INCENTIVIZING FEDERATED LEARNING: AGREEMENT IS ENOUGH

Anonymous authors
Paper under double-blind review

ABSTRACT

Federated learning enables collaborative training on private data, but attracting clients with valuable data requires incentives for effortful training and honest reporting. Computing these rewards must be inexpensive enough for every training round. Knowledge-Free Correlated Agreement (KFCA) addresses this need with fixed Dasgupta–Ghosh agreement-minus-chance scoring, avoiding correlation estimation and penalizing unilateral label flips under categorical and task assumptions. Federated LLM tuning, industrial inspection, and runtime comparisons assess rewards and efficiency; a smart-contract prototype illustrates auditable execution.

1 INTRODUCTION

Federated learning (FL) enables multiple clients to collaboratively train a model without sharing raw data (Brendan McMahan et al., 2017). Assessing and incentivizing client contributions remains difficult: updates are hard to verify, yet rewards must discourage low-effort participation and malicious reporting Zeng et al. (2021); Zhan et al. (2021); Ali et al. (2021); Tu et al. (2021); Witt et al. (2022). Classical contribution measures such as Shapley value require a verifiable utility, often from a public test set, and can be computationally expensive. Appendix A.1 comprehensively compares FL incentive mechanisms by their information requirements, objectives, computation, and guarantees (Table 2).

For frequent rewards without reliable validation data, we consider multi-task peer prediction (MTPP) the most promising approach: it elicits information from repeated client reports, with conditional reporting guarantees and inexpensive sampled peer comparisons. Correlated Agreement (CA) by Shnayder et al. (2016) is a prominent example applied in FL (Lv et al., 2021; Liu & Wei, 2020). Implementations that calculate a report-correlation matrix Δ from collected reports face two practical issues: estimating correlations adds scoring work, and recalculating the rewarded pattern from manipulated reports can preserve rewards for label flipping.

Knowledge-Free Correlated Agreement (KFCA) builds on the agreement-minus-chance rule of Dasgupta & Ghosh (2013) and its multi-signal characterization by Shnayder et al. (2016). Under the categorical-world condition, matching predictions or update signs occur more often than chance, and mismatches less often; equality checks then replace correlation estimation (Section 2). Our contribution is the FL pipeline, conditional incentive analysis, and experiments on computational cost, reward behavior, and applicability. Section 3 maps predictions and updates to the score and distinguishes unilateral flips from shared relabelings.

We examine categorical structure in four-domain federated LLM fine-tuning and non-IID client data (Section 4), then compare computation and reward allocations with Shapley and empirical-CA baselines (Section 5.1) and apply KFCA to industrial PCB inspection (Section 5.2). Section 6 presents a smart-contract scoring prototype and deployment outlook. Section 7 extends this prospect to governing large swarms of autonomous AI agents through inexpensive, frequent rewards.

2 FROM CORRELATION ESTIMATION TO FIXED AGREEMENT

Rewarding information without an answer key. In FL, clients train a shared model on private data, but the coordinator cannot directly verify the information gained through local training. Multi-task peer prediction (MTPP) uses other clients' reports as evidence. Here a *task* is one common

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.

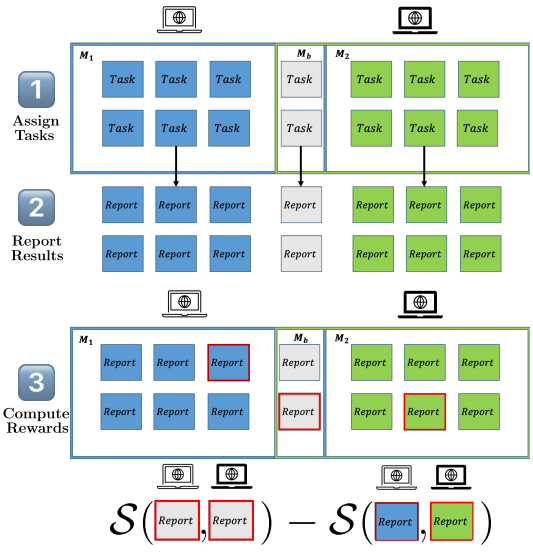


Figure 1: Peer prediction: agreement on a shared task minus agreement on distinct penalty tasks. Bonus/penalty roles are sampled after reports are fixed.

Table 1: MTPP mapped to examples of KFCA-D (public test set) and KFCA-QP (sign-quantized parameter updates) instantiations.

MTPP	**FL (KFCA-D / KFCA-QP)**
Task k	Classify a test-set sample / decide update sign at parameter p of a model update $\Delta\theta$
Latent truth Y^k	Correct class label / latent reference update sign at p
Signal Z_i^k	Local classification of an image / $q(\Delta\theta_i[p]) \in \{-1, +1\}$
Effort $e_i^k = 1$	Train local model / fine-tune LLM adapter (LoRA/DoRA)
No effort $e_i^k = 0$	Random prediction / zero or random update
Report R_i^k	Submitted label / submitted sign

scoring item: predicting the class of the same input in KFCA-D, or reporting the change at the same model-parameter coordinate in KFCA-QP. One local training run produces signals for many such tasks; a task is not a separate training run.

Formally, let $N = \{1, \ldots, n\}$ index clients and $M = \{1, \ldots, m\}$ index tasks. The latent-truth model associates each task k with one unobserved reference label $Y^k \in [L] = \{1, \ldots, L\}$, shared by all clients. This *label space* contains the possible classes for predictions, or the two update signs $\{-1, +1\}$ for parameter reports (equivalently indexed by $[2]$). For predictions, Y^k is the correct class; for updates, it models a reference direction for that coordinate, not a known unique optimal parameter value. In the analysis, truths are drawn independently from a common categorical prior, meaning $\mathbb{P}(Y^k = a) = \pi_a$ with $\sum_{a=1}^{L} \pi_a = 1$. Neither these probabilities nor the truths are needed to compute the score.

Client i obtains a signal Z_i^k—its model's prediction or local update sign—and submits R_i^k in the same label space. Truthful reporting means $R_i^k = Z_i^k$, even if the signal differs from Y^k. Paying for matches alone would also reward clients who always report the same label. MTPP therefore subtracts agreement on different tasks to estimate this chance baseline (Figure 1); Table 1 gives the FL correspondence.

Definition 2.1 (Multi-task peer prediction)**.** For a pair of clients with at least three common tasks, fix reports before choosing nonempty disjoint bonus and penalty sets $M_b, M_1, M_2 \subseteq M$. For each $k \in M_b$, independently draw $u_k \in M_1$ and $v_k \in M_2$ uniformly. Given a pairwise scoring function $\mathcal{S}$, average $\mathcal{S}(R_i^k, R_j^k) - \mathcal{S}(R_i^{u_k}, R_j^{v_k})$ over bonus tasks. The bonus compares the same input or coordinate across clients; the penalty compares two different inputs or coordinates. Under the independent, identically distributed task model, the penalty estimates the score expected from each client's overall label frequencies.

What correlation estimation supplies. For a fixed pair of clients, let Z_i, Z_j denote their signals on the same randomly drawn task. Define the excess of their joint probability over independence:

$$\Delta(a, b) = \mathbb{P}(Z_i = a, Z_j = b) - \mathbb{P}(Z_i = a)\mathbb{P}(Z_j = b). \tag{1}$$

The first term measures how often the clients produce labels a, b on the same input or coordinate; the product is the baseline from their separate label frequencies. A positive entry means that this pair occurs more often than chance predicts. Correlated Agreement (CA) uses $\mathcal{S}_{\mathrm{CA}}(a, b) = \mathbb{1}\{\Delta(a, b) > 0\}$

to select rewarded pairs (Shnayder et al., 2016), where $\mathbb{1}\{\cdot\}$ is 1 when its condition holds and 0 otherwise. In practice, CA calculates an empirical matrix $\widehat{\Delta}$ from joint and individual label frequencies in the collected reports, then uses its signs to select rewarded pairs; peers can also be sampled. What happens if the sign pattern is specified by structure instead?

Definition 2.2 (Categorical-world condition (Shnayder et al., 2016))**.** On a common label space $[L]$, $L \geq 2$, suppose every label has positive probability for each client. Observing label a from one client makes every different label less likely for the other on the same task: $\mathbb{P}(Z_j = b \mid Z_i = a) < \mathbb{P}(Z_j = b)$ for all $a \neq b$.

For binary update signs, this means that seeing one client increase a parameter makes a decrease by its peer less likely than usual. For multiclass predictions, seeing one class must make every other class less likely. A single latent truth or a finite label space alone does not ensure this condition. Its equivalence to a matrix sign pattern follows from

$$\Delta(a, b) = \mathbb{P}(Z_i = a)\big[\mathbb{P}(Z_j = b \mid Z_i = a) - \mathbb{P}(Z_j = b)\big]. \tag{2}$$

Positive label probabilities make the condition equivalent to negative off-diagonal entries. Every row of Δ sums to zero, so its diagonal must then be positive. Conversely, negative off-diagonal entries give the conditional inequalities above. Thus

$$\boxed{\Delta(a, a) > 0, \quad \Delta(a, b) < 0 \ (a \neq b)} \quad \Longrightarrow \quad \boxed{\mathcal{S}_{\mathrm{CA}}(a, b) = \mathbb{1}\{a = b\}.} \tag{3}$$

Thus, if the condition holds, CA rewards matching predicted classes or update signs and no mismatches; no correlation matrix needs to be calculated for scoring.

The design trade-off. A full-report calculation of $\widehat{\Delta}$ counts label pairs across all collected task reports for each compared client pair and subtracts the product of their individual label frequencies, following Eq. (1). CA then constructs its score matrix from the positive entries. KFCA replaces this calculation during scoring with the assumption that matching labels occur more often, and different labels less often, than independent reports would predict. The resulting fixed agreement score is the multi-signal Dasgupta–Ghosh (MSDG) construction (Dasgupta & Ghosh, 2013; Shnayder et al., 2016). "Knowledge-free" means that payments do not require this empirical matrix; calculating it remains useful for checking the assumption.

Section 4 examines matrices calculated from recorded reports in federated LLM finetuning: all displayed matrices across four domains and ten rounds exhibit the categorical sign pattern (Appendix Figure 11), with report-definition qualifications discussed there. The same section examines non-IID client data through measured update-sign matrices under Dirichlet label skew (Figure 5), including sensitivity to zero updates; Appendix A.9 gives a sufficient-condition proof and experimental details. These results support the correlation assumption in the evaluated FL settings. The reporting guarantees additionally require the task model in Section 3.2; a single correct class label or a hypothesized reference update direction alone does not establish them.

3 KFCA: FL Construction and Incentive Guarantees

3.1 Reports and sampled-peer execution

At communication round t, the coordinator distributes the current shared model parameters, or *checkpoint*, θ_t. Each participating client i starts from these parameters, trains on its private data to obtain $\theta_{i,t}$, and returns the update $\Delta\theta_{i,t} = \theta_{i,t} - \theta_t$. The coordinator combines these updates, for example by a data-weighted average (FedAvg), to form the next global model. KFCA computes reward scores alongside this aggregation.

KFCA-D uses the locally trained model's predicted labels $Z_i^k \in [L]$ on shared unlabeled inputs. KFCA-QP instead uses $Z_i^k = q(\Delta\theta_{i,t}[k]) \in \{-1, +1\}$ from updates already communicated for training (Figure 2). Coordinate k must refer to the same parameter in every client's model or trainable adapter. We set $q(x) = +1$ for $x \geq 0$ and -1 otherwise: the report records whether that parameter increased or stayed unchanged $(+1)$, or decreased (-1), discarding the change's magnitude. A submitted report R_i^k may differ from the obtained signal Z_i^k. These constructions provide a common label space; categorical correlation remains a separate assumption.

Definition 3.1 (KFCA score)**.** Using $\mathcal{S}(r, s) = \mathbb{1}\{r = s\}$ in Definition 2.1, client i receives pair score

$$\widehat{E}_{ij} = \frac{1}{|M_b|} \sum_{k \in M_b} \left(\mathbb{1}\{R_i^k = R_j^k\} - \mathbb{1}\{R_i^{u_k} = R_j^{v_k}\} \right), \qquad \widehat{E}_i = \frac{1}{p} \sum_{j \in P_i} \widehat{E}_{ij}, \tag{4}$$

where P_i contains p distinct peers sampled uniformly from the other participating clients. The sets and penalty indices are those of Definition 2.1.

The first term rewards matching predictions or update signs on the same task; the second subtracts matches on different tasks. For example, if each client reports $+1$ on 80% of tasks, independent tasks match with probability $0.8^2 + 0.2^2 = 0.68$. The score rewards agreement above this baseline, rather than merely frequent positive updates. Two constant reporters receive zero. Each comparison requires equality checks and subtraction, giving $\mathcal{O}(npm)$ scoring work for n clients and m scoring tasks at a fixed peer budget p. KFCA-D additionally needs prediction computation; KFCA-QP extracts signs from received updates. Appendix A.11 gives the algorithms. Reports must be fixed before random peers and bonus/penalty roles are revealed.

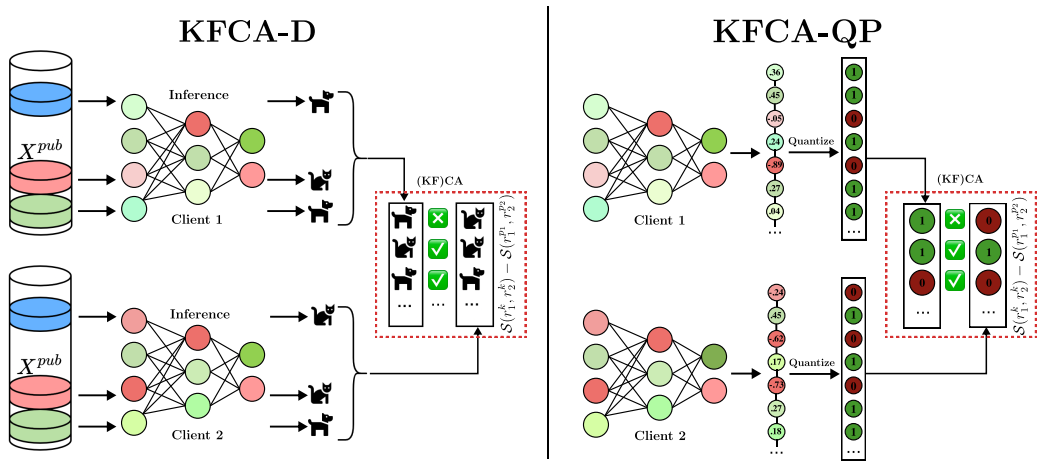


Figure 2: KFCA-D scores shared-input predictions; KFCA-QP scores update signs. Both apply the same agreement-minus-penalty rule.

3.2 Conditional reporting guarantees and label flipping

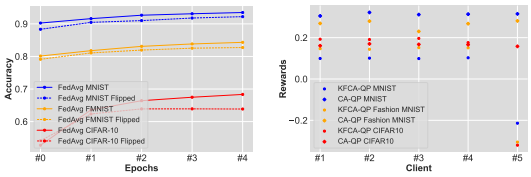
For the analysis, tasks are independent draws from a common distribution and clients use the same signal-to-report function $f_i : [L] \to [L]$ on every task, so $R_i^k = f_i(Z_i^k)$. For update signs, examples are reporting the obtained sign, reversing it, or always reporting $+1$. Appendix A.2 gives the full model. The categorical condition alone does not establish these task assumptions for neural-network coordinates. Subtracting chance agreement yields

$$E(f_i, f_j) = \sum_{a,b \in [L]} \Delta(a, b)\, \mathbb{1}\{f_i(a) = f_j(b)\}. \tag{5}$$

Proposition 3.2 (Categorical agreement (Shnayder et al., 2016))**.** *Under the task model and categorical condition, truthful reporting maximizes Eq.* (5) *over deterministic strategy pairs. Equality holds precisely when $f_i = f_j = \pi$ for a permutation π of $[L]$.*

In Eq. (5), $\Delta(a, b)$ is the excess frequency of signal pair (a, b), and the indicator selects pairs that the reporting strategies turn into matching reports. Truth includes every positive diagonal term and no negative off-diagonal term. Any maximizing strategy pair must preserve that separation, yielding the shared-permutation characterization; Appendix A.5.1 gives the proof. Truth is the unique best response to a truthful peer, while shared relabelings remain equally rewarded. In FL, this favors faithfully reporting the prediction or update sign obtained from local training; it does not assert that this prediction is correct or that the update improves global accuracy. A shared permutation renames all classes identically, or reverses both clients' update signs, preserving every equality check. This is the inherited MSDG guarantee applied to the FL reports, not a new scoring theorem.

Why fixing the score matters. Fixed categorical CA uses the same score as KFCA. A score matrix calculated from collected reports can instead adapt to manipulated reports. In the balanced binary example of Appendix A.4.2, truthful signals have positive diagonal correlations. One client's flip makes the reported correlations positive off the diagonal: recalculating the matrix continues to reward the inverted correspondence. Against an honest peer, population payoffs are $+0.5$ for honesty and -0.5 for flipping under fixed agreement, but $+0.5$ for both when the matrix is recalculated. Separate calibration and scoring tasks do not remove a systematic flip across both sets. Figure 3 reports the tested sign-flip comparison; its interpretation concerns the implementation that recalculates the matrix, not all CA formulations.



(a) Global accuracy (b) Rewards

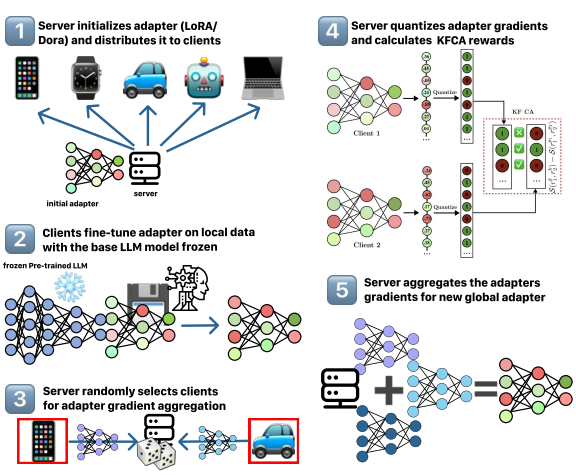
Figure 3: Sign-flip experiment: KFCA-QP assigns the flipping client a lower reward; the tested learned-CA baseline does not separate it comparably. A fixed categorical CA score equals KFCA (Appendix A.4).

Proposition 3.3 (Binary label-flip robustness)**.** *Assume the two latent labels are equally likely and signals are conditionally independent given the latent truth, each with the same error probability $\alpha < 1/2$ for either label. If a sampled peer flips with probability $\lambda < 1/2$ and otherwise reports truthfully, reporting the obtained signal (identity) is the unique best deterministic response.*

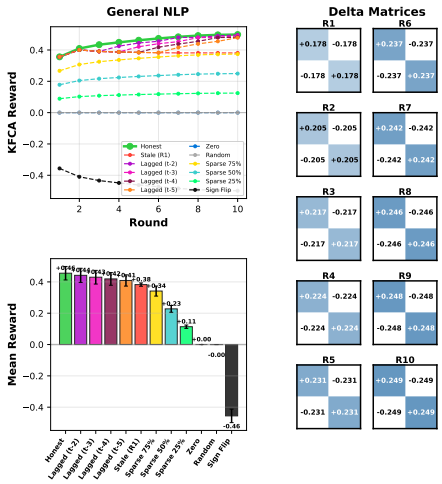
Appendix A.6 compares identity, flipping and constant reports. This binary result does not remove shared-permutation equilibria or give a general multiclass guarantee. Training once produces many reports, at cost c_i; utility is expected payment minus this cost. Payments $b + \beta \widehat{E}_i$, with fixed base payment b and score-to-payment multiplier $\beta > 0$, support effort only when the expected reward gain exceeds its additional cost. Appendix A.10 analyzes independent uninformative shirking; correlated temporal alternatives require separate evaluation.

4 Does Federated Learning Exhibit Categorical Structure?

Federated LLM fine-tuning enables collaboration when privacy, security, or governance constraints prevent pooling finance, healthcare, and enterprise-code data. We evaluate KFCA-QP using leading January 2026 FlowerTune configurations for general NLP, finance, medicine, and code (Gao et al., 2025; Flower Labs, 2026).



(a) Per-round pipeline.



(b) Correlations and rewards (general NLP).

Figure 4: Federated LLM adapter tuning: (a) pipeline; (b) general-NLP diagnostics.

Clients fine-tune LoRA/DoRA adapters with frozen base LLMs and communicate updates for FedAvg (Hu et al., 2022; Liu et al., 2024); KFCA-QP scores binary reports without shared evaluation inputs or extra model evaluations (Figure 4a). Runs use ten rounds, IID equally sized client splits, and two active clients per round. Base models have 7–8B parameters and use adapter ranks of 8–32 across 10–20 registered clients. Scores do not reweight FedAvg. Appendix A.13 gives settings and downstream scores.

Empirical correlations. Each empirical 2×2 matrix $\widehat{\Delta}$ summarizes pairs of $-1/+1$ reports at corresponding adapter coordinates, after subtracting the chance baseline. Positive diagonal entries mean matching signs occur more often than chance; negative off-diagonal entries mean opposite signs occur less often. All recorded matrices across four domains and ten rounds exhibit this categorical pattern; Figure 4b shows general NLP and Appendix Figure 11 shows the complete results.

Reward behavior. The recorded diagnostics give honest reports the highest mean reward, constant/random reports near-zero rewards, and sign flips negative rewards; sparse and stale reports retain intermediate rewards. Appendix Figure 12 shows all four domains; Appendix A.13 gives attack definitions and numerical summaries.

These observations support categorical agreement for the recorded reports. Their interpretation as local-update signs remains subject to the audit distinguishing adapter states from their changes in Appendix A.14.1. Reporting incentives and incentives to perform fresh training are distinct; the latter additionally depend on training costs and feasible alternatives.

Beyond IID client data. Non-IID training data need not violate the categorical condition: clients can favor different labels while retaining a shared informative signal.

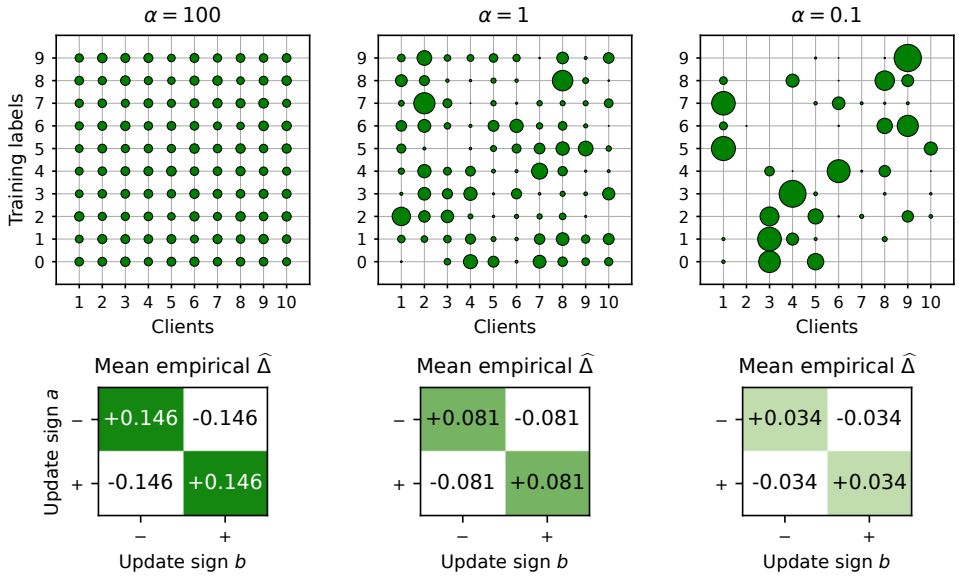


Figure 5: Toy MNIST experiment under increasing non-IID label skew ($\alpha = 100, 1, 0.1$, left to right), for seed 11. Top: training-label allocations; bubble area indicates example count. Bottom: mean empirical $\widehat{\Delta}$ of binary update signs over 45 client pairs; shared green shades encode positive diagonals, and negative off-diagonals are white. All 45 pairs retain positive diagonals and negative off-diagonals at each concentration; zeros are mapped to $+1$. Appendix A.9 gives the protocol and five-seed results.

A sufficient model explains this compatibility. Suppose signals are independent given a latent reference Y (a class or update sign): client i returns Y with probability $q_i > 0$ and otherwise samples from its own label distribution μ_i, independently of Y. With at least two labels and $\pi_a = \Pr(Y = a) > 0$ for every label, subtracting chance agreement cancels the client-specific preferences,

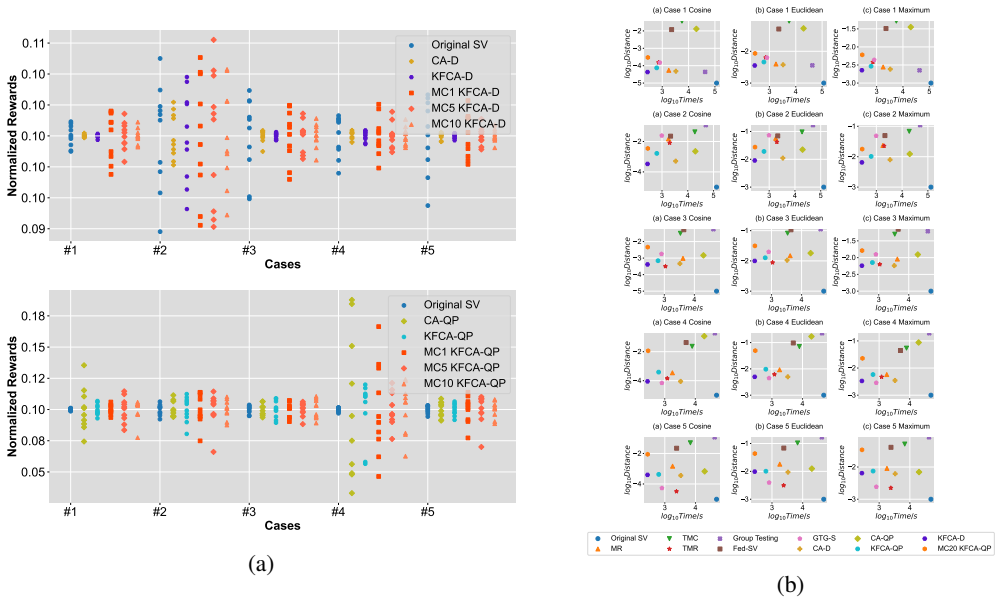
giving $\Delta_{ij}(a,b) = q_i q_j(\pi_a \mathbb{1}\{a = b\} - \pi_a \pi_b)$ (Appendix A.9). Diagonals are positive and off-diagonals negative: agreement beyond chance reflects shared information despite heterogeneous label preferences. This is a sufficient model; the scoring-task assumptions of Section 3.2 remain necessary for the reporting guarantee.

MNIST provides a controlled *toy illustration* of this compatibility. Smaller α produces more uneven training-label allocations across clients; the scored reports remain parameter-update signs. In the illustrated seed, stronger skew reduces correlation magnitude without reversing its sign. Under severe Dirichlet label skew ($\alpha = 0.1$), all 225 client-pair checks across five seeds retain categorical signs using full update-sign vectors, with zeros mapped to $+1$. Excluding coordinates where either update is zero leaves 133/225 passing checks (Figure 5). Thus categorical structure can persist under severe non-IID data, but the empirical conclusion depends on the report representation.

5 Computational Efficiency and Industrial Application

5.1 Scaling reward computation

The practical test is whether repeated scoring can stay inexpensive as participation grows. KFCA scores categorical reports in $\mathcal{O}(npm)$ work for n clients, p sampled peers per client, and m report coordinates. Holding p and m fixed makes this cost linear in participation: adding a client adds a bounded number of report comparisons, without requiring comparison with every other client. Across T rounds, the scoring work is $\mathcal{O}(\sum_{t=1}^{T} n_t p_t m_t)$. The peer budget therefore controls recurring computation, while report length captures the cost of larger updates or prediction sets. Appendix Figure 9 measures the client/peer-count scaling. Figure 6 compares scoring cost and reward allocations against Shapley baselines across five MNIST cases.



(a) (b)

Figure 6: (a) Sum-normalized reward allocations: shared-input predictions (top) and quantized updates (bottom). (b) Complete computation–fidelity comparison for all five MNIST cases (rows), with cosine, Euclidean, and maximum-coordinate distance to exact Shapley values (columns). Both axes in (b) use $\log_{10}$ scales; lower and further left indicate smaller distance and less computation.

The saving comes from the operations that scoring avoids. KFCA applies a fixed agreement rule without coalition model evaluation or correlation-matrix estimation. KFCA-QP derives reports from updates already communicated for training; KFCA-D additionally requires predictions on shared inputs. These report-construction costs remain relevant. Appendix Table 2 summarizes the information and computation each approach requires. Learned CA can also sample peers,

so eliminating correlation estimation is a separate saving from limiting peer comparisons. The comparisons measure scoring time and reward fidelity in the recorded implementations; they do not measure smart-contract execution costs.

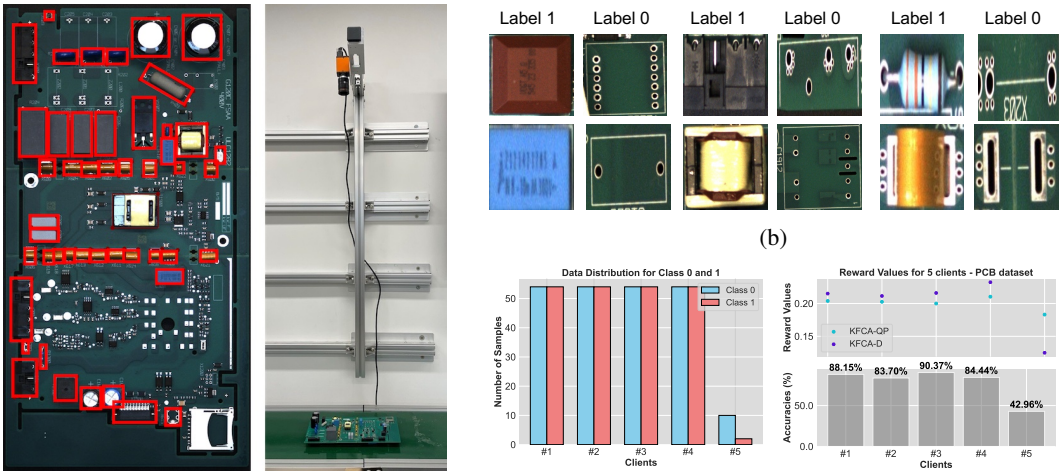
Benchmark and baselines. We use MNIST with a 21,840-parameter CNN and ten clients formed from five silos, keeping exact Shapley value (SV) computable. Five cases cover IID data, label skew, quantity skew, label noise, and feature noise. Comparisons include exact SV, GTG-Shapley, TMC-Shapley, Group Testing, MR, Fed-SV, TMR, CA-D/QP, and sampled KFCA variants. Appendix A.12.6 defines the methods; Appendix A.12.7 specifies partitions, normalization, and stopping criteria.

Results. Figure 6a compares the distributions of sum-normalized rewards. Figure 6b shows the complete computation–fidelity trade-off across all five cases, using cosine, Euclidean, and maximum-coordinate distance to exact SV. KFCA combines low computation time with reward allocations close to the reference in these settings, and computes scores substantially faster than the evaluated CA variants. Appendix Figure 10 provides individual-client allocations. Closeness to Shapley values is an empirical comparison, not an equivalence of incentive objectives.

Isolating the cost of correlation estimation. On an Apple M4 Pro, median scoring times for two general-NLP clients (19.3 million adapter coordinates each) are 123 ms for fixed scoring, 371 ms for estimating and applying a CA matrix, and 154 ms for applying a cached matrix. These matched CPU measurements hold reports and scoring indices fixed, separating estimation cost from peer sampling. Both methods remain linear in report length at fixed alphabet and peer budgets; caching narrows the saving. These are scoring-kernel times, excluding training and report construction (Appendix A.14). The following industrial example examines reward behavior beyond the LLM setting.

5.2 Real-world PCB inspection

Printed circuit board (PCB) inspection provides a physical FL application: cameras capture assembled components and a classifier detects missing ones. We train LeNet with FedAvg across five clients on 672 images, distinguishing correct assembly from missing components (LeCun et al., 1989). Figure 7 shows the hardware, component samples, client data, and measured rewards and accuracies.



(a) (b) (c)

Figure 7: Real-world PCB inspection. (a) Board and camera. (b) Component images. (c) Client data (left) and KFCA-D/QP rewards and accuracies (right). Client 5 has fewer, more imbalanced samples and receives lower rewards.

KFCA-QP gives similar rewards to clients 1–4 and less to client 5; KFCA-D shows the same broad separation. Clients 1–4 achieve 83.70–90.37% accuracy, versus 42.96% for client 5, whose samples are also fewer and more imbalanced. KFCA-QP computes these rewards from communicated updates

without using evaluation labels. This makes scoring applicable when inspection data are distributed across sites and a shared evaluation set is difficult to organize.

6 Blockchain Execution: An Implementation Outlook

KFCA reduces scoring to equality checks and arithmetic, making decentralized blockchain rewards a practical design target: contracts need neither model evaluations nor correlation estimation to score submitted reports. Figure 8 outlines a potential KFCA-D deployment: clients train and predict off-chain, commit reports before random pairing, reveal them for verification, and receive score-based payments. Our Solidity/EVM prototype implements the scoring arithmetic. Appendix A.15 checks each step, covering report hiding, randomness, verification, aggregation, and funded payouts. Completing these protocol components and measuring gas costs and end-to-end performance remain implementation work.

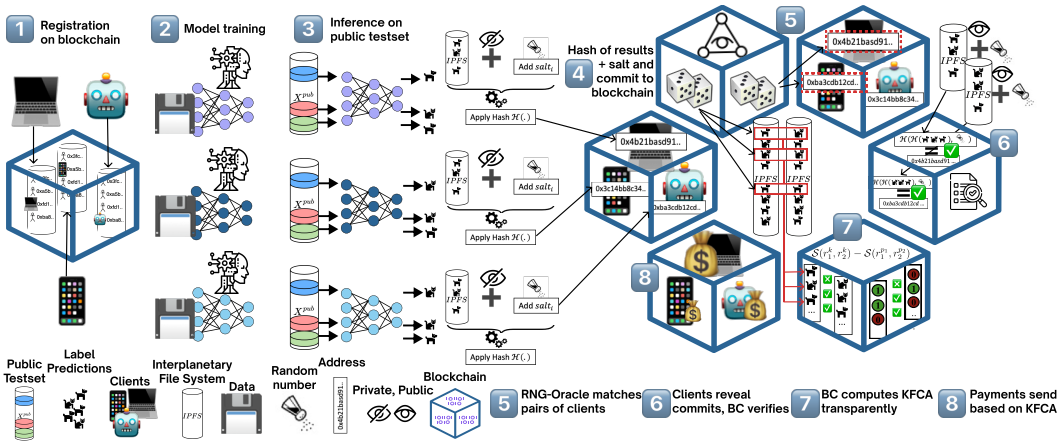


Figure 8: Potential KFCA-D blockchain workflow. (1–3) Register, train locally, and predict labels on shared unlabeled inputs. (4–6) Commit reports, sample peers/tasks, and reveal and verify. (7–8) Calculate scores and transfer funded rewards. Transparency of blockchain requires a commit-reveal scheme. Appendix A.15 specifies the steps and prototype limitations.

7 Discussion and Outlook

The central trade-off is between calculating correlation signs from reports and assuming a categorical structure that makes those signs known. When the assumption and task model apply, an established agreement principle supplies conditional reporting guarantees with simple scoring. The FL implementations connect this principle to predictions and updates, and the experiments examine its structural premise, computational costs, and reward behavior. Fixed scoring can penalize unilateral flips that an adaptive matrix continues to reward. It does not distinguish shared permutations, measure marginal model value, or by itself ensure that fresh training pays for its cost. A practical deployment should therefore measure correlations in its chosen report representation, choose a peer budget that fits each training round, and calibrate payments against the participating clients' training and communication costs.

Toward autonomous-agent collectives. KFCA could provide an inexpensive reward mechanism for governing large swarms of autonomous AI agents. Comparable categorical reports, such as independent checks of candidate solutions, could support frequent payments that guide agents' token and tool spending (Appendix A.16). This extension requires validating the report assumptions, resistance to copying and shared-model errors, and whether rewards advance independently assessed goals.

REFERENCES

Asad Ali, Inaam Ilahi, Adnan Qayyum, Ihab Mohammed, Ala Al-Fuqaha, and Junaid Qadir. Incentive-driven federated learning and associated security challenges: A systematic review. *TechRxiv*, 2021.

Anna Arias-Duart, Pablo Agustin Martin-Torres, Daniel Hinjos, Pablo Bernabeu-Perez, Lucia Urcelay Ganzabal, Marta Gonzalez Mallo, Ashwin Kumar Gururajan, Enrique Lopez-Cuena, Sergio Alvarez-Napagao, and Dario Garcia-Gasulla. Automatic evaluation of healthcare LLMs beyond question-answering. In *Proceedings of the 2025 Conference of the Nations of the Americas Chapter of the Association for Computational Linguistics: Human Language Technologies (Volume 2: Short Papers)*, pp. 108–130, Albuquerque, New Mexico, April 2025. Association for Computational Linguistics. ISBN 979-8-89176-190-2. doi: 10.18653/v1/2025.naacl-short.10. URL https://aclanthology.org/2025.naacl-short.10/.

Jacob Austin, Augustus Odena, Maxwell Nye, Maarten Bosma, Henryk Michalewski, David Dohan, Ellen Jiang, Carrie Cai, Michael Terry, Quoc Le, and Charles Sutton. Program synthesis with large language models. *arXiv preprint arXiv:2108.07732*, 2021.

Juan Benet. Ipfs - content addressed, versioned, p2p file system, 2014.

H. Brendan McMahan, Eider Moore, Daniel Ramage, Seth Hampson, and Blaise Agüera y Arcas. Communication-efficient learning of deep networks from decentralized data. In *Proc. AISTATS*, 2017.

Federico Cassano, John Gouwar, Daniel Nguyen, Sydney Nguyen, Luna Phipps-Costin, Donald Pinckney, Ming-Ho Yee, Yangtian Zi, Carolyn Jane Anderson, Molly Q Feldman, Arjun Guha, Michael Greenberg, and Abhinav Jangda. MultiPL-E: A scalable and polyglot approach to benchmarking neural code generation. *IEEE Transactions on Software Engineering*, 49(7), 2023.

Sahil Chaudhary. Code alpaca: An instruction-following llama model for code generation. https://github.com/sahil280114/codealpaca, 2023.

Mark Chen, Jerry Tworek, Heewoo Jun, Qiming Yuan, Henrique Ponde de Oliveira Pinto, Jared Kaplan, Harri Edwards, Yuri Burda, Nicholas Joseph, Greg Brockman, et al. Evaluating large language models trained on code. *arXiv preprint arXiv:2107.03374*, 2021.

Anirban Dasgupta and Arpita Ghosh. Crowdsourced judgement elicitation with endogenous proficiency. In *Proc. of ACM International Conference on World Wide Web (WWW)*, pp. 319–330, 2013.

Li Deng. The MNIST database of handwritten digit images for machine learning research [best of the web]. *IEEE Signal Processing Magazine*, 29(6):141–142, 2012.

Ningning Ding, Zhixuan Fang, and Jianwei Huang. Incentive mechanism design for federated learning with multi-dimensional private information. In *Proc. of IEEE International Symposium on Modeling and Optimization in Mobile, Ad Hoc, and Wireless Networks (WiOPT)*, 2020.

Flower Labs. Flowertune llm leaderboard. https://flower.ai/benchmarks/llm-leaderboard/, 2026. Accessed: 2026-01-29.

Yan Gao et al. FlowerTune: A cross-domain benchmark for federated fine-tuning of large language models. *arXiv preprint arXiv:2506.02961*, 2025.

Amirata Ghorbani and James Zou. Data shapley: Equitable valuation of data for machine learning. In *Proc. of the International Conference on Machine Learning (ICML)*, 2019.

Tianyu Han, Lisa C Adams, Jens-Michalis Papaioannou, Paul Grundmann, Tom Oberhauser, Alexander Löser, Daniel Truhn, and Keno K Bressem. Medalpaca–an open-source collection of medical conversational ai models and training data. *arXiv preprint arXiv:2304.08247*, 2023.

Dan Hendrycks, Collin Burns, Steven Basart, Andy Zou, Mantas Mazeika, Dawn Song, and Jacob Steinhardt. Measuring massive multitask language understanding. In *Proceedings of the International Conference on Learning Representations (ICLR)*, 2021.

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. In *International Conference on Learning Representations*, 2022. URL https://openreview.net/forum?id=nZeVKeeFYf9.

Ruoxi Jia, David Dao, Boxin Wang, Frances Ann Hubis, Nick Hynes, Nezihe Merve Gürel, Bo Li, Ce Zhang, Dawn Song, and Costas J. Spanos. Towards efficient data valuation based on the shapley value. In *Proc. of International Conference on Artificial Intelligence and Statistics (AISTATS)*, 2019.

Di Jin, Eileen Pan, Nassim Oufattole, Wei-Hung Weng, Hanyi Fang, and Peter Szolovits. What disease does this patient have? a large-scale open domain question answering dataset from medical exams. *Applied Sciences*, 11(14):6421, 2021.

Qiao Jin, Bhuwan Dhingra, Zhengping Liu, William Cohen, and Xinghua Lu. PubMedQA: A dataset for biomedical research question answering. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pp. 2567–2577, 2019.

Pang Wei Koh and Percy Liang. Understanding black-box predictions via influence functions. In *Proc. of International Conference on Machine Learning (ICML)*, 2017.

Mengsha Kou, Xiaoyu Xia, Ziqi Wang, Ibrahim Khalil, Runkun Luo, Jingwen Zhou, and Minhui Xue. WinFLoRA: Incentivizing client-adaptive aggregation in federated LoRA under privacy heterogeneity. *arXiv preprint arXiv:2602.01126*, 2026. doi: 10.48550/arXiv.2602.01126. URL https://arxiv.org/abs/2602.01126.

Yann LeCun, Bernhard Boser, John S Denker, Donnie Henderson, Richard E Howard, Wayne Hubbard, and Lawrence D Jackel. Backpropagation applied to handwritten zip code recognition. *Neural computation*, 1(4):541–551, 1989.

Qinbin Li, Yiqun Diao, Quan Chen, and Bingsheng He. Federated learning on non-iid data silos: An experimental study. *arXiv preprint arXiv:2102.02079*, 2021.

Shih-Yang Liu, Chien-Yi Wang, Hongxu Yin, Pavlo Molchanov, Yu-Chiang Frank Wang, Kwang-Ting Cheng, and Min-Hung Chen. DoRA: Weight-decomposed low-rank adaptation. In *International Conference on Machine Learning*, 2024. Oral presentation.

Yang Liu and Jiaheng Wei. Incentives for federated learning: a hypothesis elicitation approach. *arXiv:2007.10596*, 2020.

Zelei Liu, Yuanyuan Chen, Han Yu, Yang Liu, and Lizhen Cui. Gtg-shapley: Efficient and accurate participant contribution evaluation in federated learning. *ACM Trans. Intell. Syst. Technol.*, 13(4), may 2022a. ISSN 2157-6904. doi: 10.1145/3501811. URL https://doi.org/10.1145/3501811.

Zelei Liu, Yuanyuan Chen, Yansong Zhao, Han Yu, Yang Liu, Renyi Bao, Jinpeng Jiang, Zaiqing Nie, Qian Xu, and Qiang Yang. Contribution-aware federated learning for smart healthcare. *Proc. of AAAI Conference on Artificial Intelligence*, 36(11):12396–12404, Jun. 2022b.

Hongtao Lv, Zhenzhe Zheng, Tie Luo, Fan Wu, Shaojie Tang, Lifeng Hua, Rongfei Jia, and Chengfei Lv. Data-free evaluation of user contributions in federated learning. In *Proc. of IEEE International Symposium on Modeling and Optimization in Mobile, Ad hoc, and Wireless Networks (WiOpt)*, 2021.

Macedo Maia, Siegfried Handschuh, André Freitas, Brian Davis, Ross McDermott, Manel Zarrouk, and Alexandra Balahur. WWW'18 open challenge: Financial opinion mining and question answering. In *Companion Proceedings of the The Web Conference 2018*, pp. 1941–1942, 2018.

Pekka Malo, Ankur Sinha, Pekka Korhonen, Jyrki Wallenius, and Pyry Takala. Good debt or bad debt: Detecting semantic orientations in economic texts. *Journal of the Association for Information Science and Technology*, 65(4):782–796, 2014.

Ankit Pal, Logesh Kumar Umapathi, and Malaikannan Sankarasubbu. MedMCQA: A large-scale multi-subject multi-choice dataset for medical domain question answering. *Proceedings of the Conference on Health, Inference, and Learning*, 2022.

Baolin Peng, Chunyuan Li, Pengcheng He, Michel Galley, and Jianfeng Gao. Instruction tuning with gpt-4. *arXiv preprint arXiv:2304.03277*, 2023.

Lloyd S Shapley. A value for n-person games. *Contributions to the Theory of Games*, 2(28):307–317, 1953.

Victor Shnayder, Arpit Agarwal, Rafael Frongillo, and David C. Parkes. Informed truthfulness in multi-task peer prediction. In *Proc. of ACM Conference on Economics and Computation (EC)*, 2016.

Tianshu Song, Yongxin Tong, and Shuyue Wei. Profit allocation for federated learning. In *Proc. of IEEE International Conference on Big Data*, 2019.

Xuezhen Tu, Kun Zhu, Nguyen Cong Luong, Dusit Niyato, Yang Zhang, and Juan Li. Incentive mechanisms for federated learning: From economic and game theoretic perspective. *arXiv:2111.11850*, 2021.

Tianhao Wang, Johannes Rausch, Ce Zhang, Ruoxi Jia, and Dawn Song. A principled approach to data valuation for federated learning. *Federated Learning: Privacy and Incentive*, pp. 153–167, 2020a.

Tianhao Wang, Johannes Rausch, Ce Zhang, Ruoxi Jia, and Dawn Song. A principled approach to data valuation for federated learning, 2020b.

Shuyue Wei, Yongxin Tong, Zimu Zhou, and Tianshu Song. Efficient and fair data valuation for horizontal federated learning. *Federated Learning: Privacy and Incentive*, pp. 139–152, 2020.

L. Witt, U. Zafar, K. Shen, F. Sattler, D. Li, S. Wang, and W. Samek. Decentralized and incentivized federated learning: A blockchain-enabled framework utilising compressed soft-labels and peer consistency. *IEEE Transactions on Services Computing*, 17(4):1449–1464, jul 2024. ISSN 1939-1374. doi: 10.1109/TSC.2023.3336980.

Leon Witt, Mathis Heyer, Kentaroh Toyoda, Wojciech Samek, and Dan Li. Decentral and incentivized federated learning frameworks: A systematic literature review. *IEEE Internet of Things Journal*, 10(4):3642–3663, 2022. doi: 10.1109/JIOT.2022.3231363.

Hongyang Yang, Xiao-Yang Liu, and Christina Dan Wang. Fingpt: Open-source financial large language models. *FinLLM Symposium at IJCAI 2023*, 2023.

Yi Yang, Mark Christopher Siy Uy, and Allen Huang. FinBERT: A pretrained language model for financial communications. *arXiv preprint arXiv:2006.08097*, 2020.

Rongfei Zeng, Chao Zeng, Xingwei Wang, Bo Li, and Xiaowen Chu. A comprehensive survey of incentive mechanism for federated learning, 2021.

Yufeng Zhan, Jie Zhang, Zicong Hong, Leijie Wu, Peng Li, and Song Guo. A survey of incentive mechanism design for federated learning. *IEEE Transactions on Emerging Topics in Computing*, pp. preprint, 2021.

Zhebin Zhang, Dajie Dong, Yuhang Ma, Yilong Ying, Dawei Jiang, Ke Chen, Lidan Shou, and Gang Chen. Refiner: A reliable incentive-driven federated learning system powered by blockchain. *VLDB Endowment*, 14(12):2659–2662, 2021.

Yang Zhao, Jun Zhao, Linshan Jiang, Rui Tan, Dusit Niyato, Zengxiang Li, Lingjuan Lyu, and Yingbo Liu. Privacy-preserving blockchain-based federated learning for IoT devices. *IEEE Internet of Things Journal*, 8(3):1817–1829, 2021.

A Additional details

A.1 FL incentive mechanisms and related work

The introduction motivates a lightweight score for FL systems that use report-based rewards; smart contracts are one optional execution setting. Table 2 makes the relevant trade-offs explicit: what each approach observes, what it rewards, and what computation it requires. FL incentive design includes resource procurement, contribution valuation, model auditing, and information elicitation. These objectives require different observations and need not induce the same client behavior. The comparison includes foundational valuation tools and peer-prediction mechanisms used in FL. The categories overlap: for example, similarity screening can feed a reputation system, and a contribution score can feed a payment rule.

A.1.1 Resource allocation and reputation

Contract-based mechanisms can elicit private cost information subject to incentive-compatibility constraints, rather than simply assuming truthful self-reports (Ding et al., 2020). Their assumptions about observable resources and client types must be distinguished from guarantees about the information contained in an update. Reputation mechanisms can combine history with model-based screening; for example, Zhao et al. (2021) use distance-based filtering and update acceptance to determine reputation and rewards. Such criteria assess consistency with other updates and need not measure marginal improvement on a particular downstream task.

A.1.2 Evaluation-based contribution measures

Leave-one-out measures the utility change from removing a client from the full coalition. Shapley value averages marginal changes over coalitions and satisfies allocation axioms for the specified utility; the two measures can differ substantially for redundant contributions (Ghorbani & Zou, 2019). Exact Shapley evaluation can require all 2^n coalition utilities, while gradient reconstruction, Monte Carlo sampling, and truncation reduce computation (Liu et al., 2022a). These methods still require a meaningful utility, commonly performance on evaluation data, and valuation alone is not a strategic-participation guarantee. Appendix A.12 gives the definitions and FL implementations.

Influence functions approximate training-example effects using derivatives of a fitted model (Koh & Liang, 2017); they should not be identified with exact client leave-one-out evaluation. Validator-based systems address a different problem: Refiner audits submitted models using loss on requester-provided validation datasets (Zhang et al., 2021). Their costs therefore include model evaluation as well as coordination.

A.1.3 Peer prediction and categorical agreement

Peer prediction uses correlations among reports to elicit information without observing the corresponding ground-truth labels. FL instantiations include PTSFD on compressed soft-label reports (Witt et al., 2024), CA on hypotheses evaluated on shared unlabeled inputs (Liu & Wei, 2020), and pairwise CA on quantized model-update coordinates (Lv et al., 2021). Knowledge requirements differ: learned CA estimates correlation signs from reports, whereas DG/MSDG uses a fixed agreement score under categorical structure (Dasgupta & Ghosh, 2013; Shnayder et al., 2016). Both use a penalty for chance agreement. The categorical condition and the associated strong-truthfulness result, up to shared permutations, are established by Shnayder et al. (2016); KFCA uses this scoring principle in the FL instantiations studied here.

Related FL constructions already exploit this structure: Liu & Wei (2020) identify an identity score matrix for conditionally independent, informative binary classifiers, and Lv et al. (2021) score quantized model-update reports with sampled peers. Building on these results, this paper analyzes when fixed scoring remains informative for FL reports and measures computation and reward behavior across models and adapters. In this paper, "knowledge-free" specifically denotes eliminating correlation estimation from reward computation, whereas detail-free CA can learn those correlations without prior distribution knowledge.

Table 2: Representative FL incentives and related valuation methods. Information requirements, computation, and guarantees depend on the stated model; the rows are not a ranking of methods.

Approach and examples	**Information and objective**	**Computation and scope**
Resource contracts (Ding et al., 2020)	Allocate data quantities and payments under private client costs and delays.	Contract optimization; incentive guarantees depend on the type model and observable resource quantities.
Reputation and similarity screening (Zhao et al., 2021)	Use update distances and acceptance history to select and reward clients.	Pairwise distance/filtering work; interpretation depends on the similarity criterion and Byzantine assumptions.
Leave-one-out and Shapley valuation (Ghorbani & Zou, 2019; Liu et al., 2022a)	Measure marginal gains in a chosen utility, often evaluated on labeled data. Shapley averages across coalitions.	Exact Shapley uses up to 2^n distinct coalition utilities; reconstruction, sampling and truncation reduce work. Leave-one-out uses one coalition context.
Influence-function valuation (Koh & Liang, 2017)	Approximate the effect of training examples on a chosen test loss; a related valuation tool rather than an FL payment rule.	Derivative/Hessian computations and approximation assumptions; no inherent strategic-reporting guarantee.
Validator-based auditing: Refiner (Zhang et al., 2021)	A committee evaluates model loss on requester-provided validation data.	Repeated model evaluation and validation/committee assumptions; the evaluation objective determines what is rewarded.
Peer consistency: PTSFD (Witt et al., 2024)	Reward peer consistency of compressed soft-label reports in federated distillation.	Report comparison and smart-contract execution; guarantees depend on the peer-prediction model.
Empirical CA: predictions and update reports (Liu & Wei, 2020; Lv et al., 2021)	Use shared unlabeled inputs or quantized updates; estimate correlation signs for reference peers.	Estimation plus scoring. Peers can be sampled; all-client-pairs estimation is not required by the mechanism family.
DG/MSDG; KFCA instantiations (Dasgupta & Ghosh, 2013; Shnayder et al., 2016)	Reward agreement minus a chance-agreement penalty; no correlation estimation in scoring under categorical structure.	Linear in compared reports. Strong truthfulness is conditional and permits shared permutations; effort also depends on payment versus cost.
Federated LoRA incentives: WinFLoRA (Kou et al., 2026)	Estimate update noise and reward cleaner contributions through aggregation weights under privacy preferences.	Leave-one-out PCA/SVD and client adaptation; targets a noise–utility trade-off, not monetary contribution valuation.

Comparing computational cost. For n clients, p sampled peers per client, m reports and L report symbols, a dense histogram implementation of pairwise CA takes $\mathcal{O}(np(m + L^2))$ work to estimate correlations and score reports, while fixed agreement scoring takes $\mathcal{O}(npm)$. This accounting follows the sampled-peer structure of Lv et al. (2021); the quadratic dependence on clients arises when p grows with n, not from a universal requirement to examine every client pair. Runtime comparisons should use matched peer/report budgets and include report construction, any calibration, and estimation refresh costs. For fixed L, removing estimation can reduce work without changing asymptotic scaling in n or m.

Scope of incentive claims. A discrete report alphabet does not itself establish categorical correlation. Moreover, truthful reporting, effortful training, and marginal contribution to learning are distinct properties. A client that reuses a correlated old update may receive less payment yet save enough training cost to increase its utility; valuable minority-domain data can also disagree with majority reports. These considerations motivate separate analysis of report structure, payment minus effort cost, and downstream value.

A.1.4 Incentives for federated LLM tuning

Recent work also studies incentives specifically for federated LoRA. WinFLoRA estimates noise in adapter updates through leave-one-out principal component analysis and uses aggregation weights to influence clients' privacy–performance choices (Kou et al., 2026). Its reward is influence on the aggregated model, whereas KFCA computes a peer-agreement score that can inform payments. The relevant comparison therefore includes the incentive objective, assumptions, and scoring cost, as well as applicability to LLM adapters.

A.2 Formal assumptions for multi-task peer prediction

We use the multi-task peer-prediction model of Dasgupta & Ghosh (2013); Shnayder et al. (2016) to state conditional incentive guarantees. Let $N = \{1, \dots, n\}$ be the clients and $M = \{1, \dots, m\}$ the tasks. The following assumptions apply to the analysis; the FL mapping does not by itself prove them.

1. **Payment and utility.** Clients are risk-neutral and have utility $U_i = \mathbb{E}[\text{payment}_i] - c_i(e_i)$, with $c_i(1) > c_i(0) = 0$. Scores can be translated into payments, but a positive score alone does not show that training covers its cost.
2. **Tasks and reports.** The tuples $(Y^k, Z_1^k, \dots, Z_n^k)$ are independent draws from a common task distribution. Each reporting strategy is the same signal-to-report function across tasks. Reports are fixed before the random bonus/penalty assignment, with nonempty disjoint task sets. Thus a client's strategy cannot depend on whether its report will enter a bonus or penalty comparison.
3. **Effort and signals.** Each task has latent truth $Y^k \in [L]$. Under effort, client i receives a signal according to a channel $P_i(a \mid y)$. Under no effort, the model uses a baseline $Q_i(a)$ independent of Y^k and of other clients' signals. The channels may differ across clients. Informativeness or diagonal dominance of a channel is not a substitute for the pairwise categorical condition in Definition 2.2.
4. **Conditional independence where used.** The effort and noise calculations assume

$$\mathbb{P}(Z_i^k = a, Z_j^k = b \mid Y^k = y, e_i^k = e_j^k = 1) = P_i(a \mid y) P_j(b \mid y).$$

   The categorical reporting result itself uses the pairwise Δ and the task assumptions, without needing this latent-variable factorization. For the partial-effort lemma, effort choices are independent of the truth and of one another.

Reporting strategies. A randomized reporting strategy is a stochastic matrix $F_i(r \mid a)$, with nonnegative entries and $\sum_r F_i(r \mid a) = 1$. Clients randomize independently conditional on their signals. Deterministic functions $f_i : [L] \to [L]$ are its extreme points, with $F_i(r \mid a) = \mathbb{1}\{r = f_i(a)\}$. Expected reward is a convex combination of rewards for deterministic strategy pairs, so the same maximum is attained by a deterministic strategy pair.

A strategy is *uninformed* when its report distribution does not depend on its signal; a deterministic uninformed strategy is a constant function. Otherwise it is *informed*. Truthful reporting is $f_i(a) = a$. A nontrivial permutation is informed, even though it is not truthful. These distinctions concern dependence on the received signal, not whether the report contains fresh training information or improves the global model.

Scope for FL. Training once produces many correlated predictions or parameter coordinates, whereas the abstract model uses independent, identically distributed tasks and a restricted reporting function. Consequently, coordinate-wise score measurements and categorical-sign checks should be read as empirical evidence for a chosen representation. They do not establish resistance to every strategy using task identity, public training history, shared randomness, or coordinated update construction.

A.3 Expected score

Consider a fixed scoring function $\mathcal{S} : [L] \times [L] \to \mathbb{R}$ and two independently randomized reporting strategies F_1, F_2. Under the task assumptions in Appendix A.2, a bonus comparison uses the same-task joint distribution, while a penalty comparison uses the product of the respective marginals. The expected score per comparison is

$$\begin{aligned} E(F_1, F_2) &= \sum_{a,b} \mathbb{P}(Z_1 = a, Z_2 = b) \sum_{r,s} F_1(r \mid a) F_2(s \mid b) \mathcal{S}(r, s) \\ &\quad - \sum_{a,b} \mathbb{P}(Z_1 = a)\mathbb{P}(Z_2 = b) \sum_{r,s} F_1(r \mid a) F_2(s \mid b) \mathcal{S}(r, s) \qquad (6) \\ &= \sum_{a,b} \Delta(a, b) \sum_{r,s} F_1(r \mid a) F_2(s \mid b) \mathcal{S}(r, s). \qquad (7) \end{aligned}$$

For deterministic strategies this becomes

$$E(f_1, f_2) = \sum_{a,b} \Delta(a, b) \mathcal{S}(f_1(a), f_2(b)).$$

Substituting $\mathcal{S}(r, s) = \mathbb{1}\{r = s\}$ gives Eq. (5). Independent uninformed reporting by either client gives zero: its strategy factors out of the relevant sum, and every row and column of Δ sums to zero. This statement does not apply to signal-independent reports that nevertheless share task-dependent public information or coordinated randomness.

The expression is an expectation under a fixed task distribution. Disjoint penalty indices prevent use of the same task in a penalty pair; they do not on their own establish independence or identical marginals for arbitrary model coordinates.

A.4 Correlated Agreement: definition, cost, and label flipping

CA generalizes the agreement-minus-penalty approach by choosing which report pairs count as agreement using their correlation signs (Shnayder et al., 2016). We distinguish the fixed-score incentive calculation from learning a score matrix from submitted reports.

Definition A.1 (Correlated Agreement (Shnayder et al., 2016))**.** For a fixed signal correlation matrix Δ, the CA score is

$$\mathcal{S}_{\mathrm{CA}}(a, b) = \mathbb{1}\{\Delta(a, b) > 0\}.$$

The multi-task score subtracts the score on independent penalty tasks from the score on a shared bonus task, as in Definition 2.1.

Definition A.2 (Informed truthfulness)**.** A pairwise mechanism is informed-truthful if truthful reporting maximizes expected reward over reporting strategy pairs, and its reward is strictly larger than the reward of any pair containing an uninformed strategy.

Theorem A.3 (Fixed-matrix CA (Shnayder et al., 2016))**.** *Under the task assumptions of Appendix A.2, let $\Delta \neq 0$ and fix the CA score using that signal matrix. Then CA is informed-truthful.*

The proof is reproduced in Appendix A.5. Under the categorical-world condition, its score is exactly the identity-agreement score used by DG/MSDG and the KFCA instantiations. This specialization and its stronger characterization of maximizing profiles are inherited from Shnayder et al. (2016).

A.4.1 Estimating correlations and comparing cost

When the signal distribution is unknown, learned CA estimates the relevant correlation signs from reports. This requires observations for the compared pairs or reference populations, but not inherently every pair of clients. With n clients, p peers per client, m reports and alphabet size L, a dense pairwise histogram implementation takes $\mathcal{O}(np(m + L^2))$ work to estimate correlations and score reports. Fixed agreement scoring takes $\mathcal{O}(npm)$. All-pairs estimation sets p proportional to n; sampling a bounded number of peers avoids that quadratic dependence on clients (Lv et al., 2021).

Comparisons should match peer/report budgets and account for report construction and estimation refresh. At fixed L, omitting estimation can save work without changing asymptotic scaling in n or m. Correlation estimation does not by itself require access to every client's report, rule out decentralized processing, or make online updating impossible. Its information and deployment costs depend on the particular implementation.

A.4.2 A binary label-flipping example

For deterministic strategies and a fixed CA score, expected reward is

$$E(f_1, f_2) = \sum_{a,b} \Delta(a, b)\mathcal{S}_{\mathrm{CA}}(f_1(a), f_2(b)). \tag{8}$$

Consider perfectly agreeing, balanced binary signals, with

$$\Delta^+ = \begin{pmatrix} 0.25 & -0.25 \\ -0.25 & 0.25 \end{pmatrix}, \qquad \mathcal{S}^+ = \begin{pmatrix} 1 & 0 \\ 0 & 1 \end{pmatrix}.$$

With this score fixed, truthful reporting gives 0.5, unilateral flipping gives -0.5, and a shared flip gives 0.5. The same calculation applies to KFCA: it penalizes unilateral flipping relative to a truthful peer, but also retains shared-permutation equivalence.

A different calculation arises if a score is re-estimated from the currently submitted reports. Balanced sequences such as $(1, 0, 1, 0, 1, 0)$ and $(0, 1, 0, 1, 0, 1)$ have reported correlation matrix

$$\Delta^- = \begin{pmatrix} -0.25 & 0.25 \\ 0.25 & -0.25 \end{pmatrix}.$$

Estimating a new score from this matrix selects off-diagonal agreement and produces population expected reward 0.5 for these anticorrelated reports. This illustrates how relearning the scoring rule can reward an inverted correspondence. It is a different scoring-rule comparison from a unilateral deviation against a fixed $\mathcal{S}^+$, and does not contradict Theorem A.3. The finite sequences illustrate the empirical frequencies; the reward values use the corresponding population formula with independent penalty tasks.

Consequently, empirical CA/KFCA attack comparisons depend on which reports train the score matrix, whether it is fixed during the deviation, and which reference peers are used. A universal claim that CA cannot penalize a single label flipper would conflate these cases.

A.5 Informed truthfulness of fixed-matrix CA

We reproduce the fixed-matrix argument underlying the CA result of Shnayder et al. (2016). The scoring matrix in Definition A.1 is fixed from the signal distribution; the proof does not describe an estimator that changes its scoring matrix in response to a deviation.

Proof of Theorem A.3. For truthful reports, Eq. (7) gives

$$E(\mathrm{id}, \mathrm{id}) = \sum_{\Delta(a,b)>0} \Delta(a, b).$$

For any deterministic pair f_1, f_2, the score selects some entries of Δ with coefficients in $\{0, 1\}$, hence

$$E(f_1, f_2) = \sum_{a,b} \Delta(a, b)\mathbb{1}\{\Delta(f_1(a), f_2(b)) > 0\} \leq \sum_{\Delta(a,b)>0} \Delta(a, b).$$

Equality requires including every strictly positive entry and excluding every strictly negative entry. If either strategy is constant, zero row or column sums give expected reward zero. When $\Delta \neq 0$, its zero sums imply at least one positive entry, so truthful reward is strictly positive. Thus equality cannot hold for a profile with an uninformed player. The same conclusions for independent randomized strategies follow by taking convex combinations of deterministic rewards. □

No symmetry of Δ is needed for this pairwise calculation. The stronger characterization of all maximizers under the categorical sign pattern is Proposition 3.2; the remaining equivalence under shared permutations is discussed in Appendix A.7.

A.5.1 Categorical agreement

Proof of Proposition 3.2. Truthful reporting includes every positive diagonal term of Δ and no negative off-diagonal term. Any maximizing profile must do the same: it requires $f_1(a) = f_2(a)$ for every a, and $f_1(a) \neq f_2(b)$ whenever $a \neq b$. Thus the functions coincide and are injective, hence form a shared permutation. Conversely, every shared permutation selects exactly the diagonal terms. □

A.6 Robustness to label flipping

A.6.1 Binary case

Proof of Proposition 3.3. Let $Y \in \{0, 1\}$ have a uniform prior. Conditional on Y, clients receive independent signals with the same symmetric error rate $\alpha < 1/2$. A sampled peer flips its signal with probability λ and otherwise reports it truthfully; this choice is independent of the task and signal. Both types have uniform report marginals.

The match probability with a truthful peer is $A = (1-\alpha)^2 + \alpha^2$; with a flipping peer it is $1 - A = 2\alpha(1-\alpha)$. The truthful focal client's penalty expectation is $1/2$, giving

$$\begin{aligned} E_{\mathrm{id}} &= (1-\lambda)A + \lambda(1-A) - \tfrac{1}{2} \\ &= (1-2\lambda)\left(A - \tfrac{1}{2}\right) = \tfrac{1}{2}(1-2\lambda)(1-2\alpha)^2. \end{aligned} \tag{9}$$

There are exactly four deterministic binary signal-to-report functions: identity, flipping, constant zero, and constant one. Flipping the focal report exchanges match and mismatch probabilities while preserving its uniform marginal, so

$$E_{\mathrm{flip}} = -E_{\mathrm{id}}.$$

Either constant function agrees with the peer with probability $1/2$ on both bonus and penalty tasks, hence has expected reward zero. For $\lambda < 1/2$, $E_{\mathrm{id}} > 0$ and identity strictly exceeds all three alternatives. Independent randomization over these functions cannot improve the payoff. At $\lambda = 1/2$ all four payoffs are zero; for $\lambda > 1/2$ flipping is preferred. □

Here λ is the probability that the sampled peer belongs to the flipping group. Sampling without replacement from a finite population translates this into the fraction among the focal client's eligible peers, rather than automatically the fraction among all clients. The guarantee assumes the same signal quality for honest and flipping peers. It does not cover arbitrary malicious information sources, task-dependent strategies, or collusion based on shared observations outside this signal model.

A.6.2 Multiclass expected score

The following calculation expresses the expected score of a truthful client through class priors and conditional report probabilities. It is not a general best-response or coalition-robustness theorem.

Let $\pi_k = \mathbb{P}(Y = k)$, let $\alpha_{k\ell}$ be the conditional report probabilities of a truthful client, and let $\widetilde{\alpha}_{k\ell}$ describe a malicious peer. The focal client and truthful peers share channel α for this calculation. Assume their reports are independent conditional on Y. Define

$$h_\ell = \sum_k \pi_k \alpha_{k\ell}, \qquad b_\ell = \sum_k \pi_k \widetilde{\alpha}_{k\ell}, \qquad q_\ell = (1-\lambda)h_\ell + \lambda b_\ell.$$

The same-task match probability with a truthful or malicious peer is respectively

$$A = \sum_{k,\ell} \pi_k \alpha_{k\ell}^2, \qquad B = \sum_{k,\ell} \pi_k \alpha_{k\ell} \widetilde{\alpha}_{k\ell}.$$

The penalty pairs a truthful focal-client report, whose marginal is h, with a peer report of marginal q. Therefore it is $\sum_\ell h_\ell q_\ell$, giving

$$\begin{aligned} E_{\text{truth}} &= (1-\lambda)A + \lambda B - \sum_\ell h_\ell q_\ell \\ &= (1-\lambda)\left(A - \sum_\ell h_\ell^2\right) + \lambda\left(B - \sum_\ell h_\ell b_\ell\right). \end{aligned} \tag{10}$$

The focal-client expression is linear in λ for fixed channels. Replacing its penalty by $\sum_\ell q_\ell^2$ would instead use a population-mixture marginal for the focal client. A positive truthful score distinguishes it from an independent uninformed baseline; proving truthful optimality additionally requires comparing each alternative reporting function. No universal multiclass 50% threshold follows from this calculation.

A.7 Shared permutations and unilateral deviations

Under the categorical-world condition, Proposition 3.2 characterizes the maximizing strategy pairs as $f_1 = f_2 = \pi$, where π is a permutation of the report alphabet. For any two reports,

$$\mathbb{1}\{\pi(r_1) = \pi(r_2)\} = \mathbb{1}\{r_1 = r_2\}.$$

Consequently a shared permutation preserves both the bonus and penalty comparisons, including their realized values. The numeric labels in the joint distribution may change; it is the agreement score that is invariant. Truthful reporting is one maximizing profile, not the unique joint profile.

A truthful reference peer. Holding the second client's strategy fixed at identity, equality in Proposition 3.2 requires the first client to use identity as well. Truthful reporting is therefore the unique best deterministic response to a truthful peer within the stated strategy class. If every client instead uses the same permutation, the corresponding coordinated profile is also a best-response equilibrium whenever the relevant pairs are categorical. This distinction is standard in the categorical DG/MSDG setting (Shnayder et al., 2016).

A mixture of truthful and flipping peers. Proposition 3.3 compares deviations against a specified mixture in the uniform, symmetric binary model. It shows that a minority of signal-flipping peers does not make a focal client prefer flipping or constant reporting. It does not remove the shared-permutation profiles from the unrestricted population game or prove that all multiclass minority deviations are unprofitable. The reporting guarantee therefore remains qualified by shared-permutation equivalence.

A.8 Constructing discrete FL reports

The practical pipeline maps FL outputs into a common report alphabet and then applies the same agreement-minus-penalty score. Write this as

$$s_i^k \xrightarrow{T_i} \widetilde{Z}_i^k \in [L] \xrightarrow{\text{sampled-peer scoring}} \widehat{E}_i.$$

The two instantiations in Algorithm 2 use the following mappings.

1. **KFCA-D: predicted labels.** Clients evaluate their locally trained models on common input examples and use the predicted class as the discrete report. The class vocabulary and input index provide shared semantics. Ground-truth labels for these examples are not needed by the agreement score.
2. **KFCA-QP: update signs.** For each client, model or adapter increments relative to the shared checkpoint are mapped to sign reports at corresponding coordinates. A common checkpoint, parameter indexing and sign encoding identify which coordinates and symbols are compared. This removes the requirement for shared evaluation examples while retaining the usual update communication of the FL implementation.

These are report-construction choices. They do not establish that all transformed signal pairs have positive diagonal and negative off-diagonal correlation. The inherited guarantee applies when the induced distribution of $\widetilde{Z}_i, \widetilde{Z}_j$ satisfies Definition 2.2, together with the task assumptions. Empirical checks of the resulting $\widehat{\Delta}$ are evidence about the evaluated settings, rather than a proof that an arbitrary FL output can be made categorical.

Why informativeness alone is insufficient. With conditionally independent signals given a latent truth, the covariance identity is

$$\widetilde{\Delta}(a,b) = \mathrm{Cov}_Y\big(P_i(\widetilde{Z}_i = a \mid Y), P_j(\widetilde{Z}_j = b \mid Y)\big).$$

Informativeness does not fix the signs of these covariances. Common report semantics likewise do not rule out shared confusions or misaligned update directions. The prediction/sign implementation does not require a maximum-a-posteriori decoder based on unknown channels, and it does not regularize an estimated Δ to change the data distribution. In particular, a sign-preserving transformation of an estimated matrix cannot enforce a missing categorical sign pattern.

Scope of the practical extension. KFCA can still compute a score when the categorical condition fails, but the corresponding truthfulness guarantee is unavailable. Quantization supports efficient scoring and comparison across the FL applications studied here; its information loss and incentive consequences remain representation-dependent. Claims of knowledge-free scoring refer to using a fixed agreement score without estimating a correlation-sign matrix for payment. They do not assert that validity of the signal assumptions is known without empirical or modeling work.

A.9 Categorical correlation under non-IID client data

Non-IID training data are compatible with KFCA whenever the induced reports retain categorical correlation. Identical client training distributions are not required; the IID-task assumption in Appendix A.2 concerns scoring tasks. We give a sufficient condition and test the empirical sign pattern under Dirichlet label skew.

Sufficient condition. Let $Y \in [L]$ have probabilities $\pi_a > 0$, with $L \geq 2$. Conditional on the trained channels, suppose clients are independent given Y and

$$P(Z_i = a \mid Y = y) = q_i \mathbb{1}\{a = y\} + (1 - q_i)\mu_i(a), \qquad q_i > 0, \tag{11}$$

where $q_i \leq 1$ and each μ_i is an arbitrary client-specific fallback distribution.

Proposition A.4 (Heterogeneous label preferences)**.** *Under Eq.* (11)*, every pair is categorical, independently of its fallback distributions:*

$$\Delta_{ij}(a,b) = q_i q_j\big(\pi_a \mathbb{1}\{a = b\} - \pi_a \pi_b\big). \tag{12}$$

Proof. Expand $P(Z_i = a, Z_j = b)$ by conditioning on Y and subtract the product of the marginals $q_i\pi_a + (1 - q_i)\mu_i(a)$. All terms containing μ_i or μ_j cancel. Hence $\Delta_{ij}(a,a) = q_i q_j \pi_a(1 - \pi_a) > 0$ and $\Delta_{ij}(a,b) = -q_i q_j \pi_a \pi_b < 0$ for $a \neq b$. □

Together with the task assumptions in Appendix A.2, this gives the conditional reporting guarantee in Proposition 3.2, with truthful expected score $q_i q_j(1 - \sum_a \pi_a^2) > 0$. This extends the noisy-observation example of Shnayder et al. (2016, Section 4.1) to arbitrary local label preferences. Informativeness, conditional independence, and the channel form remain assumptions; a Dirichlet partition alone does not establish them for learned reports.

Toy MNIST illustration. We select a balanced subset of 12,000 MNIST training examples with seed 4242. Following Li et al. (2021), we allocate its 1,200 examples per label across ten clients using class-wise $\mathrm{Dirichlet}(\alpha \mathbf{1}_{10})$ proportions and multinomial counts, without rebalancing. We report $\alpha \in \{100, 1, 0.1\}$ and all five prespecified seeds (11, 22, 33, 44, 55). Smaller α induces stronger label and quantity skew. In one communication round, clients train a 784–64–10 ReLU MLP for two epochs (Adam, learning rate 10^{-3}, batch size 64, no weight decay or dropout). Training uses one CPU thread; clients share an initialization, paired across concentrations within each seed, and their

training partitions are disjoint. Reports are signs of the 50,890 actual increments $\theta_i - \theta_0$, including biases, with $q(0) = +1$.

All $\binom{10}{2} \times 5 = 225$ pair checks per concentration satisfies the strict empirical categorical condition; Figure 5 aligns allocations with measured matrices. For binary reports, $\widehat{\Delta} = d \left(\begin{smallmatrix} 1 & -1 \\ -1 & 1 \end{smallmatrix} \right)$, so the condition is $d > 0$. At $\alpha = 0.1$, excluding coordinates where either increment is zero leaves 133/225 passing pairs (versus 225/225 at $\alpha = 100, 1$). Shared zero coordinates therefore contribute to the full-coordinate result. Pair checks are dependent, and these observations establish compatibility for the evaluated representation, not robustness to arbitrary heterogeneity or a training-convergence guarantee. The recorded partitions, increments, and independent covariance validation are retained with the experiment.

A.10 Effort and shirking

Consider the binary-effort model in Appendix A.2. Client i exerts effort with probability η_i, independently of the latent truth, other effort choices, and signal-generation randomness. Under effort its signal channel is $P_i(a \mid Y)$; otherwise it draws independently from $Q_i(a)$, which does not depend on Y.

Lemma A.5 (Covariance under independent shirking)**.** *Under conditional independence given Y,*

$$\Delta(a, b) = \eta_1 \eta_2 \operatorname{Cov}_Y \big(P_1(a \mid Y), P_2(b \mid Y) \big) = \eta_1 \eta_2 \Delta_{\inf}(a, b),$$

where $\Delta_{\inf}$ is the signal correlation matrix when both clients exert effort.

Proof. Conditional report probabilities are $\eta_i P_i(a \mid Y) + (1 - \eta_i) Q_i(a)$. Conditional independence makes their covariance over Y equal to $\Delta(a, b)$. The Q_i terms are constant in Y and disappear from the covariance, leaving the product $\eta_1 \eta_2$. □

If $\Delta_{\inf}$ is categorical and $\eta_1 \eta_2 > 0$, the sign pattern is preserved and the expected truthful score is scaled by $\eta_1 \eta_2$. If either probability is zero, $\Delta = 0$ and the expected score is zero. This result distinguishes informative training from the stipulated independent baseline.

A utility comparison additionally includes effort costs. If a score is converted into payment with positive multiplier β, then under this model exerting effort is preferred to independent shirking only when its expected score gain, multiplied by β, exceeds the incremental training cost. Replaying a correlated earlier update, following a public update trajectory, or copying another client need not produce the independent baseline Q_i and is not covered by this lemma. Empirical reward separation for such strategies must therefore be assessed separately from its theoretical effort conclusion.

A.11 KFCA-QP and KFCA-D algorithms

In round t, let M be the scoring task-index set: common input examples for KFCA-D or corresponding parameter coordinates for KFCA-QP. Each client produces reports $R_{i,t}^k$ before the bonus and penalty subsets are sampled. The subsets M_b, M_1, M_2 are nonempty and pairwise disjoint. Here M_1, M_2 denote the first and second clients' penalty sets, as in Definition 2.1. For each client, sample p distinct peers uniformly from the other participating clients. Let $N_t \subseteq N$ be the active set in round t, with $n_t = |N_t| \geq 2$ and $1 \leq p \leq n_t - 1$. In Algorithm 1, N denotes this active set.

The following algorithm implements the normalized score in Eq. (4). Write $\widehat{E}_{i,t}$ for the round-indexed client score $\widehat{E}_i$. Its output is a score; translating it into a payment requires specifying the payment scale and any transfer rule.

For the binary formulation, write $q(x) = +1$ for $x \geq 0$ and $q(x) = -1$ for $x < 0$. This specifies a tie convention for the algorithm below. The same coordinate indexing and encoding are used for every compared client.

Scoring runs alongside FedAvg. This algorithm does not use $\widehat{E}_{i,t}$ to reweight FedAvg or remove clients. In round t, scoring costs $\mathcal{O}(n_t p m)$ for $m = |M|$; model training, report construction and any downstream payment execution are separate costs. Sampling disjoint task sets implements the score, while its interpretation through the signal matrix also requires the task-distribution assumptions in Appendix A.2.

Algorithm 1 Calculate Client i's Score at Round t

Require: Reports $\{R^k_{u,t}\}_{u\in N, k\in M}$; nonempty disjoint sets M_b, M_1, M_2; client i; peer count p
Ensure: Normalized score $\widehat{E}_{i,t}$

$P_i \leftarrow$ sample p distinct clients uniformly from $N \setminus \{i\}$
$\widehat{E}_{i,t} \leftarrow 0$
for each peer $j \in P_i$ **do**
  for each bonus task $k \in M_b$ **do**
    $u \leftarrow \mathrm{Unif}(M_1)$; $v \leftarrow \mathrm{Unif}(M_2)$, independently
    $\widehat{E}_{i,t} \leftarrow \widehat{E}_{i,t} + \mathbb{1}\{R^k_{i,t} = R^k_{j,t}\} - \mathbb{1}\{R^u_{i,t} = R^v_{j,t}\}$
  end for
end for
$\widehat{E}_{i,t} \leftarrow \widehat{E}_{i,t}/(p|M_b|)$
return $\widehat{E}_{i,t}$

Algorithm 2 KFCA Scoring Alongside FedAvg

Require: Clients N; active sets $\{N_t\}_{t=0}^{T-1}$; rounds T; initial model θ_0; method in $\{\text{KFCA-D}, \text{KFCA-QP}\}$; task indices M with $|M| \geq 3$; peer count p
Ensure: Global model θ_T and scores $\{\widehat{E}_{i,t} : i \in N_t\}$

for $t = 0$ to $T-1$ **do**
  for each client $i \in N_t$ **do**
    $\theta_{i,t} \leftarrow$ train locally from θ_t
    if method is KFCA-D **then**
      $R^k_{i,t} \leftarrow$ predicted label of $\theta_{i,t}$ on common input k, for all $k \in M$
    else if method is KFCA-QP **then**
      $\Delta\theta_{i,t} \leftarrow \theta_{i,t} - \theta_t$ on the scored model/adapter parameters
      $R^k_{i,t} \leftarrow q(\Delta\theta_{i,t}[k])$, for all $k \in M$
    end if
  end for
  After reports are fixed, sample nonempty disjoint subsets $M_b, M_1, M_2 \subset M$
  for each client $i \in N_t$ **do**
    $\widehat{E}_{i,t} \leftarrow$ Algorithm 1 using active set N_t, these reports, and task subsets
  end for
  $\theta_{t+1} \leftarrow \mathrm{FedAvg}(\{\theta_{i,t}\}_{i\in N_t})$
end for
return $\theta_T, \{\widehat{E}_{i,t}\}$

A.12 Shapley valuation and comparison details

A.12.1 Definition and axioms

The Shapley value allocates coalition utility by averaging each client's marginal utility over possible participation orders. Let N be the set of clients, $n = |N|$, and $v : 2^N \to \mathbb{R}$ a coalition utility. For client $i \in N$,

$$\phi_i(v) = \frac{1}{n!} \sum_{\Pi \in \mathcal{P}(N)} \left[v\left(S_i^\Pi \cup \{i\}\right) - v\left(S_i^\Pi\right) \right], \tag{13}$$

where $\mathcal{P}(N)$ contains all $n!$ permutations and S_i^Π contains the clients preceding i. For example, if $\Pi = (4, 2, 1, 3)$, then $S_3^\Pi = \{4, 2, 1\}$. Shapley valuation satisfies four allocation axioms (Shapley, 1953):

1. **Efficiency:** $\sum_{i\in N} \phi_i(v) = v(N) - v(\emptyset)$, which reduces to $v(N)$ when $v(\emptyset) = 0$.
2. **Symmetry:** If $v(S \cup \{i\}) = v(S \cup \{j\})$ for every $S \subseteq N \setminus \{i, j\}$, then $\phi_i(v) = \phi_j(v)$.
3. **Null player:** If $v(S \cup \{i\}) = v(S)$ for every $S \subseteq N \setminus \{i\}$, then $\phi_i(v) = 0$.
4. **Additivity:** For coalition utilities u and v, $\phi_i(u + v) = \phi_i(u) + \phi_i(v)$ for every client i.

These axioms characterize an allocation relative to the chosen utility. They do not, by themselves, establish incentives for fresh training or truthful reporting.

A.12.2 Shapley value in federated learning

Let $\mathbf{A}$ denote a training procedure initialized at θ_{init}, using clients' datasets $D^S = \{D^i : i \in S\}$. This notation allows data to remain distributed. With an evaluation function V and held-out dataset D^{test}, one possible coalition utility is

$$v(S) = V\left(\mathbf{A}(\theta_{\text{init}}, D^S), D^{\text{test}}\right). \tag{14}$$

Larger utility means better performance, for example accuracy or negative loss. Evaluation data measure utility; they are not used to train the coalition model.

A.12.3 Example: Shapley value in FL with three clients

Consider an illustrative three-client MNIST game, $N = \{1, 2, 3\}$, with initial accuracy $v(\emptyset) = 0.1$ and the coalition accuracies in Table 3. These are example values, not measurements from the experiments:

Table 3: Illustrative coalition accuracies $v(S)$ for the three-client example.

$\mathbf{S}$	$\mathbf{v(S)}$
$\emptyset$	0.1
$\{1\}$	0.7
$\{2\}$	0.75
$\{3\}$	0.8
$\{1, 2\}$	0.85
$\{1, 3\}$	0.9
$\{2, 3\}$	0.95
$\{1, 2, 3\}$	0.98

For three clients, marginal contributions to the empty or two-client coalition have weight $1/3$; each marginal contribution to a one-client coalition has weight $1/6$. Thus,

$$\begin{aligned}
\phi_1 &= \tfrac{1}{3}(0.60) + \tfrac{1}{6}(0.10) + \tfrac{1}{6}(0.10) + \tfrac{1}{3}(0.03) = \tfrac{73}{300}, \\
\phi_2 &= \tfrac{1}{3}(0.65) + \tfrac{1}{6}(0.15) + \tfrac{1}{6}(0.15) + \tfrac{1}{3}(0.08) = \tfrac{88}{300}, \\
\phi_3 &= \tfrac{1}{3}(0.70) + \tfrac{1}{6}(0.20) + \tfrac{1}{6}(0.20) + \tfrac{1}{3}(0.13) = \tfrac{103}{300}.
\end{aligned}$$

Their sum is $0.88 = v(N) - v(\emptyset)$, as required by efficiency.

A.12.4 Computational and evaluation requirements

Exact Shapley calculation for a general utility can require all 2^n coalition utilities. Although Equation 13 contains $n!$ orderings, those orderings reuse coalitions and do not require $n!$ separate training runs. Ten clients give $2^{10} = 1{,}024$ coalitions, including the empty coalition whose utility can be specified directly. Whether each nonempty coalition requires retraining depends on how its utility is defined and computed. Existing methods reduce this work through reconstruction, truncation and sampling:

1. **Model reconstruction.** Reconstruct coalition models from recorded updates instead of retraining them (Song et al., 2019; Wei et al., 2020; Wang et al., 2020b; Liu et al., 2022a):

$$\widetilde{v}(S) = V\left(\theta^{\text{global}} + \sum_{i \in S} \frac{|D^i|}{\sum_{j \in S} |D^j|} \Delta\theta^i, D^{\text{test}}\right), \qquad S \neq \emptyset. \tag{15}$$

   This reconstruction-based utility need not equal the utility of a separately retrained coalition. It requires the corresponding updates and model evaluations.

2. **Truncation.** Skip low-impact rounds or stop a sampled permutation when its utility is sufficiently close to a reference utility (Ghorbani & Zou, 2019; Liu et al., 2022a). This trades computation against approximation error; marginal utility need not decrease monotonically for every learning problem.

3. **Sampling.** Estimate the permutation average from sampled orders (Ghorbani & Zou, 2019; Jia et al., 2019; Wang et al., 2020a; Liu et al., 2022a):

$$\widehat{\phi}_i = \frac{1}{K}\sum_{k=1}^{K}\left[v\left(S_i^{\Pi_k}\cup\{i\}\right) - v\left(S_i^{\Pi_k}\right)\right], \qquad \Pi_k \sim \mathrm{Unif}(\mathcal{P}(N)). \tag{16}$$

   The sample count depends on variance, precision and the stopping rule. The number of model evaluations also depends on how many coalitions are visited per sampled permutation.

Accuracy-based Shapley valuation requires an agreed evaluation dataset and metric. Their choice determines which contributions are valued. Evaluation and payment can be distributed among validators; a single central authority is not intrinsic to Shapley valuation. These requirements distinguish performance valuation from peer-prediction scoring, which targets a different incentive objective.

A.12.5 Scalability of KFCA: wall-clock measurements

With n clients, p peers per client and m scored tasks, direct categorical scoring takes $\mathcal{O}(npm)$ work, excluding report construction and diagnostics. Figure 9 reports the existing measurements on an Apple Silicon M2 Pro with 16 GB RAM: KFCA-QP scores a CNN with 5,280 parameters, and KFCA-D scores reports on 10,000 inputs. Fixed p makes scoring work linear in n. Averaging independent peer scores of variance σ^2 would give standard error $\sigma/\sqrt{p}$; shared clients and tasks can introduce dependence, so this expression is not an unconditional guarantee for the experiments. Learned CA can also sample peers, and comparisons should distinguish estimation costs from the number of peer comparisons.

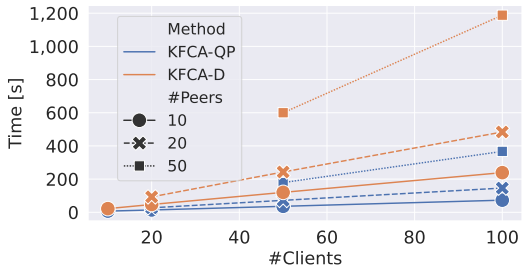


Figure 9: Computation time as a function of client and peer counts. KFCA-QP on a simple CNN with 5,280 parameters and KFCA-D on a 10,000-input dataset, measured on a single Apple Silicon M2 Pro (16 GB RAM).

A.12.6 Comparison methods

1. **Shapley Value:** Original SV calculation for FL client contribution evaluation, based on Equation 13.

2. **GTG-Shapley** (Liu et al., 2022a;b): Uses coalition-model reconstruction with gradient updates from clients, guided sampling, in-round truncation of client permutations, and between-round truncation that drops an entire round of SV calculation if the remaining marginal utility or accuracy gain is small.

3. **TMC-Shapley** (Ghorbani & Zou, 2019): Uses Monte Carlo permutation sampling with truncation when further coalition evaluations have small estimated impact.

4. **Group Testing** (Jia et al., 2019): Estimates pairwise Shapley differences and recovers an allocation by solving a feasibility problem.

5. **MR** (Song et al., 2019): In each round of the FL process, reconstructs the model of a subset of participants using their gradient updates. The final SV for a participant is obtained by summing up their SVs across all rounds.

6. **Fed-SV** (Wang et al., 2020b): A group testing-based estimation approach. Performance of subsets used for estimating Shapley differences is evaluated on a sub-model reconstructed using participants' model parameters, and Shapley values are independently estimated in each round and subsequently aggregated.

7. **TMR** (Wei et al., 2020): A gradient update-based method for SV calculation, incorporating a decay factor to include SV from previous rounds and a truncation factor to omit unimportant sub-model reconstructions.

8. **Correlated Agreement (CA-QP)** (Lv et al., 2021): Estimates a correlation score matrix from quantized update reports; the original method permits sampling peers.

9. **CA-D** (Liu & Wei, 2020): Estimates correlation structure from classifiers' predictions on shared evaluation inputs. These inputs need not carry ground-truth labels for peer scoring.

10. **KFCA-QP:** Applies the inherited categorical agreement-minus-chance score to quantized update reports.

11. **KFCA-D:** Applies that score to predictions on shared inputs under the categorical condition in Definition 2.2; equivalently, $\mathcal{S}(a, b) = \mathbb{1}\{a = b\}$.

12. **MC KFCA-D/QP:** Subsamples scoring tasks, using predictions or parameter coordinates, respectively.

A.12.7 Experimental settings and reproducibility

MNIST comparison setup. We follow the federated evaluation protocol of Wei et al. (2020) and use a simple CNN (21,840 parameters) trained on MNIST (Deng, 2012). The recorded setup describes five silos and 10 clients, with client pairs sampled from each silo (clients 1–2 from silo 1, clients 3–4 from silo 2, etc.). This is a different CNN from the 5,280-parameter timing model.

Client heterogeneity/noise cases. We evaluate five settings:

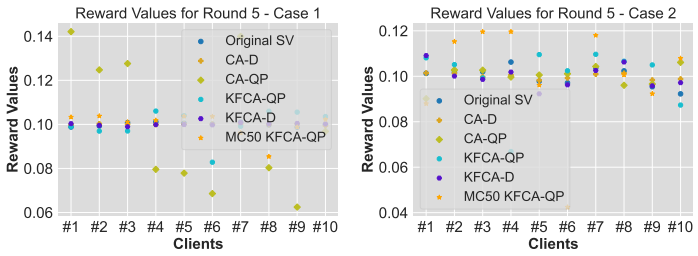
1. **Case 1 (i.i.d., same size):** Each client receives 10,840 images sampled to be balanced across digits.

2. **Case 2 (label skew, same size):** Each client still has 10,840 images. Clients 1–2 have 80% digits `1/2` and 20% spread over remaining digits; clients 3–4 are skewed to `3/4`, etc.

3. **Case 3 (size skew, i.i.d. labels):** All clients have the same label distribution, but different dataset sizes with recorded size levels: 10% (clients 1–2), 15% (3–4), 20% (5–6), 25% (7–8), 30% (9–10).

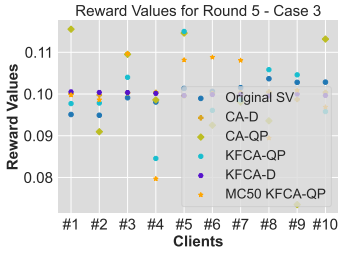
4. **Case 4 (label noise, same size):** We flip labels at increasing rates: 0% (clients 1–2), 5% (3–4), 10% (5–6), 15% (7–8), 20% (9–10).

5. **Case 5 (feature noise, same size):** We add Gaussian feature noise at recorded levels: 0% (clients 1–2), 5% (3–4), 10% (5–6), 15% (7–8), 20% (9–10).

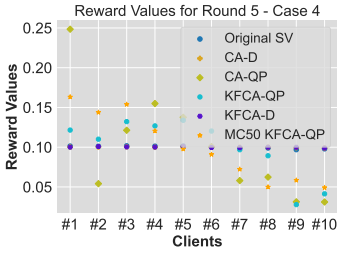


(a) Case 1

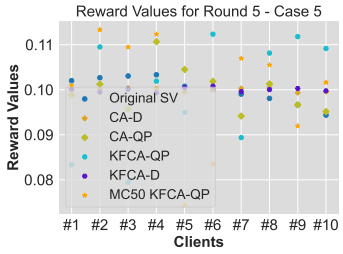
(b) Case 2



(c) Case 3



(d) Case 4



(e) Case 5

Figure 10: Per-client rewards across the five MNIST settings

Reproducibility is limited by unspecified client-data overlap, the denominator for Case 3 size percentages, and the Gaussian-noise parameterization in Case 5. In particular, a corrupted-feature fraction and a noise magnitude represent different perturbations. The archived preparation notebook instead uses additive Gaussian standard deviations $0, 0.25, 0.5, 0.75, 1$ for its five silos; its correspondence to the plotted runs has not been verified. We therefore retain the recorded case labels rather than infer a noise distribution from those percentages.

Implementation provenance. The archived peer-scoring code differs from Algorithm 1 in quantization and penalty-index handling; some Monte Carlo routines also accumulate running pair totals rather than individual comparison scores. The code revision used for each historical figure has not been established. These plots therefore compare recorded implementations, while Appendix A.14 isolates scoring costs with matched reports and indices.

Reward normalization and stopping rule. For the distance comparisons, each method's reward vector is normalized by its sum, which must be nonzero. The reported metrics are cosine distance, Euclidean distance, and the maximum coordinate difference from the reference Shapley allocation. The Shapley estimators use a mean relative-change stopping threshold of 0.05 over the preceding $K = 10$ estimates (Liu et al., 2022a):

$$\frac{1}{nK}\sum_{k=1}^{K}\sum_{i=1}^{n}\frac{|\phi_i^h - \phi_i^{h-k}|}{|\phi_i^h|} < 0.05, \qquad K = 10.$$

Here h indexes successive estimates. This expression is defined when its denominators are nonzero; the recorded configuration does not specify handling of zero entries.

Methods compared. We compare KFCA to CA variants and efficient Shapley estimators; the baselines are described in Appendix A.12.6. Figure 6b presents the complete five-case comparison, and Figure 10 shows per-client rewards. Closeness to Shapley values is an empirical comparison; it does not establish Shapley axioms or incentive compatibility for peer scores.

A.13 Federated LLM fine-tuning: settings and results

This appendix supplies the configurations, evaluation suites, categorical-condition measurements, and attack definitions for Section 4. We follow the four FlowerTune configurations for general NLP, finance, medicine, and code (Gao et al., 2025). The base LLM is frozen while clients train LoRA/DoRA adapters locally and communicate adapter updates for FedAvg aggregation. For a weight matrix $W \in \mathbb{R}^{d\times k}$, LoRA represents the weight adaptation as $\Delta W = BA$ (absorbing its scaling factor into the factors), where $A \in \mathbb{R}^{r\times k}$ and $B \in \mathbb{R}^{d\times r}$, with $r \ll \min(d, k)$; DoRA additionally decomposes weight magnitude and direction (Hu et al., 2022; Liu et al., 2024).

A.13.1 Recorded experimental setup

Table 4 summarizes the four runs; Tables 5 and 6 report the recorded training settings and per-suite results. All downstream scores describe the federated training runs in which rewards are evaluated. Benchmark evaluation uses the listed labeled suites; KFCA-QP itself scores the communicated adapter updates without those labels. These scores establish the performance of the FedAvg configurations in which reward scores are computed, rather than a gain caused by changing aggregation or participation. Each run has ten communication rounds with two clients participating per round. The sign reports serve reward computation, while adapter updates also serve their usual aggregation role.

Table 4: FlowerTune results after ten rounds. Scores average accuracy within each domain's suite, except code, which uses pass@1; they are not a shared metric across domains. Communication totals adapter uploads and downloads.

Domain	Base model	Clients	Adapter rank	Score	Comm.
General NLP	InternLM3-8B-Instruct	20	LoRA 16	69.25	2.9 GB
Finance	Qwen2.5-7B	20	LoRA 32	84.58	30.1 GB
Medical	Llama3.1-Aloe-Beta-8B	20	LoRA 32	63.57	2.0 GB
Code	Qwen3-8B	10	LoRA/DoRA 8	65.27	3.5 GB

Table 5: Recorded FlowerTune federated adapter-tuning setup. All runs use FedAvg, 4-bit quantization, sequence length 512, and three local epochs capped at ten local steps per round. Client splits are IID and equally sized.

Setting	**General NLP**	**Finance**
Base model	InternLM3-8B-Instruct	Qwen2.5-7B
Training data	alpaca-gpt4 (Peng et al., 2023)	fingpt-sentiment-train (Yang et al., 2023)
Clients	20	20
Participation fraction	0.1	0.1
Clients per round	2	2
Adapter	LoRA ($r = 16$, $\alpha = 32$)	LoRA ($r = 32$, $\alpha = 64$)
Target modules	q/k/v/o projections	All projections and MLP
Batch $\times$ accumulation	1×4	16×1
Maximum learning rate	5×10^{-5}	5×10^{-4}
Minimum learning rate	10^{-6}	5×10^{-5}

Setting	**Medical**	**Code**
Base model	Llama3.1-Aloe-Beta-8B	Qwen3-8B
Training data	medical-flashcards (Han et al., 2023)	code-alpaca-20k (Chaudhary, 2023)
Clients	20	10
Participation fraction	0.1	0.2
Clients per round	2	2
Adapter	LoRA ($r = 32$, $\alpha = 128$)	LoRA+DoRA ($r = 8$, $\alpha = 16$)
Target modules	q_proj and v_proj	All projections and MLP
Batch $\times$ accumulation	16×2	2×4
Maximum learning rate	10^{-4}	5×10^{-5}
Minimum learning rate	10^{-5}	5×10^{-6}

Table 6: FlowerTune evaluation suites and downstream performance after 10 federated rounds (FlowerTune configurations). **Comm.** is total communicated adapter payload (upload+download) aggregated over all rounds.

Domain	Eval datasets	Metric	Comm.	Avg.	Suite scores (labeled)
General NLP	MMLU (STEM/Hum./Soc. Sci.) (Hendrycks et al., 2021)	Acc.	2.9 GB	69.25	STEM 66.13; Soc. Sci. 80.76; Hum. 60.87
Finance	FPB (Malo et al., 2014), FIQA (Maia et al., 2018), TFNS (Yang et al., 2020)	Acc.	30.1 GB	84.58	FPB 85.89; FIQA 83.22; TFNS 84.63
Medical	PubMedQA (Jin et al., 2019), MedMCQA (Pal et al., 2022), MedQA (Jin et al., 2021), CareQA (Arias-Duart et al., 2025)	Acc.	2.0 GB	63.57	PubMedQA 74.80; MedMCQA 55.39; MedQA 59.31; CareQA 64.79
Code	MBPP (Austin et al., 2021), HumanEval (Chen et al., 2021), MultiPL-E (JS/C++) (Cassano et al., 2023)	Pass@1	3.5 GB	65.27	MBPP 56.20; HumanEval 70.73; MultiPL-E JS 68.32; C++ 65.84

A.13.2 Report definitions and empirical correlation

The intended KFCA-QP construction scores local adapter increments relative to the received checkpoint, $\Delta\theta_{i,t} = \theta_{i,t}^{\text{local}} - \theta_t$, with one task per parameter coordinate and report $R_{i,t}^k = q(\Delta\theta_{i,t}[k]) \in \{-1, +1\}$. Section 3 specifies $q(0) = +1$ for this construction. The recorded four-domain diagnostics do not establish whether they used local-increment signs or adapter-state signs, and their zero-coordinate convention is undocumented; Appendix A.14.1 examines this distinction. Figure 11 therefore documents the correlation pattern of the recorded reports, rather than a verified application of the intended update definition. Every displayed matrix has positive diagonals and negative off-diagonals. Neither the binary encoding nor these finite-coordinate checks establish the population task assumptions.

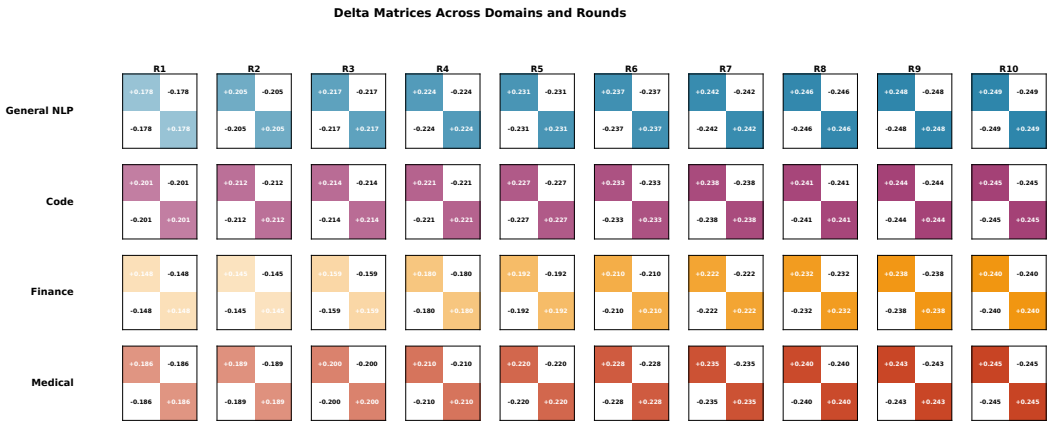


Figure 11: Recorded report-correlation matrices across four domains and ten rounds. The state-versus-increment distinction is examined in Appendix A.14.1.

A.13.3 MANIPULATED REPORTS

The following reported attack definitions describe transformations evaluated against an honest peer from the same communication round. They specify the intended interventions; reproducing the historical plots also requires recovering their report representation and matching participant identities across rounds.

Temporal manipulation.

- **Stale update (R1):** Reuse the round-1 report instead of the current report. This models a potential alternative to repeated training; the diagnostic alone does not establish a training-cost saving.
- **Lagged update ($t-k$):** Reuse a report from round $t-k$ in round t, with $k \in \{2,3,4,5\}$, when the referenced report is available.

Uninformative reports.

- **Zero attack:** The client submits an all-zero update. Under the fixed quantization convention this produces a constant report, whose same-task agreement is canceled by the penalty baseline in expectation.
- **Random attack:** The client submits $\Delta\theta_{s,t} \sim \mathcal{N}(0, \sigma^2 I)$, with the noise scaled to match the honest update's variance.

Partial or reversed signals.

- **Sparse attack ($\rho\%$):** The client retains the honest values for $\rho\%$ of coordinates and replaces the rest with random values, for $\rho \in \{25, 50, 75\}$.
- **Sign flip:** The client negates every binary report, $R^k_{s,t} \leftarrow -R^k_{s,t}$. On nonzero update coordinates, this is equivalent to reversing the underlying update sign.

A.13.4 REWARD TRAJECTORIES AND MEAN SCORES

Status of the historical diagnostics. Figures 11 and 12 show the recorded diagnostics, with reported means below. The checkpoint audit in Appendix A.14.1 distinguishes adapter-state signs from local-update signs and does not reproduce these ten-round means. Until the original analysis pipeline establishes the representation used in each plot, the historical scores should not be treated as verified local-update incentive measurements. The matched replay in Appendix A.14.1 uses an explicit, auditable update definition.

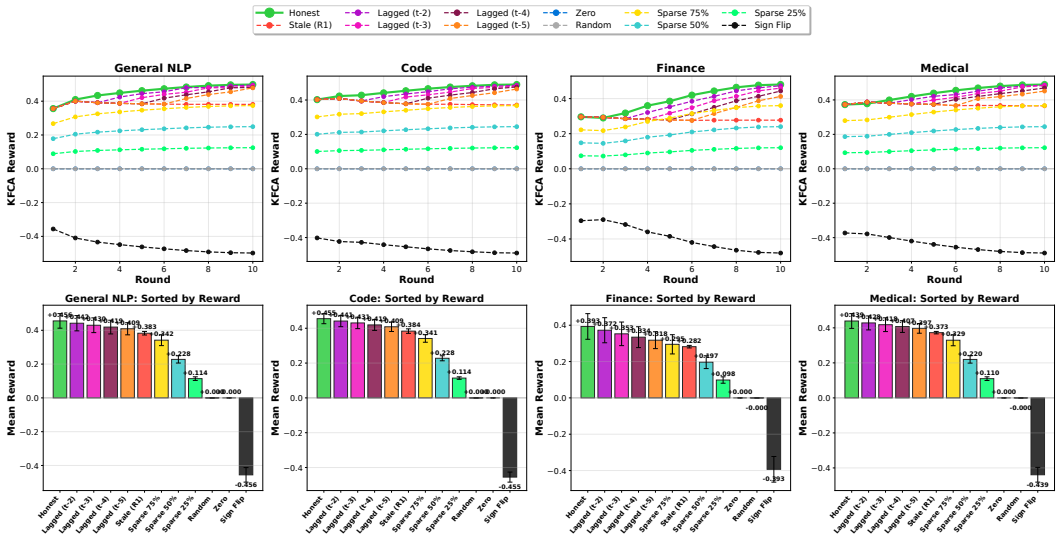


Figure 12: Recorded rewards: per-round trajectories (top) and strategy means (bottom). Report-definition audit: Appendix A.14.1.

Figure 12 shows ten-round reward trajectories (top) and four-domain mean rewards (bottom). Honest reports have the highest mean score in every domain: 0.456 for general NLP, 0.455 for code, 0.393 for finance, and 0.439 for medicine. Sign flips receive the corresponding negative scores; zero and random reports score approximately zero. Sparse reports retain approximately 75%, 50%, or 25% of honest reward when the same fraction of coordinates remains honest, consistent with the values in the plotted bars.

In these recorded diagnostics, temporal manipulation retains substantial reward but reduces it relative to current honest reports. The mean stale-R1 scores are 0.383, 0.384, 0.282, and 0.373 for general NLP, code, finance, and medicine, respectively, corresponding to approximately 72–85% of the honest mean. Lagged-$t-2$ scores are 0.442, 0.441, 0.372, and 0.428 in the same domain order. Longer lags generally receive lower rewards. These results characterize the scoring rule's response to the tested reports: stale signals can still agree with current peers, so a reduced score is not itself a guarantee that repeated training maximizes utility after accounting for its cost.

A.14 Matched Scoring and Effort Audit

Question and controlled comparison. We isolate the cost of estimating a binary correlation score from the cost of applying it. All methods use the same reports, directed peers, and bonus/penalty indices. Half the coordinates form a calibration set; the remainder is split into disjoint bonus and two penalty sets. Learned scoring estimates a 2×2 joint histogram on calibration coordinates and uses its correlation signs on the held-out scoring coordinates. Fixed scoring uses equality on the same indices. A fixed identity-matrix lookup is numerically identical and separates the choice of score from lookup overhead. This is a scoring-kernel ablation, not a reproduction of the complete cross-fitted Pairwise Correlated Agreement protocol: fixed scoring leaves the calibration budget unused, and each method could use its total budget differently.

Synthetic scaling check. Balanced binary latent signals with independent 15% errors provide a controlled test of computation. The six (n, m, p) configurations are $(10, 10^3, 2)$, $(10, 10^4, 2)$, $(10, 10^5, 2)$, $(50, 10^4, 2)$, $(100, 10^4, 2)$, and $(10, 10^4, 5)$. We use seeds 1103, 2207, 3301, 4409, and 5501, one warmup, and nine interleaved timing repetitions. Timings use the Apple M4 Pro environment specified below, with numerical-library thread counts fixed at one. At $n = 10, m = 10^4, p = 2$, median fixed, refreshed-learned, and cached scoring times are 0.265, 0.974, and 0.432 ms. Report construction, common sampling, and correlation estimation take 0.031, 0.507, and 0.542 ms separately. Learned and cached scores match fixed scores on these categorical draws. These synthetic results test arithmetic and scaling, not FL learning or strategic participation. At fixed alphabet and

peer count, both optimized approaches are linear in report length; the saving is a constant factor and depends on cache refresh frequency.

A.14.1 SAME-CLIENT ADAPTER REPLAY

We independently replay the recorded general-NLP checkpoints for two clients appearing in both rounds 4 and 10 of the historical run. Each state contains 19,267,584 coordinates in 384 adapter tensors. The round convention is checked against earlier client averages: the mean of the round-1 client states matches `global_round_2`, and the mean of the round-2 client states matches `global_round_3`, with relative ℓ_2 errors below 3×10^{-8}. Thus `global_round_t` is the received, pre-training checkpoint for client round t. We align tensor keys and compute $q(\theta_{i,t}^{\mathrm{local}} - \theta_t)$; this differs from $q(\theta_{i,t}^{\mathrm{local}})$.

The stale action in round 10 submits that same client's own round-4 increment. Each unilateral replay is evaluated against the unchanged current honest peer. Table 7 reports full empirical same-task agreement minus the product-of-marginals chance baseline, independently checked as twice the binary covariance. Three sampled-task seeds (1103, 2207, 3301) reproduce the ordering. These are correlated model coordinates from one training run, not independent training replicates or an equilibrium simulation.

Table 7: Same-client round-10 replay. Both clients' actual update signs receive higher scores when they reuse their own round-4 increment. Adapter-state signs give a different ordering. Values are full-coordinate empirical diagnostics; A and B denote the two recurring clients.

Report	**Client**	**Fresh**	**Own stale R4**	**Fresh − stale**
Update signs	A	0.05596	0.07615	−0.02019
Update signs	B	0.05596	0.06695	−0.01099
State signs	A	0.49857	0.45496	+0.04360
State signs	B	0.49857	0.45479	+0.04378

What the replay establishes. The fresh and stale update comparisons all have positive binary covariance: the categorical sign condition holds empirically, yet stale updates score more. This does not contradict the reporting result, which restricts the task distribution and signal-to-report strategies; correlated temporal replay falls outside the independent-shirking model. State-sign rewards are much larger and resemble the scale of the historical diagnostics. That resemblance does not identify their original analysis code. Resolving that provenance is necessary before using the historical plots as local-increment evidence. The replay establishes a specific failure of freshness ranking and supplies a reproducible update-based benchmark.

Measured scoring cost. On an Apple M4 Pro CPU with Python 3.13.15 and NumPy 2.3.4, the full-coordinate two-client kernel takes approximately 123 ms for equality, 153 ms for lookup using the currently estimated matrix, 215 ms for estimation, and 371 ms for refreshing and scoring together. A matrix cached from round 4 takes 154 ms to apply and gives the same scores here; its calibration cost must be included when amortizing across rounds. These medians use three coordinate-sampling seeds and five interleaved timing repetitions after one warmup. Construction for round 10 separately takes about 18 ms to load states, 7 ms to subtract/flatten, and 23 ms to extract signs in one observed pass. Common random-index generation is excluded from scoring times. These are host-memory measurements, not training-time, gas, or end-to-end throughput results. A cache can also specialize an identity matrix to equality, further narrowing recurring differences.

A.14.2 PAYMENT THRESHOLDS AND MISSING COST MEASUREMENTS

Let H denote fresh training and A a feasible cheaper action, with expected scores r_H, r_A and costs C_H, C_A over the same participation horizon. For payments $b + \beta r$ with a common report-independent baseline and $\beta > 0$,

$$U_H - U_A = \beta(r_H - r_A) - (C_H - C_A). \tag{17}$$

A positive gap $g = r_H - r_A$ supports fresh training against this action only when $\beta > (C_H - C_A)/g$, for positive saved cost. If $g \leq 0$ and $C_H > C_A$, no positive multiplier repairs that comparison. The

latter case applies to Table 7 if reusing the stored increment is cheaper. It is a conditional utility conclusion; we have no matched client training-cost logs.

For comparison, applying this arithmetic to the *rounded historical* ten-round honest and stale-R1 means yields Table 8. These are descriptive sensitivity thresholds, conditional on their report definition and matched actions, not verified update-based effort results. Each mean is rounded to three decimals, so the score-gap rounding uncertainty is at most 0.001. No confidence intervals or hardware costs are inferred.

Table 8: Conditional payment multiplier per unit of saved cost, using historical means. Fresh training beats the alternative only above the displayed multiplier, if its score gap and action comparison apply. This table does not resolve the historical report-definition discrepancy.

Domain	**Honest − stale R1**	$\beta_{\min}/(C_H - C_A)$	**Rounding range**
General NLP	0.073	13.70	13.51–13.89
Code	0.071	14.08	13.89–14.29
Finance	0.111	9.01	8.93–9.09
Medicine	0.066	15.15	14.93–15.38

Requirements for a measured effort experiment. The recorded checkpoints and round metadata identify reports and participants, but not client training durations or economic costs. File timestamps are not training measurements. A matched utility experiment must record device-synchronized training and report-generation time, hardware/energy or price rates, and communication costs for the same clients and rounds. Stale actions must include the initial training needed to obtain the stored report. Lagged or sparse reports do not automatically save training. Finally, an adaptive participation experiment must measure client choices and downstream learning under a fixed reward budget; frozen-peer replay alone cannot establish those outcomes.

A.15 Blockchain integration and prototype

Figure 8 presents a possible deployment of KFCA-D, with an auditable payment rule executed by a smart contract (Witt et al., 2022; 2024). It is an integration design, not a measured decentralized FL system. The eight steps below follow the figure and distinguish report integrity from evidence of useful training.

1. **Registration and task specification.** Before training, fix the participant-admission policy, shared checkpoint, input indices, report encoding, aggregation rule, peer budget, deadlines, and funded payment schedule. Clients authenticate control of their registered addresses; an address alone does not establish a distinct eligible participant. A permissioned deployment can use an approved client list.
2. **Local training.** Clients train from the common checkpoint on private data and produce updates for the agreed off-chain aggregation procedure. Blockchain records and rewards contributions; it does not train the model or replace aggregation. Update submission and the next checkpoint must be linked to the same task and round as the scored reports.
3. **Report construction.** In KFCA-D, clients predict labels on the same public, unlabeled inputs. The figure's "public testset" supplies scoring inputs, not ground-truth evaluation labels. In KFCA-QP, this step instead extracts signs of aligned update coordinates, so no public inputs are needed. Constructing a report does not itself certify that it came from the submitted model.
4. **Commitment and report availability.** Fix each report before revealing peers or bonus/penalty roles. A possible commitment is

$$c_{i,t} = H(\mathrm{encode}(\mathrm{task}, t, i, H(R_{i,t}), s_{i,t}))\,,$$

where $R_{i,t}$ is the canonically encoded report vector, $s_{i,t}$ a fresh secret salt, and i the registered address. Binding the task, round, and address prevents reusing another context's commitment; it does not prevent committing a copied report. Keep reports inaccessible to

other clients until all commitments close, by withholding them or storing encrypted data. IPFS can provide content-addressed storage (Benet, 2014), but public plaintext remains readable: a salted commitment alone does not implement the figure's hidden-report icons. Specify storage availability and reveal deadlines.

5. **Random pairing and task selection.** After commitments close, obtain verifiable randomness that participants cannot predict or choose when committing. Sample distinct peers excluding the scored client and nonempty disjoint sets M_b, M_1, M_2 as in Definition 2.1. The figure's RNG oracle therefore selects both client pairs and task roles. Specify oracle verification, finality, and timeout handling; block metadata alone is not a substitute for this randomness assumption.
6. **Reveal and verification.** Clients reveal their report vectors and salts; an off-chain content identifier can locate the data. The verifier checks the commitment, round, address, vector length, and label encoding. An EVM contract cannot retrieve an IPFS object itself: report data must be supplied in transaction inputs, or selected entries accompanied by proofs against a separately specified vector commitment. The latter is an alternative design, not supplied by a whole-report hash. Reject invalid or missing reveals under rules fixed before participation; replacement peers must not be chosen opportunistically after observing reports.
7. **Scoring.** Compute Eq. (4) from verified entries with the committed report indices and fixed integer scaling. Record the sampling seed, indices, and score so the calculation can be reproduced. Consistency with a commitment proves neither fresh training nor improvement of the global model. Aggregation still uses the submitted model updates under the rule in Step 1, rather than the prediction labels; KFCA scoring alone neither reweights FedAvg nor filters updates.
8. **Transfer.** Convert the signed score into a payment using the funded rule fixed at registration, with protection against repeated claims. For example, since $\widehat{E}_i \in [-1, 1]$, a base payment $b \geq \beta > 0$ makes $b + \beta\widehat{E}_i$ nonnegative without clipping the score. The budget must cover the permitted payouts; negative token transfers are not meaningful. Specify the handling of non-reveals and any deposits in advance. This transfer rule preserves the score ordering, but fresh-training incentives still depend on the reward gain relative to training cost.

Prototype scope and implementation audit. The Solidity artifact (`legacy-experiments/blockchain/contracts/KFCA.sol`) accepts report arrays and returns signed scores using equality comparisons and integer scaling. Its entry point is a `view` function: it implements neither report commitments and reveals nor funded transfers. Its pseudo-random choices hash block metadata and the caller, rather than consuming a verified randomness-oracle output.

Source inspection identifies a penalty-indexing mismatch: the code draws positions within `partition2` but uses those positions directly as report indices, without mapping them through `partition2`. Penalty comparisons can therefore reuse bonus coordinates. The prototype also uses one penalty pool with distinct sampled indices, whereas Algorithm 1 specifies two disjoint penalty sets. The indexing and sampling rules require correction or reconciliation, and the resulting scores must be checked against Eq. (4) before claiming an implementation of the stated protocol. The artifact demonstrates feasible scoring arithmetic, not a validated end-to-end implementation. Sampling, data verification, and storage also cost gas; no gas, consensus-latency, or complete-system throughput measurements are reported.

A.16 Prospective incentives for autonomous agents

The same incentive problem extends beyond training: autonomous agents spend inference tokens and use tools to pursue goals, while a coordinating system must decide which work to reward. In a large swarm, repeatedly evaluating every contribution with another model could make reward computation a substantial part of the operating budget. KFCA suggests a lightweight alternative wherever useful agent work can be expressed through comparable categorical reports.

From learning updates to goal-directed work. A possible application is distributed verification: agents independently investigate candidate solutions and report whether specified requirements are satisfied. Across a batch of comparable checks, KFCA could reward agreement on the same check

minus chance agreement across different checks. The reports would be committed before peer sampling and bonus/penalty assignment. This transfers the existing scoring principle to agent outputs; its applicability depends on the resulting signal structure. Different agent roles would require suitable task groupings and report definitions.

Token and tool costs. Let a_i denote an agent's work policy, $T_i(a_i)$ its inference-token usage, and $C_i^{\text{tools}}(a_i)$ its tool cost. Using a common accounting unit, a prospective utility is

$$U_i(a_i) = \mathbb{E}\Big[b + \beta \widehat{E}_i(a_i) - \kappa_i T_i(a_i) - C_i^{\text{tools}}(a_i)\Big], \tag{18}$$

where $\widehat{E}_i$ is the KFCA score, $\beta > 0$ converts score to reward, b is a report-independent baseline, and κ_i is the token price. An agent's controller could optimize expected net reward to choose tasks, allocate reasoning, or stop when further work is not worthwhile. Productive effort requires its expected reward gain to exceed its additional cost. With fixed peer budget p, scoring m reports from each of n agents retains the $\mathcal{O}(npm)$ comparison cost; generating independent reports and operating the agents have separate costs.

Toward autonomous swarms. This is a proposed extension, not an evaluated agent result. The categorical condition and task assumptions must be checked, particularly when agents share models, histories, or outputs. Informative agreement alone does not establish goal achievement: common errors, copying, and shared permutations can separate reward from success. A deployment would therefore need goal-relevant checks and independent outcome validation. The opportunity is to make frequent reward decisions inexpensive enough to support large swarms that allocate their own effort toward specified goals, with success assessed jointly by task outcomes and total operating cost.