Practical Confidential Clustering for Federated Learning using Efficient Secret-Sharing

Cédric Prigent, Sonia Ben Mokhtar
CNRS, INSA Lyon, Université Lyon 1, LIRIS, UMR5205
{cedric.prigent, sonia.ben-mokhtar}@liris.cnrs.fr

***Abstract*—Federated Learning (FL) often suffers from statistical heterogeneity, where diverging data distributions prevent a single global model from achieving high utility for all participants. Clustered Federated Learning (CFL) addresses this problem by grouping similar clients to train personalized sub-models, yet performing such clustering while ensuring strict confidentially with minimal communication overhead remains a significant challenge. This paper introduces CCFL, a framework for Confidential Clustering that executes clustering logic, including distance computations and centroid updates, entirely within the encrypted domain using Replicated Secret Sharing (RSS). CCFL orchestrates a unified pipeline that integrates client-side feature extraction, an anonymization server for oblivious shuffling, and Multi-Party Computation (MPC)-based clustering to ensure both input privacy and participant unlinkability. By employing a ring-based Euclidean distance optimization, CCFL significantly reduces interactive rounds, achieving up to $46.7\times$ computational speedup under realistic LAN and WAN constraints. Our one-shot clustering approach decouples heavy cryptographic computation from the training loop, allowing the system to scale seamlessly to thousands of clients with negligible edge overhead. Finally, downstream evaluations confirm that CCFL achieves a superior privacy-utility trade-off, matching or outperforming state-of-the-art personalized FL paradigms without exposing sensitive feature geometries.**

***Index Terms*—Federated Learning, Personalization, Confidential Clustering, Multi-Party Computation**

I. Introduction

Federated Learning (FL) has emerged as a cornerstone for training machine learning models on decentralized, sensitive data. However, the practical utility of FL is often hampered by statistical heterogeneity, where diverging local data distributions (*e.g.*, covariate or label shifts) prevent a single global model from achieving high utility for all participants [1].

To address this, Personalized FL seeks to balance collective knowledge with local adaptation. Clustered Federated Learning (CFL) has emerged as one such solution by grouping clients with similar objectives to train shared personalized sub-models [2–10]. Existing CFL protocols typically rely on client-side gradients [3, 4] or latent features [7, 8] to measure similarity. While effective, these approaches expose significant privacy vulnerabilities: evaluating these metrics in plaintext allows honest-but-curious servers to perform model inversion, membership inference, or property inference attacks [11]. Alternatively, some frameworks shift the clustering burden to the clients, which may be computationally prohibitive for resource-constrained edge devices [5, 6, 9].

While the domain of confidential clustering is well-established in centralized settings, its application within the constraints of federated networks remains largely unexplored – partly because existing foundations are primarily tailored to low-dimensional data points where the computational cost of privacy-preserving primitives, such as Homomorphic Encryption (HE) and Multi-Party Computation (MPC), remains manageable [12–18]. However, FL requires scaling these methods to high-dimensional model updates or feature vectors [2, 3, 7, 8]. This transition from compact data to high-dimensional, dynamic environments causes the overhead of distance computation to scale quadratically, creating a computational and communication bottleneck that our work specifically addresses.

To date, the literature on privacy-preserving CFL has predominantly relied on perturbation-based techniques, such as Differential Privacy (DP) [8, 19–21]. While DP offers formal mathematical bounds, it imposes a rigid privacy-utility trade-off where injected noise often degrades clustering quality and downstream accuracy [22–24]. In addition, in domains where the distribution itself is sensitive (*e.g.*, revealing personal habits or health categories), local record-level DP may offer insufficient protection due to correlated behaviors and dependent data points [25]. This creates a fundamental dilemma: while local record-level DP fails to protect the aggregate distribution, applying DP at the distribution or client-level would significantly exacerbate cluster drift, severely compromising the model's personalization utility.

In this work, we bridge this gap by re-engineering confidential clustering specifically for the FL context. We shift the burden of cryptographic computation from resource-constrained edge devices to an optimized, high-availability computing cluster consisting of three non-colluding parties. To overcome the communication bottlenecks of MPC, our framework replaces per-round clustering with a one-shot, high-efficiency operation, enabling practical, large-scale deployment without sacrificing personalization utility. To achieve this, we integrate three core technologies into a unified pipeline: (1) client-side feature extraction to project raw data into a low-dimensional space, (2) oblivious shuffling via an intermediate anonymization server to decouple client identities from their features, and (3) MPC-based clustering to perform distance evaluations and centroid updates entirely within the encrypted domain.

Our core contributions are summarized as follows:

- **Efficient Confidential Clustering Protocol:** We propose a framework that strategically combines $(2, 3)$-RSS, a

ring-based Euclidean distance optimization, and mini-batching into a unified pipeline. This orchestration addresses the primary bottlenecks of MPC, specifically, reducing circuit depth and minimizing communication rounds, achieving up to $46.7\times$ computational speedup in diverse network conditions.
- **Decoupled One-Shot Architecture:** We introduce a seamless, one-shot clustering pipeline that weaves confidential clustering into the FL lifecycle. By decoupling the heavy cryptographic computation from the per-round training loop, we ensure that personalization occurs without exposing raw features or imposing multi-model communication penalties on edge devices, enabling the system to scale to thousands of clients.
- **Robust and Practical Deployment:** We provide a production-oriented deployment using three physical nodes of the Grid'5000 [26] distributed testbed, demonstrating high resilience across realistic LAN, MAN, and cross-border WAN network profiles. We release our complete implementation, including configuration files, deployment scripts, and encrypted evaluation primitives, to support reproducibility and future research.[1]
- **State-of-the-Art Privacy-Utility Trade-off:** We evaluate our framework against global, personalized, and clustered FL baselines, proving that CCFL sustains near-cleartext model convergence across diverse non-IID scenarios while providing a superior utility-privacy trade-off compared to noise-based DP methods.

II. Background and Motivation

This section establishes the mathematical foundations for Clustered Federated Learning (CFL) and Replicated Secret Sharing (RSS) before formalizing the privacy-preserving problem statement.

A. Preliminaries: Clustered Federated Learning

CFL aims to partition clients into K clusters $\{C_k\}_{k=1}^K$ such that the overall model losses are minimized across the federation. We formalize CFL as the joint optimization problem of partitioning clients and training personalized models $\{\theta_k\}_{k=1}^K$:

$$\min_{\{\theta_k\}_{k=1}^K, \{C_k\}_{k=1}^K} \left(\sum_{k=1}^{K} \sum_{i \in C_k} \mathcal{L}_i(\theta_k) \right) \tag{1}$$

where $\mathcal{L}_i(\cdot)$ denotes the local loss function of client i. Existing CFL literature typically address this problem following three distinct approaches: (1) iterative similarity based on gradient trajectories [3, 4], (2) dynamic routing using local empirical risk minimization (ERM) [5, 6, 9], and (3) one-shot clustering using compact feature embeddings [7, 8].

However, these methods face a fundamental trade-off between privacy and efficiency. Iterative methods suffer from high communication latency and are often incompatible with secure aggregation, as they require access to individual client updates. Dynamic routing imposes a prohibitive communication and computation overhead for edge clients. Whereas, one-shot clustering may introduce privacy breaches by communicating plaintext embedding vectors with a central server, although providing a more cost-efficient strategy. Ultimately, the challenge lies in developing a privacy-preserving clustering mechanism that achieves the efficiency of one-shot protocols without compromising the confidentiality of the underlying client data.

B. Preliminaries: Replicated Secret Sharing

Multi-Party Computation (MPC) provides the cryptographic foundation for privacy-preserving FL. We adopt the 2-out-of-3 Replicated Secret Sharing (RSS) scheme [27, 28], which enables efficient arithmetic operations over rings $\mathbb{Z}_{2^\kappa}$. In a $(2,3)$-RSS configuration, a secret value $x \in \mathbb{Z}_{2^\kappa}$ is additively shared as a tuple $[\![x]\!] = (x_1, x_2, x_3)$ where $x = (x_1 + x_2 + x_3) \pmod{2^\kappa}$. These shares are distributed among three computing parties such that each party P_k holds two shares (e.g., P_1 holds $\{x_1, x_2\}$). This configuration provides:

- *Information-theoretic security*: No single party can reconstruct x.
- *Efficient Reconstruction*: Any two parties can reconstruct the secret by summing their shares, facilitating low-latency synchronization.
- *Constant Round Complexity*: The protocol supports multiplication operations in a single round of communication, which is critical for minimizing latency in multiplication-intensive processes such as computing the Squared Euclidean Distance (SED) metric.

C. Problem Statement: Confidential Clustering

We aim to solve the K-Means clustering optimization problem in a semi-honest setting where servers faithfully follow the protocol but may attempt to infer sensitive attributes from shared data:

$$\min_{\{\mathbf{c}_j\}_{j=1}^k} \left(\sum_{i=1}^{m} \sum_{j=1}^{k} w_{ij} \|\mathbf{x}_i - \mathbf{c}_j\|^2 \right)$$

where $w_{ij} \in \{0,1\}$ denotes cluster assignments and $\mathbf{c}_j$ is the centroid of cluster j. To ensure full confidentiality in a distributed FL environment, our framework must satisfy four constraints:

1) *Input Privacy:* Raw feature vectors $\mathbf{x}_i \in \mathbb{R}^d$ must never be shared in plaintext.
2) *Intermediate Confidentiality:* All intermediate computations (distances, centroids, and assignments) must remain encrypted in the MPC domain.
3) *Output Restriction:* Only the final cluster assignments w_{ij} may be disclosed to the aggregator.
4) *Participant Unlinkability:* The mapping between a client's identity and their cluster assignment must be protected via oblivious shuffling to prevent the leakage of group memberships.

[1]The source code repository is available anonymously at: https://anonymous.4open.science/r/MPC-C024/

Accuracy vs. Efficiency Trade-off. The fundamental tension in our approach lies between representation richness and computational overhead. High-dimensional embeddings improve clustering quality but increase the complexity of MPC operations, which scale linearly with the dimensionality d and the total number of participants m. Our goal is to minimize communication overhead and operational latency while maintaining sufficient numerical headroom to ensure convergence parity with cleartext baselines.

III. System and Threat Model

This section defines the structural architecture of our framework, outlines the adversarial conditions under which it operates, and articulates the core security goals. We formalize the non-collusion boundaries among our network actors and detail the secure multi-party pipelines that preserve complete data confidentiality and participant unlinkability.

A. Entities

Our system model is structured around four distinct logical entities designed to enforce a rigorous segregation of metadata and cryptographic keys.

Clients. A set of distributed edge peers holding private local datasets $\mathcal{D}_{1\leq i\leq N}$. Clients perform local, one-shot feature extraction on their raw data, split the resulting feature vectors into randomized secret shares, and encrypt these components using the public keys of the computing servers to ensure secure communication.

Computing Servers (CS). A cluster of three non-colluding nodes that execute multi-party Replicated Secret Sharing (RSS) protocols to drive confidential clustering. These nodes handle the computationally intensive operations of our pipeline, performing secure multi-party distance evaluations and cluster assignment calculations entirely over encrypted domains.

Anonymization Server (AS). A specialized proxy that sits between the clients and the computing backend. The AS is responsible for decoupling client network identities from their secret-shared data blocks by removing transport-layer signatures (*e.g.*, IP addresses) and performing random packet shuffling prior to backend distribution.

Training Orchestrator (TO). A centralized orchestrator responsible for global synchronization and execution logic during the subsequent personalization phase. The TO initializes cluster-specific network weights and coordinates federated training rounds. While the TO may be physically co-located with the computing infrastructure, it is logically isolated and possesses no access to the private keys or secret shares held by the CS nodes.

B. Threat Model

We adopt a semi-honest (*honest-but-curious*) adversary model where all participants (including the CSs, the AS, and the TO) are assumed to execute the protocol specifications faithfully but may passively analyze all received transcripts to infer sensitive client information. We assume the existence of a standard Public Key Infrastructure (PKI) enabling secure, end-to-end encrypted transmission channels. We assume that any active adversary has bounded computational capacity, rendering the underlying public-key encryption primitives and cryptographic signature schemes computationally unbreakable.

Under this model, the system relies on a $(2, 3)$-RSS configuration, providing formal information-theoretic security that remains robust as long as an honest majority is preserved. Specifically, we assume that at most one CS is compromised or behaves as an adversary, and that no collusion occurs between any two CSs. Because no single node holds the complete set of three shares required for reconstruction, an isolated attacker cannot recover individual client features, intermediate distances, centroids, or cluster assignments.

Furthermore, we enforce a strict non-collusion boundary between the Anonymization Server (AS) and the CSs. The AS is trusted to strip network metadata and execute oblivious shuffling, but it lacks the decryption keys needed to read the payload shares, ensuring it operates solely as a blind transport layer. Conversely, the CS nodes can decrypt their respective shares but remain completely oblivious to the original spatial alignment or client identity tracking matrices removed by the AS. Finally, our threat model focuses on confidentiality against passive adversaries; we do not consider active malicious deviations (*e.g.*, data poisoning) or side-channel exploits.

C. Security Objectives

To guarantee a robust defensive posture, our framework satisfies three primary security objectives:

Data Confidentiality (Input Privacy). No individual client feature vector, intermediate result, or cluster centroid is ever exposed in plaintext to any single computing entity. The framework guarantees that an honest-but-curious server cannot reconstruct or infer the sensitive attributes of any participant's input, maintaining privacy at rest, in transit, and during active multi-party computation.

Participant Unlinkability (Anonymity). The system breaks the cryptographic link between a client's network identity and their submitted data shares. The architecture enforces a strict separation of concerns such that an adversary monitoring the computing infrastructure cannot map specific inputs or final cluster assignments back to their original data producers, preventing profiling or targeted re-identification attacks.

Correctness and Integrity. The system guarantees that the privacy-preserving variants of non-linear operations yield mathematically accurate cluster assignments. The security bounds ensure that the introduction of randomized masking or secret sharing does not alter the convergence properties of the clustering algorithm, maintaining a negligible, cryptographically bounded probability of privacy leakage.

IV. Confidential Clustering with CCFL

Our protocol, CCFL (Confidential Clustered FL), is implemented using the MP-SPDZ framework [28], which provides tools for compiling high-level secure arithmetic into optimized

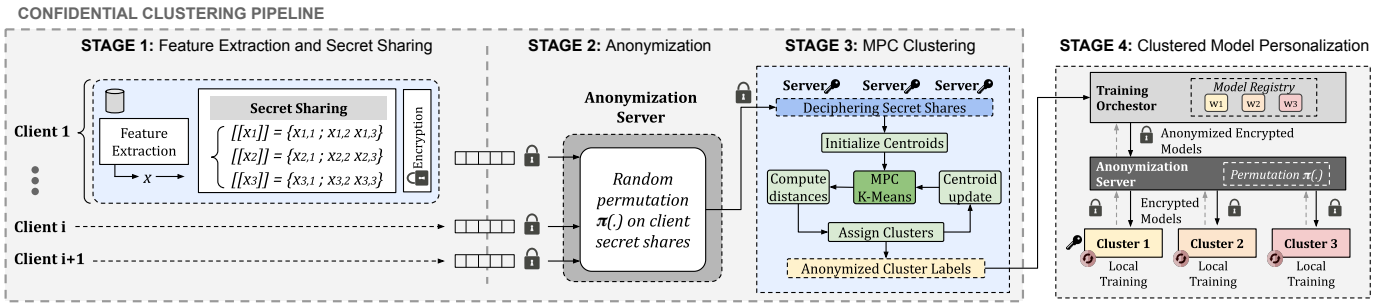


Fig. 1: CCFL confidential clustering pipeline for Federated Learning

MPC routines. We leverage these primitives to execute $(2, 3)$-RSS over a modulus ring $\mathbb{Z}_{2^\kappa}$ for highly efficient multi-party operations [29]. By relying on MP-SPDZ, we ensure that our implementation of confidential clustering primitives, including vectorized multiplication and bit-decomposition operations, follows optimized communication patterns, significantly reducing the overhead of secret-shared arithmetic.

A. Protocol Overview

The CCFL protocol, as mapped in Figure 1, operates in four distinct stages:

Stage 1: Local Extraction and Sharing. Each participant extracts representative features from its local dataset and splits them into three distinct shares using $(2, 3)$-RSS primitives. These shares are encrypted using the public keys of the respective Computing Servers, ensuring that intermediate transport proxies cannot inspect the underlying shares.

Stage 2: Anonymized Routing. The AS strips network headers from the incoming shares. It executes an oblivious shuffling routine over the arrival queue, systematically breaking the correlation between a client's physical socket connection and their data block before routing the shuffled payloads to the backend computing cluster.

Stage 3: Confidential Clustering. The CS nodes decrypt their assigned set of shares using their respective private keys. Each individual share remains mathematically blinded to the CS nodes thanks to the $(2, 3)$-RSS scheme. The CSs engage in the RSS protocol to execute distance computations and centroid updates until convergence entirely on the secret-shared data. The CS nodes then forward the final, anonymized cluster assignments to the Training Orchestrator.

Stage 4: Personalized Training. The TO initializes cluster-specific models and manages subsequent decentralized training rounds. To maintain permanent metadata anonymity, all downstream training traffic between the TO and individual clients is routed back through the AS proxy layer. While our core contribution targets confidential clustering, our architecture is aggregation-agnostic and can be seamlessly integrated with existing techniques such as Secure Aggregation or Differential Privacy (DP).

B. Local feature extraction

Each client i performs local feature extraction on its dataset $\mathcal{D}_i$ to generate an embedding vector x_i. In this work, we rely on pre-trained frozen local encoder models $f_\phi(\cdot)$ to generate compact, task-specific embedding vectors $x_i = f_\phi(\mathcal{D}_i)$. This encoder could represent various architectures, such as convolutional neural networks for image-based style extraction or RNNs/Transformers for temporal data (*e.g.*, speech recordings). By performing this mapping locally, we reduce the dimensionality of the data while preserving the latent information necessary for effective clustering [8]. Unlike prior works that transmit raw client features, thereby exposing the federation to reconstruction and membership inference attacks, our framework treats these features as sensitive private inputs that must never appear in plaintext beyond the client's device.

C. Anonymized Features secret sharing

To secure feature vectors against honest-but-curious adversaries, we employ $(2, 3)$-RSS. Let $x_i = (x_{i,1}, \ldots, x_{i,d})$ be the feature vector of client i. For each coordinate j, the client generates three additive shares $[\![x_{i,j}]\!] = (x^1_{i,j}, x^2_{i,j}, x^3_{i,j})$ such that $x_{i,j} \equiv (x^1_{i,j} + x^2_{i,j} + x^3_{i,j}) \pmod{2^k}$. Each computing party P_k receives a tuple of two shares, ensuring information-theoretic security.

Unlinkability via Oblivious Shuffling. To decouple feature vectors from client identities, CCFL embeds an intermediate anonymization phase into its communication routing matrix. Before the CSs receive their target shares, the following three steps are executed:

1) Encapsulation: Each client i encrypts each share tuple $[\![\mathbf{x}_i]\!]_k$ using the public key of the corresponding computing party P_k. This ensures that only the intended party can eventually access the underlying shares.
2) Consistent Permutation: These encrypted shares are transmitted to an intermediate *Anonymization Server* (AS), which applies a random permutation π to the batch of m client inputs. The AS applies the *identical* permutation π across all three sets of shares to preserve the algebraic validity of the RSS scheme.
3) Distribution: The AS forwards the shuffled, encrypted shares to the computing parties. By separating the knowledge of the permutation (held by the AS) from the ability to decrypt and compute (held by P_k), the system ensures that no single entity can map a resulting cluster centroid back to a specific client identity.

D. Confidential Mini-batch k-Means Clustering

Once the CSs hold the permuted secret-shared matrices $[\![X]\!]$, they execute the core CCFL clustering protocol. To optimize for MPC communication overhead, we employ a mini-batch k-Means approach. Instead of computing distances over the entire secret-shared matrices $\mathcal{X}$ in every iteration, the parties sample a mini-batch $\mathcal{B} \subset \mathcal{X}$ of size $b < m$. This reduces the communication complexity of the centroid update from $O(m \cdot K)$ to $O(b \cdot K)$ per iteration.

Distance Computation. The computing parties evaluate the private distance matrix $[\![D]\!] \in \mathbb{Z}_{2^\kappa}^{b \times K}$ for the sampled mini-batch, where each entry $[\![d_{ik}]\!]$ represents the chosen metric (*e.g.*, our optimized squared Euclidean distance) calculated between the secret-shared mini-batch vector $[\![x_i]\!]$ and the candidate cluster centroid $[\![c_k]\!]$.

Cluster Assignment. Mini-batch feature vectors are routed to their nearest target cluster by executing a vectorized secure argmin operation directly over the rows of the distance matrix $[\![D]\!]$. This yields a secret-shared membership mapping matrix $[\![A]\!] \in \{0,1\}^{b \times K}$, enabling SIMD-like parallel execution across the computing parties.

Centroid Update. Revised cluster centroids are computed as the weighted average of the embeddings within the mini-batch. We execute this through secure matrix-vector multiplication $[\![S]\!] \leftarrow [\![A]\!]^T \times [\![X_\mathcal{B}]\!]$, followed by element-wise secure division by the aggregated cluster population counts $[\![n]\!]$. Empty clusters are neutralized through secure conditional masks to prevent division-by-zero failures. To ensure resilience against timing side-channel attacks and to maintain a deterministic communication profile, the update logic is implemented in MP-SPDZ using a branch-free IF_ELSE primitive.

E. Centroid Initialization with Random Restarts

Cleartext k-means implementations often rely on the k-Means++ heuristic. However, k-Means++ requires sequential, dependent selections of points, which translates to a high number of expensive communication rounds in a MPC environment. To maximize efficiency while maintaining high-quality clustering, CCFL instead relies on a random restarts strategy. The protocol proceeds as follows:

1) **Random Selection:** The computing parties select K initial centroids $[\![c_1, \ldots, c_K]\!]$ by uniformly sampling K distinct rows from the secret-shared matrix $[\![X]\!]$ using a shared random seed. This operation is performed in a single parallelizable step.
2) **Iterative Refinement:** The parties execute the Mini-batch k-Means loop described in Section IV-D to converge on local centroids.
3) **Restart Mechanism:** To mitigate convergence to poor local minima, steps 1–2 are executed across multiple independent seeds. The parties evaluate the final secret-shared intra-cluster cost $[\![\mathcal{L}]\!]$ for each restart and extract the optimal centroid configuration using a single secure argmin operation.

By replacing k-Means++ with a non-interactive parallel selection, this strategy drastically reduces initialization time over high-latency networks while the restart mechanism maintains robust global convergence.

F. Secure k-Selection via Geometric Elbow Maximization

A fundamental challenge in clustered FL is determining the optimal number of operational clusters k without a priori access to centralized data profiles. Inspired by the curvature-maximization principles of the Kneedle algorithm [30], which evaluates deviation from a baseline secant, we compute a confidential instantiation of the Elbow Method. For a range $K \in [1, K_{\max}]$, the parties execute the confidential k-Means routine to convergence and compute the resulting secret-shared within-cluster sum of squares (WCSS), denoted by $[\![\mathcal{L}_K]\!]$:

$$[\![\mathcal{L}_K]\!] = \sum_{i=1}^{m} \sum_{\kappa=1}^{K} [\![a_{i,\kappa}]\!] \cdot [\![d(x_i, c_\kappa)]\!] \tag{2}$$

where $[\![d(\cdot)]\!]$ represents the squared euclidean distance between two points.

Rather than utilizing normalized vertical differences [30], we geometrically define the elbow point k^* as the configuration that maximizes the exact perpendicular distance to the secant line connecting the endpoints $(1, \mathcal{L}_1)$ and $(K_{\max}, \mathcal{L}_{K_{\max}})$. Let $\Delta_k = k - 1$ denote the cluster step offset and $\Delta_{\mathcal{L}_k} = \mathcal{L}_1 - \mathcal{L}_k$ denote the cumulative loss reduction relative to $k = 1$. The perpendicular distance D_k from an intermediate point $(k, \mathcal{L}_k)$ to this secant line is defined as:

$$D_k = \frac{\left| \Delta_{K_{\max}} \cdot \Delta_{\mathcal{L}_k} - \Delta_k \cdot \Delta_{\mathcal{L}_{K_{\max}}} \right|}{\sqrt{\Delta_{K_{\max}}^2 + \Delta_{\mathcal{L}_{K_{\max}}}^2}} \tag{3}$$

We observe that the denominator in D_k is a positive constant across all candidate $k \in [1, K_{\max}]$. Consequently, maximizing D_k is strictly equivalent to maximizing its numerator. Under secret sharing, let $[\![\Delta_{\mathcal{L}_k}]\!] = [\![\mathcal{L}_1]\!] - [\![\mathcal{L}_k]\!]$. To avoid the prohibitive communication overhead associated with secure multi-party division and square root operations, we simplify the optimization objective to a scale-invariant, division-free distance metric $[\![D_k]\!]$ over secret-shared fixed-point values:

$$[\![\mathcal{D}_k]\!] = \left| \Delta_{K_{\max}} \cdot [\![\Delta_{\mathcal{L}_k}]\!] - \Delta_k \cdot [\![\Delta_{\mathcal{L}_{K_{\max}}}]\!] \right| \tag{4}$$

To determine the final inflection point k^* under strict privacy guarantees, the parties execute an oblivious maximum search protocol directly over the secret-shared distance array $[\![\mathcal{D}]\!]$:

$$k^* = \underset{k \in [1, K_{\max}]}{\mathrm{argmax}} [\![\mathcal{D}_k]\!] \tag{5}$$

By evaluating the distance metric using native ring operations (additions, scalar multiplications, and secret-shared absolute values) and executing the argmax within an oblivious circuit, only the final integer index k^* is revealed. The intermediate loss trajectory $[\![\mathcal{L}_k]\!]$ and distance profiles remain entirely secret, eliminating side-channel leakage regarding global data variance or geometric cluster separability.

TABLE I: Fixed-Point Ring Arithmetic Configurations for L_2^2

Ring (κ)	**Bits** (k)	**Frac** (f)	**Security** (σ)	d_{max}	N_{max}	**Limitation**
64-bit	8	6	**40**	$2^8 (= 256)$	$2^{16} (\sim 6.5 \times 10^4)$	Insufficient Precision
64-bit	16	8	25	$2^7 (= 128)$	$2^{23} (\sim 8.3 \times 10^6)$	Compromised Security
128-bit	32	16	**40**	$2^{24} (\sim 1.6 \times 10^7)$	$2^{56} (\sim 7.2 \times 10^{16})$	**None (Target)**

V. Metric Selection and Numerical Constraints over $\mathbb{Z}_{2^\kappa}$

Transitioning k-means to an MPC environment necessitates a trade-off between numerical precision, dimensionality d, and cryptographic integrity. To maintain a statistical security baseline of $\sigma = 40$ while normalizing features to $[-1, 1]$, we conduct a design space analysis of fixed-point primitives, summarized in Table I. This analysis demonstrates that standard 64-bit rings fail to simultaneously satisfy high-dimensional requirements and robust security, requiring the use of 128-bit rings for production-scale FL.

A. Protocol Correctness and Statistical Security Margins

To ensure protocol correctness and prevent information leakage, MPC fixed-point operations rely on probabilistic truncation. The ring size κ must provide a sufficient noise buffer σ to accommodate the cumulative bit-width of the operations while ensuring that the truncation does not leak sensitive data. The operational feasibility is governed by two phase constraints, where k is the total bit-width: (1) the *Routing Constraint* (distance computation), where squaring coordinates and summing across d dimensions yields $(2k+\lceil \log_2(d) \rceil)+\sigma \leq \kappa$, and (2) the *Aggregation Constraint* (centroid update), where summing N cluster points yields $(k+\lceil \log_2(N) \rceil)+\sigma \leq \kappa$. The absolute system constraint is defined by the most restrictive phase:

$$\max(2k + \lceil \log_2(d) \rceil, k + \lceil \log_2(N) \rceil) + \sigma \leq \kappa \qquad (6)$$

Because the multiplication step in the routing phase imposes a dominant $2k$ penalty, the routing constraint typically represents the primary bottleneck for high dimensional embeddings.

B. Cryptographic Circuit Optimization via Partial SED

The standard Euclidean distance between an embedding x and a candidate cluster centroid c is defined as $d_{L_2}(x, c) = \sqrt{\sum_{i=1}^{d} (x_i - c_i)^2}$. In an MPC setting, evaluating non-linear operations such as the square root requires deep, highly interactive circuits that are prohibitively expensive over high-latency networks. To mitigate this, CCFL adopts the Squared Euclidean Distance (SED), which eliminates the square root primitive entirely.

To further reduce the circuit complexity, we expand the SED as $\sum_{i=1}^{d} x_i^2 + \sum_{i=1}^{d} c_i^2 - 2\sum_{i=1}^{d} x_i c_i$ and discard the input magnitude term $\sum_{i=1}^{d} x_i^2$. While this term is identical for all centroids and thus does not affect the $\arg\min_c$ operation, its removal provides a significant system-level advantage in the encrypted domain. We define the *partial squared Euclidean distance* as:

$$d'_{L_2^2}(x, c) = \sum_{i=1}^{d} c_i^2 - 2\sum_{i=1}^{d} x_i c_i. \qquad (7)$$

In a fixed-point MPC setting over $\mathbb{Z}_{2^\kappa}$, computing $\sum_{i=1}^{d} x_i^2$ requires d interactive multiplications of secret-shared values per client. Because fixed-point multiplication doubles the fractional bit-width, each product must be followed by an interactive truncation protocol across the computing servers to prevent ring overflow. Discarding this term eliminates d interactive multiplications and their associated truncation protocols for every client per iteration.

Furthermore, the term $\sum_{i=1}^{d} c_i^2$ is client-invariant. The computing servers can pre-compute this value once per iteration for each cluster and reuse the resulting secret shares for all m clients. Consequently, the only client-dependent multiplications required during the routing phase are the cross-products $x_i c_i$, which amount to d multiplications per client–cluster pair. The factor 2 in the cross-term is a local shift and incurs no network cost. Overall, these optimizations significantly reduce the online communication bottleneck by minimizing the number of interactive rounds required for distance computation.

Despite these network-level savings, evaluating the cross-products remains strictly bound by the *Routing Constraint* defined in Equation 6. Because the partial SED still requires accumulating d double-width ($2k$) products prior to truncation, it dictates the system's maximum scalable dimensionality. Under this mathematical bound, standard 64-bit rings ($\kappa = 64$) lack the capacity to securely support practical fractional precision for high-dimensional embeddings, necessitating deployment of our system over 128-bit rings ($\kappa = 128$).

VI. Privacy & Security Analysis

We evaluate the privacy guarantees of CCFL under the honest-but-curious adversary model using the simulation-based paradigm [31].

A. Formal Guarantees

Theorem 1 (Input Information-Theoretic Privacy)**.** *Under a $(2, 3)$-RSS configuration, the view of any single Computing Server (CS_k) reveals zero mutual information regarding the plaintext feature vector $\mathbf{x} \in \mathbb{Z}_{2^\kappa}^d$, achieving information-theoretic 1-privacy.*

Proof Sketch. Since shares are sampled uniformly at random from $\mathbb{Z}_{2^\kappa}^d$ subject to the sum constraint, any two shares held by CS_k are uniformly distributed over $\mathbb{Z}_{2^\kappa}^{2d}$, yielding zero statistical advantage over any distinct vector $\mathbf{x}'$. □

TABLE II: Training Tasks Configurations and Hyperparameters.

Scenario	Encoder e	Latent Dim d_e	Trained Model (p)	Classes	Clients	Cli/Rnd	Rounds	Partitioning scheme
Rotated MNIST [32]	VAE	360	LeNet-5 (85K)	10	100	50	50	Dir($\alpha = 1.0$)
PACS [33]	VAE	70	ResNet-18 (11.2M)	7	40	40	50	Dir($\alpha = 10.0$)
OfficeHome [34]	VAE	150	ResNet-18 (11.2M)	15	40	40	50	Dir($\alpha = 1.0$)
SpeechCommands [35]	HuBERT	500	DS-CNN (39K)	10	400	100	250	Natural IDs

Theorem 2 (Participant Unlinkability)**.** *Assuming non-collusion between the Anonymization Server (AS) and the Computing Server cluster (CS), CCFL achieves participant unlinkability against polynomial-time adversaries.*

Proof Sketch. The AS receives PKE-encrypted ciphertexts (IND-CPA secure) and applies a random permutation π to the batch of m inputs. Because the permutation π is independent of the content of the shares and the CS nodes possess no knowledge of the permutation mapping, the mapping between a client's network identity and their secret-shared data remains uniform over $M!$ permutations. □

B. Trust Boundary & Collusion Analysis

To evaluate system robustness under boundary breaches, we analyze degraded trust scenarios:

1) **Curious AS.** If the AS acts as an adversary without colluding with any single CS, the threat degrades to standard traffic analysis. The AS observes participant online timing but learns zero information regarding feature coordinates, distance metrics, or final cluster assignments, as these are protected by PKE-encryption during transit.
2) **Collusion Between AS and One CS.** If the AS colludes with CS_1, the coalition can map specific IP addresses id_i to their secret shares. However, because CS_1 holds only two of the three RSS shares, input privacy remains intact. The colluding adversary learns *membership mapping* (*i.e.*, which client is assigned to which cluster), but the raw feature vectors $\mathbf{x}_i$ and global centroids remain completely hidden.

VII. Experimental Evaluation

This section presents a fine-grained evaluation of CCFL computational overhead as well as its downstream utility compared to state-of-the-art FL competitors. To account for experimental variability, we execute each experiment across 3 independent random seeds, and report the mean test accuracy.

A. Experimental Setup

System Benchmarking. To isolate the intrinsic costs of our cryptographic primitives, we deploy several micro-benchmarks using synthetic random data corresponding to the different training tasks later explored in the empirical evaluation, as well as additional hypothetical configurations to better study system scalability. Our system is deployed on the Grid'5000 [26] distributed testbed, relying on three isolated physical compute instances to host the non-colluding $(2, 3)$-RSS computing servers. We use the EnOSlib library [36] to inject network profiles (LAN, MAN, and WAN) with varying round-trip times (RTT) and bandwidth throttles, as detailed in Table III.

TABLE III: Network Emulation Profiles

Topology	Egress Delay	Link Rate	Packet Loss	RTT
LAN	0.5 ms	10 Gbps	0.0%	$<$ 1 ms
MAN	7.0 ms	1 Gbps	0.01%	$\sim$ 14 ms
WAN	35.0 ms	100 Mbps	0.1%	$\sim$ 70 ms

Training tasks. We evaluate our framework on four distinct tasks: three image classification tasks and one keyword spotting task. The image tasks include Rotated MNIST [32], where clients are partitioned into four clusters based on image rotation; PACS [33] and OfficeHome [34], which distribute clients across four visually distinct domains. The keyword spotting task, SpeechCommands [35], partitions clients by unique speaker identities to model organic acoustic heterogeneity. Detailed hyperparameter configurations for all scenarios are provided in Table II.

Baselines. We benchmark CCFL against several baselines, including global aggregation (FedAvg [37], FedProx [38]); local personalization (Ditto [39]); dynamic clustering (FedMix [9], IFCA [5]) and one-shot feature clustering (FC [8]). Both FedMix and IFCA introduce a severe $\mathcal{O}(K)$ communication and computational overhead per round, as participating edge devices must either maintain or download all K global model candidates to evaluate local losses or compute gradients. To reflect the practical memory and bandwidth constraints of resource-constrained edge devices, we cap FedMix and IFCA at $K = 3$ and scale FedMix local training steps to $\frac{s}{K}$ to ensure a balanced total computation budget across all compared methods. Local feature extraction for FC and CCFL use frozen models from Hugging Face as detailed in Table II.

B. MPC performance & Complexity Analysis

We first isolate the execution profiles of our framework under different network environments and algebraic configurations. We deploy our system for a fixed clustering configuration ($n_{clients} = 100$, $dim = 500$, $k = 4$) and compare distance metrics costs under varying emulated network configurations. We report the latency of MPC operations for one complete iteration of confidential K-Means in Figure 2.

Algebraic domains: Ring vs Field. Across all primitives and topologies, implementations over modulus ring domains $\mathbb{Z}_{2^\kappa}$ using $(2, 3)$-Replicated Secret Sharing (RSS) consistently outperform prime field domains $\mathbb{Z}_p$ using Shamir Secret Sharing. Under LAN conditions, computing standard euclidean

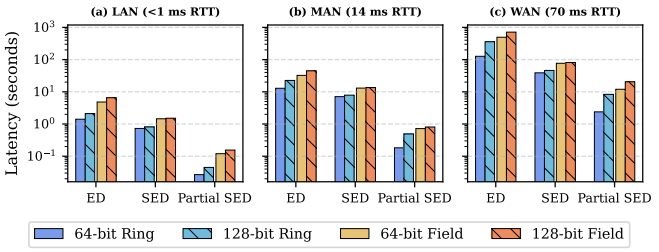


Fig. 2: Latency of MPC operations on distributed Grid'5000 nodes with emulated network constraints.

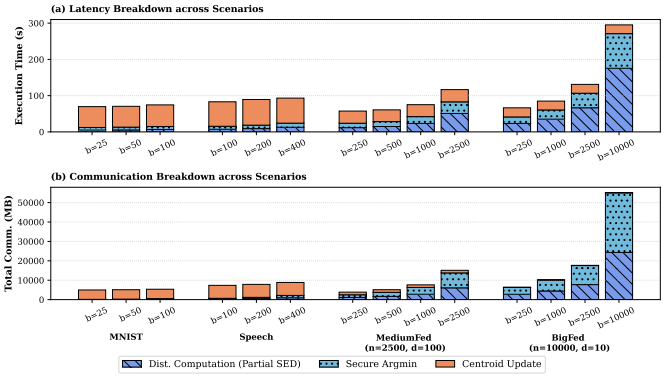


Fig. 3: Impact of batch size on performance of the confidential K-Means routine with n_{iter}=10 and K_{max}=5 (LAN).

distance (ED) over a 128-bit ring requires only 2.1 s compared to 6.5 s in a 128-bit field (3.1× speedup). While transitioning from LAN (1 ms RTT) to cross-border WAN (70 ms RTT) induces a ∼171× network latency penalty on the 128-bit ring ED computation (swelling to 359.1 s), we still observe a 1.98× speedup over the field alternative (712.8 s).

Distance Metrics Performance. The empirical results reveal a clear performance hierarchy governed by mathematical complexity. ED is the most expensive metric to compute due to non-linear square root operations requiring interactive secure comparisons. Transitioning to Squared Euclidean Distance (SED) optimizes latency by removing the square root operator, reducing 128-bit ring LAN execution to 0.8 s (2.6× speed up). The proposed *partial SED* optimization, which discards the centroid-invariant input magnitude component, achieves the best performance. In a 64-bit ring under LAN conditions, the partial SED executes in a negligible 0.027 s. When expanded to a 128-bit ring for high-dimensional data configurations, it executes in 0.045 s over LAN and scales to a manageable 8.3 s under severe WAN constraints. This achieves a substantial 46.7× speedup over LAN and a 43.3× speedup over WAN over unoptimized ED, demonstrating that our algebraic decomposition effectively removes the primary communication bottleneck of confidential distance computation.

Mini-batch size impact. We present additional results concerning batch size impact for confidential K-Means in Figure 3. First experiments on the MNIST and Speech-Commands scenarios demonstrated minor improvements when relying on mini-batch K-Means as opposed to full batch K-Means. While mini-batching significantly reduces distance computation time, reducing latency by up to 46% on Speech-Commands, it resulted in minor improvement as the main computation and communication burden is dominated by centroid update operations for those scenarios. On the contrary, the MediumFed and BigFed scenarios presented very different outcomes. When scaling to massive federation of devices ($n = 2500$ and $n = 10000$ respectively), we observe major gain provided by the mini-batch approach, especially when the dimension of data points $d \ll n$ as observed for the BigFed scenario (achieving 2.24× speedup for a batch size $b = 2500$, and up to 4.42× speedup for a batch size $b = 250$ while reducing total communication by up to 8.6×).

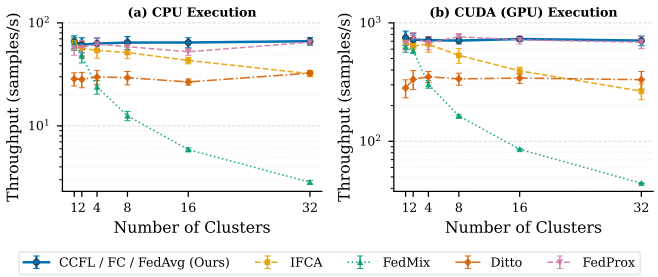


Fig. 4: Client training throughput for ResNet-18 on PACS data.

> **Summary:** We show that by combining efficient scheme such as $(2, 3)$-RSS with distance metric optimization, confidential clustering can run seamlessly in real-world setting using MPC. The proposed partial SED metric improves latency by up to 46.7× compared to conventional ED, while relying on mini batches to run K-Means turns out to be an additional significant optimization as the system scales to massive federation of devices.

C. Client-Side Overhead & Scalability

Theoretical cost. We provide theoretical per-client clustering and training overhead for CCFL and competing baselines in Table IV. CCFL achieves complexity decoupling by performing feature extraction as a one-shot operation. While the encoder parameters p_e may be substantial, the cost $(O)(p_e \cdot i)$ is amortized over the entire training lifecycle, occurring only once before the training rounds begin. In contrast, dynamic clustering methods impose a recurring overhead in every round. Additionally, by fixing a low-dimensional embedding space, CCFL ensures that the communication cost for cluster assignment $(3 \cdot d_e)$ remains significantly lower than the cost of transmitting full model updates (p).

Empirical training throughput. We measure and compare client training throughput using both CPUs (i9-13950HX) and GPUs (RTX 3500 Mobile Ada Generation) from a commodity laptop computer to emulate realistic client hardware, as depicted in Figure 4. Although GPU execution running one order of magnitude faster than the CPU execution, we observe

TABLE IV: Theoretical Per-Client Overhead Comparison.

Strategy	One-shot Clustering Phase			Training Phase		
	Computation	Memory	Communication	Computation	Memory	Communication
FEDAVG [37]	N/A	N/A	N/A	$\mathcal{O}(R \cdot p \cdot t)$	$\mathcal{O}(p)$	$\mathcal{O}(2 \cdot R \cdot p)$
FEDPROX [38]	N/A	N/A	N/A	$\mathcal{O}(R \cdot p \cdot t)$	$\mathcal{O}(p)$	$\mathcal{O}(2 \cdot R \cdot p)$
DITTO [39]	N/A	N/A	N/A	$\mathcal{O}(2 \cdot R \cdot p \cdot t)$	$\mathcal{O}(2 \cdot p)$	$\mathcal{O}(2 \cdot R \cdot p)$
IFCA [5]	N/A	N/A	N/A	$\mathcal{O}(R \cdot (K \cdot p \cdot i + p \cdot t))$	$\mathcal{O}(K \cdot p)$	$\mathcal{O}(R \cdot (K+1) \cdot p)$
FEDMIX [9]	N/A	N/A	N/A	$\mathcal{O}(R \cdot K \cdot p \cdot t)$	$\mathcal{O}(K \cdot p)$	$\mathcal{O}(2 \cdot R \cdot K \cdot p)$
FC [8]	$\mathcal{O}(p_e \cdot i)$	$\mathcal{O}(p_e)$	$\mathcal{O}(d_e)$	$\mathcal{O}(R \cdot p \cdot t)$	$\mathcal{O}(p)$	$\mathcal{O}(2 \cdot R \cdot p)$
CCFL (Proposed)	$\mathcal{O}(p_e \cdot i)$	$\mathcal{O}(p_e)$	$\mathcal{O}(3 \cdot d_e)$	$\mathcal{O}(R \cdot p \cdot t)$	$\mathcal{O}(p)$	$\mathcal{O}(2 \cdot R \cdot p)$

Notation: R: number of rounds; p: number of model parameters; t: training FLOPs; K: number of clusters; i: inference FLOPs; d_e: embedding dimension; p_e: number of encoder parameters.

similar throughput evolution for every strategy. CCFL and FC which decouple clustering from actual model training, achieve constant throughput similar to FEDAVG. In comparison IFCA's and FEDMIX's throughput degrades as the number of cluster increases. IFCA's throughput slows down by up to 61% on GPU and 49% on CPU when scaling from a single cluster to 32 clusters, while FEDMIX's drastically drops by 93% on GPU and 95% on CPU. DITTO's dual model training strategy for local personalization doubles computation overhead for FL clients, achieving 329 samples/second on GPU and 29 samples/second on CPU which is more than 50% throughput degradation compared to CCFL, FC, and FEDAVG for training ResNet-18 on PACS images.

> **Summary:** CCFL achieves complexity decoupling by moving clustering logic to a dedicated computing cluster during a one-shot pre-processing phase. This eliminates the $\mathcal{O}(K)$ per-round overhead inherent in online dynamic clustering methods (*e.g.*, IFCA, FEDMIX). This ensures that the system remains scalable to high number of clusters without imposing a multi-model resource penalty on resource-constrained edge devices, and makes CCFL uniquely suitable for scenarios where computation and network are primary bottlenecks.

D. Utility & Convergence Analysis

We evaluate the classification accuracy of personalized client models built upon our MPC clusters, comparing convergence against personalized cleartext baselines and privacy-preserving DP baselines. The resulting accuracy achieved by each strategy is compiled in Table V.

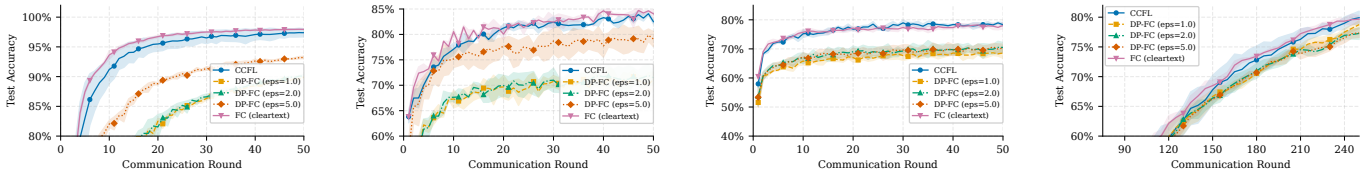
Absolute model utility results. Our confidential feature clustering approach effectively mitigates the gradient interference that degrades standard global models in non-IID settings. On the Rotated MNIST task, CCFL achieves top accuracy of 97.38% which is almost matching its cleartext counterpart (FC), while outperforming all other methods. These results translate to the OfficeHome and SpeechCommands scenarios where CCFL achieve the best performance (78.32% and 79.91% respectively). The PACS scenario presents slightly different results, where FC still achieves top performance (84.05%), while CCFL undergoes a slight accuracy drop (82.50%), although achieving the second best performance (which is likely attributable to arithmetic precision nuances inherent in MPC-based clustering). IFCA remains a strong competitor achieving the second best performance across scenarios as a cleartext baseline, while FEDMIX struggles to bring any good results under limited computation budget. DITTO dual-model local personalization approach does not work very well under strong covariate-shift as observed across Rotated MNIST, PACS, and OfficeHome. On the other hand, the poor performance on SpeechCommands could be a consequence of local model staleness, due to partial client participation in consecutive FL rounds.

Comparisons against confidential baselines. We present model convergence curves for privacy-preserving strategies in Figure 5. Overall, CCFL achieves convergence profiles nearly identical to the cleartext FC baseline. In contrast, the stochastic noise required for ϵ-DP acts as a perturbation on the high-dimensional feature space, blurring the geometric boundaries between clusters. This cluster drifting leads to significant utility degradation: at $\epsilon = 1.0$, accuracy drops by 7.5%, 11.3%, and 9.7% for MNIST, PACS, and OfficeHome, respectively. While relaxing the privacy budget to $\epsilon = 5.0$ improves utility, it still struggles with 8% accuracy degradation on the worst scenario (OfficeHome). These results demonstrate that CCFL provides a superior utility-privacy trade-off by leveraging confidential computation to preserve geometric integrity, whereas DP relies on noise-based obfuscation that inherently compromises clustering accuracy.

> **Summary:** CCFL effectively bridges the gap between high-utility FL and confidential computation. Our evaluation shows that CCFL maintains near-cleartext accuracy across diverse non-IID tasks while providing a scalable architecture that avoids the multi-model overhead of dynamic clustered FL approaches. By outperforming DP baselines in utility, CCFL proves to be a viable framework for privacy-preserving, large-scale distributed learning.

VIII. RELATED WORK

Our framework sits at the intersection of Clustered Federated Learning (CFL) and Confidential Computing pipelines. We contextualize our design choices by reviewing the struc-



(a) Rotated MNIST (b) PACS (c) OfficeHome (d) Speech

Fig. 5: CCFL model convergence compared to DP baselines.

TABLE V: Test accuracies across benchmarks

	Non-confidential baselines						Confidential baselines			Ours
Scenario	FedAvg	FedProx	Ditto	IFCA	FedMix	FC	DP-FC (ϵ=1.0)	DP-FC (ϵ=2.0)	DP-FC (ϵ=5.0)	CCFL
Rotated MNIST [32]	91.38	91.50	91.95	96.11	92.56	**97.94**	89.90	90.11	93.22	**97.38**
PACS [33]	77.39	77.45	74.66	81.12	67.23	**84.05**	71.19	71.48	79.09	**82.50**
OfficeHome [34]	76.60	76.21	64.00	73.48	37.06	**78.07**	68.57	70.50	70.35	**78.32**
Speech [35]	72.52	72.27	47.88	78.86	66.10	**79.66**	78.75	77.27	77.35	**79.91**

tural limitations of contemporary plaintext and privacy-preserving clustering strategies for FL.

Plaintext Clustered FL. CFL methodologies generally fall into two categories. The first uses *gradient similarity* to iteratively partition clients via bi-partitioning or greedy agglomerative heuristics [3, 4, 40]. While numerically robust, evaluating weight trajectories across large client populations incurs high computational latency and delays convergence. More fundamentally, they are incompatible with standard secure aggregation: computing pairwise similarities requires access to individual client updates, whereas secure aggregation reveals only their sum to the server. The second category executes dynamic cluster routing via local Empirical Risk Minimization (ERM) [5, 6, 9], where the server distributes K global models to every client in each training round. This imposes a prohibitive transmission penalty of $\mathcal{O}(K)$ that scales poorly with model depth or cluster cardinality. To bypass these iterative costs, recent works rely on local feature extraction [7, 8, 10], using image style extraction or deploying local encoders to generate compact embedding vectors for one-shot offline clustering. CCFL inherits this one-shot efficiency but eliminates the central privacy vulnerabilities native to plaintext embedding accumulation.

Privacy-Preserving Clustering. Translating canonical algorithms into secure distributed primitives is a deeply studied domain [12–18, 21]. For instance, k-FED [13] splits the k-means optimization by synchronizing local client centroids through a central coordinator, which risks leaking regional data distributions via inversion attacks. To achieve absolute confidentiality, alternative frameworks deploy MPC to evaluate distances with information-theoretic security [14–16], or use secure two-party computation (2PC) for private density-based clustering [17]. While faster than Homomorphic Encryption alternatives [18], their absolute overhead remains highly restrictive in high-dimensional spaces. Our system bridges this gap by performing secure fixed-point optimizations over modulus rings ($\mathbb{Z}_{2^\kappa}$), leveraging highly responsive $(2,3)$-RSS configurations to make multi-party distance computation practical over dimensionally-constrained feature vectors.

Confidential Clustered FL. Unlike private data-point partitioning, confidential CFL aims to cluster entire client distributions without exposing local gradients or feature geometries. Some approaches use perturbation techniques, such as randomized quantization and Differential Privacy (DP), to mask feature matrices [8, 19, 20], or employ oblivious shuffling wrappers to decouple user identities from updates [41]. However, perturbation methods still leak noisy centroids and rough feature shapes to the server. Recent frameworks like EBS-CFL [42] propose cryptographic variants of ERM clustering (IFCA), but wrapping multi-model verification loops in secure primitives heavily amplifies edge communication. Conversely, systems like FED-EHP [43] execute confidential model aggregation during training but still leak plaintext features during the initial assignment loop. To date, only [44] deploys an MPC pipeline for confidential clustering within FL, but it is strictly limited to bi-clustering for malicious client detection. In contrast, CCFL is the first unified, confidential clustered FL framework for multi-cluster personalized model assignments, guaranteeing input privacy, participant unlinkability, and numerical integrity without the ensemble-scale communication penalties of dynamic clustering.

IX. Conclusion

In this work, we presented CCFL, a practical framework for Confidential Clustered FL that addresses the challenge of statistical heterogeneity without the prohibitive overhead of traditional dynamic clustering methods. By decoupling the clustering logic from the per-round training loop, CCFL bypasses the $\mathcal{O}(K)$ communication bottleneck of dynamic approaches, making it uniquely scalable for large-scale federations. While components such as Replicated Secret Sharing and K-Means clustering are well-established, their integration into a low-latency, privacy-preserving FL pipeline remains a significant engineering challenge. CCFL solves this by

synthesizing these building blocks with targeted optimizations, specifically, a partial squared euclidean distance metric to minimize circuit complexity and a one-shot clustering phase to decouple clustering from the training loop. By orchestrating these elements into a unified pipeline, we provide a system that is not only theoretically secure but also practically viable, achieving up to $46.7\times$ speedups in high-latency environments while maintaining state-of-the-art privacy-utility trade-off across diverse non-IID benchmarks.

Acknowledgment

This work was supported by a French government grant managed by the Agence Nationale de la Recherche under the France 2030 program, reference "ANR-23-PECL-0007" (PEPR CLOUD - STEEL project). Experiments presented in this paper were carried out using Grid5000, one of the precursor infrastructures of the SLICES-FR testbed, operated by a scientific interest group including 14 major French academic institutes and Universities (see https://slices-fr.eu/).

AI Disclosure Statement

Google's Gemini was used to assist in writing (proofreading, improving readability) and code optimization. All output was reviewed by the authors, and generated scripts were thoroughly tested, validated, and integrated. The authors retain full responsibility for the content of this manuscript.

References

[1] H. Zhu, J. Xu, S. Liu, and Y. Jin, "Federated learning on non-iid data: A survey," *Neurocomputing*, vol. 465, pp. 371–390, 2021, ISSN: 0925-2312. DOI: https://doi.org/10.1016/j.neucom.2021.07.098

[2] C. Prigent, A. Costan, G. Antoniu, and L. Cudennec, "Enabling federated learning across the computing continuum: Systems, challenges and future directions," *Future Generation Computer Systems*, vol. 160, pp. 767–783, 2024, ISSN: 0167-739X. DOI: https://doi.org/10.1016/j.future.2024.06.043

[3] F. Sattler, K.-R. Müller, and W. Samek, "Clustered federated learning: Model-agnostic distributed multitask optimization under privacy constraints," *IEEE Transactions on Neural Networks and Learning Systems*, vol. 32, no. 8, pp. 3710–3722, 2021. DOI: 10.1109/TNNLS.2020.3015958

[4] M. Mehta and C. Shao, "A greedy agglomerative framework for clustered federated learning," *IEEE Transactions on Industrial Informatics*, vol. 19, no. 12, pp. 11 856–11 867, 2023. DOI: 10.1109/TII.2023.3252599

[5] A. Ghosh, J. Chung, D. Yin, and K. Ramchandran, "An efficient framework for clustered federated learning," *IEEE Transactions on Information Theory*, vol. 68, no. 12, pp. 8076–8091, 2022. DOI: 10.1109/TIT.2022.3192506

[6] Y. Ruan and C. Joe-Wong, "Fedsoft: Soft clustered federated learning with proximal local updating," *Proceedings of the AAAI Conference on Artificial Intelligence*, vol. 36, no. 7, pp. 8124–8131, Jun. 2022. DOI: 10.1609/aaai.v36i7.20785

[7] D. Shenaj, E. Fani, M. Toldo, D. Caldarola, A. Tavera, U. Michieli, M. Ciccone, P. Zanuttigh, and B. Caputo, " Learning Across Domains and Devices: Style-Driven Source-Free Domain Adaptation in Clustered Federated Learning," in *2023 IEEE/CVF Winter Conference on Applications of Computer Vision (WACV)*, Los Alamitos, CA, USA: IEEE Computer Society, Jan. 2023, pp. 444–454. DOI: 10.1109/WACV56688.2023.00052

[8] C. Prigent, M. Chelli, A. Costan, L. Cudennec, R. Schubotz, and G. Antoniu, "Efficient resource-constrained federated learning clustering with local data compression on the edge-to-cloud continuum," in *2024 IEEE 31st International Conference on High Performance Computing, Data, and Analytics (HiPC)*, 2024, pp. 255–265. DOI: 10.1109/HiPC62374.2024.00033

[9] M. Reisser, C. Louizos, E. Gavves, and M. Welling, "Federated mixture of experts," *CoRR*, vol. abs/2107.06724, 2021. arXiv: 2107.06724. [Online]. Available: https://arxiv.org/abs/2107.06724

[10] D. M. Jimenez-Gutierrez, M. Hassanzadeh, D. Solans, M. Elbamby, N. Kourtellis, A. Anagnostopoulos, I. Chatzigiannakis, and A. Vitaletti, "Clust-psi-pfl: A population stability index approach for clustered non-iid personalized federated learning," in *2026 IEEE International Parallel and Distributed Processing Symposium (IPDPS)*, 2026, pp. 732–745. DOI: 10.1109/IPDPS65963.2026.00067

[11] J. Zhao, S. Bagchi, S. Avestimehr, K. Chan, S. Chaterji, D. Dimitriadis, J. Li, N. Li, A. Nourian, and H. Roth, "The federation strikes back: A survey of federated learning privacy attacks, defenses, applications, and policy landscape," *ACM Comput. Surv.*, vol. 57, no. 9, Apr. 2025, ISSN: 0360-0300. DOI: 10.1145/3724113

[12] A. Hegde, H. Möllering, T. Schneider, and H. Yalame, "Sok: Efficient privacy-preserving clustering," *Proceedings on Privacy Enhancing Technologies*, 2021. DOI: 10.2478/popets-2021-0068

[13] D. K. Dennis, T. Li, and V. Smith, *Heterogeneity for the win: One-shot federated clustering*, 2021. arXiv: 2103.00697 [cs.LG]. [Online]. Available: https://arxiv.org/abs/2103.00697

[14] P. Mohassel, M. Rosulek, and N. Trieu, "Practical privacy-preserving k-means clustering," *Proceedings on Privacy Enhancing Technologies*, vol. 2020, pp. 414–433, 2020. [Online]. Available: https://api.semanticscholar.org/CorpusID:204772074

[15] M. Upmanyu, A. M. Namboodiri, K. Srinathan, and C. V. Jawahar, "Efficient privacy preserving k-means clustering," in *Intelligence and Security Informatics*, H. Chen, M. Chau, S.-h. Li, S. Urs, S. Srinivasa, and G. A. Wang, Eds., Berlin, Heidelberg: Springer Berlin Heidelberg, 2010, pp. 154–166, ISBN: 978-3-642-13601-6. DOI: 10.1007/978-3-642-13601-6_17

[16] W. Wei, C. Tang, and Y. Chen, "Efficient privacy-preserving k-means clustering from secret-sharing-based secure three-party computation," *Entropy*, vol. 24, no. 8, 2022, ISSN: 1099-4300. DOI: 10.3390/e24081145

[17] B. Bozdemir, S. Canard, O. Ermis, H. Möllering, M. Önen, and T. Schneider, "Privacy-preserving density-based clustering," in *Proceedings of the 2021 ACM Asia Conference on Computer and Communications Security*, ser. ASIA CCS '21, Virtual Event, Hong Kong: Association for Computing Machinery, 2021, pp. 658–671, ISBN: 9781450382878. DOI: 10.1145/3433210.3453104

[18] A. Jäschke and F. Armknecht, "Unsupervised machine learning on encrypted data," in *Selected Areas in Cryptography – SAC 2018*, C. Cid and M. J. Jacobson Jr., Eds., Cham: Springer International Publishing, 2019, pp. 453–478. DOI: 10.1007/978-3-030-10970-7_21

[19] S. Malekmohammadi, A. Taik, and G. Farnadi, "Differentially private clustered federated learning," *Transactions on Machine Learning Research*, 2025, ISSN: 2835-8856. [Online]. Available: https://openreview.net/forum?id=JSsko0a4yr

[20] Y. Li, S. Wang, C.-Y. Chi, and T. Q. S. Quek, "Differentially private federated clustering over non-iid data," *IEEE Internet of Things Journal*, vol. 11, no. 4, pp. 6705–6721, 2024. DOI: 10.1109/JIOT.2023.3312852

[21] J. Scott, C. H. Lampert, and D. Saulpic, "Differentially private federated k-means clustering with server-side data," in *Forty-second International Conference on Machine Learning*, 2025. [Online]. Available: https://openreview.net/forum?id=EFLPHl5RGJ

[22] A. Diaa, T. Humphries, and F. Kerschbaum, "FastLloyd: Federated, accurate, secure, and tunable k-Means clustering with differential privacy," in *34th USENIX Security Symposium (USENIX Security 25)*, Seattle, WA: USENIX Association, Aug. 2025, pp. 2733–2752, ISBN: 978-1-939133-52-6. [Online]. Available: https://www.usenix.org/conference/usenixsecurity25/presentation/diaa

[23] M. Yang, L. Huang, and C. Tang, "K-means clustering with local distance privacy," *Big Data Mining and Analytics*, vol. 6, no. 4, pp. 433–442, 2023. DOI: 10.26599/BDMA.2022.9020050

[24] J. Imola, A. Epasto, M. Mahdian, V. Cohen-Addad, and V. Mirrokni, "Differentially private hierarchical clustering with provable approximation guarantees," in *Proceedings of the 40th International Conference on Machine Learning*, A. Krause, E. Brunskill, K. Cho, B. Engelhardt, S. Sabato, and J. Scarlett, Eds., ser. Proceedings of Machine Learning Research, vol. 202, PMLR, 23–29 Jul 2023, pp. 14 353–14 375. [Online]. Available: https://proceedings.mlr.press/v202/imola23a.html

[25] C. Liu, S. Chakraborty, and P. Mittal, "Dependence makes you vulnerable: Differential privacy under dependent tuples," English (US), in *23rd*

Annual Network and Distributed System Security Symposium, NDSS 2016, ser. 23rd Annual Network and Distributed System Security Symposium, NDSS 2016, Publisher Copyright: © 2016 Internet Society.; 23rd Annual Network and Distributed System Security Symposium, NDSS 2016 ; Conference date: 21-02-2016 Through 24-02-2016, The Internet Society, 2016. DOI: 10.14722/ndss.2016.23279

[26] D. Balouek, A. Carpen Amarie, G. Charrier, F. Desprez, E. Jeannot, E. Jeanvoine, A. Lèbre, D. Margery, N. Niclausse, L. Nussbaum, O. Richard, C. Pérez, F. Quesnel, C. Rohr, and L. Sarzyniec, "Adding virtualization capabilities to the Grid'5000 testbed," in *Cloud Computing and Services Science*, ser. Communications in Computer and Information Science, I. I. Ivanov, M. van Sinderen, F. Leymann, and T. Shan, Eds., vol. 367, Springer International Publishing, 2013, pp. 3–20, ISBN: 978-3-319-04518-4. DOI: 10.1007/978-3-319-04519-1_1

[27] T. Araki, J. Furukawa, Y. Lindell, A. Nof, and K. Ohara, "High-throughput semi-honest secure three-party computation with an honest majority," in *Proceedings of the 2016 ACM SIGSAC Conference on Computer and Communications Security*, ser. CCS '16, Vienna, Austria: Association for Computing Machinery, 2016, pp. 805–817, ISBN: 9781450341394. DOI: 10.1145/2976749.2978331

[28] M. Keller, "Mp-spdz: A versatile framework for multi-party computation," in *Proceedings of the 2020 ACM SIGSAC Conference on Computer and Communications Security*, ser. CCS '20, Virtual Event, USA: Association for Computing Machinery, 2020, pp. 1575–1590, ISBN: 9781450370899. DOI: 10.1145/3372297.3417872

[29] A. Baccarini, M. Blanton, and C. Yuan, "Multi-party replicated secret sharing over a ring with applications to privacy-preserving machine learning," *Proceedings on Privacy Enhancing Technologies*, 2023. DOI: 10.56553/popets-2023-0035

[30] V. Satopaa, J. Albrecht, D. Irwin, and B. Raghavan, "Finding a "kneedle" in a haystack: Detecting knee points in system behavior," in *2011 31st International Conference on Distributed Computing Systems Workshops*, 2011, pp. 166–171. DOI: 10.1109/ICDCSW.2011.20

[31] Y. Lindell, "How to simulate it – a tutorial on the simulation proof technique," in *Tutorials on the Foundations of Cryptography: Dedicated to Oded Goldreich*, Y. Lindell, Ed. Cham: Springer International Publishing, 2017, pp. 277–346, ISBN: 978-3-319-57048-8. DOI: 10.1007/978-3-319-57048-8_6

[32] Y. Lecun, L. Bottou, Y. Bengio, and P. Haffner, "Gradient-based learning applied to document recognition," *Proceedings of the IEEE*, vol. 86, no. 11, pp. 2278–2324, 1998. DOI: 10.1109/5.726791

[33] D. Li, Y. Yang, Y.-Z. Song, and T. M. Hospedales, "Deeper, broader and artier domain generalization," in *2017 IEEE International Conference on Computer Vision (ICCV)*, 2017, pp. 5543–5551. DOI: 10.1109/ICCV.2017.591

[34] H. Venkateswara, J. Eusebio, S. Chakraborty, and S. Panchanathan, "Deep hashing network for unsupervised domain adaptation," in *2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2017, pp. 5385–5394. DOI: 10.1109/CVPR.2017.572

[35] P. Warden, "Speech commands: A dataset for limited-vocabulary speech recognition," *arXiv preprint arXiv:1804.03209*, 2018. DOI: 10.48550/arXiv.1804.03209

[36] R.-A. Cherrueau, M. Delavergne, A. van Kempen, A. Lebre, D. Pertin, J. R. Balderrama, A. Simonet, and M. Simonin, "Enoslib: A library for experiment-driven research in distributed computing," *IEEE Transactions on Parallel and Distributed Systems*, vol. 33, no. 6, pp. 1464–1477, 2022. DOI: 10.1109/TPDS.2021.3111159

[37] H. B. McMahan, E. Moore, D. Ramage, S. Hampson, and B. A. y Arcas, "Communication-efficient learning of deep networks from decentralized data," in *International Conference on Artificial Intelligence and Statistics*, 2016. [Online]. Available: https://api.semanticscholar.org/CorpusID:14955348

[38] T. Li, A. K. Sahu, M. Zaheer, M. Sanjabi, A. Talwalkar, and V. Smith, "Federated optimization in heterogeneous networks," *Proceedings of Machine learning and systems*, vol. 2, pp. 429–450, 2020. DOI: 10.48550/arXiv.1812.06127

[39] T. Li, S. Hu, A. Beirami, and V. Smith, "Ditto: Fair and robust federated learning through personalization," in *International conference on machine learning*, PMLR, 2021, pp. 6357–6368. DOI: 10.48550/arXiv.2012.04221

[40] C. Briggs, Z. Fan, and P. Andras, "Federated learning with hierarchical clustering of local updates to improve training on non-iid data," in *2020 International Joint Conference on Neural Networks (IJCNN)*, 2020, pp. 1–9. DOI: 10.1109/IJCNN48605.2020.9207469

[41] Y. Wei, S. Zhao, C. Zhao, Z. Liu, Z. Chen, and M. Zhao, *Fedcaprivacy: Privacy-preserving heterogeneous federated learning with anonymous adaptive clustering*, 2025. arXiv: 2503.23292 [cs.CR]. [Online]. Available: https://arxiv.org/abs/2503.23292

[42] Z. Li, H. Bao, M. Guan, H. Pan, C. Huang, and H.-N. Dai, "Ebs-cfl: Efficient and byzantine-robust secure clustered federated learning," in *Proceedings of the Thirty-Ninth AAAI Conference on Artificial Intelligence and Thirty-Seventh Conference on Innovative Applications of Artificial Intelligence and Fifteenth Symposium on Educational Advances in Artificial Intelligence*, ser. AAAI'25/IAAI'25/EAAI'25, AAAI Press, 2025, ISBN: 978-1-57735-897-8. DOI: 10.1609/aaai.v39i17.34046

[43] S. Han, J. Pan, S. Zhao, S. Ren, Z. Wang, S. He, K. Ren, Z. Qin, and X. Chen, "Fed-ehp: Efficient and heterogeneous privacy-preserving personalized federated learning," *IEEE Transactions on Dependable and Secure Computing*, vol. 23, no. 2, pp. 3206–3221, 2026. DOI: 10.1109/TDSC.2025.3634446

[44] T. Yuhang, W. Rui, Q. Yanqi, P. Emmanouil, and L. Kaitai, "Flvoogd: Robust and privacy preserving federated learning," in *Proceedings of The 14th Asian Conference on Machine Learning*, E. Khan and M. Gonen, Eds., ser. Proceedings of Machine Learning Research, vol. 189, PMLR, Dec. 2023, pp. 1022–1037. [Online]. Available: https://proceedings.mlr.press/v189/yuhang23a.html