In this appendix, we provide additional details and results that were not included in the main paper but we evaluated during the study. This appendix contains the following items,

- Appendix A The detailed description of the CFedDC algorithm.
- Appendix B A detailed description of the experimental setup used for the experiments in this paper.
- In Appendix C, we present additional experiments conducted during this study that were not included in the main paper.

A. Algorithm

We divided the approach of CFedDC into three components,

- Identification of DR and DL users from the user pool (see Algorithm 1).
- Local model updates performed independently by each user (see Algorithm 2).
- The Clustered Federated daisy-chaining strategy (see Algorithm 3), which forms the core of CFedDC by integrating clustering, daisy-chaining, and federated aggregation.

A.1. DR and DL identification

To identify DR and DL users (see Algorithm 1), the server computes the fraction of training samples contributed by each user with respect to the total number of samples available across all users. Based on their data contribution fractions, these users are then sorted in descending order based. Starting from the user with the highest contribution, we sequentially add users to the DR group until the cumulative contribution reaches or exceeds a predefined threshold Δ (e.g., for our experiments it is 50% of the total data). Once this threshold is met, the remaining users are categorized as DL.

Algorithm 1 DR/DL User indentification

Require: Set of users $U = \{u_1, u_2, \ldots, u_N\}$ where each user u_i has d_i training samples; threshold $\Delta \in (0, 1]$
Ensure: Sets of users U_{DR} and U_{DL}
Compute total number of samples: $D \leftarrow \sum_{i=1}^{N} d_i$
for each user $u_i \in U$ **do**
    Compute data fraction $f_i \leftarrow \frac{d_i}{D}$
Sort users in descending order of f_i: $f_{(1)} \geq f_{(2)} \geq \cdots \geq f_{(N)}$
Initialize $U_{\text{DR}} \leftarrow \emptyset, U_{\text{DL}} \leftarrow \emptyset, cumulative \leftarrow 0$
for $i = 1$ to N **do**
    $U_{\text{DR}} \leftarrow U_{\text{DR}} \cup \{u_{(i)}\}$
    $cumulative \leftarrow cumulative + f_{(i)}$
    if $cumulative \geq \Delta$ **then**
        break
$U_{\text{DL}} \leftarrow U \setminus U_{\text{DR}}$
    return $U_{\text{DR}}, U_{\text{DL}}$

A.2. Local Update

In Algorithm 2, each DR user initializes its local model using the cluster-head model, or the global model if its cluster affiliation is unknown. This typically occurs at the begining of the federated training or when a new user joins midway through the training process. DL users initialize their model with the peer DR model. Each user while local update compute the Equation (4) (Line 6) to update their local model.

Algorithm 2 Local Update

1: **Initialization:** $\theta_i^{t,s} = w^t, \theta_i^t = \theta_i^{t-1}$ ▷ $\theta_i^{t,s}$ is the current local model, θ_i^{t-1} is the previous local model, w^t is the current global model
2: **Input:** w^t, S, D_i, x, y
3: **Output:** θ_i^t
4: **for** local step $s \leftarrow 0, 1, \ldots, S-1$ **do**
5: Compute gradient of local loss:

$$\nabla F_i(\theta_i^{t,s}) = \frac{1}{|D_i|} \sum_{(x,y) \in D_i} \nabla F_i(\theta_i^{t,s}; x, y)$$

6: Update local model using gradient descent:

$$\theta_i^{t+1} = \theta_i^t - \eta(\nabla F_i(\theta_i^t) + (\theta_i^t - m_i^t))$$

7: Set $\theta_i^t \leftarrow \theta_i^{t,S}$
8: Send updated local model θ_i^t to the server

A.3. CFedDC: Clustered Federated daisy-chaining

The objective of designing Algorithm 3 is to tackle challenges arising from the variability in privacy scores among users, which leads to data heterogeneity, and the issue of limited annotations contributed by a large number of users. CFedDC addresses these challenges through two key strategies: (1) employing user clustering along with user- and cluster-specific L2 regularization to alleviate data heterogeneity, and (2) introducing an efficient daisy-chaining mechanism that transfers trained models from data-rich (DR) to data-less (DL) users, followed by few-shot fine-tuning to address data scarcity. Initially, Algorithm 3 sets up the global model weights w^0, cluster weights w_c^0 for C clusters, the number of global communication rounds T, and local update steps L. Each user shares their data availability with the server, based on which the server partitions them into DR and DL groups. In each global round t, subsets of DR and DL users are sampled. Each selected user receives either the global model w^t or their associated cluster model $w_{c(i)}^t$, depending on whether their cluster assignment is known.

DR users perform local updates on the received model to obtain updated parameters θ_i^{t+1}. The server then extracts representations from the last k layers of each DR user's model and flattens them to vectors z_i, which are clustered into C groups. Based on this, each DR user is assigned to a cluster. In parallel, each selected DL user evaluates the loss on their validation data across the cluster heads and chooses the best-fitting cluster. From that cluster, a DL user selects a DR peer at random, copies its model, and fine-tunes it on

its own data for L steps.

After all updates, the server performs cluster-wise model aggregation by averaging the updated models from all users assigned to each cluster. Finally, the global model is updated as a weighted combination of the cluster heads. This daisy-chaining and clustering mechanism facilitates effective knowledge transfer from data rich to data- users while preserving personalization and ensuring robustness under data heterogeneity.

Algorithm 3 CFedDC

Initialize: $w_0^0 = w_1^0 = \ldots = w_C^0 = w^0, T, L$ ▷ Initialize C clusters, initialize Global iterations, and Local iterations
All users broadcast their data availability to the server.
DR, DL = Divide_users(N)
for $t \leftarrow 0$ **to** T **do**
  $DR_\kappa, DL_\delta \leftarrow$ Select_users($\mathcal{N}$) ▷ Select subset of DR and DL users
  send $\begin{cases} w^t_{c(i)}, & \text{if user } i \text{ knows its cluster } c(i) \\ w^t, & \text{otherwise} \end{cases}$
  for $i \in DR_\kappa$ **in parallel do**
    $\theta_i^{t+1} \leftarrow$ Local_Update$\Big(\begin{cases} w^t, & \text{if } Lg_i \notin C, \\ w^t_{c(i)}, & \text{otherwise} \end{cases}\Big)$
  for $i \in DR_\kappa$ **in parallel do**
    $\mathbf{z}_i \leftarrow \text{vec}\big(\text{LastKLayers}(\theta_i^{t+1})\big)$ ▷ flatten the weights of the last k layers
  $\{c(i)\}_{i \in DR_\kappa} \leftarrow \text{Cluster}\big(\{\mathbf{z}_i\}_{i \in DR_\kappa}, C\big)$ ▷ e.g. K-means into C clusters
  for $i \in DL_\delta$ **in parallel do**
    $c^\star(i) \leftarrow \underset{c \in \{1,\ldots,C\}}{\arg\min}\ \mathcal{L}\big(\mathcal{V}_i;\, w_c^{t+1}\big)$ ▷ choose cluster whose head model minimises loss on i's validation data
    $j \leftarrow \text{RandomPick}\big(\{\, k \in DR_\kappa \mid c(k) = c^\star(i) \,\}\big)$ ▷ pick a data-rich peer from the same cluster
    $\theta_i^{t,0} \leftarrow \theta_j^{t+1}$ ▷ receive (copy) the trained model from user j
    $\theta_i^{t+1} \leftarrow \text{Local\_Update}\big(\theta_i^{t,0},\, L\big)$ ▷ fine-tune on i's own data for L local steps
  for $c = 1$ **to** C **in parallel do**
    for $i \in \mathcal{N}_c$ **do**
      $\mathbf{w}_c^{t+1} \leftarrow \sum_{i=1}^{\mathcal{N}_c} r_i\, \boldsymbol{\theta}_i^{t+1}$ ▷ Cluter update
  $\mathbf{w}^{t+1} \leftarrow \sum_{c=1}^{C} r_c\, \mathbf{w}_c^{t+1}$ ▷ Global update

A.4. Convergence Discussion

The convergence of CFedDC can be discussed based on the framework of proximal-based federated optimization. By reformulating the local objective with both consensus and inertia regularizers, CFedDC effectively controls client drift, a common challenge in heterogeneous FL settings. This is analogous to FedProx[13], which adds a proximal term to stabilize updates, and pFedMe[23], which leverages a Moreau envelope formulation to guarantee convergence under smooth and convex losses. In CFedDC, the time-varying interpolation parameter $\lambda_t \in [\lambda_{min}, \lambda_{max}]$ balances stability and innertia that ensures the local updates remain bounded while allowing adaptation to individual users. Under standard assumptions of L-smoothness and bounded variance, the proximal gradient step used in CFedDC converges to a stationary point at a rate of $O(1/T)$, where T is the number of communication rounds. The empirical convergence curves validate this theoretical expectation, showing faster and more stable convergence compared to FedDC, particularly for data-limited users benefiting from daisy-chaining.

B. Experimental Setup

We simulated a federated learning environment to benchmark CFedDC against several state-of-the-art FL algorithms. Each model was trained over 30 global iterations, with 5 local iterations per round. For clustering within CFedDC, we employed the k-means algorithm.

B.1. Hyperparameter settings:

Here we describe the details of hyperparameter used in this paper for the experiments

- For the experiments corresponding to Table 2, 4, 5, Figure 6, and 7, the hyperparameters are set as follows: $\lambda_t \in [0.4, 0.6]$, $\kappa = 1.0$, $\delta = 1.0$, learning rate = 0.05, number of clusters = 2, and the optimizer used is SGD.
- For the experiments shown in Figure 8, the following configuration is used: fixed $\lambda_t = 0.5$, with κ and δ taking values from the set $0.1, 0.5, 0.8, 1.0$; the learning rate is 0.05, the number of clusters is 2, and the optimizer is SGD.
- For the experiments shown in Figure 9 the following configuration is used: fixed $\lambda_t = 0.5$, with κ *and* $\delta = 1.0$ learning rate is 0.05, the number of clusters is 2, and the optimizer is SGD, $\lambda_t = 0.5$.
- For the experiments illustrated in Figure 10a, 10b, 15a, and 15b, the setup includes $\kappa = 1.0$ and $\delta = 1.0$. When λ_t is time-varying across global rounds, its values are chosen from $[0.1, 0.9]$, $[0.2, 0.8]$, $[0.3, 0.7]$, $[0.4, 0.6]$. For the fixed case, $\lambda_t = 0.5$, and when the regularization terms are omitted from the loss function, $\lambda_t = 0$. The learning rate is 0.05, the number of clusters is 2, and SGD is used as the optimizer.

B.1.1. Hardware used:

All experiments were conducted on an NVIDIA A100 Tensor Core GPU with 40GB of high-bandwidth memory.

B.2. DR/DL Identification

In Figure 11, we present the identification of DR and DL users within the DIPA2 dataset. A total of 442 users participated, out of which 114 users (U_{DR}) were responsible for 50% of the total annotations. The remaining 328 users (U_{DL}) contributed the other half.

B.3. PIONet

In Figure 12, we provide a detailed illustration of the PIONet architecture introduced in the main paper, including layer configurations, dimensions, and activation functions.

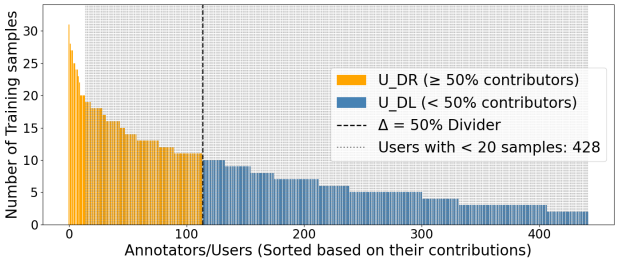


Figure 11. Samples distribution across users

B.4. Privacy risk score distribution across annotations

In Figure 13, we analyze the distribution of privacy risk scores of the annotations done by the 442 users independently. Annotations were mentioned using a 7-point Likert scale ranging from -3 to 3, where -3 indicates very low privacy risk and 3 represents very high privacy risk.

C. Experiments

In this appendix we have detailed analysis of the performance of DR and DL users.

C.1. Performance of DR and DL across State-of-the-art

Here we carry out the experiments separately for DR and DL users.

Performance of DR Users: As shown in Table 4, our proposed model CFedDC achieves a CMAE of 1.02 and an MAE of 0.88, indicating strong performance in privacy score prediction. It consistently outperforms the nonclustered federated baselines, where FedAvg reports CMAE and MAE of 1.06 and 1.00 respectively, and FedProx yields 1.05 and 0.98. It also surpasses the isolated (siloed) training setup, which results in CMAE of 1.16 and MAE of 1.04. When compared with FedMEM, which employs clustered personalization and achieves a CMAE of 1.04 and an MAE of 0.88, CFedDC offers a slight improvement in CMAE while maintaining the same MAE. Although FedDC obtains the lowest CMAE and MAE (0.99 and 0.85), CFedDC delivers nearly comparable performance, highlighting the effectiveness of combining clustering and daisy-chaining in federated learning.

Performance of DL Users: As observed in Table 5, our proposed model CFedDC achieves a CMAE of 0.63 and an MAE of 0.59, which demonstrate a strong performance in predicting privacy scores. It clearly outperforms the non-clustered federated approaches such as FedAvg has CMAE and MAE 0.84, 0.81 respectively and FedProx got CMAE and MAE 0.81 and 0.78 respectively, as well as the isolated (Siloed) training setting without model/data sharing which has CMAE and MAE 1.05, and 1.03 respectively. Compared to FedMEM, which uses clustered personalization gives CMAE of 0.69 and MAE of 0.67, FedDC gives CMAE of 0.64 and MAE of 0.6. Therefore, for DL users CFedDC achieves the best performance among all compared methods.

Table 4. Performance analysis of PIONet on siloed and federated setup for DR users

Model	**Privacy score**		
		CMAE	**MAE**
Siloed	No sharing	1.16	1.04
Federated			
FedAvg [15]	Non-Clustered	1.06	1.00
FedProx [13]	Non-Clustered	1.05	0.98
FedMEM [2]	Clustered	1.04	0.88
FedDC [11]	daisy-chaining	**0.99**	**0.85**
CFedDC	Clustering and daisy-chaining	1.02	0.88

Table 5. Performance analysis of PIONet on siloed and federated setup for DL users

Model	**Privacy score**		
		CMAE	**MAE**
Siloed	No sharing	1.05	1.03
Federated			
FedAvg [15]	Non-Clustered	0.84	0.81
FedProx [13]	Non-Clustered	0.81	0.78
FedMEM [2]	Clustered	0.69	0.67
FedDC [11]	daisy-chaining	0.64	0.6
CFedDC	Clustering and daisy-chaining	**0.63**	**0.59**

C.2. Effect of partial participation on DR users

varying participation of DR: We begin by analyzing the effect of partial participation of DR users in federated learning while keeping full participation of DL users (i.e., $\delta = 1$). In Figure 14a, we evaluate the impact of varying the participation factor κ for DR users on the model's performance, measured using CMAE.

From the figure, we observe that increasing κ from 0.1 (10% participation) to 0.5 (50% participation) leads to a significant improvement in performance, with CMAE dropping from 1.19 to 0.98. However, further increasing κ to 0.8 (80% participation) and 1.0 (100% participation) results in only marginal changes in performance (CMAE of 1.00 and 1.05, respectively).

Therefore, from the results we can say a participation rate of 50% for DR users per FL round is sufficient to achieve strong performance, making the training process more efficient without requiring full user involvement.

Varying participation of DL users: Next, we analyze the effect of partial participation of DL users in federated learning while keeping full participation of DR users fixed (i.e., $\kappa = 1$). From the figure, we observe that increasing δ from

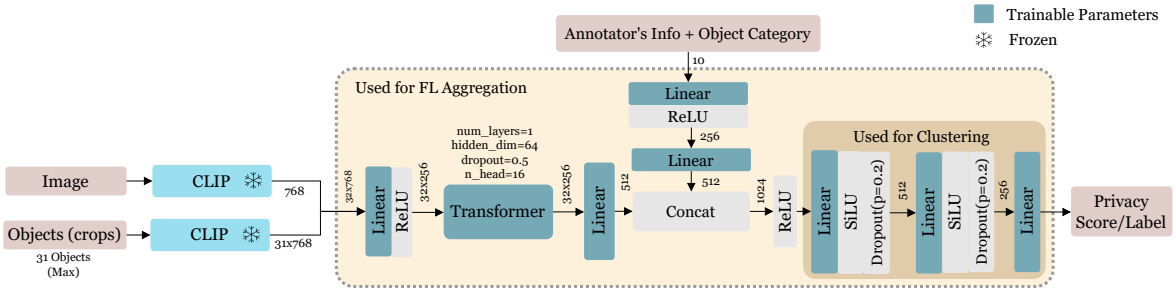


Figure 12. Detailed PIONet Architecture

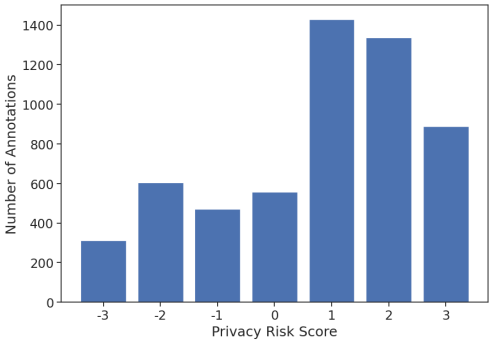
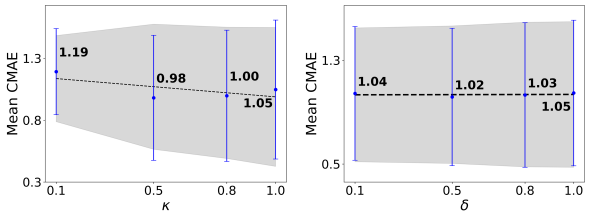


Figure 13. Privacy Risk Score distribution across annotations

0.1 (10% participation) to 0.5 (50% participation) leads to a slight improvement in performance, with CMAE decreasing from 1.04 to 1.02. However, further increasing δ to 0.8 (80% participation) and 1.0 (100% participation) results in only marginal changes in CMAE, which remains at 1.03 and 1.05, respectively. These results indicate that the participation of DL users doesn't affect the performance of DR users.



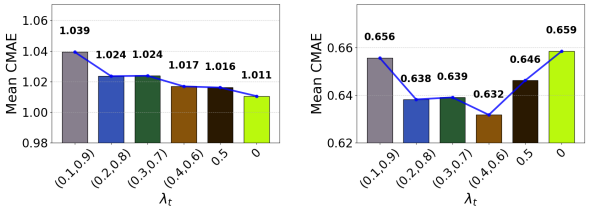
(a) Variable κ and constant δ (b) Constant κ and variable δ

Figure 14. Effect of κ and δ on DR users

C.3. Ablation study on consensus and intertia regularizer (λ_t)

In Figure 10, we analyze the impact of the hyperparameter λ_t on the performance of CFedDC for DR (Figure 15a) and DL (Fig. 15b). As shown in Figure 15a, increasing λ_{min} while decreasing λ_{max} leads to a reduction in CMAE. When $\lambda_{min} = 0.4$ and $\lambda_{max} = 0.6$, the DR users achieve their best performance with a time-varying λ_t. For a fixed $\lambda_t = 0.5$, or when setting $\lambda_t = 0$ (effectively removing the hyperparameter's influence from the loss function), the DR users show performance comparable to that achieved with $\lambda_{min} = 0.4$ and $\lambda_{max} = 0.6$.

similarly in Figure 15b, for DL users a similar trend is observed for time-varing λ_t. The CMAE is minimum for $\lambda_{min} = 0.4$ and $\lambda_{max} = 0.6$. A different trend is observed when using a fixed $\lambda_t = 0.5$ or setting $\lambda_t = 0$, where the CMAE increases compared to the time-varying case.



(a) Data Rich users (b) Data Limited users

Figure 15. Performance of RF and RL on different values of lambda within a range of 0 to 1

C.4. Statistical Significance test

We performed Mann–Whitney U test comparing FedDC and CFedDC over five independent runs, each evaluated across 30 global rounds. The reported p-values for all runs (ranging from $2.15 \times e^{-}6$ to 0.001) are well below the 0.05 threshold, indicating that the performance differences between FedDC and CFedDC are statistically significant. The corresponding U statistics belongs to 129 to 223. That quantify the magnitude of separation between CFedDC and FedDC, supports that CFedDC offers a meaningful improvement over FedDC.

Table 6. Statistical significance analysis between FedDC and CFedDC across 5 independent runs, evaluated at each of the 30 global rounds

Mann-Whitney U test	run 1	run 2	run 3	run 4	run 5
p-value	$5.85e^{-06}$	$2.15e^{-06}$	$1.42e^{-05}$	$4.08e^{-05}$	0.001
Statistics	143	129	156	172	223

C.5. Performance of CFedDC on FMNIST dataset

The results in Table 7 show that CFedDC (CFedDC) achieves 99.52% accuracy on the FMNIST dataset that significantly outperform pFedMe, which reaches 96.3%. This shows the effectiveness of CFedDC in leveraging clustering and daisy-chaining for highly accurate and personalized learning, even under heterogeneous client distributions.

Table 7. Performance of CFedDC on FMNIST with 50 clients (8 Data-Rich, 42 Data-Limited) organized into 2 clusters. Global rounds 20, Local iters 5

Method	FMnist (Accuracy)
CFedDC	99.52
pFedMe	96.3 [1]