Federated Learning Beyond Uniform Participation: Probing and Diversity-Aware Client Selection

Adriano De Cesare (*s*333044), *Mattia Cappellino* (*s*327277), *Alessia Bevilacqua* (*s*327984)

Abstract

Federated Learning (FL) has emerged as a paradigm for training machine learning models across distributed clients without sharing raw data. Despite its promise, FL faces challenges due to statistical heterogeneity (non-IID client data) and system heterogeneity (variable resources and participation). This paper presents a systematic study of these challenges through experiments on CIFAR-100 and Shakespeare datasets, implementing centralized and federated baselines with varying degrees of heterogeneity. We analyze the impact of client participation patterns and non-IID data distributions on FedAvg performance. Finally, inspired by prior work on client selection, we propose two diversity-aware selection strategies: one leveraging label distribution, entropy, and inter-client distance, and another lightweight approach that is probe-informed and fairness-aware, both demonstrating improved convergence in heterogeneous settings. The code can be found at our GitHub repository GitHub: Scrivane/AML_project_5

Or alternatively at the link https://github.com/Scrivane/AML_project_5

1. Introduction

1.1. Federated Learning and its Challenges

Federated learning is a machine learning approach where many clients collaboratively train a model under the coordination of a central server [5]. The main advantage is that the raw data remains on the client devices, preserving privacy and reducing the need to transfer sensitive information. Only model updates are sent to the server.

However, federated learning also introduces new challenges. As the number of participating devices grows, communication between the clients and the server can become a significant bottleneck. Another challenge is that client data, such as that collected from mobile phones, is often non-IID (not independent and identically distributed) and unbalanced, since it depends on how each user interacts with their device.

1.2. Baselines and personal contributions

In this paper we analyze the standard FL scenarios, comparing a centralized baseline with a federated one. We also study the impact of client participation (uniform vs. skewed) as well as the effects of heterogeneous data distributions on the federated baseline. Since we noticed that these two aspects are critical for effective federated learning, we then propose two client selection strategies that can mitigate these challenges.

2. Methodologies

Datasets. We use two datasets to evaluate our methods: CIFAR-100 and Shakespeare. CIFAR-100 [11] is an image classification dataset consisting of 60,000 32x32 color images in 100 classes, with 600 images. Shakespeare [10] is a text dataset derived from the works of Shakespeare, used for next-character prediction tasks. Each sample is comprised of a text of 80 characters (x), a next character (y) and a label which contains info about the play title and the character name that is telling the line .

Splits. In CIFAR-100 examples are already split between train(50000 samples) and test(10000 samples). To get the validation set we randomly sampled 5000 samples from the training set. We ended up with 45000 samples for training, 5000 for validation and 10000 for testing. In Shakespeare, we split using 90% of the samples for training, 5% for validation, and 5% for testing.

Centralized Baseline. We implemented a centralized baseline model for both datasets in order to provide a performance reference. For CIFAR-100, we adopted a convolutional neural network inspired by the **LeNet architecture**, consisting in two convolutional layers with ReLU activations and max-pooling operations, followed by three fully connected layers. The final layer outputs logits over the 100 classes. For the Shakespeare dataset, we employed a character-level recurrent neural network which consists of an embedding layer, a two-layer **Long-Short-Term Memory(LSTM)** with dropout, and a linear output layer with output dimension equal to the vocabulary size. Given a sequence of characters, it predicts the next character at the end of the sequence.

Training on CIFAR and on Shakespeare was performed using cross-entropy loss and stochastic gradient descent, updating parameters via backpropagation .

Federated Baseline - FedAvg. For Federated Learning we adopt the FedAvg algorithm [7] , in which a central server coordinates distributed clients. The server initializes a global model, transmits it to a subset of clients, and aggregates their locally updated models through weighted averaging. This procedure is repeated across multiple communication rounds. For our experiments, we impose a step budget of 8000 for CIFAR and 2000 for Shakespeare, constraining the product of local steps J and the number of global rounds.

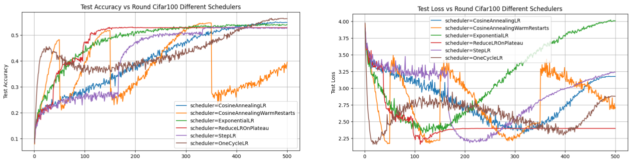
Label heterogeneity. To replicate the label heterogeneity of real-world federated learning scenarios, we fixed the number of classes per client, using values of $N_c \in \{1, 5, 10, 50\}$. For Cifar we used class of the images to do the split , for shakespeare we created a custom dataset where class is a character of a play , so we assigned together also same characters from different plays.

Participation skewness. To replicate real-world federated learning scenarios, we also analyzed the impact of skewed client participation. Using a Dirichlet distribution with parameter γ, we generated non-uniform client participation probabilities. A smaller γ value results in a more skewed distribution, where a few clients are selected more frequently than others.

3. Experiments

3.1. Centralized Baselines

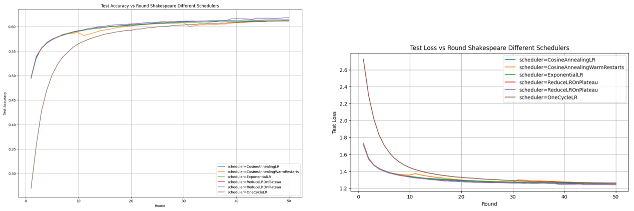
We tested different schedulers for the centralized baseline to find the one that gives the best performance: StepLR, MultiStepLR, ReduceLROnPlateau, ExponentialLR, OneCycleLR, CosineAnnealingLR and CosineAnnealingWarmRestarts. The best one turned out to be OneCycleLR for Cifar to get a better result in a few steps and also to get the best result in the end. We used 500 rounds for Cifar-100 and 50 rounds on Shakespeare because the model was already showing good accuracy and small improvements in both of them.



(a) Comparison between different schedulers accuracy on CIFAR dataset.

(b) Comparison between different schedulers loss on Shakespeare dataset.

For Shakespeare the best one after 50 rounds turned out to be ReduceLROnPlateau.



(a) Comparison between different schedulers on Shakespeare dataset.

(b) Comparison between different schedulers on Shakespeare dataset.

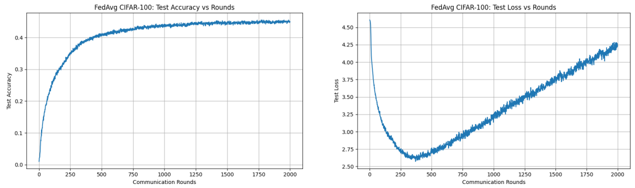
Figure 2. CIFAR-100 Federated Baseline: Test Accuracy and Test Loss over communication rounds.

There isn't a big difference between different schedulers max accuracy anyway , max difference is around 1% for shakespeare and 5% for cifar.

3.2. Federated Learning Training

To establish a baseline for the federated setting, we implemented Federated Averaging (FedAvg) [1] on CIFAR-100 with K = 100 IID clients. In each round, 10% of clients were randomly sampled, each performing 4 local SGD steps before weighted averaging at the server. Training ran for 2000 rounds with a fixed learning rate.

Results. As shown in Fig. 3, FedAvg converges steadily under this configuration. The model reached a **final test accuracy** of 44.92%, with a peak of 45.48% near 2000 rounds. The test loss decreased monotonically during the initial phase before plateauing, suggesting that further improvements could be achieved through optimization strategies such as learning-rate schedules or adaptive client sampling.



(a) Test accuracy vs. communication rounds

(b) Test loss vs. communication rounds

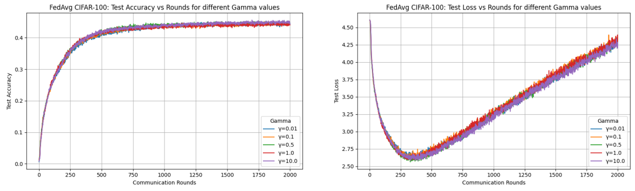
Figure 3. CIFAR-100 Federated Baseline: Test Accuracy and Test Loss over communication rounds.

3.3. Impact of Client Participation

To evaluate the effect of client participation on federated learning, we kept the same setup as in the baseline experiments and varied the client sampling scheme.

Dirichlet-skewed sampling Instead of uniform participation, clients were assigned selection probabilities drawn from a **Dirichlet (γ) distribution**, where smaller γ values induce greater skew. We evaluated $\boldsymbol{\gamma \in 0.01, 0.1, 0.5, 1, 10}$.

Results. As we can see in Fig.. 4, final performance was consistent across all configurations, with test accuracy in the range of 44–45%. The **highest result** was obtained with $\boldsymbol{\gamma = 10}$ (45.56%), while even the most skewed case ($\gamma = 0.01$) reached a comparable level (45.02%). These findings suggest that FedAvg is relatively robust to different client participation distributions in this setting.



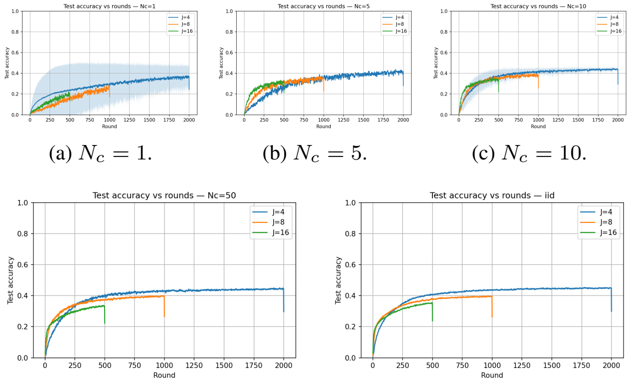
(a) Test accuracy vs. communication rounds for different values of γ (0.01, 0.1, 0.5, 1, 10).

(b) Test loss vs. communication rounds for different values of γ (0.01, 0.1, 0.5, 1, 10).

Figure 4. CIFAR-100 FedAvg with Skewed Sampling: Impact of varying γ on test accuracy and test loss over communication rounds.

3.4. Heterogeneous distribution

Experimental goal and protocol. We studied how label heterogeneity across clients affects FedAvg performance. [2, 3, 7].Experiments fix the number of clients to $K = 100$ and client fraction $C = 0.1$. We generated several non-IID shardings of the CIFAR-100 training set by constraining the number of distinct labels available to each client: $N_c \in \{1, 5, 10, 50\}$. [2, 3] An IID baseline (random split into K shards) is included for comparison. For each sharding we run FedAvg with local epochs $J \in \{4, 8, 16\}$. To keep compute comparable, rounds are scaled according to a fixed local-step budget: if J_0, R_0 are the canonical epoch/round pair, then for a new J we set $R_{\text{scaled}} = \lfloor (J_0 \cdot R_0)/J \rfloor$. [12, 14, 15]



(a) $N_c = 1$.

(b) $N_c = 5$.

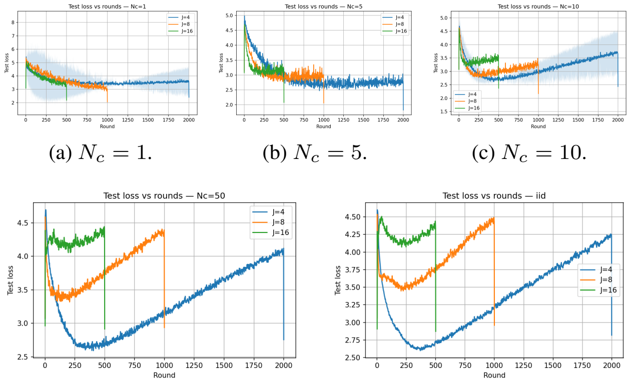
(c) $N_c = 10$.

(d) $N_c = 50$.

(e) IID baseline.

Figure 5. Test accuracy vs rounds separated by label-support N_c. Each plot contains curves for different local epoch budgets J ($J = 4, 8, 16$). Smaller N_c (more skew) results in slower convergence and lower final accuracy. [2, 4, 6]

Per-N_c learning curves (accuracy).



(a) $N_c = 1$.

(b) $N_c = 5$.

(c) $N_c = 10$.

(d) $N_c = 50$.

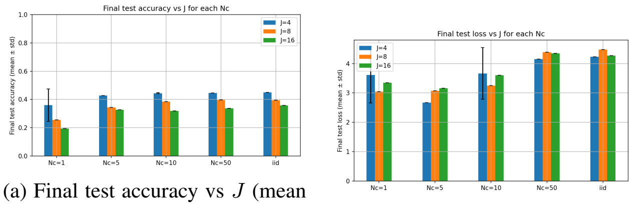
(e) IID baseline.

Figure 6. Test loss vs rounds separated by label-support N_c. Loss curves mirror accuracy behavior: greater label skew yields higher loss and noisier trajectories.

Per-N_c learning curves (loss).

Apparent loss-accuracy mismatch We observed an empirical mismatch in several CIFAR runs [2, 5]: around early training the test cross-entropy loss decreases (from ≈ 4.5 to ≈ 2.5) but then gradually increases again (reaching ≈ 4.0)

while top-1 test accuracy still slowly rises. This is not a plotting bug — it reflects model calibration and heterogeneity effects. Cross-entropy is sensitive to the predicted probabilities: a small number of highly confident wrong predictions can dominate the average loss, even if many other examples become correctly classified (so accuracy increases). In the federated setting this can arise from selection bias and client drift: the aggregator repeatedly sees updates from fast/majority clients, which drives the model to be overconfident on those classes while performance on rare classes degrades (or becomes overconfidently wrong), increasing average loss. [4, 6, 8]



(a) Final test accuracy vs J (mean ± std).

(b) Final test loss vs J (mean ± std).

Figure 7. Summary plots showing how final accuracy / loss vary with local work J for each N_c.

Final-summary plots (aggregate across seeds).

Short interpretation. The per-N_c accuracy and loss plots (Figs. 5, 6) clearly show degradation in both metrics as the per-client label support N_c decreases. [2, 3] The final-summary plots (Fig. 7) quantify the effect: IID and $N_c = 50$ typically achieve the highest final accuracy and lowest loss; $N_c = 10$ shows moderate degradation; $N_c = 5$ and $N_c = 1$ suffer the largest performance drop and greatest variance. Mechanistically, small N_c concentrates client data on a few classes, producing biased, high-variance local updates; aggregated updates are therefore noisier and the global model converges more slowly and to a worse point. Increasing J exacerbates drift for strongly non-iid shardings (not that harmful for IID, but worsen a lot for small N_c), explaining the differing curves across J.

Results and interpretation. There is a notable difference: both accuracy and loss show clear deterioration as N_c decreases: IID and $N_c = 50$ perform similarly, $N_c = 10$ degrades moderately, while $N_c = 5$ and $N_c = 1$ show the worst final accuracy and highest loss.

This happens because small N_c produces many clients whose local data covers very few classes: local objectives become biased, local gradients are high-variance and concentrated on limited features, and aggregation averages noisy updates. This produces slower decrease in loss and lower final accuracy. Increasing local epochs J amplifies client drift (larger steps away from the global model between synchronizations), which typically helps IID settings (reduced communication overhead) but harms strongly non-iid settings; thus the interaction between N_c and J explains the differences seen in both accuracy and loss. [4, 6]

Heterogeneous distribution on Shakespeare

Experimental setup. We repeated the heterogeneous-distribution protocol on the Shakespeare dataset. As for CIFAR-100 we fixed $K = 100$ clients and a participation fraction $C = 0.1$, and compared IID splits with artificial non-IID shardings parameterised by the number of distinct labels per client $N_c \in \{1, 5, 10, 50\}$. The model is run with local steps $J \in \{4, 8, 16\}$ while scaling the number of rounds to keep a fixed local-step budget (500, 250, 125 global rounds respectively for $J = 4, 8, 16$). The evaluation metric is top-1 accuracy for next-token prediction. [5, 10]

Observed invariance and immediate diagnosis. Contrary to CIFAR-100, the Shakespeare experiments show almost no sensitivity to N_c: across the tested shardings the reported top-1 accuracy converges to roughly $37\% - 38\%$ and the cross-entropy loss hovers near 2.3–2.4. This near-invariance is not a bug but an expected outcome when the global test distribution is heavily dominated by a small set of frequent tokens. [1, 3] In our case a trivial majority-class predictor (always predict the most frequent next letter) already attains a comparable top-1 score; therefore changes to per-client label support have little effect on the global top-1 metric. In short, the model can achieve the dominant-token baseline regardless of N_c, masking any improvements on rare tokens.

Why this differs from CIFAR-100. The CIFAR experiments measure a multi-class image classification problem where class labels are (approximately) balanced in the global test set and improving minority-class performance visibly changes top-1 accuracy and loss. Shakespeare is different for two related reasons: (i) the target distribution for next-letter prediction is highly skewed (few tokens dominate), and (ii) role-based shards retain many of these frequent tokens, so even non-IID partitions expose the aggregator repeatedly to the same high-frequency signals. Consequently (a) global top-1 accuracy is an insensitive summary under this modality, and (b) cross-entropy stabilises around the entropy of the dominant distribution.

Concluding note. In summary, Shakespeare does *not* contradict the CIFAR-100 findings: non-IID sharding and larger local work still increase heterogeneity and client drift. However, because the modality and test distribution differ, the standard global summaries (top-1 / mean loss) are less informative .

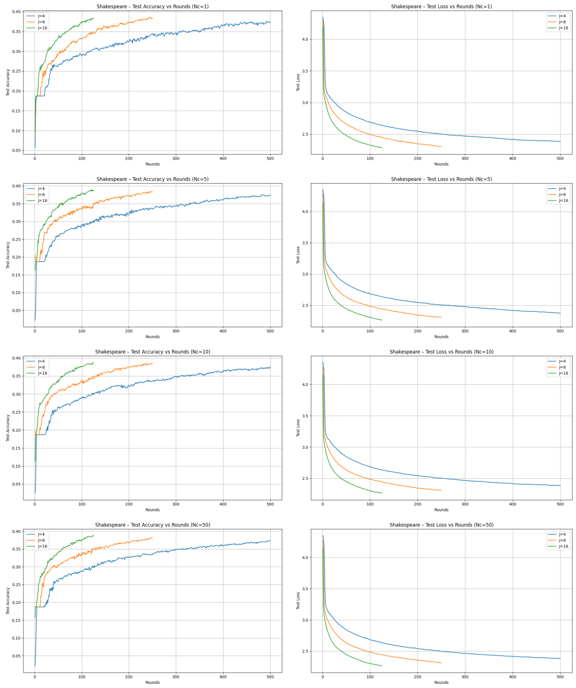


Figure 8. Test accuracy and loss over rounds for Shakespeare under different N_c values.

4. Personal contribution: Client selection under variable client heterogeneity

Motivation. Federated Learning (FL) typically assumes uniform client participation, but in real deployments clients differ widely in activity and in local data heterogeneity (label support, sample counts). Participation heterogeneity interacts with statistical heterogeneity and can slow convergence, increase variance across clients, and produce unfair outcomes (some clients never contribute meaningfully). In this part of our personal contribution we study client selection: how non-uniform participation affects FedAvg, and whether a lightweight, probe-informed + fairness-aware selection policy can speed early convergence while avoiding long-term starvation of "weak" clients. [8,9,13] In particular, we explore two selection strategies:

- **Probe + fairness hybrid** selection
- **Diversity Score** selection

Sharder: randomized per-client class support. To emulate realistic heterogeneity we introduce a *sampled sharder* that constructs per-client shards by sampling the number of classes per client from a broad distribution, selecting classes with a preference for labels that still have remaining examples, and allocating target examples across the chosen classes. This produces a population that mixes narrow clients (few classes) and broad clients (many classes), increasing variance in per-client updates compared to fixed-N_c schemes commonly used in FL benchmarks.

4.1. Selection policy: probe + fairness hybrid

We implement a lightweight hybrid policy combining:

1. **Probe utility:** each round we sample a small candidate pool C_t and run a tiny probe (one or two minibatches) on each candidate to produce a *probe model* $\mathbf{w}_k^{\text{probe}}$. The probe norm is

$$\text{probe\_norm}_k^t = \left\| \mathbf{w}_k^{\text{probe}} - \mathbf{w}^t \right\|_2 ,$$

which estimates how large an update client k would produce in the near term.

2. **Fairness score:** we compute an inverse-frequency and recency score per candidate to avoid starving clients. Inverse frequency uses $1/(1 + \text{count}_k)$ where count_k is times selected so far; recency is $t - \text{last\_selected}_k$. These components are normalized and blended. Both inverse-frequency and recency components are min–max normalized (and so is their blended fairness score) before blending with the normalized probe score; this ensures α mixes comparably-scaled quantities

3. **Blended score and soft exploration:** the final score is a convex blend

$$s_k = (1 - \alpha)\, p_k + \alpha\, f_k,$$

where p_k is the normalized probe score, f_k is the fairness score, and α (code: `beta_fairness`) trades speed for equity. A small fraction ϵ of the m client slots is reserved for probabilistic soft exploration using a temperatureed softmax over s_k. We also force-select any client not chosen for a long gap (anti-starvation).

Probe cost is limited by using few minibatches and by reusing the probed model state if that client is selected (so we do not retrain from scratch for the probe work).

Algorithms and hyperparameters. Local training can use FedProx regularization:

$$\mathcal{L}_k^{\text{prox}}(\mathbf{w}) = \mathcal{L}_k(\mathbf{w}) + \frac{\mu}{2} \|\mathbf{w} - \mathbf{w}^t\|_2^2,$$

where μ penalizes drift from the global model (typical values tested: $\mu \in \{0.01, 0.05, 0.1\}$). On the server side we also use FedAvgM (server momentum) with velocity $\mathbf{v}^t$:

$$\mathbf{v}^{t+1} = \beta\, \mathbf{v}^t + \Delta_{\text{agg}}^t, \qquad \mathbf{w}^{t+1} = \mathbf{w}^t + \eta_s\, \mathbf{v}^{t+1},$$

We observed that overly large η_s together with large per-client deltas can amplify instability. In our experiments we apply server momentum and then scale velocity by `server_lr` before updating the global model; conservative choices ($\eta_s \leq 1.0$, e.g. 1.0 or 0.5 depending on clipping) with delta clipping (`clip_norm` ≈ 10.0) and moderate momentum ($\beta \in [0.8, 0.95]$) improved robustness. **Note:** some earlier runs with `server_lr`=0.5 showed instability in our setup and were discarded; we treat `server_lr`=1.0 with clipping as a more stable baseline.

Analysis: why some clients are strong and others weak. A handful of interacting causes explain the observed client disparity:

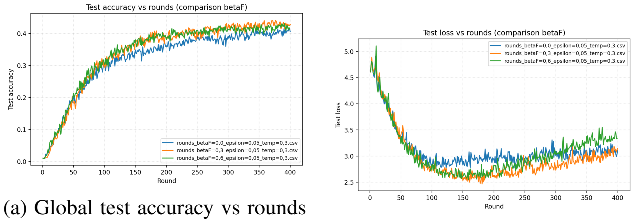
- **Label support and sample count:** clients with many classes expose the model to more diverse training signal and tend to obtain higher local accuracy and produce larger, more informative updates.
- **Statistical idiosyncrasy:** clients with narrow or unique label mixes can either help (by providing missing labels) or harm (by producing noisy, high-variance updates) global performance.
- **Selection bias and deterministic top-k:** a deterministic policy that always picks top probe-norm clients concentrates training on a small subset (fast learners), accelerating short-term progress but starving others. Over time this increases population variance and can reduce final generalization.
- **Optimization interactions:** large per-client deltas combined with high server momentum or large server scaling (η_s) can create instability (overshoot) unless deltas are clipped or the server learning is conservative.

Deterministic top-k behavior (pros and cons). Selecting the highest probe norms every round is attractive because it picks clients that promise large immediate improvement (fast convergence in early rounds). However:

- **Positive:** faster early reduction in global loss and fewer rounds-to-threshold for moderate skew.
- **Negative:** deterministic exploitation leads to a bimodal selection frequency distribution (some clients selected very often; many rarely selected) which increases fairness issues and can prevent the model from seeing crucial rare labels. It also increases the chance of unstable aggregation when idiosyncratic clients dominate.

Additional results: hyperparameter sweeps and selection statistics The plots in this section summarize sensitivity to three selection hyperparameters: the fairness blend weight β (code: `beta_fairness`), the exploration fraction ϵ, and the softmax temperature T (code: `temp`). For each hyperparameter we show two side-by-side figures: (left) global test accuracy across rounds, and (right) test loss across rounds.

Sweep: fairness weight β.

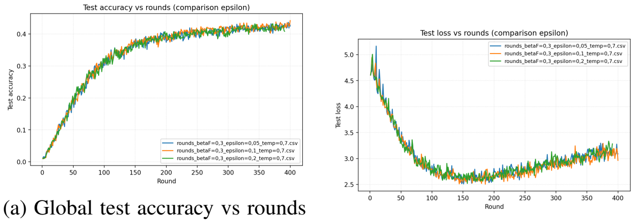


(a) Global test accuracy vs rounds (varying β).

(b) Test loss vs rounds (varying β).

Figure 9. Test accuracy and loss over various fairness weight β

Sweep: exploration fraction ϵ

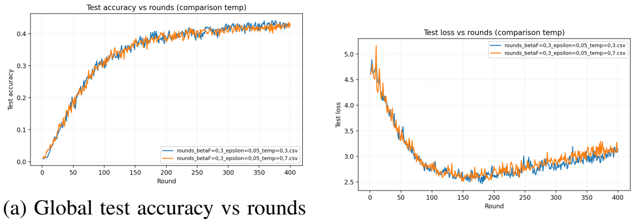


(a) Global test accuracy vs rounds (varying ϵ).

(b) Test loss vs rounds (varying ϵ).

Figure 10. Test accuracy and loss over various exploration fraction ϵ.

Sweep: softmax temperature T

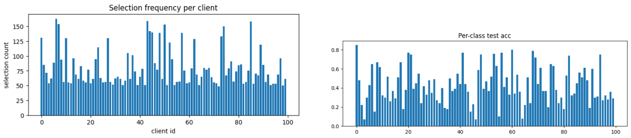


(a) Global test accuracy vs rounds (varying T).

(b) Test loss vs rounds (varying T).

Figure 11. Test accuracy and loss over various sampling temperature T.

Selection frequency and per-client accuracy histograms



(a) Client selection frequency (times selected).

(b) Per-client test accuracy at final round.

Figure 12. (Left) Selection frequency indicates skew and starvation. (Right) Per-client accuracy highlights spread between strong and weak clients.

Takeaway. Probe-based selection accelerates learning by prioritizing informative clients, but deterministic top-k selection starves many clients and risks instability; blending probe utility with simple fairness and soft exploration yields strong practical gains—faster convergence without starving weak clients.

4.2. Selection policy: Diversity Score sampling

To quantify client diversity, we defined a diversity score that integrates both intra-client and inter-client information.

Intra-client score. At the intra-client level, we combined two factors:

- **Structural diversity:** the number of distinct classes present in the local dataset of client i (N_c);
- **Distributional diversity:** the entropy of the local class distribution (H), computed through Shannon entropy.

For each client, we represent the local data as a frequency vector of length C, where the k-th entry corresponds to the number of samples belonging to class k. Normalizing this vector yields class probabilities p_k, which are then used to compute entropy as

$$H = -\sum_{k=1}^{C} p_k \log(p_k), \quad \text{where } p_k = \frac{n_k}{\sum_{j=1}^{C} n_j},$$

- n_k = the number of samples belonging to class k in that client's local dataset
- $\sum_{j=1}^{C} n_j$ = the total number of samples for that client.

Entropy is maximized when all classes present in a client are equally represented, and decreases as the distribution becomes more imbalanced.

These two components were combined into a **normalized diversity measure** for each client i:

$$D_i = \lambda \cdot \frac{N_c}{C} + (1-\lambda) \cdot \frac{H}{\log(C)},$$

where C is the total number of classes in the dataset and $\lambda \in [0,1]$ balances the contribution of structural and distributional diversity.

Inter-client score. To further incorporate inter-client dissimilarity, we extended the definition of the score by computing the mean Jensen–Shannon (JS) divergence of client i with respect to all other clients (JS_i). The final score is then given by:

$$\text{Score}_i = \alpha \cdot D_i + (1-\alpha) \cdot JS_i,$$

where $\alpha \in [0,1]$ controls the relative importance of intra-client versus inter-client diversity.

Experiments In our experiments on diversity score, we evaluate our LeNet-like architecture on CIFAR-100 by varying α, which directly controls the degree of statistical heterogeneity between clients. Since this diversity is already established by the sampled sharder and represents the main challenge in federated learning, focusing on α allows us to systematically study its impact on training dynamics while keeping λ fixed at 0.5 to balance intra-client diversity.

To better contextualize the effect of our client-selection method, we also compare results under our heterogeneous sharding scheme (random number of classes per client, with balanced participation) against fixed-N_c baselines from previous experiments, where each client has exactly N_c classes.

Experiments settings. CIFAR-100 dataset is divided in: 45k/5k/10k train/val/test split. FL settings: $K = 100$, client fraction $C = 0.1$ per round, local epochs $J = 4$, SGDM (LR=0.01, WD=1e-4, momentum=0.9).

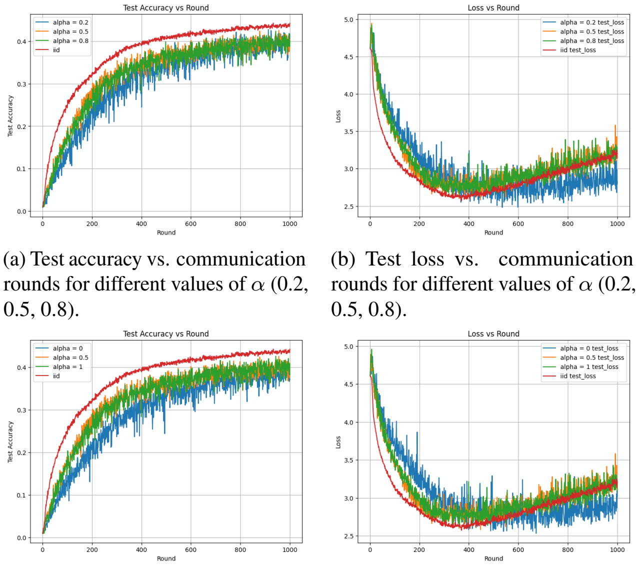
Results. Across our experiments, we consistently observed the following trends:

1. **Effect of varying α.** Performance differences across values of α were modest, yet several patterns emerged. (See Fig. 13a and 13c)

   (a) **For $\alpha = 0$**, which relies purely on **inter-client dissimilarity** (via Jensen–Shannon divergence), the model peaked at 40.7% but converged to a lower final accuracy of 37.1%.

   (b) **Intermediate values ($\alpha = 0.2, 0.5$)** that balance inter- and intra-client dissimilarity yielded more stable convergence, with final accuracies of 39.8% and 40.3% and peaks at 42.7% and 42.6%.

   (c) **Higher values ($\alpha = 0.8, 1$)**, emphasizing **intra-client dissimilarity**, yielded no further gains: final accuracies remained in the 40.0 – 40.9% range, with slightly higher validation losses suggesting less stable training, a trend mirrored by the final test losses. (Fig. 13b and 13d)



(a) Test accuracy vs. communication rounds for different values of α (0.2, 0.5, 0.8).

(b) Test loss vs. communication rounds for different values of α (0.2, 0.5, 0.8).

(c) Test accuracy vs. communication rounds for different values of α (0, 0.5, 1).

(d) Test loss vs. communication rounds for different values of α (0, 0.5, 1).

Figure 13. Impact of varying α on test accuracy and test loss over communication rounds.

Overall, moderate values **($\boldsymbol{\alpha = 0.2 – 0.5}$)** provided the **best trade-off between stability and accuracy**, confirming the value of combining both diversity signals for client selection.

2. **Comparison with fixed N_c settings.** Experiments using diversity-aware client selection consistently outperformed configurations with low fixed label counts ($N_c = 1$ and $N_c = 5$), achieved slightly better results than moderate label allocations ($N_c = 10$), and remained slightly below the best-performing scenario ($N_c = 50$), which approaches the IID baseline. This indicates that selecting clients based on label diversity can effectively mitigate the impact of non-IID distributions while remaining practical for heterogeneous client settings. By adapting naturally to varying label distributions, diversity-based selection balances accuracy and realism: it achieves near-optimal performance without requiring large, uniform label allocations per client, making it suitable for real-world federated learning scenarios.

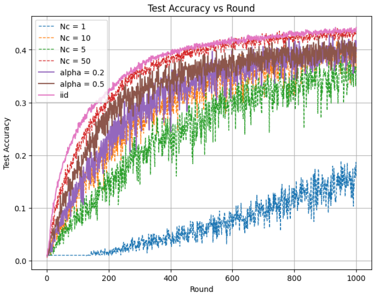


Figure 14. Test accuracy of our proposed Diversity Score client selection method compared with the fixed-Nc non-iid baseline.

5. Conclusion

In this work, we compared centralized and federated training baselines on CIFAR-100 and Shakespeare, confirming the sensitivity of FedAvg to both data and participation heterogeneity. To address these challenges, we explored client selection strategies. We proposed a Probe + fairness hybrid, which accelerated early convergence while avoiding client starvation, and a Diversity Score policy, which improved stability under heterogeneous splits and skewed participation. Overall, our results show that lightweight, fairness-aware selection can enhance both efficiency and equity, paving the way for more robust deployments of federated learning.

References

[1] Caldas et al. Leaf: A benchmark for federated settings, 2019. 4

[2] Hsu et al. Measuring the effects of non-identical data distribution for federated visual classification, 2019. 3, 4

[3] Hsu et al. Federated visual classification with real-world data distribution. In *ECCV*, 2020. 3, 4

[4] Karimireddy et al. SCAFFOLD: Stochastic controlled averaging for federated learning. In *ICML*, 2020. 3, 4

[5] Kairouz et al. Advances and open problems in federated learning. *Foundations and Trends in Machine Learning*, 2021. 1, 3, 4

[6] Li et al. Federated optimization in heterogeneous networks, 2018. 3, 4

[7] McMahan et al. Communication-efficient learning of deep networks from decentralized data. In *AISTATS*, 2017. 2, 3

[8] Németh et al. A snapshot of the frontiers of client selection in federated learning, 2022. 4, 5

[9] Pfeiffer et al. Federated learning for computationally-constrained heterogeneous devices: A survey. *ACM Computing Surveys*, 2023. 5

[10] S. Caldas et al. Leaf: A benchmark for federated settings, 2019. 1, 4

[11] Alex Krizhevsky. Learning multiple layers of features from tiny images. 2009. 1

[12] Sam McCandlish. An empirical model of large-batch training, 2018. 3

[13] Sashank J. et al Reddi. Adaptive federated optimization, 2021. 5

[14] Sebastian U. Stich. Don't use large mini-batches, use local sgd, 2020. 3

[15] Yang You, Igor Gitman, and Boris Ginsburg. Large batch training of convolutional networks, 2017. 3