CONCORDIA: SELF-IMPROVING SYNTHETIC TABLES FOR FEDERATED LLMS

Anonymous authors
Paper under double-blind review

ABSTRACT

Adapting large language models (LLMs) to tabular prediction in regulated cross-silo settings is difficult when institutions cannot share raw records, validation sets, synthetic tables, generators, or downstream model updates. This challenge is amplified by non-IID client distributions, extreme class imbalance, and limited communication budgets, where only one or two rounds may be feasible. Synthetic tables offer a natural on-premise training substrate, but existing pipelines often treat generation as static or optimize proxy objectives disconnected from the evolving downstream learner. We propose CONCORDIA, a tri-level limited-shot framework that turns private validation utility into an online signal for refining client-local synthetic generation. Each client performs local LoRA adaptation on synthetic tables, learns a capacity-limited utility scorer from private validation feedback to reweight synthetic samples, and uploads only this scorer. The server pools and redistributes scorers within each task, and each client uses the pooled scorer feedback to refine its own generator with group-relative policy optimization (GRPO). Across finance and healthcare tabular benchmarks, CONCORDIA improves early-round MCC under severe heterogeneity and imbalance, with the clearest gains on the long-tail Travel task where positives are near 0.01; the finance tasks are single-client streams and measure local self-improvement, while healthcare evaluates multi-client scorer pooling. Generator audits further suggest that scorer-guided refinement changes the support of synthetic training data, preserving coherent feature dependencies in finance and emphasizing multi-feature risk patterns in healthcare.

1 INTRODUCTION

What can be federated when neither data nor model updates can move? This question is central to adapting large language models (LLMs) for tabular prediction under compliance-constrained, limited-shot cross-silo federated learning (FL), where institutions may participate in only one or two communication rounds and cannot export raw records, validation sets, synthetic tables, generator parameters, or high-capacity model updates. (Rieke et al., 2020; Chua et al., 2024; Stoisser et al., 2025; Li et al., 2021; Kang et al., 2023) Such restrictions arise naturally in regulated domains such as finance and healthcare, where institutions are independently governed and differ in schema, feature semantics, decision policies, and label prevalence. (Yan et al., 2025; Wang et al., 2024; Nam et al., 2024) The difficulty is further amplified by extreme class imbalance: the outcomes that matter most, such as default, fraud, insurance claims, or adverse clinical events, are often rare, making local adaptation prone to majority-class collapse. (Weng et al., 2025; Akinseloyin et al., 2025) Thus, the central challenge is not how to aggregate federated model updates, but how to extract and coordinate utility-relevant learning signals when the usual objects of federation are non-exportable.

Prior work breaks at exactly the point where our setting becomes hard. Standard federated adaptation can coordinate non-IID clients, but it does so by exchanging gradients, adapters, LoRA deltas, or other task-model updates. (Tan et al., 2023; Collins et al., 2021; Yi et al., 2023; Babakniya et al., 2023; Fan et al., 2023; Cheng et al., 2025; Ye et al., 2024b;a; Guo et al., 2025b; Wang et al., 2025) Synthetic-data FL reduces raw-data exposure, but it often shares synthetic samples, fixes the generator, or optimizes generation with proxy objectives disconnected from the downstream learner. (Lomurno & Matteucci, 2024; Abacha et al., 2024; Goetz & Tewari, 2020; Hu et al., 2022; Xin et al., 2022) LLM-based tabular synthesis improves schema-conditioned generation, but plausibility is not the

Reviewers: please read the Reviewer Guidelines (iclr.cc/Conferences/2027/ReviewerGuidelines) and the AI Policy for Reviewers (iclr.cc/Conferences/2027/AIPolicyForReviewers). If you used AI to expand, edit, or polish your review, please provide the input text to the LLM. Better still, consider skipping the LLM and submitting your original text: we, and the authors, are much more interested in your unedited thoughts than in what an LLM has to say. AI-assisted or not, you are putting your name and reputation behind your review: LLM-generated falsehoods, hallucinations or misrepresentations are subject to disciplinary action, which may include desk-rejecting all papers you have authored.

same as usefulness: a realistic table may still teach the learner little about rare validation-critical cases. (Borisov et al., 2022; Solatorio & Dupriez, 2023; Schmidhuber & Kruschwitz, 2024) Federated RLHF and GRPO-style post-training show how distributed preferences can guide LLM optimization, but they mainly use those signals to update a shared model or policy. (Guo et al., 2025a; Wu et al., 2025; Zhu et al., 2026)

To address it, we propose Concordia, a tri-level limited-shot federated loop that turns private validation utility into an online signal for refining on-premise synthetic training distributions under heterogeneity. (Ren et al., 2018) The three levels are jointly necessary: local LoRA (Hu et al., 2021) adaptation defines the evolving client model state, utility modeling distills private validation feedback into a capacity-limited scorer that reweights synthetic mini-batches toward utility-critical and often minority cases, and federated coordination stabilizes this utility proxy across non-IID clients when only one or two communication rounds may be feasible. Concretely, each client alternates LoRA updates on locally generated synthetic batches with intermittent scorer refresh on its private validation set, so the proxy tracks what improves utility as the model and synthetic distribution drift over rounds. At the federated level, only scorer parameters are exchanged: the server pools and redistributes diverse scorers within each task to reduce client-specific bias and high-variance utility noise, yielding a stable relative preference signal that each client uses to update its on-premise generator online with GRPO. Unlike fixed synthetic or decoupled pipelines, Concordia closes the missing loop by letting validation utility continuously reshape the synthetic distribution seen by the next limited-shot round, while keeping records, validation sets, synthetic tables, generators, and LoRA weights strictly on premise.

We evaluate Concordia on compliance-constrained, limited-shot federated tabular prediction across finance and healthcare. The evaluation is designed to test whether one or two allowable communications can improve downstream utility when clients train only on local synthetic tables. The benchmarks cover extreme class imbalance, including Travel Insurance with positives near 0.01[1], and cross-site medical shift, where VA Long Beach is held out as an unseen evaluation site. (Detrano et al., 1989) Under this restricted setting, naive single-node adaptation often collapses in minority-heavy regimes. Concordia yields its largest gains in exactly these cases: on Travel, MCC (Matthews, 1975) improves from **0.00** for the single-node baseline to **7.07** after one round and **21.78** after two rounds, showing that validation-guided refinement of the local synthetic distribution can recover minority-sensitive signal from otherwise near-degenerate local training (Travel is a single-client stream, so this gain reflects local self-improvement under the scorer-only protocol rather than cross-client pooling). It also improves early-round utility on German Credit (Hofmann, 1994) and LendingClub[2], while the medical benchmark shows non-trivial but heterogeneous transfer to the unseen VA site. To explain these gains, we audit the generators across rounds. The analysis shows that high-utility synthetic samples move closer to real positive neighborhoods and cover hard positives more effectively, suggesting that Concordia does not merely enforce valid table format, but reshapes the synthetic support toward validation-critical regions under non-IID imbalance.

2 Problem Statement

We study *limited-shot cross-silo federated learning* for adapting LLMs to tabular prediction under strict data-isolation rules in regulated deployments. A system consists of N clients and a coordinating server with a small communication budget R, typically one or two rounds, since each round may require institutional approval, secure execution, audit logging, and coordination across independently governed organizations. Client i holds a private dataset $\mathcal{D}_i = \{(x, y)\}$ and a private validation set $\mathcal{V}_i$, neither of which can be shared.

We do not argue that synthetic tables should replace local real-data training whenever unrestricted local supervision suffices; rather, client-local synthetic tables serve as an auditable on-premise training substrate whose distribution can be refined by private validation utility while raw records, synthetic samples, generators, and task-model updates all stay local.

Motivating setting and heterogeneity. In privacy-sensitive domains such as finance and healthcare, institutions differ in schema, feature semantics, decision policies, and label prevalence, while the

[1] https://www.kaggle.com/datasets/mhdzahier/travel-insurance

[2] https://www.kaggle.com/datasets/wordsforthewise/lending-club

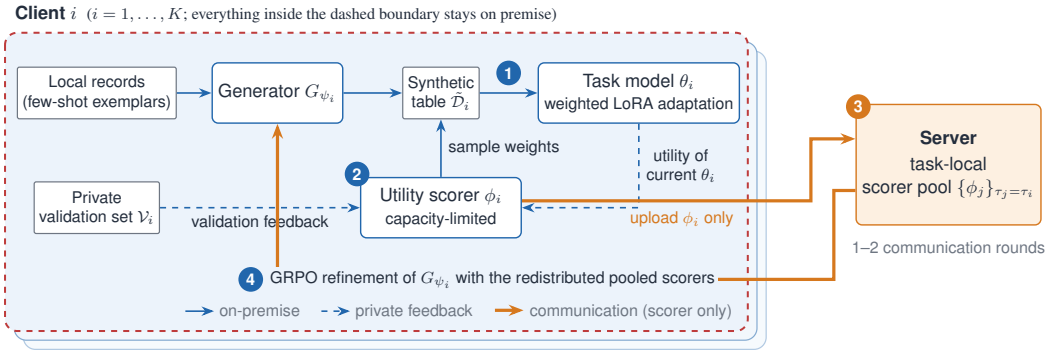

Figure 1: Overview of Concordia. Raw records, private validation sets, synthetic tables, generators, and LoRA updates never leave a client; only the capacity-limited utility scorer is uploaded, pooled within a task, and redistributed. Each round: (1) local LoRA adaptation on scorer-weighted synthetic tables, (2) scorer refresh from private validation feedback, (3) within-task scorer pooling, (4) on-premise generator refinement with GRPO using the pooled scorers.

labels that matter most, such as default, fraud, insurance claims, or adverse clinical outcomes, are often rare. (Rieke et al., 2020) We denote the client-specific positive rate by p_i and allow it to vary widely across clients, including $p_i = 0$. When $p_i = 0$ or p_i is extremely small, naive local learning contains little minority signal and can collapse to majority-only behavior. Accordingly, we emphasize long-tail utility after local adaptation, including worst-client behavior, rather than only average accuracy.

Tabular task and LLM interface. Each example consists of a mixed-type feature vector x and a label $y \in \mathcal{Y}$. Client i serializes x into text with a schema-conditioned template $\mathcal{T}_i$, allowing an LLM predictor to use feature descriptions, schema information, and task instructions. We allow both single-task federation, where clients share the same prediction target and label space, and multi-task federation, where each client has a task identifier τ_i. Federated coordination is performed within each task to avoid ill-posed cross-task aggregation and negative transfer.

Communication and threat model (compliance-constrained FL). We study a federated deployment where institutions cannot export data-dependent high-capacity artifacts. At round r, the server broadcasts a shared backbone $\theta^{(r)}$, while client i uploads only a lightweight utility proxy, namely scorer parameters:

$$m_i^{(r)} = \phi_i^{(r)}. \tag{1}$$

Raw records, validation sets, synthetic tables, generator parameters, and parameter-efficient task-model updates such as LoRA or adapter weights are never transmitted. Thus, unlike standard FL, the server does not aggregate client model updates; it only pools and redistributes utility proxies. We consider adversaries that may infer client-specific records or label properties from communicated artifacts, and view scorer sharing as a reduced-exposure channel rather than a formal privacy guarantee. This communication boundary defines the feasible method class in our main experiments: methods that transmit gradients, full model updates, LoRA deltas, synthetic samples, or generator parameters require a less restrictive export policy and are treated only as policy-violating oracle baselines.

Synthetic training and local utility proxy. Since centralized supervision is prohibited, client i maintains an on-premise generator G_{ψ_i} that produces labeled synthetic examples $\tilde{z} = (\tilde{x}, \tilde{y})$ consistent with its schema and task. Labels are assigned by a client-local interface $\mathcal{A}_i$, such as internal risk policies or clinical coding rules; this interface may be noisy and is never shared. Neither synthetic tables nor generator parameters ψ_i are uploaded.

Given the broadcast backbone $\theta^{(r)}$, client i performs parameter-efficient local adaptation on synthetic mini-batches. A lightweight scorer $S_{\phi_i^{(r)}}$ induces weights $w_{\phi_i^{(r)}}(\tilde{z})$ to prioritize utility-critical, often minority, samples:

$$\Delta\theta_i^* := \arg\min_{\Delta\theta_i} \mathbb{E}_{\tilde{z}\sim G_{\psi_i}} \left[w_{\phi_i}(\tilde{z}) \ell(\tilde{z}; \theta + \Delta\theta_i) \right]. \tag{2}$$

Algorithm 1 Concordia: limited-shot scorer-pooling protocol

Input: clients $\{(\mathcal{D}_i, \mathcal{V}_i, \tau_i, G_{\psi_i^{(0)}})\}_{i=1}^N$, task backbones $\{\theta_\tau\}$, budget R
for $r = 0, \ldots, R-1$ **do**
  Server samples clients $\mathcal{C}_r$ and broadcasts θ_{τ_i} to each $i \in \mathcal{C}_r$
  for each client $i \in \mathcal{C}_r$ in parallel **do**
    Adapt local LoRA $\Delta\theta_i$ on scorer-weighted synthetic batches from $G_{\psi_i^{(r)}}$
    Refresh scorer S_{ϕ_i} using private validation feedback from $\mathcal{V}_i$
    Upload only scorer parameters ϕ_i
  end for
  Server forms within-task pools $\mathcal{S}_r(\tau) = \{S_{\phi_j} : j \in \mathcal{C}_r(\tau)\}$
  for each client $i \in \mathcal{C}_r$ in parallel **do**
    Receive $\mathcal{S}_r(\tau_i)$ and refine $G_{\psi_i^{(r)}}$ with pooled scorer feedback
  end for
end for
Output: client-local generators $\{G_{\psi_i^{(R)}}\}_{i=1}^N$

The scorer is refreshed on premise using private validation feedback from $\mathcal{V}_i$ under the current adapted state, so the weighting signal follows the evolving learner rather than a fixed data-quality proxy.

Objective and tri-level structure. Given a shared backbone θ, our goal is to learn client-local generators $\{\psi_i\}_{i=1}^N$ such that local adaptation on synthetic tables yields high imbalance-aware validation utility under a small communication budget. Conceptually, this can be written as

$$\min_{\{\psi_i\}} \frac{1}{N} \sum_{i=1}^{N} \mathcal{L}_{\text{val}}^{(i)}(\theta + \Delta\theta_i^*(\theta, \psi_i, \phi_i^*(\theta, \psi_i)); \mathcal{V}_i)\,, \tag{3}$$

where $\mathcal{L}_{\text{val}}^{(i)}$ is aligned with imbalanced evaluation, such as $-\text{MCC}$, and ϕ_i^* is the scorer learned on premise from $\mathcal{V}_i$. This objective couples three components: local adaptation on synthetic data, validation-guided scorer learning, and federated scorer pooling for generator refinement. Because validation sets, generators, and adapted model updates are non-exportable, Concordia optimizes this objective through an online surrogate: clients upload only $\phi_i^{(r)}$, the server pools and redistributes these scorers, and each client uses the resulting relative preference signal to update its own generator locally.

3 Proposed Method

3.1 Overview

Concordia is a limited-shot federated adaptation framework that aligns on-premise synthetic tabular generation with private validation utility under restricted communication. Given a small communication budget R, each round couples three operations. First, clients adapt LLM predictors locally with parameter-efficient updates on weighted synthetic mini-batches generated on premise. Second, clients refresh lightweight utility scorers using private validation feedback, so the scorer tracks which synthetic samples are useful for the current learner. Third, the server pools and redistributes only scorer parameters within each task, and each client uses the pooled scorer feedback to refine its own generator. Raw records, validation sets, synthetic tables, generator parameters, and LoRA updates never leave the client. Algorithm 1 gives the high-level protocol; full pseudocode and implementation-level objectives are provided in Appendix B.[3]

[3]The appendix includes the detailed scorer-refresh rule, rank-normalized pooled reward, and clipped GRPO objective used for generator refinement.

3.2 Weighted Local Adaptation on Synthetic Tables

At round r, client i receives the approved task backbone θ_{τ_i} and samples synthetic examples $\tilde{z} = (\tilde{x}, \tilde{y})$ from its local generator $G_{\psi_i^{(r)}}$. For readability, let $G_i^{(r)} := G_{\psi_i^{(r)}}$ and $w_i := w_{\phi_i}$. The client adapts the predictor using a local parameter-efficient update $\Delta\theta_i$, such as LoRA. This update is used only on premise and is never transmitted.

The local adaptation objective is

$$\min_{\Delta\theta_i} \mathbb{E}_{\tilde{z}\sim G_i^{(r)}} \left[w_i(\tilde{z}) \ell(\tilde{z}; \theta_{\tau_i} + \Delta\theta_i) \right], \tag{4}$$

where $\ell(\cdot)$ is the supervised loss on serialized tabular examples, and $w_i(\tilde{z})$ is induced by the local utility scorer.

For a mini-batch $\tilde{B}$, we first compute the clipped raw weight

$$a_i(\tilde{z}) = \mathrm{clip}\left(\frac{S_{\phi_i}(\tilde{z}) + 1}{2}, w_{\min}, w_{\max} \right). \tag{5}$$

We then normalize within the batch:

$$Z_i(\tilde{B}) = \sum_{\tilde{z}' \in \tilde{B}} a_i(\tilde{z}'). \tag{6}$$

$$w_i(\tilde{z}) = \frac{a_i(\tilde{z})}{Z_i(\tilde{B})}. \tag{7}$$

Thus, the scorer affects local adaptation only by reweighting synthetic samples, prioritizing utility-critical and often minority-relevant cases.

3.3 Validation-Aligned Scorer Learning

Each client maintains an exportable utility scorer S_{ϕ_i} that maps a synthetic example to a scalar score. To respect the communication boundary, ϕ_i denotes only the capacity-limited scorer parameters allowed to leave the silo, such as a small scorer head or low-rank scorer module on top of a fixed encoder. Task adapters, generator parameters, synthetic tables, and validation data are never transmitted.

Given a serialized synthetic example $\tilde{z}$, the scorer predicts

$$S_{\phi_i}(\tilde{z}) = \tanh(\mathrm{Head}_{\phi_i}(h(\tilde{z}))) \in [-1, 1], \tag{8}$$

where $h(\tilde{z})$ is a fixed or locally available representation.

Every m local steps, client i evaluates the adapted state on its private validation set:

$$q_i^{(t)} = \mathcal{L}_{\mathrm{val}}^{(i)}(\theta_{\tau_i} + \Delta\theta_i^{(t)}; \mathcal{V}_i), \tag{9}$$

where $\mathcal{L}_{\mathrm{val}}^{(i)}$ is imbalance-aware, such as $1 - \mathrm{MCC}$. The validation-improvement signal is

$$\delta_i^{(t)} = q_i^{(t-m)} - q_i^{(t)}. \tag{10}$$

Because the exact hypergradient would require differentiating through the full local adaptation trajectory, we use a first-order online surrogate. We first define the average scorer value on the recent mini-batch:

$$\bar{S}_{\phi_i}(\tilde{B}_{i,t}) = \frac{1}{|\tilde{B}_{i,t}|} \sum_{\tilde{z} \in \tilde{B}_{i,t}} S_{\phi_i}(\tilde{z}). \tag{11}$$

The scorer objective is

$$\begin{aligned} \mathcal{J}_{\phi_i}^{(t)} = &- \mathrm{sg}(\hat{\delta}_i^{(t)}) \bar{S}_{\phi_i}(\tilde{B}_{i,t}) \\ &+ \lambda_{\mathrm{reg}} \|\phi_i - \phi_i^{(0)}\|_2^2. \end{aligned} \tag{12}$$

Here $\mathrm{sg}(\cdot)$ denotes stop-gradient, and $\hat{\delta}_i^{(t)}$ is a normalized version of $\delta_i^{(t)}$ over recent scorer refreshes on the same client. This surrogate is biased relative to the exact tri-level hypergradient, but it is lightweight, online, and aligned with the limited-shot setting. We restrict scorer capacity and update budget per round to reduce memorization and leakage risk.

3.4 FEDERATED SCORER POOLING

After local adaptation and scorer refresh, client i uploads only ϕ_i. For each task τ, the server forms a within-task scorer pool

$$\mathcal{S}_r(\tau) = \{S_{\phi_j} : j \in \mathcal{C}_r(\tau)\}. \tag{13}$$

The pool is redistributed to clients with the same task identifier. Pooling is task-local rather than cross-task, avoiding ill-posed aggregation across unrelated label spaces. It reduces dependence on a single noisy client signal and provides a stabler relative preference under non-IID distributions, sparse minority labels, and heterogeneous validation feedback.

3.5 CLIENT-LOCAL GENERATOR REFINEMENT

Each client refines its own generator using the pooled scorer set. For a condition c, client i samples a group of G candidates:

$$\{\tilde{x}_{c,1}, \ldots, \tilde{x}_{c,G}\} \sim G_i^{(r)}(\cdot \mid c). \tag{14}$$

For each scorer $S \in \mathcal{S}_i$, scores are normalized within the candidate group:

$$\tilde{s}_{c,j}^{S} = \mathrm{NormGroup}(\{S(\tilde{x}_{c,k})\}_{k=1}^{G})_j. \tag{15}$$

By default, $\mathrm{NormGroup}$ maps within-group ranks to $[-1, 1]$. The pooled preference is

$$\bar{s}_{c,j} = \mathrm{RobustAgg}_{S \in \mathcal{S}_i}(\tilde{s}_{c,j}^{S}), \tag{16}$$

where $\mathrm{RobustAgg}$ is the mean by default.

Each client also computes a deterministic format reward $R_{\mathrm{fmt}}(c, \tilde{x}) \in [-1, 1]$. The total reward is

$$r_{c,j} = \tanh(\beta \bar{s}_{c,j}) + \lambda_{\mathrm{fmt}} R_{\mathrm{fmt}}(c, \tilde{x}_{c,j}), \tag{17}$$

and the group-relative advantage is

$$A_{c,j} = r_{c,j} - \frac{1}{G} \sum_{k=1}^{G} r_{c,k}. \tag{18}$$

The client then updates $G_i^{(r)}$ with a clipped GRPO objective using $A_{c,j}$ as the group-relative preference signal. For compactness, the main text denotes this objective as

$$\mathcal{J}_{\mathrm{GRPO}} = -\mathbb{E}_{c,j}\left[\min(\rho_{c,j} A_{c,j}, \bar{\rho}_{c,j} A_{c,j})\right] + \lambda_{\mathrm{KL}} \mathcal{D}_{\mathrm{KL}}, \tag{19}$$

where $\rho_{c,j}$ is the likelihood ratio between the updated and round-start generators, $\bar{\rho}_{c,j}$ is its clipped version, and $\mathcal{D}_{\mathrm{KL}}$ denotes the conditional KL to the round-start generator. The update is performed entirely on premise, and the server never observes generated tables or generator parameters.

4 EXPERIMENTS

We evaluate Concordia under the limited-shot, compliance-constrained setting defined in Section 2. The experiments are designed to answer three questions. First, can scorer-only communication improve tabular LLM adaptation when clients train only on on-premise synthetic tables? Second, are the gains strongest in long-tail regimes where local learning tends to collapse? Third, does validation-aligned scorer feedback change the support of synthetic data rather than merely improving surface validity?

Our benchmarks cover two regulated cross-silo regimes. The finance regime stresses extreme class imbalance and heterogeneous financial objectives, including Travel Insurance with a positive rate near 0.01. The healthcare regime stresses cross-site shift: Cleveland, Hungarian, and Switzerland participate in limited-shot coordination, while VA Long Beach is held out as an unseen evaluation site. (Detrano et al., 1989)

Dataset	Split (Tr/Val/Te)	Features	Pos. Ratio
Finance			
German Credit	700 / 100 / 200	21	0.29
Lending Club	9,417 / 1,345 / 2,691	21	0.20
Travel Insurance	8,865 / 1,266 / 2,534	9	0.015
Healthcare			
Cleveland	231 / 41 / 31	13	0.45
Hungarian	224 / 40 / 30	13	0.35
Switzerland	93 / 17 / 13	13	0.94
VA Long Beach	153 / 27 / 20	13	0.72

Table 1: Summary of tabular datasets used in our experiments. **Pos. Ratio** is the fraction of positive-class records in the training split under a single convention: for Heart Disease the positive class is always `heartdisease` (`num`> 0), for German Credit `bad`, for LendingClub `charged off`, and for Travel Insurance `claim`.

4.1 Experimental Settings

Datasets and clients. Table 1 summarizes the datasets and splits. In healthcare, each Heart Disease site is treated as one client under a shared prediction task. Cleveland, Hungarian, and Switzerland participate in limited-shot coordination, while VA Long Beach is held out and used only for unseen-site evaluation. (Detrano et al., 1989) In finance, German Credit, Lending Club, and Travel Insurance are treated as separate task streams because their feature spaces and label semantics correspond to different financial objectives. This design avoids ill-posed cross-task model aggregation. Because each finance dataset therefore forms a singleton task stream, the finance experiments evaluate *local* validation-guided generator refinement under the scorer-only protocol and do not involve cross-client scorer pooling; the healthcare experiments, where Cleveland, Hungarian, and Switzerland share one task identifier (N=3 clients per pool), are the setting in which multi-client scorer pooling is evaluated.

Limited-shot protocol. We simulate $R = 2$ communication rounds, matching the limited-shot regime targeted by Concordia. At each round, the server broadcasts an approved task backbone. Each client performs on-premise LoRA adaptation on synthetic tables generated by its local generator, refreshes a capacity-limited utility scorer using private validation feedback, and uploads only the scorer parameters. The server forms task-local scorer pools and redistributes them to clients, after which each client refines its own generator with GRPO. No raw records, validation sets, synthetic tables, generators, or LoRA updates are transmitted.

We emphasize that $R = 2$ is not used as an asymptotic convergence benchmark. It represents a restricted coordination budget in which each round may require governance approval, secure execution, audit logging, and cross-institution coordination. Accordingly, our primary focus is early-round utility under the allowed communication boundary.

Models and initialization. All clients initialize their generators from the same pretrained instruction-following checkpoint (Qwen3-4B-Instruct); no additional public instruction-tuning stage is used, and schema adherence is enforced by the generation prompt and the FormatReward parser (Appendix C.4). After initialization, generators are maintained and updated independently on each client. The downstream predictor uses the same backbone capacity across all methods and is adapted with the same LoRA configuration. Unless stated otherwise, all methods use the same synthetic sampling budget per round and comparable local compute.

Metrics. We report Matthews Correlation Coefficient (MCC) (Matthews, 1975), which is robust to class imbalance, as mean $\pm$ standard deviation over multiple seeds. Because strict deployments often limit communication rounds, we report one- and two-round performance directly and analyze round-to-round behavior rather than claiming long-horizon convergence.

Healthcare	Cleveland	Hungarian	Switzerland	VA
Single-node backbones				
Qwen3-4B	60.25±14.76	23.57±19.71	12.31±9.18	27.50±0.00
Qwen3-4B-Instruct	32.28±17.36	26.65±14.76	0.00±0.00	−0.10±0.00
Concordia after 1 scorer-pooling round				
Cleveland	74.69±11.70	30.86±18.78	−23.35±12.53	33.61±25.02
Hungarian	46.28±16.52	40.00±18.03	−3.29±27.31	72.76±11.16
Switzerland	0.00±0.00	−9.78±18.08	10.11±28.21	42.01±4.63
Concordia after 2 scorer-pooling rounds				
Cleveland	82.29±8.97	0.00±0.00	0.00±0.00	14.00±7.77
Hungarian	67.22±13.30	46.29±17.72	27.25±32.79	34.30±13.49
Switzerland	38.52±13.11	0.00±18.14	67.70±40.37	−9.64±7.21

Finance	German	LendingClub	Travel
Single-node backbones			
Qwen3-4B	7.93±5.48	−0.94±0.59	0.00±0.00
Qwen3-4B-Instruct	5.17±6.87	9.27±8.13	−2.85±2.86
Concordia after 1 scorer-pooling round			
German	30.44±7.06	0.00±0.00	0.00±0.00
LendingClub	0.00±0.00	79.83±1.49	0.00±0.00
Travel	0.00±0.00	0.00±0.00	7.07±1.85
Concordia after 2 scorer-pooling rounds			
German	32.93±6.00	0.00±0.00	0.00±0.00
LendingClub	0.00±0.00	90.22±1.05	0.00±0.00
Travel	0.00±0.00	0.00±0.00	21.78±5.84

Table 2: Limited-shot performance under restricted communication. Rows under Concordia report performance after one or two scorer-pooling rounds; each row is the model of the named client, and columns are evaluation datasets. Entries of exactly 0.00±0.00 correspond to single-class predictions (MCC is 0 when a model predicts only one class). Results are MCC mean ± standard deviation.

Baselines. We separate baselines into *feasible baselines* and *policy-violating reference baselines*. Feasible baselines obey the same export boundary as Concordia. **Single-node** evaluates the pretrained backbone **Qwen3-4B** (Yang et al., 2025) and its instruction-tuned variant **Qwen3-4B-Instruct** (Yang et al., 2025) without cross-client coordination. **Round 1/2** report Concordia after one and two rounds of scorer pooling and redistribution.

We also evaluate update-sharing FL methods such as FedLoRA as *policy-violating reference baselines*. These methods transmit task-model updates and therefore are not feasible under our export policy. Local supervised learners (e.g., gradient-boosted trees), imbalance-aware local training (oversampling, class-balanced or focal loss), and static tabular synthesizers (e.g., CTGAN, TVAE) act on a single client's own data and provide no signal that can cross the export boundary, so they are outside this comparison. They are included only to test whether relaxing the communication boundary to standard update aggregation is sufficient. In our setting, FedLoRA collapses to near-zero MCC on the studied tasks, indicating that update sharing alone does not resolve the combined effects of non-IID clients, extreme imbalance, and synthetic-training mismatch.

4.2 Main Results: Early Utility under Restricted Communication

Long-tail gains are largest when local learning collapses. The clearest gains occur in the most imbalanced setting. On Travel Insurance ($p \approx 0.01$), the single-node Qwen3-4B baseline obtains 0.00±0.00 MCC. After one scorer-pooling round, Concordia reaches 7.07±1.85; after two rounds, it reaches 21.78±5.84. This pattern suggests that the first round recovers a weak minority-sensitive signal, while the second round benefits from a generator that has been refined toward more informative synthetic regions.

Finance isolates local scorer-guided refinement. In the finance regime, each dataset is a singleton task stream, so the reported improvements measure local validation-guided generator refinement

under the scorer-only protocol rather than federated collaboration; they cannot be attributed to cross-client pooling or to cross-task model-update aggregation. German Credit improves from 7.93 ± 5.48 to 32.93 ± 6.00, and LendingClub improves from -0.94 ± 0.59 to 90.22 ± 1.05 after two rounds. The off-task entries remain near zero, which supports our decision not to merge task-model updates across heterogeneous financial objectives. Moreover, FedLoRA also collapses to near-zero MCC despite operating under a less restrictive policy that permits transmitting LoRA updates. This indicates that standard update sharing alone is not sufficient; the useful signal in this setting is not only how to aggregate task-model updates, but how to reshape the synthetic data that each client trains on.

Medical transfer is useful but heterogeneous. In healthcare, participating sites show clear within-site improvements, such as Cleveland from 60.25 ± 14.76 to 82.29 ± 8.97 and Hungarian from 23.57 ± 19.71 to 46.29 ± 17.72. The held-out VA site shows asymmetric transfer: the best one-round transfer reaches 72.76 ± 11.16, while some two-round refinements regress under site shift. We therefore do not claim uniform out-of-site robustness. Instead, these results indicate that limited-shot synthetic refinement can improve local utility while exposing a trade-off between in-site fitting and unseen-site generalization.

Reliability signals are visible even in a two-shot budget. While we do not claim long-horizon convergence from two rounds, several tasks show reduced seed sensitivity alongside higher mean utility, such as Cleveland where the standard deviation drops from 14.76 to 8.97, and LendingClub where it drops from 1.49 at Round 1 to 1.05 at Round 2. Conversely, variance can increase when minority signal is first recovered, as in Travel. This is expected when performance becomes sensitive to rare-event coverage, and motivates the generator audit below.

4.3 Generator Audit: What Does Scorer Feedback Change?

We audit generator snapshots across rounds under a controlled paired-generation protocol; the protocol and the paired examples are given in Appendix D.

Takeaway. Across finance and healthcare, scorer-guided refinement changes the support of synthetic training data rather than merely improving formatting. In finance, it preserves dominant-mode plausibility while maintaining functional feature dependencies. In healthcare, it emphasizes multi-feature risk conjunctions rather than single-feature shortcuts. These observations are consistent with the quantitative gains under non-IID imbalance.

5 Conclusion

We introduced Concordia, a limited-shot federated adaptation framework for LLM-based tabular prediction under strict communication constraints. In our setting, raw records, validation sets, synthetic tables, generators, and downstream LoRA updates remain on premise; only capacity-limited utility scorers are exchanged. Concordia turns private validation feedback into an online signal for refining client-local synthetic generation: each client learns a utility scorer from its private validation set, uses it to reweight synthetic batches for local LoRA adaptation, and refines its own generator with pooled scorer feedback. Across finance and healthcare, Concordia improves early-round MCC under severe heterogeneity and extreme imbalance, with its clearest gains on the long-tail Travel task where positives are near 0.01. Generator audits further suggest that scorer-guided refinement changes the support of synthetic training data, preserving plausible feature dependencies in finance and emphasizing multi-feature risk conjunctions in healthcare.

Limitations. Concordia is designed for limited-shot deployment, not for proving long-horizon convergence. Although one- and two-round evaluation is appropriate for regulated cross-silo settings where each round can be costly, it does not establish asymptotic stability. The medical results also show heterogeneous transfer to the unseen VA site, indicating that scorer-guided refinement can trade off in-site fitting and out-of-site robustness under distribution shift. In addition, the scorer update is a first-order online surrogate rather than an exact tri-level hypergradient, and its bias is controlled empirically rather than theoretically.

ETHICS STATEMENT

Concordia reduces direct exposure by keeping raw data, private validation sets, synthetic tables, generators, and LoRA updates on premise, but it is not a formal privacy guarantee. Shared scorer parameters may still leak aggregate information about client distributions or label properties through membership inference, property inference, or model inversion attacks. The framework also assumes access to a client-local labeling interface for synthetic tables; noisy or biased local rules may affect generator refinement and should be audited in deployment. Synthetic generators may also memorize local patterns or amplify historical biases, especially in rare-event domains such as credit, insurance, and clinical risk prediction. We therefore view Concordia as a utility-optimization framework under restricted communication, not as a complete privacy or fairness solution; before deployment, shared scorers should be subjected to empirical leakage tests such as membership- and property-inference probes. Responsible deployment should include access control, secure aggregation where appropriate, clipping or noising of transmitted scorer updates when formal privacy is required, subgroup fairness audits, documentation of intended use, and monitoring for harmful risk scoring against vulnerable groups.

REPRODUCIBILITY STATEMENT

The protocol, objectives and reward construction are specified in Section 3 and Appendix B.3; models, LoRA and GRPO hyper-parameters, seeds and the evaluation protocol are given in Appendix C.1–C.7; prompts and tabular serialization in Appendix E; and the released artifacts are listed in Appendix C.8.

AI USE STATEMENT

We used a generative AI assistant (Claude Code) only for polishing the writing, formatting the manuscript to the conference template, and cross-checking bibliographic entries against arXiv, ACL Anthology, Crossref and DBLP. It was not used to generate research ideas, to write the technical content of the method, to implement the core code, or to produce, alter or analyze the reported results, and it was not used for data labeling or for reviewing other submissions. The generator and task models studied in the paper (Qwen3-4B and Qwen3-4B-Instruct) are part of the method itself and are described in Section 4.1. All AI-assisted polishing was checked by the authors, and we take full responsibility for the content of this paper, including text, claims, code and results.

REFERENCES

Fatima Abacha, Sin G. Teo, Lucas C. Cordeiro, and Mustafa A. Mustafa. Synthetic data aided federated learning using foundation models, 2024. URL https://arxiv.org/abs/2407.05174.

Opeoluwa Akinseloyin, Xiaorui Jiang, and Vasile Palade. Weakly supervised active learning for abstract screening leveraging LLM-based pseudo-labeling. medRxiv preprint, 2025. URL https://www.medrxiv.org/content/10.1101/2025.08.24.25334314v1.

Sara Babakniya, Ahmed Roushdy Elkordy, Yahya H. Ezzeldin, Qingfeng Liu, Kee-Bong Song, Mostafa El-Khamy, and Salman Avestimehr. Slora: Federated parameter efficient fine-tuning of language models. *ArXiv*, abs/2308.06522, 2023. URL https://api.semanticscholar.org/CorpusID:260887495.

Vadim Borisov, Kathrin Seßler, Tobias Leemann, Martin Pawelczyk, and Gjergji Kasneci. Language models are realistic tabular data generators. *arXiv preprint arXiv:2210.06280*, 2022.

Yujun Cheng, Weiting Zhang, Zhewei Zhang, Chuan Zhang, Shengjin Wang, and Shiwen Mao. Toward federated large language models: Motivations, methods, and future directions. *IEEE Communications Surveys & Tutorials*, 27:2733–2764, 2025. URL https://api.semanticscholar.org/CorpusID:274196624.

Lynn Chua, Badih Ghazi, Yangsibo Huang, Pritish Kamath, Ravi Kumar, Daogao Liu, Pasin Manurangsi, Amer Sinha, and Chiyuan Zhang. Mind the privacy unit! user-level differential privacy for language model fine-tuning. *arXiv preprint arXiv:2406.14322*, 2024. URL https://arxiv.org/abs/2406.14322.

Liam Collins, Hamed Hassani, Aryan Mokhtari, and Sanjay Shakkottai. Exploiting shared representations for personalized federated learning. *ArXiv*, abs/2102.07078, 2021. URL https://api.semanticscholar.org/CorpusID:231924497.

Robert C. Detrano, András Jánosi, Walter Steinbrunn, Matthias Emil Pfisterer, Johann-Jakob Schmid, Sarbjit Sandhu, Kern H Guppy, Stella Lee, and Victor Froelicher. International application of a new probability algorithm for the diagnosis of coronary artery disease. *The American journal of cardiology*, 64 5:304–10, 1989. URL https://api.semanticscholar.org/CorpusID:23545303.

Tao Fan, Yan Kang, Guoqiang Ma, Weijing Chen, Wenbin Wei, Lixin Fan, and Qiang Yang. Fate-llm: A industrial grade federated learning framework for large language models. *ArXiv*, abs/2310.10049, 2023. URL https://api.semanticscholar.org/CorpusID:264145987.

Jack Goetz and Ambuj Tewari. Federated learning via synthetic data, 2020. URL https://arxiv.org/abs/2008.04489.

Tao Guo, Junxiao Wang, Fushuo Huo, Laizhong Cui, Song Guo, Jie Gui, and Dacheng Tao. On the evolution of federated post-training large language models: A model accessibility view. *arXiv preprint arXiv:2508.16261*, 2025a. URL https://arxiv.org/abs/2508.16261.

Wenkai Guo, Xuefeng Liu, Haolin Wang, Jianwei Niu, Shaojie Tang, and Jing Yuan. Can federated learning safeguard private data in LLM training? vulnerabilities, attacks, and defense evaluation. In *Findings of the Association for Computational Linguistics: EMNLP 2025*, pp. 23986–24013. Association for Computational Linguistics, 2025b. doi: 10.18653/v1/2025.findings-emnlp.1303. URL https://aclanthology.org/2025.findings-emnlp.1303/.

Hans Hofmann. Statlog (German Credit Data). UCI Machine Learning Repository, 1994. DOI: https://doi.org/10.24432/C5NC77.

Edward J. Hu, Yelong Shen, Phillip Wallis, Zeyuan Allen-Zhu, Yuanzhi Li, Shean Wang, Lu Wang, and Weizhu Chen. LoRA: Low-rank adaptation of large language models. *arXiv preprint arXiv:2106.09685*, 2021. URL https://arxiv.org/abs/2106.09685.

Shengyuan Hu, Jack Goetz, Kshitiz Malik, Hongyuan Zhan, Zhe Liu, and Yue Liu. Fedsynth: Gradient compression via synthetic data in federated learning, 2022. URL https://arxiv.org/abs/2204.01273.

Chao Huang, Jianwei Huang, and Xin Liu. Cross-silo federated learning: Challenges and opportunities. *arXiv preprint arXiv:2206.12949*, 2022. URL https://arxiv.org/abs/2206.12949.

Myeongkyun Kang, Philip Chikontwe, Soopil Kim, Kyong Hwan Jin, Ehsan Adeli, Kilian M. Pohl, and Sang Hyun Park. One-shot federated learning on medical data using knowledge distillation with image synthesis and client model adaptation. In *Medical Image Computing and Computer Assisted Intervention – MICCAI 2023*, volume 14221 of *Lecture Notes in Computer Science*, pp. 521–531. Springer, 2023. doi: 10.1007/978-3-031-43895-0_49.

Akim Kotelnikov, Dmitry Baranchuk, Ivan Rubachev, and Artem Babenko. Tabddpm: Modelling tabular data with diffusion models. In *International Conference on Machine Learning*, pp. 17564–17579. PMLR, 2023.

Qinbin Li, Bingsheng He, and Dawn Song. Practical one-shot federated learning for cross-silo setting. In Zhi-Hua Zhou (ed.), *Proceedings of the Thirtieth International Joint Conference on Artificial Intelligence, IJCAI-21*, pp. 1484–1490. International Joint Conferences on Artificial Intelligence Organization, August 2021. doi: 10.24963/ijcai.2021/205. URL https://doi.org/10.24963/ijcai.2021/205. Main Track.

Eugenio Lomurno and Matteo Matteucci. Federated knowledge recycling: Privacy-preserving synthetic data sharing, 2024. URL https://arxiv.org/abs/2407.20830.

Brian W. Matthews. Comparison of the predicted and observed secondary structure of t4 phage lysozyme. *Biochimica et biophysica acta*, 405 2:442–51, 1975. URL https://api.semanticscholar.org/CorpusID:44596673.

Jaehyun Nam, Kyuyoung Kim, Seunghyuk Oh, Jihoon Tack, Jaehyung Kim, and Jinwoo Shin. Optimized feature generation for tabular data via llms with decision tree reasoning. *ArXiv*, abs/2406.08527, 2024. URL https://api.semanticscholar.org/CorpusID:270440964.

Mengye Ren, Wenyuan Zeng, Bin Yang, and Raquel Urtasun. Learning to reweight examples for robust deep learning. In *International Conference on Machine Learning*, 2018. URL https://api.semanticscholar.org/CorpusID:4321928.

Nicola Rieke, Jonny Hancox, Wenqi Li, Fausto Milletarì, Holger R. Roth, Shadi Albarqouni, Spyridon Bakas, Mathieu Galtier, Bennett A. Landman, Klaus H. Maier-Hein, Sébastien Ourselin, Micah J. Sheller, Ronald M. Summers, Andrew Trask, Daguang Xu, Maximilian Baust, and Manuel Jorge Cardoso. The future of digital health with federated learning. *NPJ Digital Medicine*, 3, 2020. URL https://api.semanticscholar.org/CorpusID:212747909.

Maximilian Schmidhuber and Udo Kruschwitz. LLM-based synthetic datasets: Applications and limitations in toxicity detection. In *Proceedings of the Fourth Workshop on Threat, Aggression & Cyberbullying @ LREC-COLING-2024*, pp. 37–51, Torino, Italia, 2024. URL https://aclanthology.org/2024.trac-1.6/.

Aivin V Solatorio and Olivier Dupriez. Realtabformer: Generating realistic relational and tabular data using transformers. *arXiv preprint arXiv:2302.02041*, 2023.

Josefa Lia Stoisser, Marc Boubnovski Martell, Kaspar Märtens, Lawrence Phillips, Stephen M. Town, Rory M. Donovan-Maiye, and Julien Fauqueur. Query, don't train: Privacy-preserving tabular prediction from ehr data via sql queries. *ArXiv*, abs/2505.21801, 2025. URL https://api.semanticscholar.org/CorpusID:278960066.

Alysa Ziying Tan, Han Yu, Lizhen Cui, and Qiang Yang. Towards personalized federated learning. *IEEE Transactions on Neural Networks and Learning Systems*, 34(12):9587–9603, 2023. doi: 10.1109/TNNLS.2022.3160699.

Yuxin Wang, Duanyu Feng, Yongfu Dai, Zhengyu Chen, Jimin Huang, Sophia Ananiadou, Qianqian Xie, and Hao Wang. Harmonic: Harnessing llms for tabular data synthesis and privacy protection. *arXiv preprint arXiv:2408.02927*, 2024.

Zezhou Wang, Yaxin Du, Xingjun Ma, Yu-Gang Jiang, Zhuzhong Qian, and Siheng Chen. Optimizing cross-client domain coverage for federated instruction tuning of large language models. In *Findings of the Association for Computational Linguistics: EMNLP 2025*, pp. 991–1011. Association for Computational Linguistics, 2025. doi: 10.18653/v1/2025.findings-emnlp.52. URL https://aclanthology.org/2025.findings-emnlp.52/.

Futian Weng, Miao Zhu, Mike Buckle, Petr Hajek, and Mohammad Zoynul Abedin. Class imbalance Bayesian model averaging for consumer loan default prediction: The role of soft credit information. *Research in International Business and Finance*, 74:102722, 2025. doi: 10.1016/j.ribaf.2024.102722.

Feijie Wu, Xiaoze Liu, Haoyu Wang, Xingchen Wang, Lu Su, and Jing Gao. Towards federated rlhf with aggregated client preference for llms. In *International Conference on Learning Representations*, 2025. URL https://openreview.net/forum?id=mqNKiEB6pd.

Bangzhou Xin, Yangyang Geng, Teng Hu, Sheng Chen, Wei Yang, Shaowei Wang, and Liusheng Huang. Federated synthetic data generation with differential privacy. *Neurocomputing*, 468:1–10, 2022. doi: 10.1016/J.NEUCOM.2021.10.027. URL https://doi.org/10.1016/j.neucom.2021.10.027.

Lei Xu, Maria Skoularidou, Alfredo Cuesta-Infante, and Kalyan Veeramachaneni. Modeling tabular data using conditional gan. *Advances in neural information processing systems*, 32, 2019.

Biwei Yan, Kun Li, Minghui Xu, Yueyan Dong, Yue Zhang, Zhaochun Ren, and Xiuzhen Cheng. On protecting the data privacy of large language models (LLMs): A survey. *arXiv preprint arXiv:2403.05156*, 2024.

Jiahuan Yan, Jintai Chen, Chaowen Hu, Bo Zheng, Yaojun Hu, Jimeng Sun, and Jian Wu. Small models are LLM knowledge triggers for medical tabular prediction. In *International Conference on Learning Representations*, 2025. URL https://openreview.net/forum?id=WoPovNkM5h.

An Yang, Anfeng Li, Baosong Yang, Beichen Zhang, Binyuan Hui, Bo Zheng, Bowen Yu, Chang Gao, Chengen Huang, Chenxu Lv, et al. Qwen3 technical report, 2025. URL https://arxiv.org/abs/2505.09388.

Rui Ye, Rui Ge, Xinyu Zhu, Jingyi Chai, Yaxin Du, Yang Liu, Yanfeng Wang, and Siheng Chen. Fedllm-bench: Realistic benchmarks for federated learning of large language models. *ArXiv*, abs/2406.04845, 2024a. URL https://api.semanticscholar.org/CorpusID:270357469.

Rui Ye, Wenhao Wang, Jingyi Chai, Dihan Li, Zexi Li, Yinda Xu, Yaxin Du, Yanfeng Wang, and Siheng Chen. Openfedllm: Training large language models on decentralized private data via federated learning. *Proceedings of the 30th ACM SIGKDD Conference on Knowledge Discovery and Data Mining*, 2024b. URL https://api.semanticscholar.org/CorpusID:267627968.

Liping Yi, Han Yu, Gang Wang, Xiaoguang Liu, and Xiaoxiao Li. pFedLoRA: Model-heterogeneous personalized federated learning with LoRA tuning. *arXiv preprint arXiv:2310.13283*, 2023. URL https://arxiv.org/abs/2310.13283.

Zilong Zhao, Aditya Kunar, Robert Birke, and Lydia Y Chen. Ctab-gan: Effective table data synthesizing. In *Asian Conference on Machine Learning*, pp. 97–112. PMLR, 2021.

Gongxi Zhu, Hanlin Gu, Lixin Fan, Qiang Yang, and Yuxing Han. Fedgrpo: Privately optimizing foundation models with group-relative rewards from domain clients. In *Proceedings of the AAAI Conference on Artificial Intelligence*, 2026. doi: 10.1609/aaai.v40i24.39027. URL https://doi.org/10.1609/aaai.v40i24.39027.

A Related Work

A.1 LLM-based tabular learning and data synthesis.

Classical tabular synthesis has been studied with GANs, VAEs, and diffusion models. (Zhao et al., 2021; Xu et al., 2019; Kotelnikov et al., 2023) These methods can model mixed-type tabular distributions, but they often depend on rigid encodings and can struggle when schemas are high-dimensional, semantically rich, or inconsistent across institutions. Recent work uses language models to serialize tables, condition generation on schema descriptions, and produce samples with stronger semantic consistency. (Borisov et al., 2022; Solatorio & Dupriez, 2023; Schmidhuber & Kruschwitz, 2024) This direction is appealing for regulated tabular tasks because feature definitions, policy rules, and label criteria are naturally expressed in language. However, plausible synthetic tables are not necessarily useful training examples. LLM-based synthesis may leak memorized information (Yan et al., 2024), and most generation objectives remain disconnected from the downstream learner whose validation utility changes over time. Concordia builds on LLM-based tabular synthesis, but treats generation as an online decision variable: synthetic tables are refined toward private validation utility rather than optimized only for surface plausibility or schema validity.

A.2 Federated adaptation and personalization under heterogeneity.

Federated personalization addresses non-IID clients by learning client-specific models, partially shared representations, or personalized heads, reducing the bias of a single global model toward dominant clients. (Tan et al., 2023; Collins et al., 2021) Recent parameter-efficient FL methods extend this idea to LLM adaptation by communicating adapters, LoRA deltas, or other lightweight task-model updates. (Yi et al., 2023; Babakniya et al., 2023; Fan et al., 2023; Cheng et al., 2025; Ye et al., 2024b;a; Guo et al., 2025b; Wang et al., 2025) These methods are natural baselines when model updates are exportable. Our setting imposes a stricter communication boundary: downstream LoRA/adaptor updates, generator parameters, real tables, synthetic tables, and validation sets remain on premise. Therefore, the bottleneck is not only how to aggregate model updates, but how to obtain a utility-relevant training signal when such updates cannot be transmitted and rare labels may be absent or severely underrepresented on some clients. Concordia addresses this by federating lightweight validation-derived scorers rather than task-model updates.

A.3 Federated learning with synthetic data under restricted communication.

Synthetic data has been used in federated learning to reduce raw-data exposure, lower communication cost, and improve training efficiency, either by generating local surrogate data or by sharing synthetic samples under privacy constraints. (Lomurno & Matteucci, 2024; Abacha et al., 2024; Goetz & Tewari, 2020; Hu et al., 2022; Xin et al., 2022) This line of work is closely related to our motivation, but many methods still rely on at least one artifact that is unavailable in our setting, such as shareable synthetic samples, exchangeable generators, or model updates. In addition, generation is often fixed before training or guided by proxy distributional objectives. Under non-IID clients and extreme imbalance, such proxy improvements may not translate into better validation utility for the evolving local learner. Concordia instead keeps generators and synthetic tables local, and uses redistributed utility scorers as the only cross-client signal for online generator refinement.

A.4 Federated LLM post-training and preference optimization.

A growing body of work studies federated post-training for LLMs, including federated instruction tuning, federated RLHF, and GRPO-style optimization from distributed rewards or preferences. (Guo et al., 2025a; Wu et al., 2025; Zhu et al., 2026) These methods show that client-side feedback can guide LLM optimization without centralizing all preference data. However, their primary goal is usually to improve a shared model or policy through distributed preference or reward signals. Concordia differs in both object and boundary. The object being improved is not a shared downstream LLM policy, but each client's on-premise synthetic table generator. The communication boundary is also stricter: task-model updates and generator parameters are non-exportable, and only a capacity-limited scorer distilled from private validation feedback is communicated. Thus, Concordia can be viewed as federated preference coordination for synthetic-data refinement rather than federated post-training of the downstream LLM itself.

A.5 Limited-shot and communication-efficient cross-silo FL.

Cross-silo FL in healthcare, finance, and other regulated institutional settings often involves a small number of independently governed organizations, where each communication round can require approval, secure execution, audit logging, and operational coordination. This has motivated one-shot and communication-efficient FL, which studies whether useful collaboration can be achieved with very few rounds rather than relying on long-horizon iterative optimization. (Li et al., 2021; Kang et al., 2023; Huang et al., 2022) Our work follows this deployment-oriented view. The one-/two-round setting is not used as an asymptotic convergence benchmark; it is the target regime. Concordia asks whether limited communication can still improve downstream utility when the usual objects of federation, including data, synthetic tables, generators, and task-model updates, are non-exportable.

B Additional Details for Concordia

This appendix provides full implementation details for Concordia. All notation follows Section 3. For readability, we use the shorthand $G_i^{(r)} := G_{\psi_i^{(r)}}$ and $w_i := w_{\phi_i}$. Throughout the appendix, we follow the same communication boundary as the main paper: raw records, private validation sets, synthetic tables, generator parameters, and downstream LoRA/adaptor updates remain on premise. Only capacity-limited scorer parameters are uploaded.

B.1 Full Protocol

B.2 Scorer Refresh Objective

The exact hypergradient of validation utility with respect to scorer parameters would require differentiating through the full local adaptation trajectory. We instead use an online first-order surrogate. At refresh step t, client i computes

$$q_i^{(t)} = \mathcal{L}_{\text{val}}^{(i)}(\theta_{\tau_i} + \Delta\theta_i^{(t)}; \mathcal{V}_i). \tag{20}$$

Algorithm 2 Full Concordia protocol

Input: clients $\{(\mathcal{D}_i, \mathcal{V}_i, \tau_i, G_{\psi_i^{(0)}})\}_{i=1}^N$, task backbones $\{\theta_\tau\}$, budget R
Hyperparameters: local steps T, refresh interval m, group size G, temperature β, format weight λ_{fmt}, clip range ϵ
for $r = 0, \ldots, R-1$ **do**
  Server samples clients $\mathcal{C}_r$, partitions them into $\{\mathcal{C}_r(\tau)\}$, and broadcasts θ_{τ_i}.
  for each client $i \in \mathcal{C}_r$ in parallel **do**
    Initialize local adapter $\Delta\theta_i^{(0)} \leftarrow 0$ and compute initial validation loss $q_i^{(0)}$.
    for $t = 1, \ldots, T$ **do**
      Sample synthetic mini-batch $\tilde{B}_{i,t} \sim G_i^{(r)}$.
      Update local adapter on scorer-weighted synthetic loss using Eqs. (4)–(7).
      if $t \bmod m = 0$ **then**
        Compute private validation loss $q_i^{(t)}$ and improvement $\delta_i^{(t)}$ using Eqs. (9)–(10).
        Refresh scorer ϕ_i with the surrogate objective in Eq. (12).
      end if
    end for
    Upload only scorer parameters ϕ_i.
  end for
  Server forms task-local scorer pools $\mathcal{S}_r(\tau) = \{S_{\phi_j} : j \in \mathcal{C}_r(\tau)\}$ and redistributes them.
  for each client $i \in \mathcal{C}_r$ in parallel **do**
    Set $\mathcal{S}_i \leftarrow \mathcal{S}_r(\tau_i) \cup \{S_{\phi_i}\}$.
    Sample candidate groups from $G_i^{(r)}$ and compute pooled rewards/advantages using Eqs. (27)–(29).
    Update local generator with the GRPO objective in Eq. (33) to obtain $G_i^{(r+1)}$.
  end for
end for
Return: client-local generators $\{G_i^{(R)}\}_{i=1}^N$ and scorer pool dynamics.

The local validation-improvement signal is

$$\delta_i^{(t)} = q_i^{(t-m)} - q_i^{(t)}. \tag{21}$$

We normalize recent improvement signals on the same client:

$$\hat{\delta}_i^{(t)} = \frac{\delta_i^{(t)} - \mu_i}{\sigma_i + \epsilon_\delta}, \tag{22}$$

where μ_i and σ_i are running statistics over recent scorer refreshes.

For compactness, define the batch-average scorer value

$$\bar{S}_{\phi_i}(\tilde{B}_{i,t}) = \frac{1}{|\tilde{B}_{i,t}|} \sum_{\tilde{z} \in \tilde{B}_{i,t}} S_{\phi_i}(\tilde{z}). \tag{23}$$

The scorer objective is

$$\begin{aligned} \mathcal{J}_{\phi_i}^{(t)} = &- \operatorname{sg}(\hat{\delta}_i^{(t)}) \bar{S}_{\phi_i}(\tilde{B}_{i,t}) \\ &+ \lambda_{\text{reg}} \|\phi_i - \phi_i^{(0)}\|_2^2. \end{aligned} \tag{24}$$

Here $\operatorname{sg}(\cdot)$ denotes stop-gradient. If recent adaptation improves validation utility, the scorer is encouraged to assign higher scores to the recent synthetic batch. If validation utility decreases, similar samples receive lower scores in future weighting and generator refinement. This surrogate is biased relative to the exact tri-level hypergradient, but avoids unrolling the local adaptation trajectory and is compatible with limited-shot communication.

B.3 Pooled Preference and Reward Construction

For condition c, client i samples a group of G candidates:

$$\{\tilde{x}_{c,1}, \ldots, \tilde{x}_{c,G}\} \sim G_i^{(r)}(\cdot \mid c). \tag{25}$$

For each scorer $S \in \mathcal{S}_i$, scores are normalized within the group:

$$\tilde{s}_{c,j}^S = \operatorname{NormGroup}(\{S(\tilde{x}_{c,k})\}_{k=1}^G)_j. \tag{26}$$

By default, $\mathrm{NormGroup}$ maps group ranks to $[-1, 1]$, removing scorer-specific scale differences while preserving relative preference.

The pooled preference score is

$$\bar{s}_{c,j} = \mathrm{RobustAgg}_{S \in \mathcal{S}_i}(\tilde{s}^S_{c,j}), \tag{27}$$

where $\mathrm{RobustAgg}$ is the mean by default. Median or trimmed mean can be used when scorer outliers are a concern.

The deterministic format reward $R_{\mathrm{fmt}}(c, \tilde{x}_{c,j}) \in [-1, 1]$ checks schema validity, required fields, label constraints, numeric ranges, and output structure. The total reward is

$$r_{c,j} = \tanh(\beta \bar{s}_{c,j}) + \lambda_{\mathrm{fmt}} R_{\mathrm{fmt}}(c, \tilde{x}_{c,j}). \tag{28}$$

The group-relative advantage is

$$A_{c,j} = r_{c,j} - \frac{1}{G} \sum_{k=1}^{G} r_{c,k}. \tag{29}$$

In implementation, advantages are normalized and clipped within each group before generator update.

B.4 GRPO Generator Update

Let $G_i^{(r)}$ be the reference generator at the start of round r, and let G_{ψ_i} denote the generator being optimized. For candidate $\tilde{x}_{c,j}$, define the likelihood ratio

$$\rho_{c,j} = \frac{G_{\psi_i}(\tilde{x}_{c,j} \mid c)}{G_i^{(r)}(\tilde{x}_{c,j} \mid c)}. \tag{30}$$

The clipped ratio is

$$\bar{\rho}_{c,j} = \mathrm{clip}(\rho_{c,j}, 1 - \epsilon, 1 + \epsilon). \tag{31}$$

We use the conditional KL regularizer

$$\mathcal{D}_{\mathrm{KL}} = D_{\mathrm{KL}}(G_{\psi_i}(\cdot \mid c) \| G_i^{(r)}(\cdot \mid c)). \tag{32}$$

The client minimizes the clipped group-relative objective

$$\begin{aligned} \mathcal{J}_{\mathrm{GRPO}}(\psi_i) = -\mathbb{E}_{c,j}\Big[\min(&\rho_{c,j} A_{c,j}, \\ &\bar{\rho}_{c,j} A_{c,j})\Big] + \lambda_{\mathrm{KL}} \mathcal{D}_{\mathrm{KL}}. \end{aligned} \tag{33}$$

The KL term constrains the refined generator to remain close to the round-start generator. The update is performed entirely on premise, and neither generated tables nor generator parameters are transmitted.

B.5 Communication and Leakage Controls

Only the capacity-limited scorer parameters ϕ_i are uploaded. In implementation, we restrict scorer capacity, cap the number of scorer update steps per round, and optionally clip scorer updates before upload. These controls reduce direct exposure of client-specific information but do not constitute a formal privacy guarantee. Concordia is compatible with secure aggregation, access control, and differential privacy mechanisms, which can be added when formal privacy guarantees are required.

C Implementation Details

This section describes implementation details for reproducing Concordia, including the downstream model, local LoRA adaptation, client-local generators, scorers, GRPO refinement, and the limited-shot federated protocol. Unless otherwise stated, all methods use the same backbone capacity and the same per-round local compute budget.

C.1 Backbone LLM and Downstream Formulation

Backbone. We use **Qwen3-4B** as the shared backbone for the downstream tabular learner and the synthetic table generator. The backbone is initialized from the **Qwen/Qwen3-4B** repository at revision **1cfa9a7208912126459214e8b04321603b3df60c**.

Tabular task formulation. Each tabular record is serialized into a structured JSON-like representation and paired with a task instruction that specifies binary classification and the label space. The downstream model is trained using causal language modeling loss over the label token. During evaluation, the decision threshold is selected on the validation split to maximize MCC and then applied to the test split.

C.2 Federated Protocol and Two-round Schedule

Clients and rounds. We run $R = 2$ communication rounds. In healthcare, we simulate $N = 3$ training clients, Cleveland, Hungarian, and Switzerland, under a shared task identifier, with full participation in both rounds. VA Long Beach is held out for unseen-site evaluation. In finance, German Credit, Lending Club, and Travel Insurance are treated as three independent task streams because their label semantics and feature spaces correspond to different financial objectives. For a single-client task stream, the scorer pool is a singleton; the protocol still performs local scorer-guided generator refinement.

At round r, the server broadcasts the approved task backbone $\theta_\tau^{(r)}$ to participating clients of task τ. Each participating client performs local LoRA adaptation on synthetic tables, updates its scorer using private validation feedback, refines its generator via GRPO, and uploads only the capacity-limited scorer parameters. LoRA updates, generator parameters, synthetic tables, raw records, and validation sets remain on premise.

Task-aware scorer pooling. Clients are associated with a task identifier τ, where label semantics define task groups. The server does not aggregate LoRA deltas or generator parameters. Instead, it forms a within-task scorer pool

$$\mathcal{S}_r(\tau) = \{S_{\phi_j} : j \in \mathcal{C}_r(\tau)\}$$

and redistributes this pool to participating clients with the same task identifier. In healthcare, participating sites share one task identifier. In finance, each dataset is treated as an independent task stream.

C.3 LoRA Configuration and Local Optimization

LoRA modules. We apply LoRA to attention and MLP projections with rank $r = \mathbf{16}$, scaling $\alpha = \mathbf{32}$, and dropout **0**, targeting `[q_proj, k_proj, v_proj, o_proj, gate_proj, up_proj, down_proj]`. All backbone parameters remain frozen. The resulting LoRA adapter is used only locally and is never uploaded.

Local steps and optimizer. Each client performs $T = \mathbf{1500}$ local update steps per round. We use **AdamW** with learning rate $\eta = \mathbf{5e\text{-}5}$, weight decay **0**, and batch size **4** synthetic samples per step. We set the maximum sequence length to **1024** tokens and use gradient clipping with norm **1.0**.

C.4 Client-local Synthetic Table Generator

Initialization. Each client maintains a client-local generator $G_{\psi_i^{(r)}}$, initialized from the same checkpoint family as the backbone, using the **Qwen3-4B-Instruct** family. No additional public instruction-tuning stage is performed (consistent with Section 4.1); schema adherence is enforced by the generation prompt and the FormatReward parser described below.

Generation format. For each dataset, we construct a synthesis condition c consisting of (i) schema/-field list, (ii) five in-class exemplar records, and (iii) a task instruction with explicit label strings. The generator outputs a single serialized record in JSON-like format. We validate outputs using the

FormatReward parser/reward, including brace-block extraction and required key/label checks, and discard malformed samples.

Sampling parameters. We generate synthetic tables using temperature **1.0**, top-p **0.95**, and maximum new tokens **512**. Each local update step samples **4** synthetic tables, matching the LoRA batch size.

C.5 Scorer Model and Utility-weighted Training

Scorer architecture. Each client maintains a lightweight scorer S_{ϕ_i} that maps a serialized synthetic table to a scalar score. We implement the scorer by attaching a scorer LoRA adapter to the backbone and predicting a scalar from the final hidden representation. The scorer adapter uses rank $r_s = \mathbf{8}$, scaling $\alpha_s = \mathbf{16}$, dropout **0.05**, targeting `[q_proj, v_proj]`. Only this scorer adapter and scalar head are uploaded.

Input and representation. The scorer consumes only the JSON-like synthetic table string. We reuse backbone representations by extracting the last-layer hidden state with last-token pooling as the scorer input.

Weighting function. The scorer outputs a bounded value $S_{\phi_i}(x) \in [-1, 1]$ via $\tanh$. For a mini-batch X_{syn}, scores are mapped to nonnegative weights:

$$\tilde{\omega}_{\phi_i}(x) = \text{clip}\left(\frac{S_{\phi_i}(x) + 1}{2}, w_{\min}, w_{\max}\right), \tag{34}$$

$$\omega_{\phi_i}(x) = \frac{\tilde{\omega}_{\phi_i}(x)}{\sum_{x' \in X_{\text{syn}}} \tilde{\omega}_{\phi_i}(x')}. \tag{35}$$

We use $(w_{\min}, w_{\max}) = (0.05, 0.95)$.

Scorer update frequency. We refresh scorer parameters every $m = \mathbf{15}$ local LoRA steps. The validation utility is computed on the private validation set using $\mathcal{L}_{\text{val}} = 1 - \text{MCC}$. We optimize the scorer with **AdamW** and learning rate $\eta_\phi = \mathbf{5e\text{-}5}$.

C.6 Generator Refinement with GRPO

Candidate groups. For each synthesis condition c, the client samples a group of $G = \mathbf{8}$ candidate tables. Each candidate is evaluated with the scorer pool, and the ensemble reward is computed using the mean over normalized scores.

Advantages and stabilization. We compute group-relative advantages by subtracting the within-group mean. We normalize advantages within each group and clip them to **5.0**.

GRPO objective details. We use a frozen reference policy equal to the generator snapshot at the beginning of each round and apply a KL penalty to stabilize updates. The GRPO update is sample-level: each completion receives a group-relative advantage computed from the composed reward, normalized within the group. We use a PPO-style clipped objective with clip range **0.2** and KL coefficient **0.01**, kept fixed across tasks.

GRPO updates. We update the client-local generator using GRPO with learning rate $\eta_g = \mathbf{1e\text{-}6}$. Per round, we run **300** GRPO update steps over **600** synthesis conditions, using batch size 2 prompts per step, maximum prompt length **1024**, maximum completion length **512**, temperature **1.0**, and top-p **0.95**. The generator is never uploaded.

C.7 Random Seeds and Reporting

We report MCC as mean $\pm$ standard deviation over **3** random seeds, seed $\in \{0, 1, 2\}$. Each seed controls synthetic prompt construction and sampling, model/adapter initialization, data shuffling, and

Cand.	Age	Sex	RestECG	Angina	OldPeak	ST-slope	Vessels	Synth. label	$\hat{s}_{\phi 1}$	$\hat{s}_{\phi 2}$	$\hat{s}_{\phi 3}$
#1	65	M	LVH	yes	2.10	down	0	heartdisease	2.3281	4.8125	5.0000
#2	55	M	normal	no	0.00	up	1	healthy	**2.9844**	4.1250	4.0938

Table 3: **Clinical paired case with aligned scorer pool (Cleveland, G=2).** $\hat{s}_{\phi_k}$ are raw scorer-head outputs (pre-$\tanh$ logits) of the three pooled scorers.

Cand.	Agency	Product	Dur.	Dest.	Sales	Comm.	Age	Claim	$\hat{s}_{\phi 1}$	$\hat{s}_{\phi 2}$	$\hat{s}_{\phi 3}$
#1	c2b	silver_plan	60	thailand	45.00	13.50	35	no	0.9297	-5.1563	-3.7500
#2	ssi	silver_plan	60	india	25.50	7.65	55	no	0.9297	-5.1563	**-3.0625**

Table 4: **Finance paired case under scorer-guided generator refinement (Travel, G=2).** $\hat{s}_{\phi_k}$ are raw scorer-head outputs (pre-$\tanh$ logits).

GRPO sampling. For each seed, we run exactly two communication rounds and evaluate after each round. We use the same number of synthetic samples and the same local optimization steps for all methods.

C.8 Released Artifacts

We release the preprocessing scripts for the public benchmarks, the synthesis prompt templates and tabular serialization, all hyper-parameters and seeds, the scorer architecture, the GRPO configuration, and the evaluation task definitions and scripts. Raw records, validation sets, synthetic tables, client generators, and task-model updates are client-local under the protocol; for the public benchmarks used here they are regenerated by the released code.

D Generator Audit

The main results show early-round gains, but they do not by themselves explain what scorer-guided refinement changes in the synthetic training distribution. We therefore audit generator snapshots across rounds under a controlled paired-generation protocol. The goal is not to claim that individual examples prove causality, but to test whether the refinement signal moves synthetic samples toward validation-relevant structure.

Controlled paired generation protocol. For each dataset, we reuse the same few-shot prompt and identical decoding hyperparameters. We then sample paired synthetic tables from the generator snapshots used in each round. Each table is schema-valid by construction and labeled locally via the client labeling interface. To interpret refinement, we inspect the same scorer signals used for generator updates. The audit tables report, for each pooled scorer ϕ_k, the *raw scorer-head output* $\hat{s}_{\phi_k}$ (a real-valued logit) *before* the $\tanh$ squashing of Eq. (8); these raw values are therefore not confined to $[-1, 1]$, whereas the bounded scores that enter the GRPO reward are $\tanh$ transforms of their weighted fusion (Appendix B.3).

Healthcare: sharpening multi-feature risk conjunctions. For Heart Disease, the audit suggests that refinement organizes synthetic samples into clinically interpretable archetypes rather than single-feature heuristics. A high-risk profile jointly activates exercise-induced angina, elevated `OldPeak`, and adverse ST-segment slope patterns, and is labeled as `heartdisease`. Conversely, a low-risk profile exhibits `OldPeak`≈ 0, favorable slope, and normal ECG patterns, and is labeled as `healthy`. The key point is that the label is supported by a conjunction of features rather than one attribute in isolation, which is the type of structure that may transfer better under site shift.

Finance: preserving functional dependencies under skew. Travel Insurance is an extreme long-tail setting, and the few-shot context is dominated by `Claim_Status=no`. Under such skew, an unrefined generator can increase surface variety while remaining utility-irrelevant. In contrast, the refined generator preserves majority-mode plausibility while maintaining feature-level regularities

that matter for learning. In our paired examples, the generator maintains a stable commission-to-sales relationship, `Commission`$= 0.3\times$`Sales`, while varying destinations, duration, and age. This suggests that refinement does not merely diversify strings or enforce formatting, but encourages internally consistent feature dependencies.

E Prompts and Tabular Serialization

This section documents the tabular serialization format and the client-local prompting templates used for synthetic table generation.

E.1 Serialization Format

Each record is serialized into a JSON-like format with a single record per sample. We use a fixed key order to avoid format drift across rounds. Missing values are encoded as `nan`. Categorical values are represented as unquoted tokens and continuous values are represented with four decimal places.

Task-specific instructions.

Heart disease

```
Predict whether the patient has heart disease using the features below: Age, Sex, ChestPainType, RestingBloodPressure, Cholesterol, FastingBloodSugar, RestingECG, MaxHeartRate, ExerciseInducedAngina, OldPeak, PeakExerciseSTSegmentSlope, NumberOfVesselsColored, and ThalassemiaStatus. Directly respond with 'heartdisease' if the patient has heart disease or 'healthy' if they do not.
```

Credit risk

```
Predict whether the loan applicant is a credit risk using the features below. Directly respond with 'bad' if the applicant is a bad credit risk or 'good' if they are a good credit risk.
```

Loan status

```
Predict the loan status using the features below. Directly respond with 'fullypaid' if the loan is fully repaid or 'chargeoff' if the loan is charged off.
```

Insurance claim

```
Predict whether the customer will file an insurance claim based on the features below. Directly respond with 'yes' if the customer is likely to make a claim, or 'no' if the customer is unlikely to make a claim.
```

E.2 Synthesis Condition

For each client, the synthesis condition includes the schema, in-class exemplar records, and the task specification. The task specification defines the prediction target and the label format, e.g., `healthy/heartdisease`, `good/bad`, `fullypaid/chargeoff`, and `yes/no`.

E.3 Generator Prompt Template

We use the following client-local prompt template with five in-class exemplars:

> **System:** You are a data synthesis assistant. Output only one valid JSON-like object that matches the schema.
> **User:** Here are five entries of tabular data in JSON format, each consisting of n features. Each feature is described as "`"feature name": value`". The target feature *label* is a classification task.
> **User:** `sample one:` ...(5 exemplars)
> **User:** Directly generate only one new final sample in JSON format that approximates the key patterns observed in the provided samples.
> **Assistant:** `{ ... }`

We apply a strict parser/reward, FormatReward, to the output and discard malformed generations.

F Communication and Compute Cost

F.1 Communication Payload

Under the export boundary of Concordia, each client uploads only the capacity-limited scorer parameters ϕ_i. Downstream LoRA deltas, generator parameters, synthetic tables, raw records, and validation sets are never transmitted.

In our implementation, the communicated scorer consists of a scorer LoRA adapter and a scalar scorer head on top of the shared public backbone. Using bf16 transmission, 2 bytes per parameter, for Qwen3-4B with $L = 36$ and hidden size $d = 2560$, the scorer payload is

$$P_{\mathrm{s}} = 4Lr_s d = 2{,}949{,}120, \tag{36}$$

and the scalar head has

$$P_{\mathrm{head}} = d + 1 = 2{,}561. \tag{37}$$

Thus, the per-client scorer uplink is

$$2(P_{\mathrm{s}} + P_{\mathrm{head}}) = 5.90 \text{ MB} \quad (5.63 \text{ MiB}). \tag{38}$$

For $|\mathcal{C}_r| = 3$ participating clients, the total uplink per round is

$$3 \times 5.90 = 17.71 \text{ MB} \quad (16.89 \text{ MiB}). \tag{39}$$

When the server redistributes the scorer pool, each client receives the other $|\mathcal{C}_r| - 1$ scorers:

$$2(|\mathcal{C}_r| - 1)(P_{\mathrm{s}} + P_{\mathrm{head}}) = 11.81 \text{ MB} \quad (11.26 \text{ MiB}) \tag{40}$$

for $|\mathcal{C}_r| = 3$. We exclude one-time distribution of the public backbone checkpoint.

G Privacy Discussion and Threat Model

G.1 What Remains Local

Raw training records, private validation sets, generated synthetic tables, generator parameters, and downstream LoRA/adaptor updates remain on the client. The only learned artifact uploaded to the server is the capacity-limited scorer parameter set ϕ_i.

G.2 Scorer Sharing and Privacy Boundary

Scorers are shared as lightweight evaluators to enable validation-aligned coordination without centralizing validation data or transmitting task-model updates. A communicated scorer does not include full scorer LLM weights. All parties share the same public backbone checkpoint, and each client transmits only a scorer LoRA adapter plus a scalar scorer head.

This reduces the communicated surface compared with sharing full model updates, synthetic data, or generator parameters. However, the scorer remains a learned model component and could still leak aggregate information about client distributions or label properties under strong adversarial assumptions. We therefore treat scorer sharing as a reduced-exposure communication channel rather than a formal privacy guarantee. Secure aggregation, access control, clipping, or differential privacy can be added when formal guarantees are required.

H Mechanistic Evidence on Travel

Travel Insurance is the most imbalanced task in our study ($p \approx 0.01$) and exhibits the characteristic few-shot shape in the main results: MCC improves from 0.00 ± 0.00 for the single-node baseline to 7.07 ± 1.85 after one round and 21.78 ± 5.84 after two rounds. We use the reward-colored embedding visualization in Fig. 2 to explain why the second round contributes disproportionately: refinement changes where the generator places probability mass in synthetic space, not merely how many samples it generates.

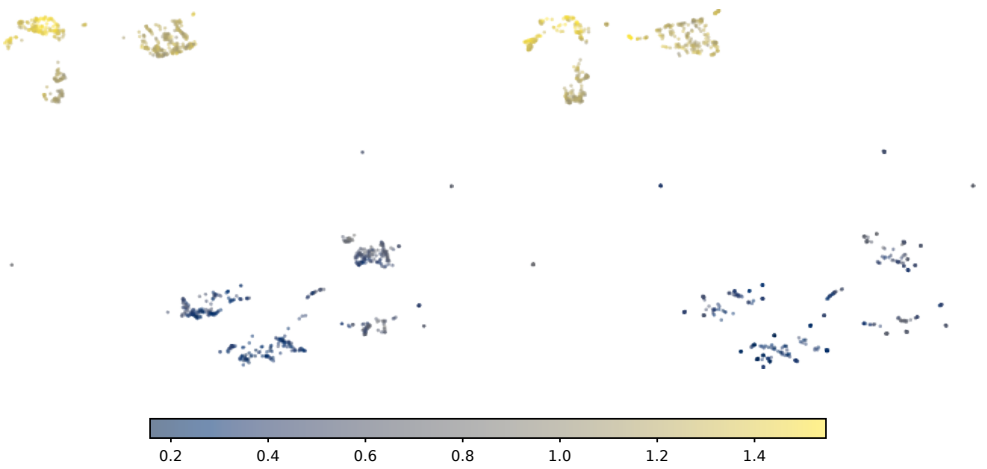

Figure 2: **Travel: reward-aligned refinement expands utility-relevant support.**

Round 0: diverse modes, but reward concentrated in a few easy regions. The round-0 generator already produces a multi-modal set of samples, visible as several separated clusters in the embedding space. However, reward is highly uneven across these modes. High-reward samples concentrate in only a small subset of clusters, while many clusters remain uniformly low-reward. Under extreme imbalance, this pattern is expected: without a utility-driven signal, synthetic sampling tends to overproduce majority-like cases that are schema-valid yet uninformative for minority-sensitive decisions.

Refinement induces within-mode shifts and across-mode activation. Comparing round 0 and round 1, Fig. 2 suggests two changes driven by reward-based refinement. First, within modes that persist across rounds, reward increases locally, indicating that refinement pushes samples toward reward-favorable subregions without collapsing the mode structure. Second, reward is no longer confined to the same few modes; modes that were previously uniformly low-reward begin to contain high-reward samples. This indicates that refinement does not merely oversample the easiest high-reward mode, but activates additional utility-relevant modes that were present but ineffective in round 0.

Why this explains the two-shot gain on the long tail. The first round adapts the predictor to the initial synthetic support, which is dominated by low-reward regions and therefore yields only a modest improvement. The second round benefits disproportionately because the generator has been reshaped to place more mass on reward-aligned regions within modes and to expose the learner to a broader set of reward-bearing modes. In a $p \approx 0.01$ regime, this distributional change enables minority-sensitive utility to improve: the learner sees more boundary-relevant and rare-event-supporting cases, rather than repeating majority-like patterns.

Implication for limited-shot cross-silo coordination. The Travel analysis highlights why one-/two-shot coordination is meaningful in privacy-constrained deployments. When the limiting factor is rare-event exposure rather than long-horizon optimization, a single refinement step that redistributes synthetic mass toward reward-aligned regions can translate into an immediate next-round gain. This mechanism is consistent with the design goal of Concordia: use validation-aligned feedback to reshape synthetic training under strict isolation, enabling progress even when direct minority supervision is scarce.