, Agnieszka Grabska-Barwinska, et al. Overcoming catastrophic forgetting in neural networks. *Proceedings of the national academy of sciences*, 114(13):3521–3526, 2017.

[32] Hao Yu, Xin Yang, Xin Gao, Yihui Feng, Hao Wang, Yan Kang, and Tianrui Li. Overcoming spatial-temporal catastrophic forgetting for federated class-incremental learning. In *Proceedings of the 32nd ACM International Conference on Multimedia*, pages 5280–5288, 2024.

[33] Di Chai, Junxue Zhang, Liu Yang, Yilun Jin, Leye Wang, Kai Chen, and Qiang Yang. Efficient decentralized federated singular vector decomposition. In *2024 USENIX Annual Technical Conference (USENIX ATC 24)*, pages 1029–1047, 2024.

[34] Di Chai, Leye Wang, Junxue Zhang, Liu Yang, Shuowei Cai, Kai Chen, and Qiang Yang. Practical lossless federated singular vector decomposition over billion-scale data. In *Proceedings of the 28th ACM SIGKDD conference on knowledge discovery and data mining*, pages 46–55, 2022.

[35] Petros Drineas, Malik Magdon-Ismail, Michael W Mahoney, and David P Woodruff. Fast approximation of matrix coherence and statistical leverage. *The Journal of Machine Learning Research*, 13(1):3475–3506, 2012.

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

[37] Ya Le and Xuan Yang. Tiny imagenet visual recognition challenge. *CS 231N*, 7(7):3, 2015.

[38] Ramprasaath R Selvaraju, Michael Cogswell, Abhishek Das, Ramakrishna Vedantam, Devi Parikh, and Dhruv Batra. Grad-cam: Visual explanations from deep networks via gradient-based localization. In *Proceedings of the IEEE international conference on computer vision*, pages 618–626, 2017.

[39] Zhuang Qi, Lei Meng, and et al. Cross-silo feature space alignment for federated learning on clients with imbalanced data. In *The 39th Annual AAAI Conference on Artificial Intelligence (AAAI-25)*, pages 19986–19994, 2025.

A Appendix

A.1 Algorithm

To illustrate the implementation, the pseudocode of the FedCBDR is provided in Algorithm 1.

Algorithm 1 FedCBDR

```
Initialize: $R$: number of communication rounds; $K$: number of clients; $t$: number of tasks; $\theta_g$: global model parameters; $\mathcal{B}_k^{pre}$: replay buffer for historical tasks on client $k$; $\mathcal{D}_k^s$: local data of task $s$ on client $k$.
for each task $s = 1$ to $t$ do
    for each communication round $r = 1$ to $R$ do
        for each client $k = 1$ to $K$ do
            Initialize local model parameters: $\theta_k \leftarrow \theta_g$
            if $s == 1$ then
                Sample a mini-batch $\zeta$ from $\mathcal{D}_k^{(1)}$, and update $\theta_k$ using Eq. 8.
            else
                Store the historical task data corresponding to globally sampled IDs into $\mathcal{B}_k^{pre}$.
                Sample a mini-batch $\zeta$ from $\mathcal{D}_k^{(s)} \cup \mathcal{B}_k^{pre}$, and update $\theta_k$ using Eq. 9.
                Compute pseudo-features based on Eq. 1, and upload them to the server.
            end if
        end for
        if $r < R$ then
            Aggregate local model parameters across clients.
        else
            Aggregate model parameters and pseudo-features from all clients using Eq. 2.
            Perform Global Sampling based on Eqs. 3–6, and send the selected sample IDs back to the corresponding clients.
        end if
    end for
end for
// Global Sampling Procedure
Form the global feature pool $X^{(i)}$ by aggregating all pseudo-features via Eq. (2).
Perform singular value decomposition (SVD) using Eq. 3 to extract key attributes.
Compute leverage scores for each client's samples using Eqs. 4–5, and normalize globally using Eq. 6.
Perform sampling and adjust the probabilities of the selected samples accordingly.
```

A.2 Datasets

Experiments are conducted on CIFAR-10, CIFAR-100, and TinyImageNet, with dataset statistics summarized in Table 5. Furthermore, a Dirichlet distribution is employed to partition data among clients for each incoming task, where a smaller Dirichlet parameter β indicates a higher level of data heterogeneity. Specifically, we explore multiple federated settings for each dataset: CIFAR-10 is evaluated under 5 and 10 clients, with 3 and 5 tasks, and Dirichlet parameters $\beta = \{0.5, 1.0\}$; CIFAR-100 is tested with 5 and 10 clients, 5 and 10 tasks, and $\beta = \{0.1, 0.5, 1.0\}$; TinyImageNet is configured with 5 and 10 clients, 10 and 20 tasks, and the same range of β values.

Table 5: Statistics of the datasets used in experiments.

Datasets	#Class	#Training	#Testing	Image Size	Federated Settings		
					Clients	Tasks	Heterogeneity
CIFAR10	10	50,000	10,000	32×32	5/10	3/5	0.5/1.0
CIFAR100	100	50,000	10,000	32×32	5/10	5/10	0.1/0.5/1.0
TinyImageNet	200	100,000	10,000	64×64	5/10	10/20	0.1/0.5/1.0

A.3 Experimental Results

A.3.1 Performance Comparison

To thoroughly verify the effectiveness of the proposed `FedCBDR`, we compare its performance against various baselines under the setting of 10 clients. Based on the original implementations, we generate 10,240 synthetic samples per task for both TARGET and LANDER. The data replay configurations for Re-Fed and `FedCBDR` follow the settings outlined in Section 5.1. The results are presented in Tables 6 and 7. Consistent with the results shown in Tables 1 and 2, `FedCBDR` achieves the best performance across all cases. Notably, `FedCBDR` **achieves over a 10% gain compared to the second-best performing method in several settings.**

Table 6: Performance comparison between `FedCBDR` and baseline methods across CIFAR-10, CIFAR-100, and TinyImageNet under varying levels of data heterogeneity (Dirichlet parameter β). Specifically, CIFAR-10 is split into 3 tasks, CIFAR-100 into 5 tasks, and TinyImageNet into 10 tasks. The number of clients is fixed at 10, and all experiments are conducted with a random seed of 2023 to ensure reproducibility. The best results are **bolded**.

Method	**CIFAR10**		**CIFAR100**			**TinyImageNet**		
	β=0.5	β=1.0	β=0.1	β=0.5	β=1.0	β=0.1	β=0.5	β=1.0
FedEWC	36.40	42.00	15.19	18.66	19.50	6.19	7.23	7.78
FedLwF	48.24	49.11	27.02	37.92	41.77	10.67	13.02	14.73
TARGET	38.23	41.11	18.34	23.59	25.71	7.45	8.29	8.87
LANDER	41.54	45.52	30.83	43.69	47.29	12.33	15.18	15.64
Re-Fed	45.49	52.22	31.81	36.40	37.95	9.28	11.48	12.10
`FedCBDR`	59.80	62.59	42.25	47.90	48.55	14.81	16.54	17.43

Table 7: Performance comparison between `FedCBDR` and baseline methods across CIFAR-10, CIFAR-100, and TinyImageNet under varying levels of data heterogeneity (Dirichlet parameter β). Specifically, CIFAR-10 is split into 5 tasks, CIFAR-100 into 10 tasks, and TinyImageNet into 20 tasks. The number of clients is fixed at 10, and all experiments are conducted with a random seed of 2023 to ensure reproducibility. The best results are **bolded**.

Method	**CIFAR10**		**CIFAR100**			**TinyImageNet**		
	β=0.5	β=1.0	β=0.1	β=0.5	β=1	β=0.1	β=0.5	β=1.0
FedEWC	20.18	23.33	6.68	10.98	12.30	3.27	4.80	4.89
FedLwF	43.31	46.79	13.82	17.79	27.80	4.50	5.71	9.07
TARGET	21.60	28.39	12.11	16.64	17.14	3.45	4.88	5.01
LANDER	27.24	32.21	10.74	25.87	31.79	4.74	12.05	13.21
Re-Fed	38.28	39.22	28.08	33.52	37.27	7.95	8.53	10.13
`FedCBDR`	51.71	59.57	37.42	43.82	45.50	11.51	14.45	15.25

A.3.2 Performance Evaluation of `FedCBDR` under Incremental Tasks

We evaluate the performance evolution of `FedCBDR` and competing methods under a 10-client setting across incremental tasks on three benchmark datasets. Specifically, CIFAR-10 is split into 3 tasks ($\beta = \{0.5, 1.0\}$), CIFAR-100 into 5 tasks ($\beta = \{0.1, 0.5, 1.0\}$), and TinyImageNet into 10 tasks ($\beta = \{0.1, 0.5, 1.0\}$). As shown in Figure 7, `FedCBDR` **consistently outperforms all baseline methods across incremental tasks, maintaining higher accuracy on both current and previous tasks throughout the training process**. Moreover, its performance degrades more slowly as the number of tasks increases.

A.3.3 Comparison of Per-Class Sample Distributions in the Replay Buffer

We further validate the capability of the proposed `FedCBDR` to balance per-class sample distributions in more complex scenarios. Specifically, we divide the CIFAR100 dataset into 10 tasks. As illustrated

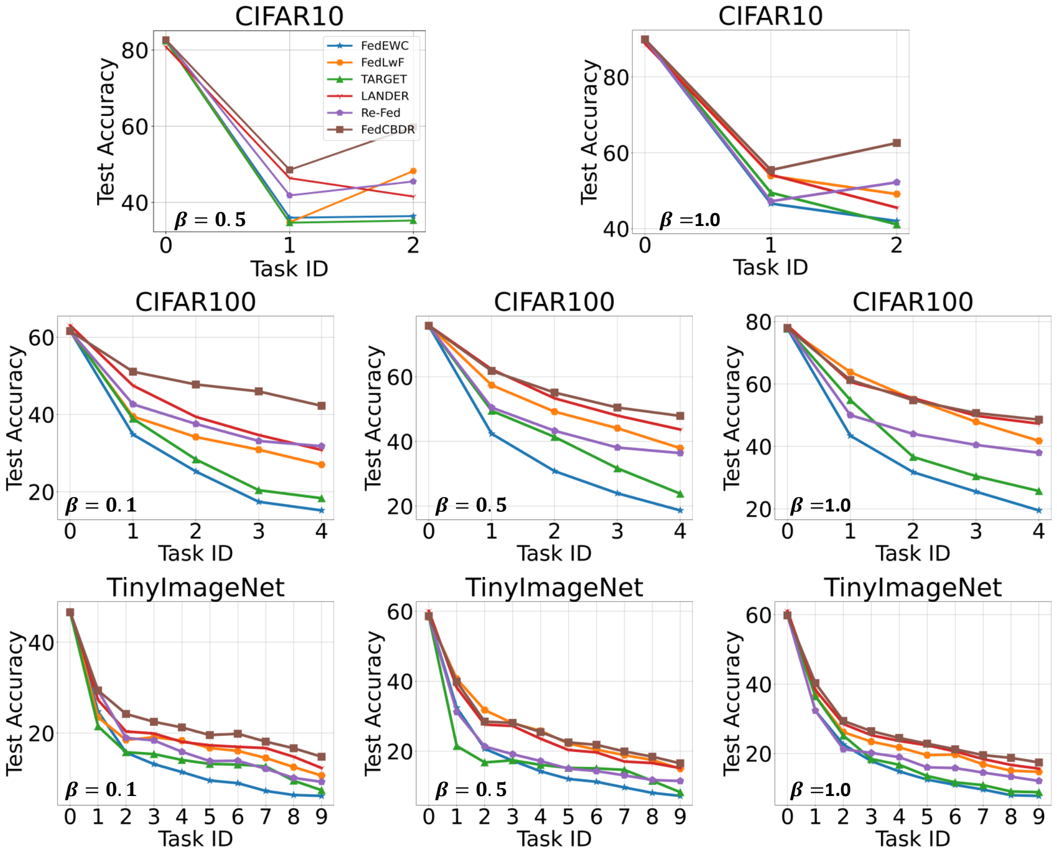


Figure 7: Comparison of per-class sample distributions in the replay buffer between `FedCBDR` and Re-Fed on the CIFAR100 dataset, conducted under a heterogeneity level of $\beta = 0.5$, with a 10-task split and 5 clients.

in Figure 8, Re-Fed exhibits substantial disparities in the number of replayed samples across classes. For example, in task 1, while classes 10 and 18 contain nearly 100 samples each, class 11 has fewer than 10. In contrast, `FedCBDR` **effectively alleviates such class imbalance, with the number of replayed samples for all classes remaining consistently close to the average (as marked by the red line)**. This contributes to more stable knowledge retention across tasks and enhances overall model generalization.

A.3.4 Quantitative Analysis of Replay Buffer Size on Test Accuracy

Table 8: Comparison of model performance with varying replay budget M per task across datasets, with the number of clients fixed at 10, and heterogeneity level $\beta = 0.5$.

Methods	**CIFAR10**			**CIFAR100**		
	M=150	M=300	M=450	M=500	M=1000	M=1500
Re-Fed	39.22	42.93	45.49	28.21	36.40	41.78
`FedCBDR`	48.15	54.62	59.80	38.34	47.90	51.14

We compare the performance of the data replay-based methods, Re-Fed and `FedCBDR`, under varying replay buffer budgets. Specifically, for CIFAR10, the buffer size is adjusted among $\{150, 300, 450\}$, while for CIFAR100, it ranges from $\{500, 1000, 1500\}$. The number of clients is set to 10, and heterogeneity level $\beta = 0.5$. As shown in Table 8, **the performance of both methods improves as the buffer size increases, with `FedCBDR` maintaining a clear advantage over Re-Fed under all settings.** This also underscores the importance of balancing per-class sample counts in the replay buffer to ensure fair representation and stable performance.

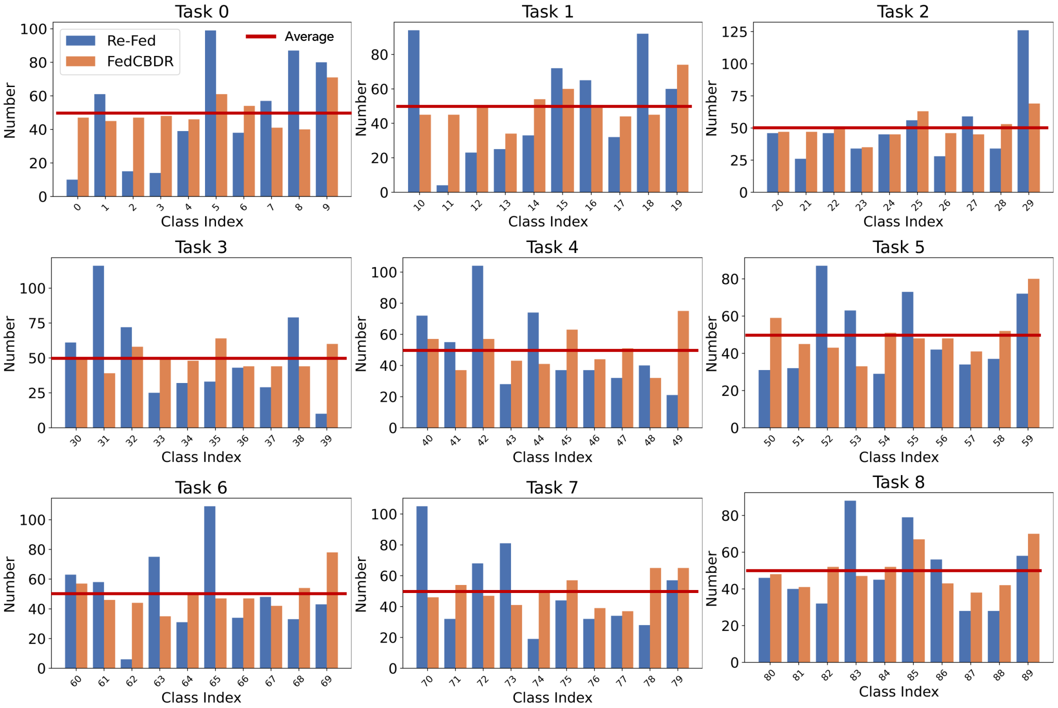


Figure 8: Comparison of per-class sample distributions in the replay buffer between FedCBDR and Re-Fed on the CIFAR100 dataset, conducted under a heterogeneity level of $\beta = 0.5$, with a 10-task split and 5 clients.

A.3.5 Evaluation on the Impact of Local Training Epochs

To assess the impact of local training intensity, we compare the performance of LANDER, Re-Fed, and FedCBDR, under varying local training epoch settings. Specifically, the evaluation is conducted on CIFAR10 divided into 3 tasks and CIFAR100 divided into 5 tasks, under a federated setting with 10 clients and a heterogeneity level of $\beta = 0.5$. As shown in Figure 9, **both GDR and GDR+TTS consistently outperform the baseline methods (LANDER and Re-Fed) across all local training epoch settings on both CIFAR10 and CIFAR100**. Moreover, **GDR+TTS achieves the highest test accuracy in every configuration**. The improvement brought by TTS highlights its necessity in alleviating class imbalance during local training. And, unlike other methods whose performance drops at 10 local epochs due to biased updates, **GDR+TTS demonstrates a sustained improvement potential.**

A.3.6 Performance Assessment of the Final Model Across Tasks

This section compares the final model performance of different methods (LANDER, Re-Fed, FedCBDR) across various tasks. Specifically, all experiments are conducted under a federated setting with 5 clients and a heterogeneity level of $\beta = 0.5$. CIFAR10 is split into 3 tasks and CIFAR100 into 5 tasks. Each task is trained for 50 communication rounds, with each client performing 2 local training epochs per round using a batch size of 128. For sample replay, LANDER synthesizes 10,240 samples per task, while Re-Fed and FedCBDR retain 150 and 1,000 real samples per task on CIFAR10 and CIFAR100, respectively. As shown in Table 9, **LANDER suffers from significant forgetting of earlier tasks**, as evidenced by its low accuracy of only 1.37% on Task 1 of CIFAR10. This indicates a severe inability to retain prior knowledge. Moreover, **LANDER also shows a noticeable decline in performance on the last task**, achieving only 57.00% on Task 5 of CIFAR100, suggesting that its generalization to new tasks is also limited under non-i.i.d. conditions. Compared to LANDER and Re-Fed, **GDR significantly enhances the retention of knowledge from most early tasks**. **This demonstrates the advantage of balanced sample replay over imbalanced sampling**. In particular, **GDR+TTS outperforms GDR alone, highlighting the effectiveness of the proposed TTS module**

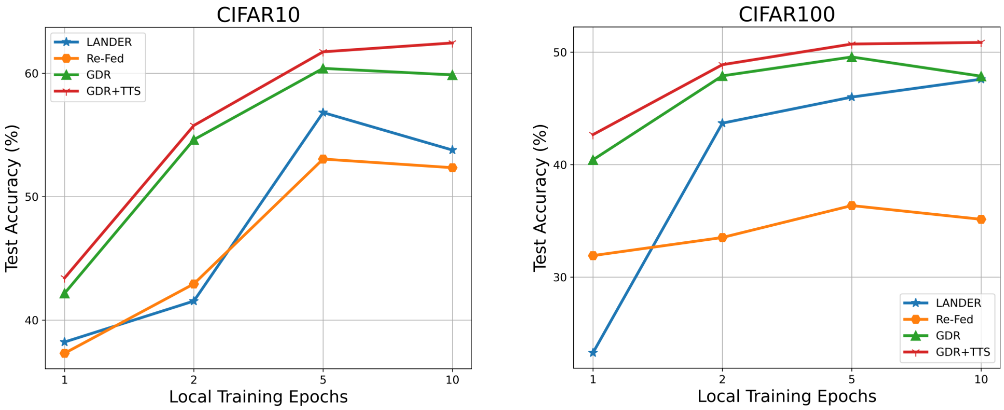


Figure 9: Comparison of the final performance of the LANDER, Re-Fed, GDR, and GDR+TTS methods under different numbers of local training epochs. GDR and TTS are the two functional modules proposed in this work, and GDR+TTS=`FedCBDR`.

in mitigating class imbalance and supporting long-term knowledge preservation under non-i.i.d. settings.

Table 9: Per-task and average accuracy (%) of different methods on CIFAR10 and CIFAR100.

	CIFAR10				CIFAR100					
	Task 1	Task 2	Task 3	ALL	Task 1	Task 2	Task 3	Task 4	Task 5	ALL
LANDER	1.37	30.00	88.32	44.74	33.95	40.90	43.70	44.45	57.00	44.00
Re-Fed	14.20	18.33	95.88	44.28	23.40	22.00	21.10	34.10	81.70	36.46
GDR	17.07	19.00	96.10	49.26	40.40	37.90	39.25	47.50	80.45	49.10
GDR+TTS	21.43	21.46	96.08	51.30	41.45	38.05	38.50	48.55	81.40	49.59

NeurIPS Paper Checklist

1. **Claims**

   Question: Do the main claims made in the abstract and introduction accurately reflect the paper's contributions and scope?

   Answer: [Yes]

   Justification: The research problem and the main contributions of this study are clearly articulated in the abstract and introduction sections.

   Guidelines:

   - The answer NA means that the abstract and introduction do not include the claims made in the paper.
   - The abstract and/or introduction should clearly state the claims made, including the contributions made in the paper and important assumptions and limitations. A No or NA answer to this question will not be perceived well by the reviewers.
   - The claims made should match theoretical and experimental results, and reflect how much the results can be expected to generalize to other settings.
   - It is fine to include aspirational goals as motivation as long as it is clear that these goals are not attained by the paper.

2. **Limitations**

   Question: Does the paper discuss the limitations of the work performed by the authors?

   Answer: [Yes]

   Justification: The conclusion and future work section include a discussion of the study's limitations.

   Guidelines:

   - The answer NA means that the paper has no limitation while the answer No means that the paper has limitations, but those are not discussed in the paper.
   - The authors are encouraged to create a separate "Limitations" section in their paper.
   - The paper should point out any strong assumptions and how robust the results are to violations of these assumptions (e.g., independence assumptions, noiseless settings, model well-specification, asymptotic approximations only holding locally). The authors should reflect on how these assumptions might be violated in practice and what the implications would be.
   - The authors should reflect on the scope of the claims made, e.g., if the approach was only tested on a few datasets or with a few runs. In general, empirical results often depend on implicit assumptions, which should be articulated.
   - The authors should reflect on the factors that influence the performance of the approach. For example, a facial recognition algorithm may perform poorly when image resolution is low or images are taken in low lighting. Or a speech-to-text system might not be used reliably to provide closed captions for online lectures because it fails to handle technical jargon.
   - The authors should discuss the computational efficiency of the proposed algorithms and how they scale with dataset size.
   - If applicable, the authors should discuss possible limitations of their approach to address problems of privacy and fairness.
   - While the authors might fear that complete honesty about limitations might be used by reviewers as grounds for rejection, a worse outcome might be that reviewers discover limitations that aren't acknowledged in the paper. The authors should use their best judgment and recognize that individual actions in favor of transparency play an important role in developing norms that preserve the integrity of the community. Reviewers will be specifically instructed to not penalize honesty concerning limitations.

3. **Theory assumptions and proofs**

   Question: For each theoretical result, does the paper provide the full set of assumptions and a complete (and correct) proof?

Answer: [NA]

Justification: This paper does not involve theoretical assumptions

Guidelines:

- The answer NA means that the paper does not include theoretical results.
- All the theorems, formulas, and proofs in the paper should be numbered and cross-referenced.
- All assumptions should be clearly stated or referenced in the statement of any theorems.
- The proofs can either appear in the main paper or the supplemental material, but if they appear in the supplemental material, the authors are encouraged to provide a short proof sketch to provide intuition.
- Inversely, any informal proof provided in the core of the paper should be complemented by formal proofs provided in appendix or supplemental material.
- Theorems and Lemmas that the proof relies upon should be properly referenced.

4. **Experimental result reproducibility**

   Question: Does the paper fully disclose all the information needed to reproduce the main experimental results of the paper to the extent that it affects the main claims and/or conclusions of the paper (regardless of whether the code and data are provided or not)?

   Answer: [Yes]

   Justification: This paper provides a detailed description of the experimental setup, including the parameter tuning ranges, and the code will be made available as supplementary material.

   Guidelines:

   - The answer NA means that the paper does not include experiments.
   - If the paper includes experiments, a No answer to this question will not be perceived well by the reviewers: Making the paper reproducible is important, regardless of whether the code and data are provided or not.
   - If the contribution is a dataset and/or model, the authors should describe the steps taken to make their results reproducible or verifiable.
   - Depending on the contribution, reproducibility can be accomplished in various ways. For example, if the contribution is a novel architecture, describing the architecture fully might suffice, or if the contribution is a specific model and empirical evaluation, it may be necessary to either make it possible for others to replicate the model with the same dataset, or provide access to the model. In general. releasing code and data is often one good way to accomplish this, but reproducibility can also be provided via detailed instructions for how to replicate the results, access to a hosted model (e.g., in the case of a large language model), releasing of a model checkpoint, or other means that are appropriate to the research performed.
   - While NeurIPS does not require releasing code, the conference does require all submissions to provide some reasonable avenue for reproducibility, which may depend on the nature of the contribution. For example
     (a) If the contribution is primarily a new algorithm, the paper should make it clear how to reproduce that algorithm.
     (b) If the contribution is primarily a new model architecture, the paper should describe the architecture clearly and fully.
     (c) If the contribution is a new model (e.g., a large language model), then there should either be a way to access this model for reproducing the results or a way to reproduce the model (e.g., with an open-source dataset or instructions for how to construct the dataset).
     (d) We recognize that reproducibility may be tricky in some cases, in which case authors are welcome to describe the particular way they provide for reproducibility. In the case of closed-source models, it may be that access to the model is limited in some way (e.g., to registered users), but it should be possible for other researchers to have some path to reproducing or verifying the results.

5. **Open access to data and code**

Question: Does the paper provide open access to the data and code, with sufficient instructions to faithfully reproduce the main experimental results, as described in supplemental material?

Answer: [Yes]

Justification: The code will be made available as supplementary material.

Guidelines:

- The answer NA means that paper does not include experiments requiring code.
- Please see the NeurIPS code and data submission guidelines (`https://nips.cc/public/guides/CodeSubmissionPolicy`) for more details.
- While we encourage the release of code and data, we understand that this might not be possible, so “No” is an acceptable answer. Papers cannot be rejected simply for not including code, unless this is central to the contribution (e.g., for a new open-source benchmark).
- The instructions should contain the exact command and environment needed to run to reproduce the results. See the NeurIPS code and data submission guidelines (`https://nips.cc/public/guides/CodeSubmissionPolicy`) for more details.
- The authors should provide instructions on data access and preparation, including how to access the raw data, preprocessed data, intermediate data, and generated data, etc.
- The authors should provide scripts to reproduce all experimental results for the new proposed method and baselines. If only a subset of experiments are reproducible, they should state which ones are omitted from the script and why.
- At submission time, to preserve anonymity, the authors should release anonymized versions (if applicable).
- Providing as much information as possible in supplemental material (appended to the paper) is recommended, but including URLs to data and code is permitted.

6. **Experimental setting/details**

   Question: Does the paper specify all the training and test details (e.g., data splits, hyperparameters, how they were chosen, type of optimizer, etc.) necessary to understand the results?

   Answer: [Yes]

   Justification: The experimental setup is clearly detailed in both the main experimental section and the appendix.

   Guidelines:

   - The answer NA means that the paper does not include experiments.
   - The experimental setting should be presented in the core of the paper to a level of detail that is necessary to appreciate the results and make sense of them.
   - The full details can be provided either with the code, in appendix, or as supplemental material.

7. **Experiment statistical significance**

   Question: Does the paper report error bars suitably and correctly defined or other appropriate information about the statistical significance of the experiments?

   Answer: [Yes]

   Justification: This paper reports the mean and standard deviation over multiple runs.

   Guidelines:

   - The answer NA means that the paper does not include experiments.
   - The authors should answer "Yes" if the results are accompanied by error bars, confidence intervals, or statistical significance tests, at least for the experiments that support the main claims of the paper.
   - The factors of variability that the error bars are capturing should be clearly stated (for example, train/test split, initialization, random drawing of some parameter, or overall run with given experimental conditions).

- The method for calculating the error bars should be explained (closed form formula, call to a library function, bootstrap, etc.)
- The assumptions made should be given (e.g., Normally distributed errors).
- It should be clear whether the error bar is the standard deviation or the standard error of the mean.
- It is OK to report 1-sigma error bars, but one should state it. The authors should preferably report a 2-sigma error bar than state that they have a 96% CI, if the hypothesis of Normality of errors is not verified.
- For asymmetric distributions, the authors should be careful not to show in tables or figures symmetric error bars that would yield results that are out of range (e.g. negative error rates).
- If error bars are reported in tables or plots, The authors should explain in the text how they were calculated and reference the corresponding figures or tables in the text.

8. **Experiments compute resources**

   Question: For each experiment, does the paper provide sufficient information on the computer resources (type of compute workers, memory, time of execution) needed to reproduce the experiments?

   Answer: [Yes]

   Justification: The computational resources utilized are described in the experimental implementation details section.

   Guidelines:

   - The answer NA means that the paper does not include experiments.
   - The paper should indicate the type of compute workers CPU or GPU, internal cluster, or cloud provider, including relevant memory and storage.
   - The paper should provide the amount of compute required for each of the individual experimental runs as well as estimate the total compute.
   - The paper should disclose whether the full research project required more compute than the experiments reported in the paper (e.g., preliminary or failed experiments that didn't make it into the paper).

9. **Code of ethics**

   Question: Does the research conducted in the paper conform, in every respect, with the NeurIPS Code of Ethics `https://neurips.cc/public/EthicsGuidelines`?

   Answer: [Yes]

   Justification: This work fully adheres to the NeurIPS Code of Ethics in all aspects.

   Guidelines:

   - The answer NA means that the authors have not reviewed the NeurIPS Code of Ethics.
   - If the authors answer No, they should explain the special circumstances that require a deviation from the Code of Ethics.
   - The authors should make sure to preserve anonymity (e.g., if there is a special consideration due to laws or regulations in their jurisdiction).

10. **Broader impacts**

    Question: Does the paper discuss both potential positive societal impacts and negative societal impacts of the work performed?

    Answer: [Yes]

    Justification: The introduction highlights the significance of multi-source collaborative modeling.

    Guidelines:

    - The answer NA means that there is no societal impact of the work performed.
    - If the authors answer NA or No, they should explain why their work has no societal impact or why the paper does not address societal impact.

- Examples of negative societal impacts include potential malicious or unintended uses (e.g., disinformation, generating fake profiles, surveillance), fairness considerations (e.g., deployment of technologies that could make decisions that unfairly impact specific groups), privacy considerations, and security considerations.
- The conference expects that many papers will be foundational research and not tied to particular applications, let alone deployments. However, if there is a direct path to any negative applications, the authors should point it out. For example, it is legitimate to point out that an improvement in the quality of generative models could be used to generate deepfakes for disinformation. On the other hand, it is not needed to point out that a generic algorithm for optimizing neural networks could enable people to train models that generate Deepfakes faster.
- The authors should consider possible harms that could arise when the technology is being used as intended and functioning correctly, harms that could arise when the technology is being used as intended but gives incorrect results, and harms following from (intentional or unintentional) misuse of the technology.
- If there are negative societal impacts, the authors could also discuss possible mitigation strategies (e.g., gated release of models, providing defenses in addition to attacks, mechanisms for monitoring misuse, mechanisms to monitor how a system learns from feedback over time, improving the efficiency and accessibility of ML).

11. **Safeguards**

    Question: Does the paper describe safeguards that have been put in place for responsible release of data or models that have a high risk for misuse (e.g., pretrained language models, image generators, or scraped datasets)?

    Answer: [NA]

    Justification: The ResNet model and datasets used in this study are all open-source.

    Guidelines:

    - The answer NA means that the paper poses no such risks.
    - Released models that have a high risk for misuse or dual-use should be released with necessary safeguards to allow for controlled use of the model, for example by requiring that users adhere to usage guidelines or restrictions to access the model or implementing safety filters.
    - Datasets that have been scraped from the Internet could pose safety risks. The authors should describe how they avoided releasing unsafe images.
    - We recognize that providing effective safeguards is challenging, and many papers do not require this, but we encourage authors to take this into account and make a best faith effort.

12. **Licenses for existing assets**

    Question: Are the creators or original owners of assets (e.g., code, data, models), used in the paper, properly credited and are the license and terms of use explicitly mentioned and properly respected?

    Answer: [Yes]

    Justification: All relevant works are properly cited, and all open-source assets are used in accordance with their licensing terms.

    Guidelines:

    - The answer NA means that the paper does not use existing assets.
    - The authors should cite the original paper that produced the code package or dataset.
    - The authors should state which version of the asset is used and, if possible, include a URL.
    - The name of the license (e.g., CC-BY 4.0) should be included for each asset.
    - For scraped data from a particular source (e.g., website), the copyright and terms of service of that source should be provided.

- If assets are released, the license, copyright information, and terms of use in the package should be provided. For popular datasets, `paperswithcode.com/datasets` has curated licenses for some datasets. Their licensing guide can help determine the license of a dataset.
- For existing datasets that are re-packaged, both the original license and the license of the derived asset (if it has changed) should be provided.
- If this information is not available online, the authors are encouraged to reach out to the asset's creators.

13. **New assets**

    Question: Are new assets introduced in the paper well documented and is the documentation provided alongside the assets?

    Answer: [Yes]

    Justification: An anonymized version of the code developed in this study is included in the supplementary materials to ensure reproducibility.

    Guidelines:

    - The answer NA means that the paper does not release new assets.
    - Researchers should communicate the details of the dataset/code/model as part of their submissions via structured templates. This includes details about training, license, limitations, etc.
    - The paper should discuss whether and how consent was obtained from people whose asset is used.
    - At submission time, remember to anonymize your assets (if applicable). You can either create an anonymized URL or include an anonymized zip file.

14. **Crowdsourcing and research with human subjects**

    Question: For crowdsourcing experiments and research with human subjects, does the paper include the full text of instructions given to participants and screenshots, if applicable, as well as details about compensation (if any)?

    Answer: [NA]

    Justification: The paper does not involve crowdsourcing nor research with human subjects.

    Guidelines:

    - The answer NA means that the paper does not involve crowdsourcing nor research with human subjects.
    - Including this information in the supplemental material is fine, but if the main contribution of the paper involves human subjects, then as much detail as possible should be included in the main paper.
    - According to the NeurIPS Code of Ethics, workers involved in data collection, curation, or other labor should be paid at least the minimum wage in the country of the data collector.

15. **Institutional review board (IRB) approvals or equivalent for research with human subjects**

    Question: Does the paper describe potential risks incurred by study participants, whether such risks were disclosed to the subjects, and whether Institutional Review Board (IRB) approvals (or an equivalent approval/review based on the requirements of your country or institution) were obtained?

    Answer: [NA]

    Justification: The paper does not involve crowdsourcing nor research with human subjects.

    Guidelines:

    - The answer NA means that the paper does not involve crowdsourcing nor research with human subjects.
    - Depending on the country in which research is conducted, IRB approval (or equivalent) may be required for any human subjects research. If you obtained IRB approval, you should clearly state this in the paper.

- We recognize that the procedures for this may vary significantly between institutions and locations, and we expect authors to adhere to the NeurIPS Code of Ethics and the guidelines for their institution.
- For initial submissions, do not include any information that would break anonymity (if applicable), such as the institution conducting the review.

16. **Declaration of LLM usage**

    Question: Does the paper describe the usage of LLMs if it is an important, original, or non-standard component of the core methods in this research? Note that if the LLM is used only for writing, editing, or formatting purposes and does not impact the core methodology, scientific rigorousness, or originality of the research, declaration is not required.

    Answer: [NA]

    Justification: LLMs were used only for grammar correction and refinement.

    Guidelines:

    - The answer NA means that the core method development in this research does not involve LLMs as any important, original, or non-standard components.
    - Please refer to our LLM policy (`https://neurips.cc/Conferences/2025/LLM`) for what should or should not be described.