Evaluating Differential Privacy in Federated Clustering Under Adversarial Attacks

Presented by:
Urvi Gupta (22B1006) Dept. of CSE
Ojas Maheshwari (22B0965) Dept. of CSE

Course: CS6007:Multi-Agent Machine Learning

Instructor: Prof. Avishek Ghosh

IIT Bombay

November 2, 2025

Problem Statement

- **Core challenge:**
  - Evaluate whether Differential Privacy can effectively protect Federated K-Means from adversarial attacks **without significantly degrading clustering quality**.
- **Key difficulties:**
  - Adding DP noise protects privacy but degrades clustering quality — finding a meaningful trade-off is challenging.
  - Simulating realistic and effective attacks that meaningfully test privacy guarantees is non-trivial.
  - Non-IID client data and varying sample sizes complicate both clustering and privacy analysis.
- **What prior work misses:**
  - Prior work focuses on theoretical privacy guarantees (ϵ, δ) but rarely tests robustness against concrete attacks.
  - Empirical evaluation under realistic attack settings is largely missing, leaving practical effectiveness uncertain.

Literature Survey - Federated Clustering

- **Federated K-Means Clustering (Garst et al., Delft University):**
  - Proposed one of the first frameworks for performing K-Means clustering in a federated setting without sharing raw data.
  - Highlighted communication efficiency and privacy concerns, but did not incorporate formal privacy mechanisms.
- **Differentially Private Federated K-Means (Scott et al., 2025):**
  - Introduced a differentially private version of federated K-Means by adding calibrated Gaussian noise on server-side updates.
  - Provided theoretical privacy guarantees (ϵ, δ) and utility analysis, but no empirical evaluation against real attacks.
- **Inverting Gradients (Geiping et al., 2020):**
  - Showed that gradients shared during federated learning can be reversed to recover the original training data.
  - Highlighted serious privacy risks in model updates, motivating similar reconstruction studies in federated clustering.

Contribution

- **Verification and Benchmarking:** Verified the FedDP-KMeans implementation of **Scott et al. (2025)** against the non-private Federated K-Means baseline.
- **Reconstruction attacks:**
  - **Data-level reconstruction:** Simulated recovery of an individual datapoint from shared client updates, comparing reconstruction quality (cosine similarity) under DP vs non-DP.
  - **Client-level reconstruction:** Attempted to reconstruct the client's aggregate update (cluster means sent to the server) to evaluate information leakage at the client level.

Definitions

Evaluating K-Means Performance

How we measure success

1. K-Means Cost

Definition: It is the **sum of squared Euclidean distances** from each data point to its assigned cluster center. Given a dataset $P = \{p_1, \ldots, p_N\}$, a set of K centers $C = \{c_1, \ldots, c_K\}$, and assignments y_i for each point p_i: $\text{Cost}(P, C) = \sum_{i=1}^{N} \|p_i - c_{y_i}\|^2$

2. K-Means Accuracy

1. For each predicted cluster j, find the *majority* true label among all points assigned to j.
2. Count how many points in cluster j have this majority true label. This is the number correct for cluster j.
3. Sum these counts across all K clusters to get the total number correct.
4. Accuracy $= \frac{\text{Total Number Correct}}{\text{Total Data Points}}$

Differential Privacy (DP)

Definition

An algorithm $A : \mathcal{P} \to \mathcal{S}$ is said to satisfy (ε, δ)-**Differential Privacy** if for all neighboring datasets P, P' and all measurable subsets $S \subseteq \mathcal{S}$,

$$\Pr[A(P) \in S] \leq e^{\varepsilon} \Pr[A(P') \in S] + \delta$$

where $\varepsilon > 0$ quantifies the privacy loss and $\delta \geq 0$ allows for a small probability of failure.

- **Neighboring Datasets:**
  - *Data-point-level DP:* P and P' differ by one individual data point.
  - *Client-level DP:* P and P' differ by all data points belonging to one client.
- **Compositionality:** If algorithms $A_1, \ldots, A_t$ are each $(\varepsilon_i, \delta_i)$-DP, their combination remains DP with parameters $(\sum_i \varepsilon_i, \sum_i \delta_i)$, enabling multi-step private pipelines.

Privacy Models (1/2): Data-point Privacy

- **What it Protects:**
  - A *single data point* (e.g., one person's record).
- **The Privacy Question:**
  - "Does the final result change if we add or remove *Jane Doe's record* from Client A's dataset?"
- **Sensitivity (L_2):**
  - The maximum impact of **one point**.
  - This is a small and generally known value (or easy to set).
  - Example: `max_data_norm`
- **Clipping:**
  - Applied to each data point (or implicitly assumed).
- **Use Case:**
  - Standard "cross-silo" FL.
  - Clients are trusted entities (like hospitals).
  - The individuals *within* the silo need privacy.

Privacy Models (2/2): Client-level Privacy

- **What it Protects:**
  - The *entire contribution* of one client (e.g., all 10,000 records from "Hospital A").
- **The Privacy Question:**
  - "Does the final result change if we add or remove *Hospital A's entire dataset* from the training?"
- **Sensitivity (L_2):**
  - The maximum impact of **one client**. This is large and unknown.
  - Clipping is mandatory to create a finite bound.
  - Example: `max_client_norm_vector`
- **Clipping:**
  - Applied to the client's *total aggregated update* before it leaves the client.
- **Use Case:**
  - Standard "cross-device" FL.
  - The client (e.g., a mobile phone) is the user and cannot be trusted.
  - This is a much stronger privacy guarantee.

Federated Differentially Private K-Means (FedDP-KMeans)

- **Goal:** Perform k-means clustering across m clients without sharing raw data, while ensuring (ε, δ)-differential privacy.
- **Setup:** Each client j has data P_j, and the server has small auxiliary data Q (may be out-of-distribution).
- **Algorithm Structure:**
  1. **FedDP-Init:** Generates DP-protected initial cluster centers using client updates and server data.
  2. **FedDP-Lloyds:** Refines centers iteratively with noised client contributions.
- **Privacy:** Gaussian noise added to client-side aggregates.

FedDP-Init — Algorithm

- **Objective:** Generate high-quality, privacy-preserving initial cluster centers before running Lloyd's iterations.
- **Step 1 — Subspace Estimation:**
  - Each client computes its local data covariance $P_j P_j^T$.
  - The server aggregates these (with Gaussian noise) to estimate top-k eigenvectors.
  - Projects data into a lower-dimensional subspace, improving signal-to-noise ratio.
- **Step 2 — Proxy Weight Computation:**
  - Server shares projection matrix Π and projected server data ΠQ.
  - Clients compute how many local points are closest to each server point (weights).
  - Noised weights are sent to server (Laplace mechanism, $(\varepsilon_2, 0)$-DP).
- **Step 3 — Center Refinement:**
  - Server runs weighted k-means on ΠQ to get projected centers.
  - Clients refine these using their local data and send noised sums/counts.
  - Server aggregates and reconstructs final DP initialization centers.

FedDP-Lloyds Algorithm

- **Goal:** Refine cluster centers under DP after FedDP-Init.
- **Key Idea:** Lloyd's updates rely only on *sums and counts* — enabling secure aggregation and DP noise addition.
- **Steps per Round:**
  - Server sends current centers ν^{t-1} to clients.
  - Client j assigns each point to the nearest center and computes:

$$m_{jr} = \sum_{p \in S_r^j} p, \quad n_{jr} = |S_r^j|$$

  - Add Gaussian noise to m_{jr} and Laplace noise to n_{jr}.
  - Server aggregates noisy stats and updates:

$$\nu_r^t = \tilde{m}_r / \tilde{n}_r$$

- **Privacy:** Overall algorithm achieves (ε, δ)-DP via composition.

Explanations and Intuition

Lloyd's Algorithm & The Importance of Initialization

- **The Problem with Lloyd's in DP:**
  - The paper identifies a counterintuitive trade-off for DP algorithms.
  - Accessing private data more often (e.g., more rounds of Lloyd's algorithm) requires more noise per step to stay within the privacy budget.
  - This can lead to *lower* accuracy, not higher.
- **The Goal:**
  - Consequently, a good initialization is crucial for achieving high accuracy.
  - This is because it allows the algorithm to avoid running many iterations of Lloyd's.
- **Iterations Required for FedDP-KMeans:**
  - **Finding:** The FedDP-Init centers are so close to the optimal ones that only very few (sometimes none at all) steps of Lloyd's algorithm are required to refine them.
  - **Conclusion:** The preference was to instead use all the budget for the initialization rather than on subsequent Lloyd's steps.

Why FedDP-Init Requires Fewer Lloyd's Steps?

More iterations $\rightarrow$ smaller budget per step $\rightarrow$ **more noise per step**.

- **k-FED: A "Local-First" Initialization**
  - k-FED (a non-private baseline) has clients run k-means *locally* and send their centers.
  - If client data is non-IID (unbalanced or different), these local centers are **highly biased**.
  - The server then clusters these biased centers, resulting in a poor global starting point that is far from the true optimum.
- **FedDP-Init: A "Global-First" Initialization**
  - FedDP-Init is designed to solve this. It uses its budget to get a high-quality start.
  - **Step 1 (Subspace)** estimates the *global* data structure from all clients.
  - **Step 3 (Refinement)** is already one complete, *globally-informed* Lloyd's step.
  - **Result:** The centers are already so close to the optimum that few (or even **zero**) further noisy FedDP-Lloyds iterations are needed.

Clipping: An Important Step in FedDP-Init

What is Clipping? (What, Why, How)

- **What is it?**
  - Clipping enforces a **maximum limit** (a "cap") on the size of a client's (or datapoint's) contribution before it is sent to the server.
- **Why is it necessary?**
  - Differential Privacy adds noise based on **sensitivity** (the max possible impact of one client).
  - A client with 1M points has a *massive* impact vs. a client with 10 points. This sensitivity is unknown and potentially infinite.
  - Clipping creates a **fixed, known sensitivity** (e.g., "no client can contribute more than 100.0"). This makes adding a finite amount of noise possible.
- **How is it implemented? (L_2 Clipping)**
  - The client calculates the L_2 norm (length) of its update vector (e.g., its local sum m_r).
  - **If `norm` $\leq$ `C`:** The vector is sent as-is.
  - **If `norm > C`:** The vector is "shrunk" by multiplying it by `C / norm`. Its original direction is thus preserved.

Parameters of FedDP-Init: Clipping Bounds

- `outer_product_clipping_bound:` Caps the client's **covariance matrix** $(P_j P_j^T)$ for Step 1 of `FedDP-Init`.
- `weighting_clipping_bound:` Caps the client's **proxy weights** (counts) for Step 2 of `FedDP-Init`.
- `center_init_clipping_bound:` Caps the L_2 norm of the client's **local mean vectors** for Step 3 of `FedDP-Init`.
- `center_init_contributed_components_clipping_bound:` Caps the L_1 norm (count) of the client's **histogram of non-empty clusters** for Step 3.
- `fedlloyds_clipping_bound` Caps the L_2 norm of the client's **vector sums (m_r)** during `FedLloyds` rounds.
- `fedlloyds_laplace_clipping_bound:` Caps the L_1 norm (count) of the client's **point counts (n_r)** during `FedLloyds` rounds.

Comparative Analysis of KFed and FedDP-Init

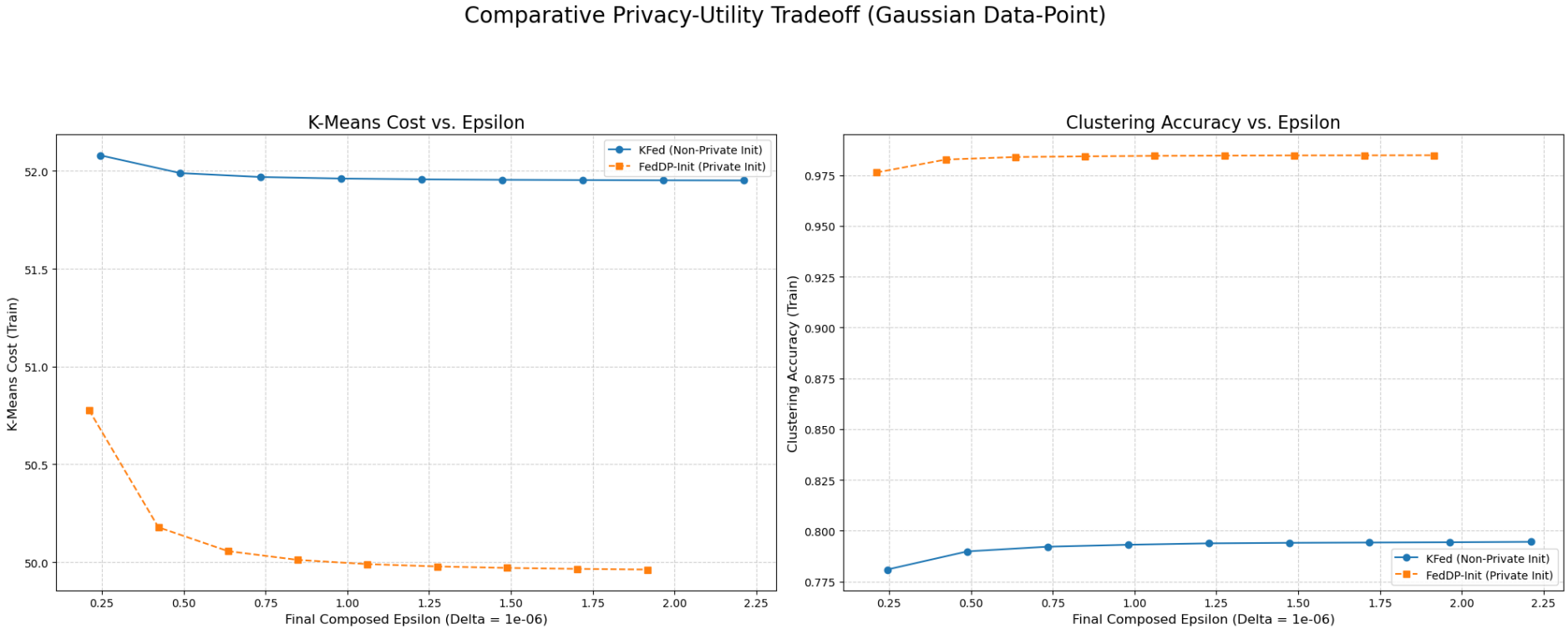
Datasets Used

- **Synthetic Gaussian Dataset:**
  - A controlled, high-dimensional dataset used to simulate a standard federated k-means environment.
  - Ideal for isolating the effect of Differential Privacy (DP) on clustering utility and privacy leakage due to its simple, predictable structure.

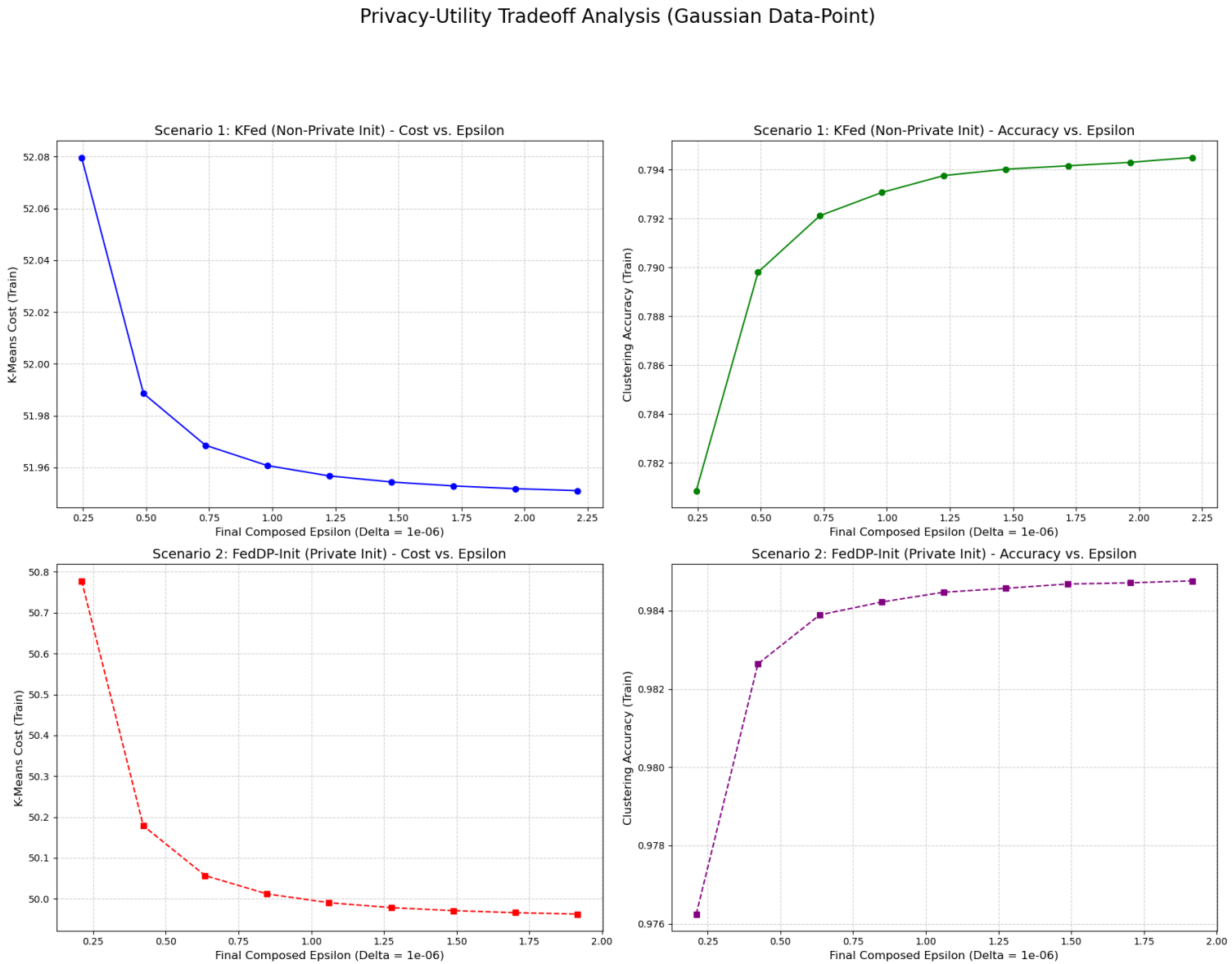
- **Folktables Dataset (US Census ACS):**
  - A real-world dataset focused on demographic and economic features.
  - Typically lower-dimensional and **non-isotropic**, providing a practical test against complex, real-world data distributions.

Gaussian Synthetic Dataset with Datapoint Privacy

Comparison of Utility vs. Privacy between KFed and FedDP-Init



Gaussian Synthetic Dataset with Datapoint Privacy

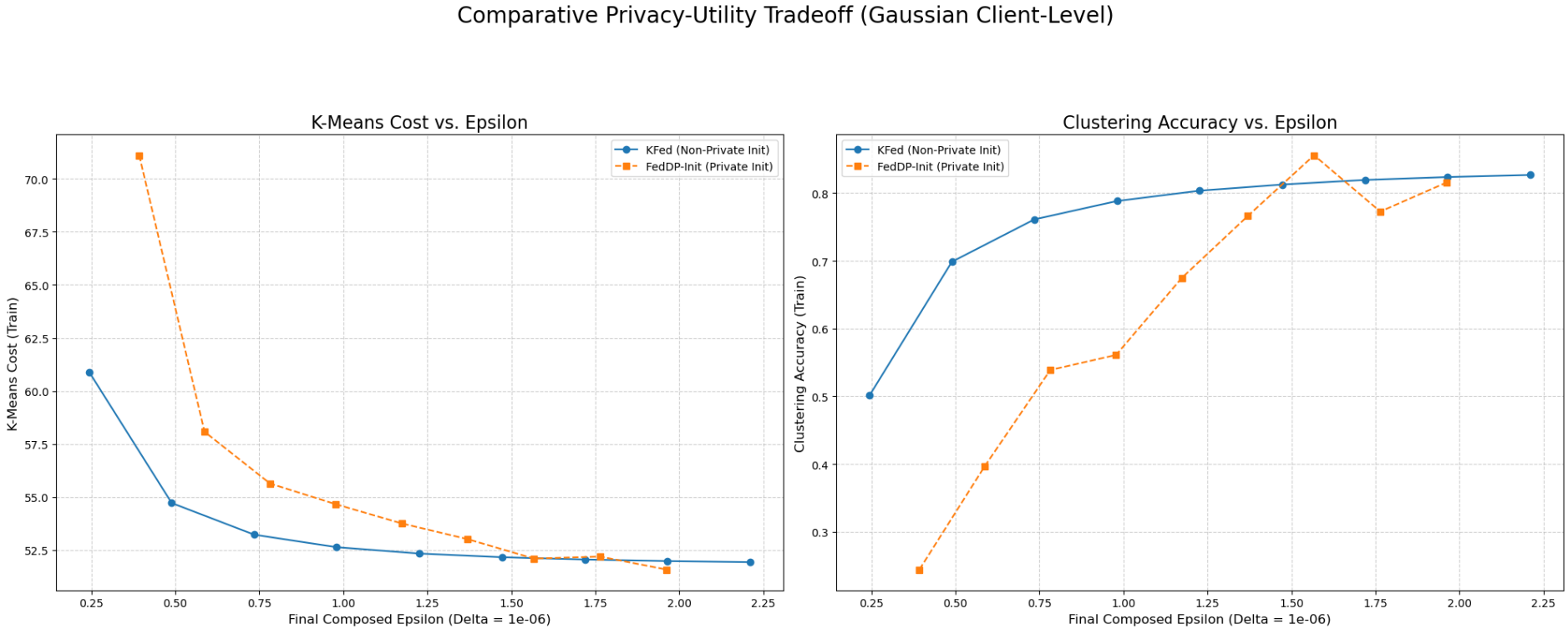


Why FedDP-Init Won with Data-Point Privacy

- **Scenario:** Low noise (data-point level).
- **FedDP-Init**
  - Runs 3 private steps to get its initialization.
  - Because the data-point noise is tiny, even 3 steps of this noise have almost no effect.
  - It produces a near-perfect initialization from the start.
- **KFed:**
  - Gets a free (non-private) initialization, which is not as good as the one FedDP-Init finds.
  - It never catches up to FedDP-Init's superior starting point.
- **Conclusion:** With low noise, the **quality of the initialization algorithm** is what matters most. The paper's FedDP-Init is simply a better algorithm.

Gaussian Synthetic Dataset with Client-Level Privacy

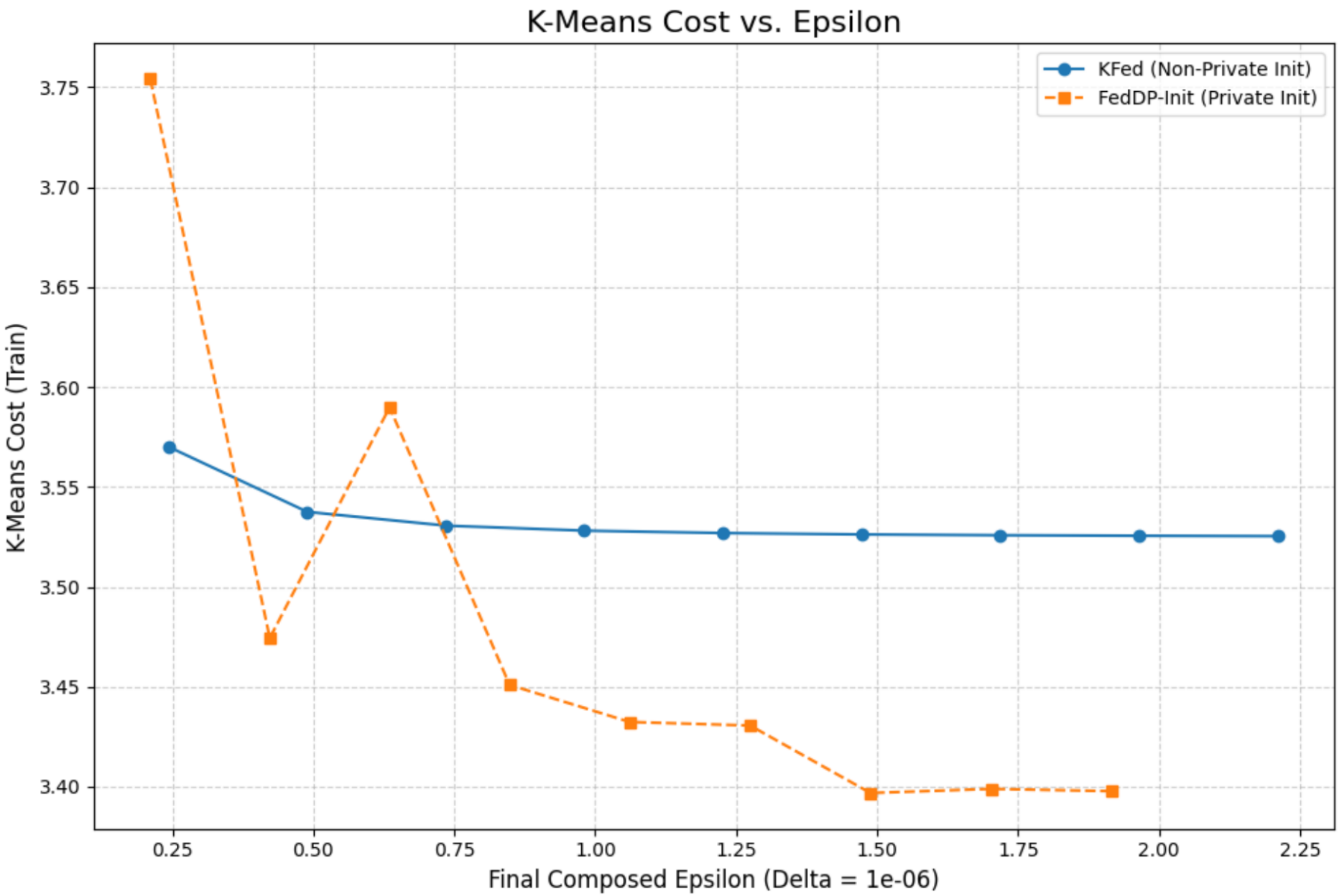
Comparison of Utility vs. Privacy between KFed and FedDP-Init



Why KFed Wins at Low Epsilon (Client-Level Privacy)

- **Scenario:** High noise (client-level).
- **FedDP-Init:**
  - At low epsilon (e.g., 0.25), it splits its tiny budget 3 ways for its 3 private steps.
  - Each step must add massive client-level noise.
  - This triply large dose of noise destroys the initialization, resulting in terrible accuracy and cost.
- **KFed:**
  - Gets its free non-private initialization.
  - It adds the massive client-level noise *only once* during refinement.
  - One dose of massive noise is less bad than three, so it performs better at low epsilon.
- However, at $\epsilon = 2.5$, FedDP-Init gave an accuracy of 97.62% that is much higher than what KFed could ever achieve. Thus, the new algorithm eventually surpasses KFed at higher epsilon.
- **Conclusion:** With high noise, the **number of private steps** is what matters most. FedDP-Init's 3-step process is a disadvantage at low epsilon.

Folktables US Dataset with Datapoint Privacy



Reconstruction Attacks

Threat Model

- **White-box setting:** full access to training code and DP mechanism.
- Observes global cluster centers after each round.
- knows the DP mechanism (Gaussian/Laplace) and its parameters (ε, δ, noise scales, clipping bounds).
- Attacker knows number of clients, number of features, and the data distribution/statistics (but **not** raw datapoints).
- With knowledge of model hyperparameters and training setup, attacker can train **shadow models** to judge whether a point (or client) was likely included.

Note: This is a strong (worst-case) attacker model used to stress-test DP; a weaker black-box attacker that only sees final centers would yield much weaker attacks and could give a false sense of robustness.

Reconstruction Attack: Overview

- **Goal:** Reconstruct **private training information** from shared, noisy client updates in Federated Learning (FL).
- **Setup:** The attacker observes the noisy client updates (total sums and counts in Federated K-Means) in each FL round.
- **Attack Levels:**
  - **Data-Level:** Recover an individual datapoint x.
  - **Client-Level:** Reconstruct a client's mean update μ_j.
- **Comparison:** Attack success is measured on **Non-Private** vs. **DP-Federated K-Means** algorithms via L2 error and cosine similarity.

Data-Level Reconstruction: Reconstructing $\mathbf{x}$

- **Objective:** Recover a single, targeted datapoint $\mathbf{x}$.
- **How the attack works**:
  1. Run federated training *once without* $\mathbf{x}$ to get baseline global centers (C_{base}).
  2. Using C_{base}, simulate two noisy updates from the target client:
     - $\tilde{u}_{\text{in}}$: update *with* the target point $\mathbf{x}$.
     - $\tilde{u}_{\text{out}}$: update *without* the target point $\mathbf{x}$.
  3. Compute the **difference** (the point's isolated contribution):

$$\Delta = \tilde{u}_{\text{in}} - \tilde{u}_{\text{out}}$$

- **Reconstruction Formula:**
  - Find the point's most probable cluster k^* (where the count difference Δ_{counts} is largest):$k^* = \text{argmax}_k(\Delta_{\text{counts}}[k])$
  - The reconstructed point is the **sum difference from that single cluster**:

$$\hat{\mathbf{x}} = \Delta_{\text{sums}}[k^*]$$

- **Metric:** Accuracy is measured by **Cosine Similarity** $\left(\frac{\mathbf{x}\cdot\hat{\mathbf{x}}}{\|\mathbf{x}\|\|\hat{\mathbf{x}}\|}\right)$.

Client-Level Reconstruction: Reconstructing μ_j

- **Objective:** Reconstruct the **mean update** ($\hat{\mu}_j$) of a target client j's entire dataset.
- **How the attack works**:
  1. Simulate a global model state trained *without* the target client j.
  2. Simulate the target client j's **full noisy local update** ($\tilde{u}_j$).
  3. **The noisy update IS the reconstructed total** (sum and count) contribution:

$$(\hat{\Sigma}_j, \hat{N}_j) = \tilde{u}_j \quad \text{(Noisy Sum and Count)}$$

- **Reconstructed Mean:** Calculate the mean from the noisy components:

$$\hat{\mu}_j = \frac{\hat{\Sigma}_j}{\hat{N}_j}$$

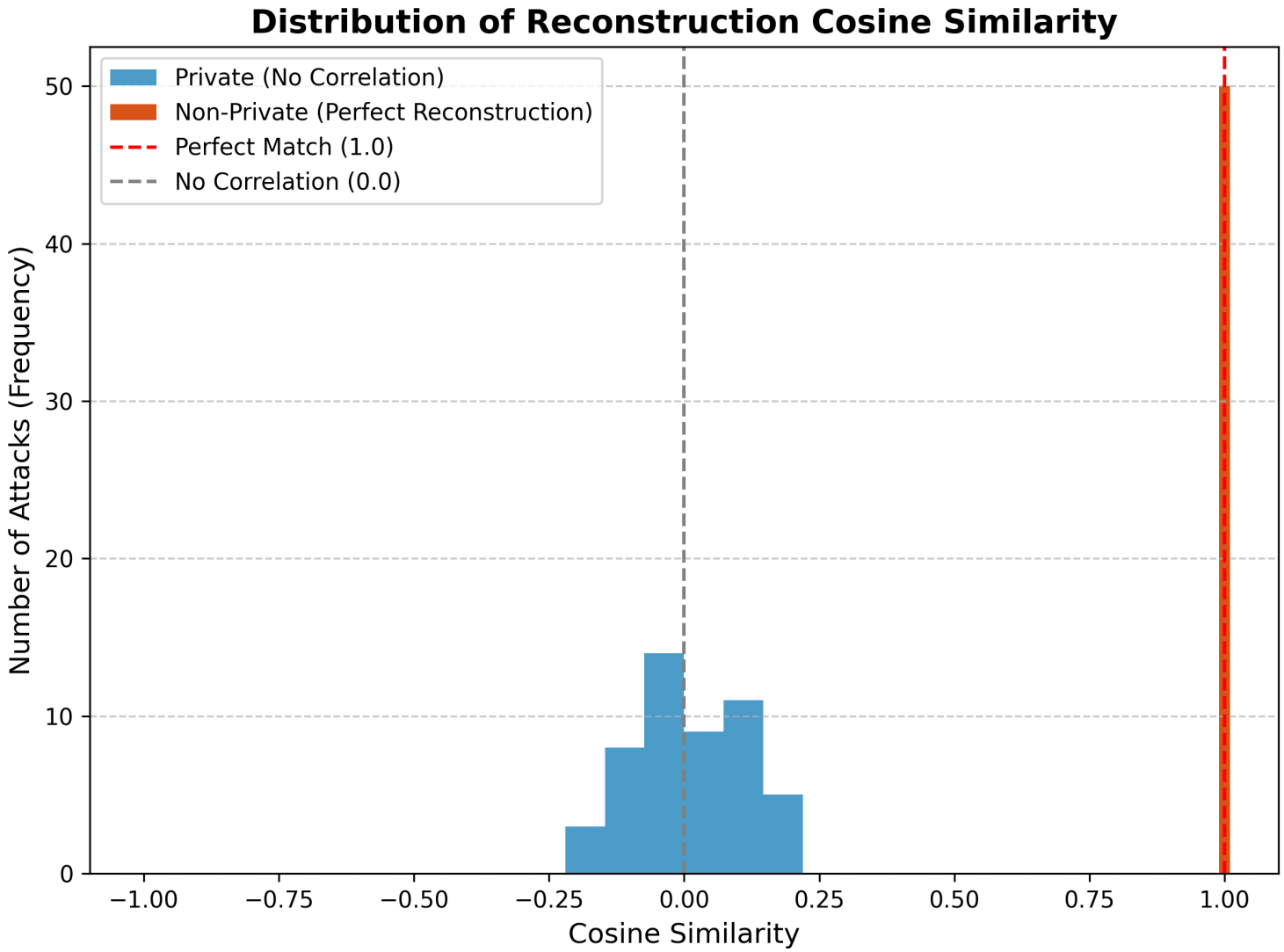
- **Privacy Effect:** Stronger Differential Privacy (smaller ε) $\rightarrow$ higher **noise distortion** in $\hat{\mu}_j \rightarrow$ lower attack accuracy.

Experiment 1

Data-point Reconstruction Attack with Gaussian Dataset

- **Goal:** Reconstruct the vector of a single target data point from a client's noisy update.
- **Model:** Federated k-means with and without differential privacy
- Clients (M): 100
- Samples per client: 1000
- Feature Size (D): 100
- Clusters (K): 10
- Attack Trials: 50 target data points
- Privacy level: data-point
- Target Epsilon (ϵ): 1.0
- Target Delta (δ): 1.0×10^{-6}
- Core Clipping (C): 11 (FedLloyds L_2 Clipping Bound)
- Mechanism: Gaussian Noise (due to L_2 clipping and $\delta \neq 0$)

Experiment 1 - Results

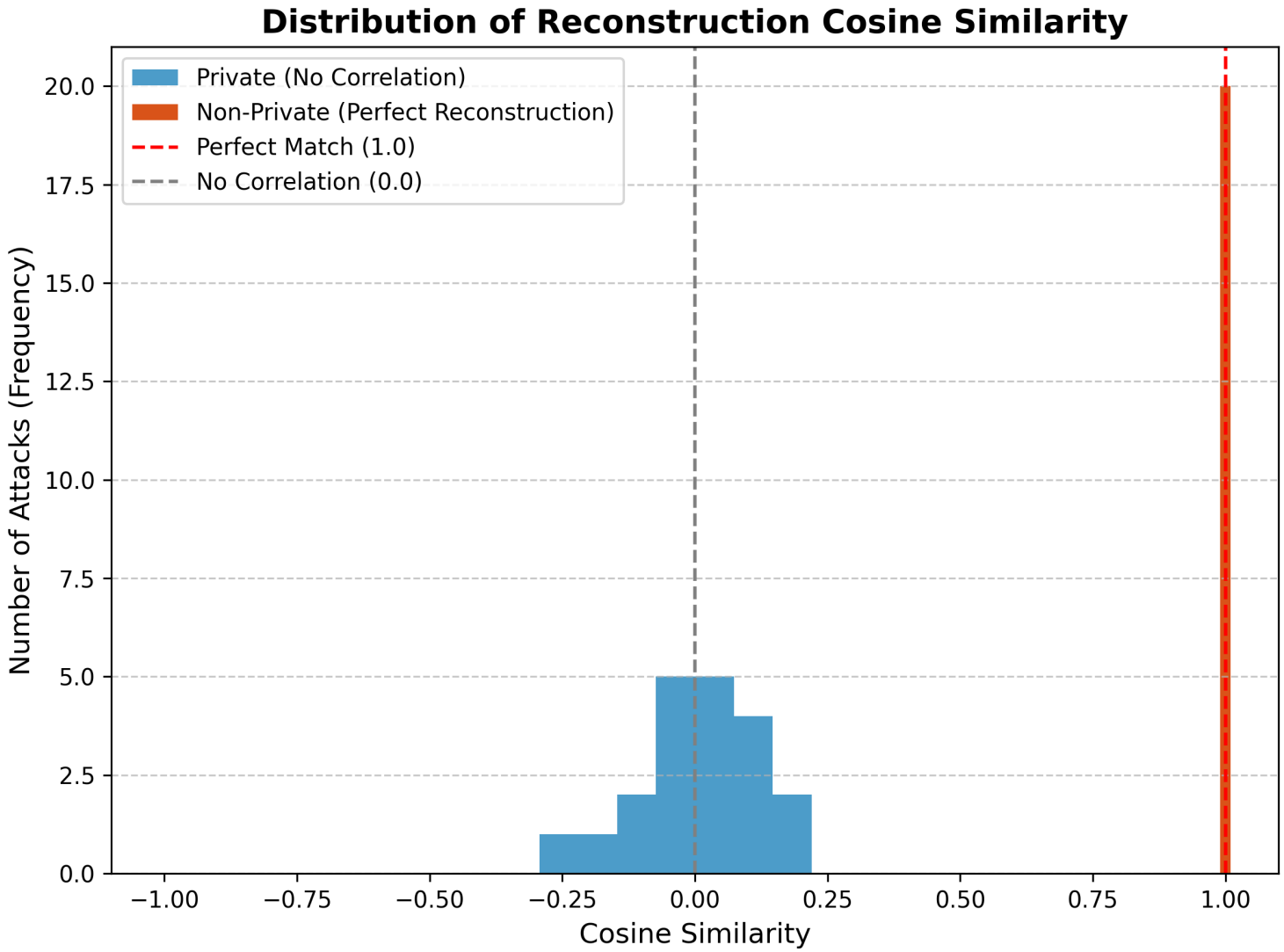


Experiment 2

Client Level Reconstruction Attack with Gaussian Dataset

- **Goal:** Reconstruct a client's mean update
- **Model:** Federated k-means with and without differential privacy
- Clients (M): 100
- Samples per client: 1000
- Feature Size (D): 100
- Clusters (K): 10
- Attack Trials: 20 clients
- Privacy level: client-level
- Target Epsilon (ϵ): 1.0
- Target Delta (δ): 1.0×10^{-6}
- Core Clipping (C): 120 (FedLloyds L_2 Clipping Bound)
- Mechanism: Gaussian Noise (due to L_2 clipping and $\delta \neq 0$)

Experiment 2 - Results

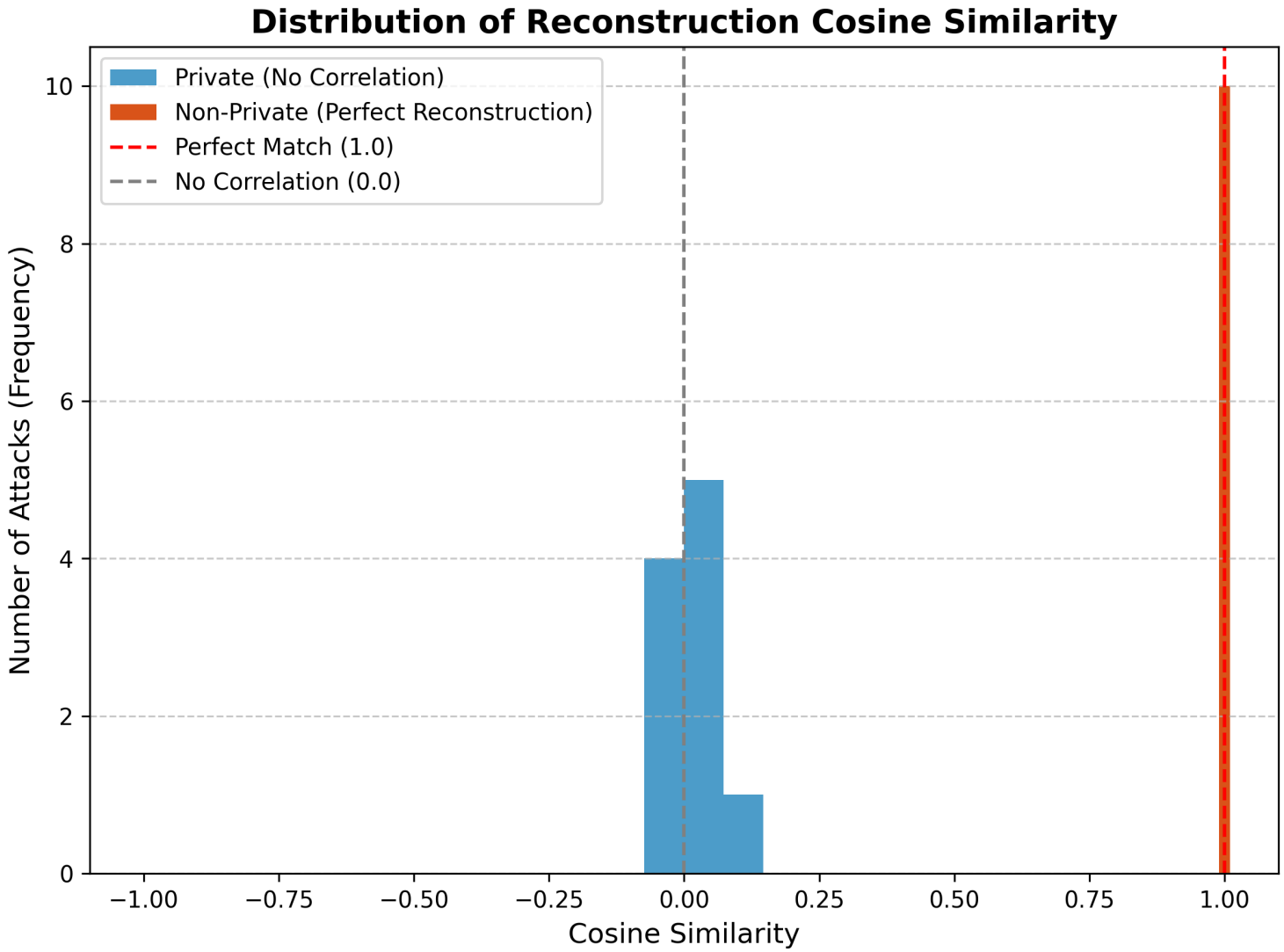


Experiment 3

Data-point Reconstruction Attack with Folktables Dataset

- **Goal:** Reconstruct the vector of a single target data point from a client's noisy update.
- **Model:** Federated k-means with and without differential privacy
- Clients (M): 51
- Samples per client: Variable
- Feature Size (D): 124
- Clusters (K): 10
- Attack Trials: 10 target data points
- Privacy level: data-point
- Target Epsilon (ϵ): 1.0
- Target Delta (δ): 1.0×10^{-6}
- Core Clipping (C): 2.65 (FedLloyds L_2 Clipping Bound)
- Mechanism: Gaussian Noise (due to L_2 clipping and $\delta \neq 0$)

Experiment 3 - Results



Conclusion: Key Takeaways & Challenges

- **Key Takeaways:**
  - Spending privacy budget on initialization (FedDP-Init) is more effective than on iterations (KFed/FedLloyds).
  - FedDP-Init provides high accuracy with very few (or even 0) Lloyd's steps.
  - FedDP-Init is highly robust to reconstruction attacks with minimal utility loss.

- **Limitations & Challenges:**
  - Client-level Folktables simulation failed due to clipping issues and sparse data.
  - Membership Inference Attacks (MIA) were inconclusive, succeeding similarly on both private and non-private models.

Future Work

- **Address Current Limitations:**
  - Develop systematic methods to estimate optimal hyperparameters (e.g., clipping bounds).
  - Investigate and design more effective defenses against Membership Inference Attacks.
- **Explore New Directions:**
  - Test the system's robustness against other types of privacy attacks (e.g., attribute inference).
  - Extend the DP-KMeans framework to different data modalities or more complex clustering algorithms.

References I

[1] S. Garst and M. Reinders.
Federated K-Means Clustering.
Delft Bioinformatics Lab, Delft University of Technology, 2023.

[2] J. Scott, C. H. Lampert, and D. Saulpic.
Differentially Private Federated k-Means Clustering with Server-Side Data.
arXiv preprint arXiv:2506.05408, 2025.

[3] R. Shokri, M. Stronati, C. Song, and V. Shmatikov.
Membership Inference Attacks Against Machine Learning Models.
In *2017 IEEE Symposium on Security and Privacy (SP)*, pages 3–18, 2017.

[4] J. Geiping, H. Bauermeister, H. Dröge, and M. Moeller.
Inverting Gradients: How Easy Is It to Break Privacy in Federated Learning?
In *Advances in Neural Information Processing Systems (NeurIPS)*, 2020.

Thank You!