RUMOR HAS IT: CORRECTING PARTICIPATION BIAS IN DECENTRALIZED LANGUAGE MODEL FINE-TUNING

Anonymous authors
Paper under double-blind review

ABSTRACT

Decentralized fine-tuning of language models can leave infrequent participants' data poorly represented. Motivated by prior user participation models that take into account model quality, we study a controlled setting in which a node becomes less likely to train and exchange its model as its loss rises relative to the network mean. This dependence can create a positive feedback loop: missed updates and model exchanges can keep the node's loss high, which can in turn further reduce its participation. To correct this imbalance using only local information, we reweight updates based on recent participation history, use past local loss as a reference to limit excessive amplification, and use disagreement with active neighbors' models as an additional signal for increasing the update weight. We call the resulting scheme RUMOR, Reference-guided Updates using Memory, Observed loss, and Reweighting. We evaluate a small transformer and seven pretrained models up to 12 billion parameters under both loss-adaptive participation and fixed participation probabilities. RUMOR substantially reduces the loss disparities associated with infrequent participation, with particularly clear benefits under strong feedback. Under weaker feedback and fixed participation probabilities, it remains competitive with aggressive participation reweighting while avoiding the severe loss increases observed for such baselines in the evaluated large-model runs.

1 INTRODUCTION

Unequal participation can systematically reduce the influence of some nodes in decentralized learning. An infrequent participant takes fewer local steps and exchanges fewer models with its neighbors, so its data can have less influence on training. This matters as language-model training and use extend across collaborating machines (Douillard et al., 2024; Jaghouar et al., 2024a; Borzunov et al., 2023). We study decentralized fine-tuning with gossip algorithms, which combine local training with model exchanges between neighboring nodes (Boyd et al., 2006; Lian et al., 2017; Ghiasvand et al., 2025).

A particularly challenging case arises when participation itself depends on model performance. Prior work has modeled settings in which poorer model performance reduces future user retention or willingness to participate (Hashimoto et al., 2018; Goetze et al., 2025), and more broadly studied feedback between model behavior and the data subsequently observed (Perdomo et al., 2020). Motivated by this direction of feedback, we study a controlled setting that we call *loss-adaptive participation*, in which a node participates less often as its recent average loss rises relative to the network mean. We do not assume that this direction is universal: in other systems, underperforming nodes may instead have greater incentive to participate. Our goal is to isolate the disengagement regime in which poor local performance reduces participation. The node then receives fewer local updates and exchanges fewer models, which can keep its loss high and further reduce participation. We also test static participation, where each node's probability stays fixed and this dependence is absent. In both settings, we minimize the equally weighted average of node losses.

An intuitive correction is to give infrequent participants a greater impact on training when they participate. An *update weight*, the scalar multiplying a node's local gradient step, provides this adjustment. Large weights can compensate for missed training but also amplify an update enough to increase loss severely. Therefore, participation frequency alone cannot determine how much amplification is useful. Recent local loss can reduce the maximum update weight, or *cap*, when loss rises above a reference from past losses.

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.

We call this method RUMOR, Reference-guided Updates using Memory, Observed loss, and Reweighting. Each node estimates how often it participates and gives infrequent updates larger weights, up to the maximum set by its local loss. Because participation counts do not describe how far its model differs from its neighbors, RUMOR also measures that disagreement. For nodes already receiving larger history-based weights, disagreement can increase the correction within the cap. These observations are local and require no network-wide loss measurement.

Our contributions connect the participation problem, the weighting rule and its evaluation.

1. We formulate participation correction for gossip fine-tuning when local performance affects future training and model exchange, and compare loss-adaptive with static participation.
2. We develop RUMOR and establish bounds on its update weight and necessary conditions for disagreement to increase it. The analysis shows how counts alone may not identify the participation rule and how averaging can make neighboring models identical despite different histories.
3. On a small transformer and low-rank adaptation (LoRA) (Hu et al., 2022) of seven pretrained models up to 12 billion parameters, we demonstrate strong correction under feedback and competitive correction under static participation. We test these findings with independent seeds and ablations, using paired absolute losses, local-adapter evaluation and communication controls to identify sources of improvement.

2 Related work

Inverse-probability weighting corrects unequal inclusion by giving less likely observations greater weight (Horvitz & Thompson, 1952). FedIPW applies this principle to server aggregation with separate enrollment and participation probabilities (Morishita, 2026). The *inverse-probability weighting (IPW)* baseline adapts this principle to local gossip updates, estimating probability from recent participation and limiting the resulting weight. Other federated-learning methods estimate correction weights from participation records (Wang & Ji, 2024; Sun et al., 2025). FedAU estimates correction weights from each client's participation intervals, which we adapt as interval-average weighting for our gossip comparison in Appendix L.2. FLOSS estimates participation probabilities from device and user attributes, including model-quality feedback, and applies inverse weighting at a server (Goetze et al., 2025). Participation counts alone do not reveal rising loss, so RUMOR uses local loss to limit the update weight.

In gossip training, correcting local updates addresses only part of the effect of participation, since inactive nodes also stop exchanging models. Reception-aware aggregation instead reweights received models to correct link failures (Hewa Kaluannakkage & Buyya, 2026), making communication correction complementary to update weighting. Our target is the equally weighted objective despite unequal participation, whereas fairness objectives can give poorly served clients greater weight (Mohri et al., 2019; Li et al., 2020). Appendix A discusses further work and these gossip adaptations.

3 Problem setting

We consider decentralized learning over N nodes on a fixed, connected communication graph. Node i has local objective $f_i(\theta)$, and the network minimizes the uniform average $f(\theta) = \frac{1}{N}\sum_{i=1}^{N} f_i(\theta)$. At round t, node i holds model θ_i^t and draws a participation indicator $\xi_i^t \sim \mathrm{Bernoulli}(\pi_i^t)$. Parameter differences and gradients use the Euclidean norm. For LoRA, θ_i^t contains the trainable factors.

The primary experiments use stochastic gradient descent (SGD) with gossip mixing (Boyd et al., 2006; Lian et al., 2017). Appendix E reports additional experiments with modified AdamW. An active node ($\xi_i^t = 1$) computes a clipped stochastic gradient g_i^t and takes the local step $\widetilde{\theta}_i^t = \theta_i^t - \eta_t w_i^t g_i^t$. Here η_t is the learning rate and $w_i^t \geq 0$ is the update weight. An inactive node takes no step, $\widetilde{\theta}_i^t = \theta_i^t$. Active nodes then mix their models over the active subgraph using Metropolis weights, a variant of the Metropolis–Hastings construction with positive self-weights (Xiao & Boyd, 2004; Xiao et al., 2005). With $\deg_i^t$ the active degree, the weight is $W_{ij}^t = 1/(1 + \max\{\deg_i^t, \deg_j^t\})$ for active neighbors and $W_{ij}^t = 0$ otherwise. The diagonal is $W_{ii}^t = 1 - \sum_{j \neq i} W_{ij}^t$, with identity rows and columns for inactive nodes. The gossip update is

$$\theta_i^{t+1} = \textstyle\sum_{j=1}^{N} W_{ij}^t \widetilde{\theta}_j^t. \tag{1}$$

By default inactive nodes neither update nor communicate. Because the graph is undirected, W^t is doubly stochastic, so mixing preserves the average of the post-update models $\widetilde{\theta}_i^t$. We evaluate the network-average model $\theta_{\text{net}}^t = \frac{1}{N}\sum_i \theta_i^t$ at the recorded evaluation rounds. We refer to data associated with less frequent participation as the *low-participation group* and to the remaining data as the *high-participation group*.

Under *static participation* each node has a fixed participation probability, $\pi_i^t = \pi_i$. In contrast, loss-adaptive participation responds to model quality through the controlled participation rule

$$\pi_i^t = \sigma\big(c_0 - \beta(\ell_i^t - \ell_{\text{net}}^t)\big)\,, \tag{2}$$

motivated by work on loss-dependent user retention (Hashimoto et al., 2018). Here $\sigma(x) = 1/(1 + e^{-x})$ is the logistic function. ℓ_i^t is an exponential moving average (EMA) of local training loss, and $\ell_{\text{net}}^t = \frac{1}{N}\sum_i \ell_i^t$ is its network mean. Each local loss EMA changes only when its node participates, so an inactive node's probability moves only through the network mean. In this rule, a higher loss relative to the network mean gives a lower participation probability. The simulator uses the network mean to draw participation indicators, but each node receives only its indicator, keeping the learning rule fully decentralized. At the mean loss, the baseline participation probability is $\sigma(c_0)$. Nonnegative *feedback strength* β sets participation's sensitivity to relative loss, allowing weaker and stronger feedback.

4 METHOD

RUMOR combines a history-based weight, bounded using local loss, with a drift-based weight from neighbor disagreement. We define these weights below and show how they combine in Figure 1.

History-based weighting with a loss-dependent cap. Node i estimates its participation probability from its own recent indicators as $\hat{\pi}_i^t = (c_i^t + 1)/(m_i^t + 2)$. The count c_i^t records participation over the preceding $m_i^t \le H$ rounds, excluding the current round, and H is the maximum window length. Windowed IPW uses this estimate to multiply the local update by $w_i^t = 1/\max(\hat{\pi}_i^t, \tau)$. The probability floor $\tau > 0$ caps the weight at $1/\tau$. A smaller cap limits amplification while reducing the correction available to infrequent participants, reflecting the bias–variance tradeoff of truncated importance sampling (Ionides, 2008). With static participation, $\hat{\pi}_i^t$ estimates π_i. Under feedback, the estimate responds slowly to changes in π_i^t because it includes earlier rounds.

Large inverse-probability weights can amplify loss increases, motivating a cap that responds to local training. RUMOR retains the participation estimate but lowers its maximum weight when recent loss rises above a local reference. The resulting *history-based weight* is $h_i^t = \min(1/\hat{\pi}_i^t, C_i^t)$, where C_i^t is the node's loss-dependent cap. Node i keeps two loss EMAs, updated only when it participates by $\ell \leftarrow (1-\alpha)\ell + \alpha L_{\text{batch}}$ for current batch loss L_{batch}. The recent-loss average ℓ_i^t uses rate α_f and also determines participation in Equation 2. The slower average $\bar{\ell}_i^t$ uses rate $\alpha_s < \alpha_f$ to reduce sensitivity to individual batches. Its lowest value, $b_i^t = \min_{r \le t} \bar{\ell}_i^r$, provides the loss reference for setting the cap. The *loss excess* and the *loss-dependent cap* are

$$D_i^t = \max\big(\ell_i^t - b_i^t,\, 0\big), \qquad C_i^t = C + (C_{\text{hi}} - C)\exp(-D_i^t/\delta). \tag{3}$$

The cap decreases continuously as ℓ_i rises above the reference. The constants $C_{\text{hi}} > C$ set the cap range, and $\delta > 0$ sets its response scale in natural-log units. The cap stays above C for every finite excess and increases again as recent loss approaches the reference. Persistent loss increases can keep the cap near C. Appendix B.1 gives initialization and numerical cap responses.

Drift-based weighting. Two nodes with the same participation count can still differ in how closely their models agree with their neighbors. RUMOR's drift-based weight uses this disagreement to supplement the history-based correction. At a participation event with active neighbors, node i reads their current mean model $\bar{\theta}_i^t$ before local updates. Its *local model disagreement* is the Euclidean distance $d_i^t = \|\theta_i^t - \bar{\theta}_i^t\|_2$. Let $k_{i,\text{obs}}^t$ count rounds since the previous such event. To scale disagreement using both its recent magnitude and neighborhood movement, two exponential moving averages at rate α_d summarize these events. The *neighbor-mean change rate* s_i averages $\|\bar{\theta}_i^t - \bar{\theta}_i^{\text{last}}\|_2/k_{i,\text{obs}}^t$, the displacement of the neighbor mean divided by the elapsed rounds. The *disagreement rate* v_i averages $d_i^t/k_{i,\text{obs}}^t$, the disagreement divided by the elapsed rounds. Node i updates s_i before selecting the

weight and v_i afterward. The scale uses current neighbor movement and earlier disagreement. The *drift-based weight* is $r_i^t = \min\left(1 + d_i^t / \max(v_i, \frac{1}{2}s_i, \epsilon_{\text{num}}), C\right)$, where the positive numerical floor ϵ_{num} prevents division by zero. Before the first estimate of v_i, the denominator is $\max(s_i/2, \epsilon_{\text{num}})$. The denominator floor $s_i/2$ limits amplification when past disagreement is absent or small. The cap C limits r_i^t. The node records the neighbor mean and observation time when it first observes active neighbors. At the next observation, it estimates s_i from that change and computes the drift-based weight.

RUMOR update rule. For an active node with an active neighbor, the update weight is

$$w_i^t = \begin{cases} \max(h_i^t, r_i^t) & \text{if } h_i^t > \gamma \text{ and } s_i \text{ is defined,} \\ h_i^t & \text{otherwise.} \end{cases} \tag{4}$$

The threshold condition $h_i^t > \gamma$ restricts disagreement-based increases to nodes whose history calls for larger updates. Taking the maximum preserves the history-based weight. Corollary 2 in Appendix K shows that disagreement can raise the weight only when $1/C < \hat{\pi}_i^t < 1/\gamma$, and never above C. Without an active neighbor, pretrained nodes use $w_i^t = h_i^t$. The small transformer can reuse earlier disagreement at these events, as specified in Appendix B.2. Using current-round neighbor models requires a second model exchange per active edge before the exchange for gossip mixing. Figure 1 shows how these corrections combine before the weighted local update. Afterward, the pre-update batch loss and current participation indicator update the statistics used in later rounds.

The participation rule compares local loss with the network mean, while the cap compares it with the minimum of the node's slower loss average. If a node's loss rises but the network mean rises faster, its participation probability can increase while its cap decreases. Conversely, a node whose loss falls more slowly than the network mean can participate less often despite its improvement.

We test three ablations of RUMOR. RUMOR, FIXED CAP replaces the loss-dependent history cap with C. RUMOR, SHORT WINDOW also fixes this cap and takes the smaller long- or short-window history weight, limiting amplification as recent participation increases. RUMOR, NO HISTORY retains r_i^t and removes history weighting, its loss-dependent cap and the threshold.

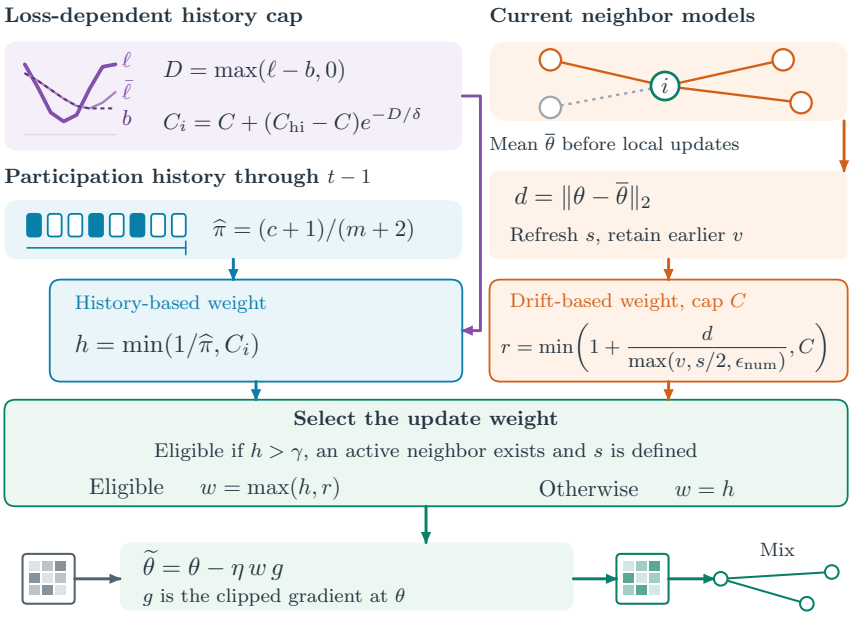


Figure 1: RUMOR at an active node, with round indices omitted. Participation history gives weight h, local loss sets its cap C_i, and active-neighbor models give drift-based weight r. These weights combine before the local update. The sketch shows recent average ℓ, slower average $\bar{\ell}$ and its minimum b. Orange marks active and gray inactive neighbors. Gradient clipping precedes weighting.

5 THE LOGIC BEHIND THE RUMOR

RUMOR relies on three fundamental mechanisms of history, loss, and disagreement information from the neighbors. Here, we provide basic theoretical evidence on why these signals play complementary roles and why relying on any one of them can be insufficient.

We first show that participation history alone may not distinguish static from loss-adaptive participation, motivating the use of local loss. We then quantify how a cap limits the correction available to infrequent participants. Finally, we show that gossip mixing can erase model disagreement even when nodes have different participation histories, motivating the retention of participation history alongside disagreement. Appendix K provides the proofs of the propositions below.

Proposition 1. *Let node i participate independently in each round with fixed probability $0 < p < 1$. Over a window of consecutive training rounds, suppose node j under loss-adaptive participation also has probability p of participating in each round, conditional on its own preceding participation indicators. Their participation-indicator sequences have the same joint law, as does any shared statistic of those sequences, including a count-based cap.*

The condition stated in Proposition 1 for node j holds, for example, if its loss remains at a fixed offset from the network mean that yields participation probability p in every round. In that case, its participation history has the same distribution as that of a node with static participation probability p. Therefore, participation history alone cannot reveal whether low participation is associated with worsening local performance; this motivates using local loss as an additional signal for controlling the update weight.

In RUMOR, the role of the local loss information is to control how much participation-based correction is allowed through the update-weight cap. The following simple argument reveals why the choice of this cap matters: Under full participation with unit weight, the scalar multiplier on the local update is one in every round. If instead $0 \leq w_i \leq C$ and $\xi_i \sim \mathrm{Bernoulli}(\pi_i)$, we have

$$\mathbb{E}[w_i \xi_i] \leq C\pi_i. \tag{5}$$

Thus, when $\pi_i < 1/C$, even always using the maximum allowed weight cannot, on average, match the update multiplier under full participation. For example, participation in 20% of rounds with cap 3.33 gives an average multiplier of at most 0.67, compared with one under full participation.

A small cap limits aggressive updates but can prevent sufficient correction for infrequent participants, whereas a larger cap permits stronger compensation. RUMOR uses local loss to adapt between these two regimes, allowing a larger cap when recent loss is close to its historical reference and reducing the cap as the loss excess grows. Corollary 1 in Appendix K extends Equation (5) to this loss-dependent cap under feedback, using $h_i^t \leq w_i^t \leq C_i^t \leq C_{\mathrm{hi}}$. When the loss excess is zero, $C_i^t = C_{\mathrm{hi}}$, so matching the full-participation multiplier requires at least $\pi_i \geq 1/C_{\mathrm{hi}}$ rather than $\pi_i \geq 1/C$. We compare RUMOR's loss-dependent cap with fixed and count-based cap rules in Appendix B.3. Appendix J further shows when the loss-dependent cap allows RUMOR's weights to exceed the base cap C.

We now turn our attention to the role of neighbor disagreement in RUMOR. In the following, we show that disagreement can provide an additional signal of missed participation, but gossip mixing can also reduce or even eliminate model differences between neighboring nodes.

Proposition 2. *If active node i has nonzero degree $\deg_i^t \geq \deg_j^t$ for every active neighbor j, Metropolis mixing contracts the distance from $\widetilde{\theta}_i^t$ to their fixed post-update mean by $1/(1 + \deg_i^t)$. A complete connected component of the active graph reaches consensus at the average of its post-update models in one step.*

Thus, model disagreement is not a persistent record of missed participation, because gossip mixing can erase it. As a result, disagreement alone cannot replace participation history. We examine this effect empirically in Appendix J.1.1, where neighboring models become nearly identical.

Together, these observations motivate RUMOR's use of participation history, local loss, and neighbor disagreement as complementary signals rather than substitutes. We next evaluate whether these design choices translate into effective participation correction across both loss-adaptive and static participation regimes.

6 EXPERIMENTS

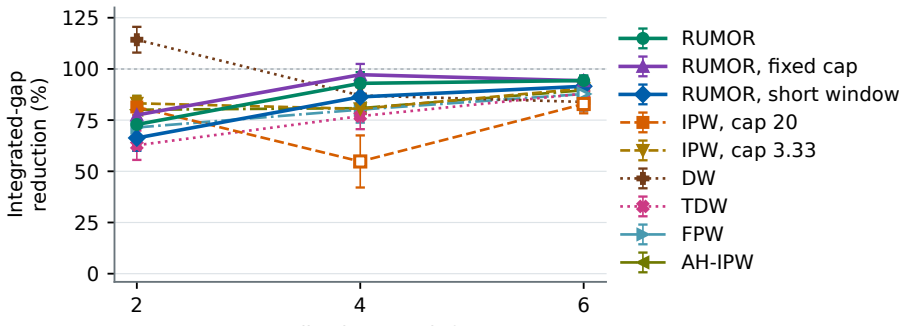
We test whether correction reduces group loss disparities and average node loss. We compare RUMOR with methods using participation history, model disagreement and their combination. Windowed IPW isolates history correction. Constant weights isolate update scale. Drift weighting (DW) and topological drift weighting (TDW) use model disagreement, while adaptive-history IPW (AH-IPW) resets discounted participation counts and filtered participation weighting (FPW) combines counts with disagreement. Appendix B.3 gives implementation details, while Appendix M derives the drift-based weights.

RUMOR uses $C = 3.33$ and $C_{\mathrm{hi}} = 20$, retained from IPW comparisons with floors 0.3 and 0.05 during method development. The larger cap permits correction at lower participation rates near the loss reference. We retain the testbeds' fast-loss and drift smoothing rates and threshold $\gamma = 1.5$. We fixed $\delta = 0.08$ nats and $\alpha_s = 0.02$ during development. At 0.08-nat excess, the cap falls to 9.5, while the slow loss average smooths its reference. Appendix B.1 gives all settings and their roles.

The small transformer uses four types of generated token sequences across sixteen nodes. Pretrained models use rank-8 LoRA across eight nodes, two per Wikipedia language. English and German form the high-participation group, and Turkish and Swahili the low-participation group. The *final group loss gap* is low-participation-group loss minus high-participation-group loss, which averages the three other groups on the small transformer. Its feedback comparisons use the *integrated group loss gap*, the area above 0.1 nats over evaluation rounds. Pretrained feedback runs and static Qwen 7B runs use the final gap. The remaining static comparisons measure *final low-participation-group loss* directly. Appendix D specifies the data assignments, evaluation timing and integration rule.

For loss metric G, the *normalized reduction* is $100(\bar{G}_{\mathrm{uncorrected}} - \bar{G}_{\mathrm{method}})/(\bar{G}_{\mathrm{uncorrected}} - \bar{G}_{\mathrm{full}})$. Zero matches the uncorrected run, 100 matches full participation, and values above 100 improve on full participation in that metric. We also report absolute losses, including when the denominator is non-positive, to distinguish gap reduction from improvement in both groups. Each $\bar{G}$ averages paired seeds, so the estimate is a ratio of mean differences. The leave-one-seed-out jackknife gives its standard error (SE), which quantifies uncertainty in this ratio. Between-method comparisons use paired differences on shared seeds. *Uniform node-average loss* averages the network-average model's loss on each node's data with equal node weights. Figure 5 shows paired loss differences for the main pretrained feedback comparisons. Appendix I gives all paired comparisons and absolute losses.

We summarize large final deficits using a 0.3-nat threshold fixed before the main RUMOR experiments. A pretrained run is *flagged* when its final low-participation-group loss exceeds matched full participation by more than 0.3 nats. On the small transformer, the criterion is a final group loss gap above 0.3 nats, versus below 0.05 under full participation. A flag records a remaining deficit and



Method	$\beta = 2$	k	$\beta = 4$	k	$\beta = 6$	k
RUMOR	72.9 (7.4)	0	93.0 (5.5)	0	94.3 (2.5)	0
RUMOR, fixed cap	77.5 (6.4)	0	97.2 (5.3)	0	94.2 (2.3)	0
IPW, cap 20	81.4 (3.8)	0	54.8 (12.7)	5	82.9 (4.5)	6

Figure 2: Small-transformer normalized integrated group loss gap reduction across feedback strengths (20 shared seeds with one leave-one-seed-out SE). The dotted line marks full participation and hollow markers mark flagged runs. Below, values are reduction (SE), and k is the number of flagged runs out of twenty. Fixed caps default to 3.33.

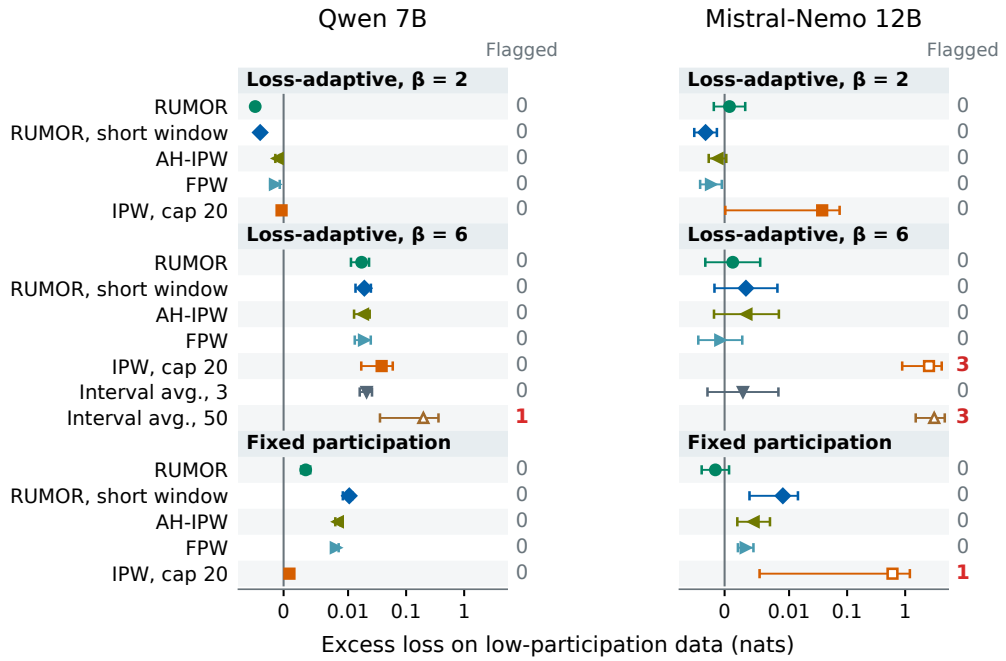


Figure 3: Final loss on low-participation data relative to full participation at 7B and 12B. Zero matches full participation. Lower loss is better. Points show means with one paired SE over ten seeds, except five at 12B and $\beta = 2$. Axes are linear between -0.01 and 0.01 nats and logarithmic beyond. Hollow markers and red counts flag excess losses above 0.3 nats. Interval-average labels give cutoffs 3 and 50.

does not require loss to increase during training. For pretrained runs, the cutoff corresponds to about 35% higher perplexity than matched full participation, since $e^{0.3} \approx 1.35$. All reported losses are finite, with deficits ranging from just above the cutoff to 13 nats, and all flagged runs remain in the reported results. Appendix D tests thresholds of 0.2 and 0.5 nats on the small transformer.

The clearest advantage of RUMOR is strong correction without the severe loss increases observed for some aggressively weighted baselines. Figure 2 shows this at stronger feedback on the small transformer, where RUMOR and its fixed-cap ablation RUMOR, FIXED CAP bring the integrated group loss gap close to full participation. RUMOR's normalized reductions are 93.0 (SE 5.5) and 94.3 (SE 2.5) at $\beta = 4$ and $\beta = 6$, with no flagged run among the forty runs using the original seeds. Both exceed windowed IPW at either cap, DW and the constant-weight control (Table 3). The independent-seed analysis in Appendix G confirms this pattern on 20 seeds excluded from method design and selection of the loss-excess scale. RUMOR's normalized integrated group loss gap reductions remain within 1.2 points of RUMOR, FIXED CAP's at both strengths. Each has one flagged run at $\beta = 6$. Its normalized final low-participation-group loss reduction under static participation exceeds RUMOR, FIXED CAP's by 12.5 points, versus 13.8 on the original seeds.

Figure 3 compares final loss on low-participation data across both feedback strengths and static participation at 7B and 12B. We compare losses directly because the 12B feedback normalization denominators are non-positive. At 12B and $\beta = 6$, RUMOR has no flagged run among ten, while windowed IPW with cap 20 and interval-average weighting with cutoff 50 each have three. The flagged IPW runs exceed matched full-participation losses by 0.40, 12.1 and 13.2 nats. Neither RUMOR, SHORT WINDOW nor interval-average weighting with cutoff 3 has a flagged run. At 12B with static participation, Figure 3 shows no RUMOR flag and one IPW flag. Both IPW group losses exceed uncorrected values (Table 4).

Figure 4 compares four pretrained models at $\beta = 2$. RUMOR, RUMOR, FIXED CAP and RUMOR, SHORT WINDOW achieve the three largest normalized final group-gap reductions on GPT-2 medium, Pythia 410M and Qwen 7B. On Qwen 0.5B, RUMOR and RUMOR, FIXED CAP achieve the two largest reductions. On all four models, RUMOR lowers final loss on low-participation data relative to windowed IPW with cap 20, AH-IPW and FPW. Against windowed IPW with cap 20, both groups

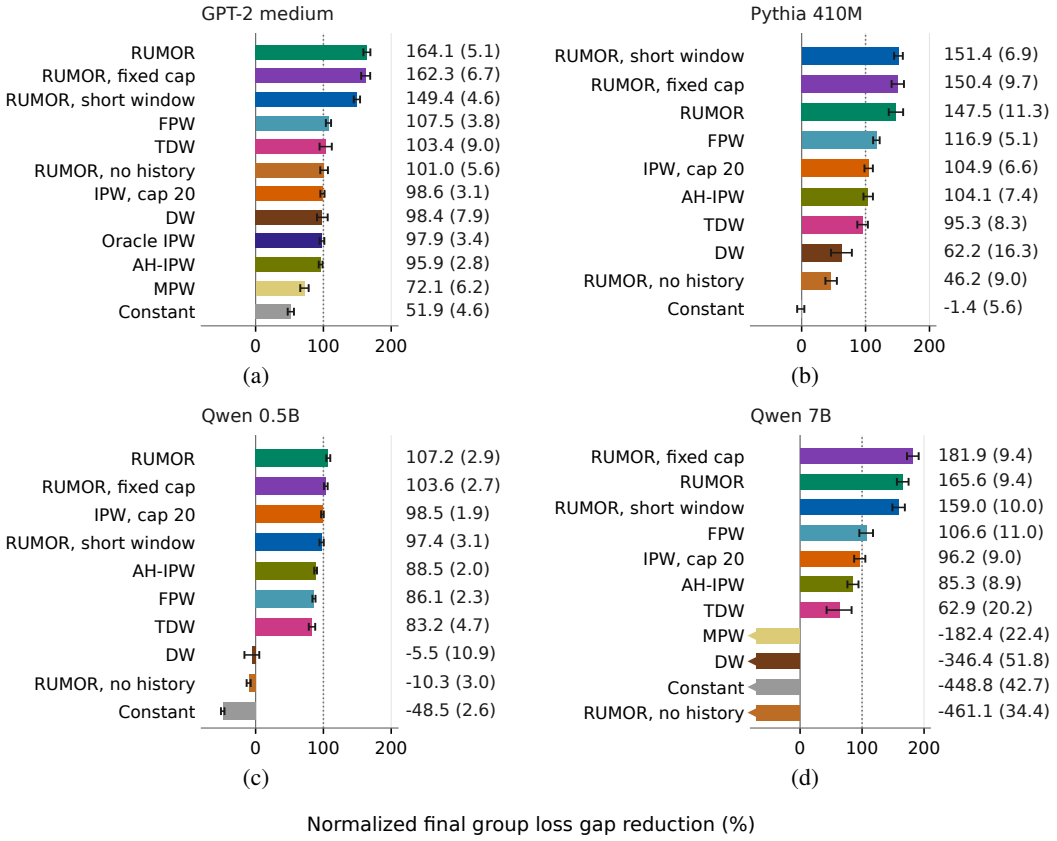


Figure 4: Loss-adaptive participation at $\beta = 2$, ten matched seeds per model. Bars show normalized final group loss gap reduction (leave-one-seed-out SE). Zero matches uncorrected training and the dotted line marks full participation. Larger values indicate greater gap reduction. Fixed caps are 3.33 unless labeled otherwise. No run is flagged. Arrows indicate off-scale bars. On Qwen 7B, 100 points represent a 0.0039-nat decrease in the group loss gap.

improve on GPT-2 medium and Qwen 7B, giving normalized gap reductions of 164.1 and 165.6 versus 98.6 and 96.2. For Pythia 410M and Qwen 0.5B, the paired losses in Figure 5 show that higher loss on high-participation data partly explains the gap reduction. Evaluating each node's adapter on five GPT-2 medium seeds keeps the ordering of RUMOR, RUMOR, FIXED CAP and windowed IPW with cap 20. Group losses stay within 0.004 nats of network-average evaluation in Appendix I.

At $\beta = 6$, Figure 5 shows that RUMOR's mean uniform node-average losses are within 0.003 nats of windowed IPW with cap 20 on the three smaller pretrained models. Although IPW reduces the gap more there (Figure 7, Appendix F.1), its uniform-loss advantages are 0.0029 nats (SE 0.0013) on GPT-2 medium and 0.0024 (SE 0.0010) on Qwen 0.5B. At 7B, RUMOR achieves the largest gap reduction and lowers uniform loss by 0.0196 nats (SE 0.0176) relative to IPW. Together with the 12B results, these comparisons show strong correction without IPW's severe loss increases. On the small transformer at $\beta = 2$, RUMOR's normalized integrated group loss gap reduction is 72.9, below RUMOR, FIXED CAP and windowed IPW with cap 20. At $\beta = 6$, flag counts depend on the loss threshold. RUMOR has eight runs above 0.2 nats versus one for RUMOR, FIXED CAP, and none above 0.5 nats versus two for windowed IPW with cap 20.

Figure 15 in Appendix L.1 shows that RUMOR retains strong correction under static participation, while its fixed-cap variants leave larger deficits. Its reductions trail windowed IPW with cap 20 by 2.4 points on GPT-2 small, 4.3 on Qwen 1.5B and 4.6 on Qwen 7B. On the small transformer, its normalized final low-participation-group loss reduction is 73.8, exceeding windowed IPW with cap 20 by 2.6 points (SE 1.4). The 14-point gain over RUMOR, FIXED CAP partly transfers loss between groups. Table 6 gives a 0.078-nat decrease on low-participation data and a 0.030-nat increase on the other groups relative to RUMOR, FIXED CAP.

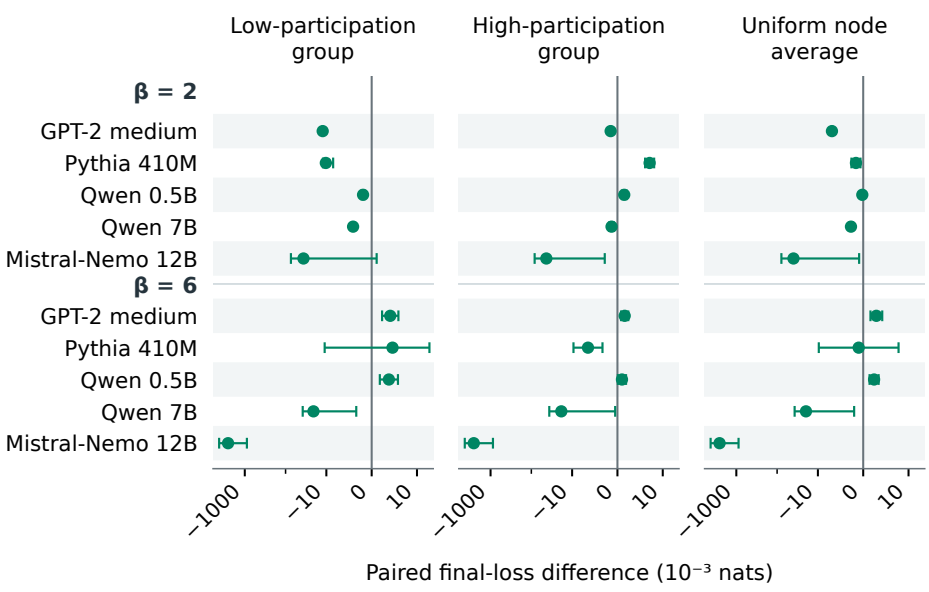


Figure 5: Paired final-loss differences, RUMOR minus windowed IPW with cap 20. Negative values favor RUMOR and positive values favor IPW. Points show means with one paired SE over ten matched seeds, except five at 12B and $\beta = 2$. Axes are linear within $\pm 10 \times 10^{-3}$ nats and logarithmic beyond. Figure 3 counts flagged runs at 7B and 12B.

The component comparisons test the roles of loss adaptation, participation history and neighbor disagreement. RUMOR, FIXED CAP fixes the cap to isolate loss adaptation, and RUMOR, SHORT WINDOW tests whether adding a shorter history window helps. Figure 10 in Appendix H.1 isolates the drift-based weight by comparing RUMOR with windowed IPW under the same loss-dependent cap. Its contribution is clearest at $\beta = 4$ on the small transformer and $\beta = 2$ on GPT-2 medium and Qwen 0.5B. RUMOR, NO HISTORY removes history correction and its cap, with mixed outcomes across regimes in Appendix H.1.

7 Discussion and Limitations

RUMOR combines strong participation correction with competitive losses across the evaluated settings. Its loss-dependent cap retains much of the fixed-cap variants' feedback correction and improves static performance. Across 120 small-transformer feedback runs using original and independent seeds, RUMOR has one flagged run versus 27 for windowed IPW with cap 20 (Appendix J.1). However, capped weights can leave participation imbalance uncorrected, possibly explaining IPW's small leads. The cap's behavior in Figure 6 in Appendix B.1 helps explain this balance. It permits stronger correction when recent loss is near its historical reference and lowers the maximum weight as the excess grows. Local loss supplements participation counts, which can have identical distributions under different participation rules, as Proposition 1 shows. The cap can remain large for a node improving more slowly than the network because it uses that node's loss history. An unusually low past loss or a lasting data shift can keep the cap small and limit correction.

Fresh neighbor models add a correction signal but require an extra model exchange and increase communication cost relative to history-only methods. Equal training rounds therefore do not establish equal communication cost. Reusing models from earlier exchanges avoids this communication cost, but yields lower gap reduction on GPT-2 medium at $\beta = 2$, as Appendix J.1 shows.

A smaller group loss gap can also reflect higher loss on high-participation data. Reweighting received models alongside RUMOR raises normalized final-gap reduction by 14 to 114 points on GPT-2 medium and Qwen 0.5B at $\beta = 2$ and 6. However, uniform node-average loss falls only on GPT-2 medium at $\beta = 2$, while higher high-participation-group loss drives most of the gap gain in the other three cases, as the paired comparisons in Appendix L.3 show. These distinctions matter especially when full participation barely changes the group gap.

AI USE STATEMENT

LLM assistants were used to supplement our literature search, assist with manuscript drafting and editing, provide additional checks of author-developed proofs, and help present the experimental results in plots. The authors reviewed the assisted work and take responsibility for the final content.

REPRODUCIBILITY STATEMENT

Sec. 3, Algorithm 1 and Appendices B–D, K and M document implementation, experiments and proofs. Code is available at `https://anonymous.4open.science/r/rumor-code/`. The repository implements RUMOR, its ablations and the comparison methods for the small-transformer and pretrained-model testbeds. It includes training and analysis code, experiment settings, software requirements and plotting scripts.

REFERENCES

Alp Berke Ardic, Hulya Seferoglu, Salim El Rouayheb, and Erdem Koyuncu. Random Walking Snakes for Decentralized Learning at Edge Networks. In *IEEE International Symposium on Local and Metropolitan Area Networks (LANMAN)*, pp. 1–6. IEEE, 2023.

Mahmoud Assran, Nicolas Loizou, Nicolas Ballas, and Mike Rabbat. Stochastic Gradient Push for Distributed Deep Learning. In *Proceedings of the 36th International Conference on Machine Learning*, volume 97 of *Proceedings of Machine Learning Research*, pp. 344–353. PMLR, 2019.

Chaouki Ben Issaid, Anis Elgabli, and Mehdi Bennis. DR-DSGD: A Distributionally Robust Decentralized Learning Algorithm over Graphs. *Transactions on Machine Learning Research*, 2022.

Stella Biderman, Hailey Schoelkopf, Quentin Gregory Anthony, Herbie Bradley, Kyle O'Brien, Eric Hallahan, Mohammad Aflah Khan, Shivanshu Purohit, USVSN Sai Prashanth, Edward Raff, Aviya Skowron, Lintang Sutawika, and Oskar van der Wal. Pythia: A Suite for Analyzing Large Language Models Across Training and Scaling. In *International Conference on Machine Learning (ICML)*, volume 202, pp. 2397–2430, 2023.

Alexander Borzunov, Dmitry Baranchuk, Tim Dettmers, Max Ryabinin, Younes Belkada, Artem Chumachenko, Pavel Samygin, and Colin Raffel. Petals: Collaborative Inference and Fine-tuning of Large Models. In *ACL System Demonstrations*, pp. 558–568, 2023.

Stephen Boyd, Arpita Ghosh, Balaji Prabhakar, and Devavrat Shah. Randomized Gossip Algorithms. *IEEE Transactions on Information Theory*, 52(6):2508–2530, 2006.

Yae Jee Cho, Jianyu Wang, and Gauri Joshi. Towards Understanding Biased Client Selection in Federated Learning. In *International Conference on Artificial Intelligence and Statistics (AISTATS)*, volume 151, pp. 10351–10375, 2022.

Martijn de Vos, Sadegh Farhadkhani, Rachid Guerraoui, Anne-Marie Kermarrec, Rafael Pires, and Rishi Sharma. Epidemic Learning: Boosting Decentralized Learning with Randomized Communication. In *Advances in Neural Information Processing Systems*, volume 36, pp. 36132–36164. Curran Associates, Inc., 2023. doi: 10.52202/075280-1567.

Arthur Douillard, Qixuan Feng, Andrei A. Rusu, Rachita Chhaparia, Yani Donchev, Adhiguna Kuncoro, Marc'Aurelio Ranzato, Arthur Szlam, and Jiajun Shen. DiLoCo: Distributed Low-Communication Training of Language Models. In *Workshop on Advancing Neural Network Training (WANT) at the International Conference on Machine Learning*, 2024.

Jeffrey M. Ede and Richard Beanland. Adaptive learning rate clipping stabilizes learning. *Machine Learning: Science and Technology*, 1(1):015011, 2020.

Yann Fraboni, Richard Vidal, Laetitia Kameni, and Marco Lorenzi. A General Theory for Client Sampling in Federated Learning. In *Trustworthy Federated Learning (FL 2022)*, volume 13448 of *Lecture Notes in Computer Science*, pp. 46–58. Springer, 2023.

Sajjad Ghiasvand, Mahnoosh Alizadeh, and Ramtin Pedarsani. Decentralized Low-Rank Fine-Tuning of Large Language Models. In *Workshop for Research on Agent Language Models (REALM)*, pp. 334–345, 2025.

David J. Goetze, Dahlia J. Felten, Jeannie R. Albrecht, and Rohit Bhattacharya. FLOSS: Federated Learning with Opt-Out and Straggler Support. In *Proceedings of the 4th Workshop on Practical Adoption Challenges of ML for Systems*, PACMI '25, pp. 69–74. Association for Computing Machinery, 2025.

Xinran Gu, Kaixuan Huang, Jingzhao Zhang, and Longbo Huang. Fast Federated Learning in the Presence of Arbitrary Device Unavailability. In *Advances in Neural Information Processing Systems*, volume 34, pp. 12052–12064, 2021.

Tatsunori B. Hashimoto, Megha Srivastava, Hongseok Namkoong, and Percy Liang. Fairness Without Demographics in Repeated Loss Minimization. In *International Conference on Machine Learning (ICML)*, volume 80, pp. 1929–1938, 2018.

Chanuka A. S. Hewa Kaluannakkage and Rajkumar Buyya. Asynchronous Decentralized Federated Learning over Lossy Wireless Links via Reception- and Age-Aware Aggregation. *arXiv preprint arXiv:2606.10774*, 2026.

D. G. Horvitz and D. J. Thompson. A Generalization of Sampling Without Replacement from a Finite Universe. *Journal of the American Statistical Association*, 47(260):663–685, 1952.

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 (ICLR)*, 2022.

Edward L. Ionides. Truncated Importance Sampling. *Journal of Computational and Graphical Statistics*, 17(2):295–311, 2008.

Sami Jaghouar, Jack Min Ong, Manveer Basra, Fares Obeid, Jannik Straube, Michael Keiblinger, Elie Bakouch, Lucas Atkins, Maziyar Panahi, Charles Goddard, Max Ryabinin, and Johannes Hagemann. INTELLECT-1 Technical Report. *arXiv preprint arXiv:2412.01152*, 2024a.

Sami Jaghouar, Jack Min Ong, and Johannes Hagemann. OpenDiLoCo: An Open-Source Framework for Globally Distributed Low-Communication Training. *arXiv preprint arXiv:2407.07852*, 2024b.

Divyansh Jhunjhunwala, Pranay Sharma, Aushim Nagarkatti, and Gauri Joshi. FedVARP: Tackling the Variance Due to Partial Client Participation in Federated Learning. In *Conference on Uncertainty in Artificial Intelligence (UAI)*, volume 180, pp. 906–916, 2022.

R. E. Kalman. A New Approach to Linear Filtering and Prediction Problems. *Journal of Basic Engineering*, 82(1):35–45, 1960.

Anastasia Koloskova, Nicolas Loizou, Sadra Boreiri, Martin Jaggi, and Sebastian U. Stich. A Unified Theory of Decentralized SGD with Changing Topology and Local Updates. In *International Conference on Machine Learning (ICML)*, volume 119, pp. 5381–5393, 2020.

Tian Li, Maziar Sanjabi, Ahmad Beirami, and Virginia Smith. Fair Resource Allocation in Federated Learning. In *International Conference on Learning Representations (ICLR)*, 2020.

Tian Li, Ahmad Beirami, Maziar Sanjabi, and Virginia Smith. Tilted Empirical Risk Minimization. In *International Conference on Learning Representations (ICLR)*, 2021.

Xiangru Lian, Ce Zhang, Huan Zhang, Cho-Jui Hsieh, Wei Zhang, and Ji Liu. Can Decentralized Algorithms Outperform Centralized Algorithms? A Case Study for Decentralized Parallel Stochastic Gradient Descent. In *Advances in Neural Information Processing Systems*, volume 30, 2017.

Ilya Loshchilov and Frank Hutter. Decoupled Weight Decay Regularization. In *International Conference on Learning Representations (ICLR)*, 2019.

Mistral AI team. Mistral NeMo. Mistral AI release announcement, July 2024. Published 18 July 2024.

Mehryar Mohri, Gary Sivek, and Ananda Theertha Suresh. Agnostic Federated Learning. In *International Conference on Machine Learning (ICML)*, volume 97, pp. 4615–4625, 2019.

Gota Morishita. Who Trains Matters: Federated Learning under Enrollment and Participation Selection Biases. *arXiv preprint arXiv:2604.26604*, 2026.

Angelia Nedić, Alex Olshevsky, and Wei Shi. Achieving Geometric Convergence for Distributed Optimization over Time-Varying Graphs. *SIAM Journal on Optimization*, 27(4):2597–2633, 2017.

Juan C. Perdomo, Tijana Zrnic, Celestine Mendler-Dünner, and Moritz Hardt. Performative Prediction. In *International Conference on Machine Learning (ICML)*, volume 119, pp. 7599–7609, 2020.

Qwen Team. Qwen2.5 Technical Report, 2024. arXiv 2412.15115, version 1.

Alec Radford, Jeffrey Wu, Rewon Child, David Luan, Dario Amodei, and Ilya Sutskever. Language Models are Unsupervised Multitask Learners. Technical report, OpenAI, 2019.

Nastaran Saadati, Zhanhong Jiang, Joshua R. Waite, Shreyan Ganguly, Aditya Balu, Chinmay Hegde, and Soumik Sarkar. DeCAF: Decentralized Consensus-And-Factorization for Low-Rank Adaptation of Foundation Models. *Neural Networks*, 202:108992, 2026.

Zhenyu Sun, Ziyang Zhang, Zheng Xu, Gauri Joshi, Pranay Sharma, and Ermin Wei. Debiasing Federated Learning with Correlated Client Participation. In *International Conference on Learning Representations (ICLR)*, 2025.

Adith Swaminathan and Thorsten Joachims. The Self-Normalized Estimator for Counterfactual Learning. In *Advances in Neural Information Processing Systems*, volume 28, pp. 3231–3239, 2015.

Jianyu Wang, Qinghua Liu, Hao Liang, Gauri Joshi, and H. Vincent Poor. Tackling the Objective Inconsistency Problem in Heterogeneous Federated Optimization. In *Advances in Neural Information Processing Systems*, volume 33, pp. 7611–7623, 2020.

Shiqiang Wang and Mingyue Ji. A Lightweight Method for Tackling Unknown Participation Statistics in Federated Averaging. In *International Conference on Learning Representations (ICLR)*, 2024.

Ming Xiang, Stratis Ioannidis, Edmund Yeh, Carlee Joe-Wong, and Lili Su. Efficient Federated Learning against Heterogeneous and Non-stationary Client Unavailability. In *Advances in Neural Information Processing Systems*, volume 37, pp. 104281–104328, 2024.

Lin Xiao and Stephen Boyd. Fast Linear Iterations for Distributed Averaging. *Systems & Control Letters*, 53(1):65–78, 2004.

Lin Xiao, Stephen Boyd, and Sanjay Lall. A Scheme for Robust Distributed Sensor Fusion Based on Average Consensus. In *Proceedings of the 4th International Symposium on Information Processing in Sensor Networks (IPSN)*, pp. 63–70, 2005.

Liyan Xie, Shaofeng Zou, Yao Xie, and Venugopal V. Veeravalli. Sequential (Quickest) Change Detection: Classical Results and New Directions. *IEEE Journal on Selected Areas in Information Theory*, 2(2):494–514, 2021.

Yikai Yan, Chaoyue Niu, Yucheng Ding, Zhenzhe Zheng, Shaojie Tang, Qinya Li, Fan Wu, Chengfei Lyu, Yanghe Feng, and Guihai Chen. Federated Optimization Under Intermittent Client Availability. *INFORMS Journal on Computing*, 36(1):185–202, 2024.

Haibo Yang, Xin Zhang, Prashant Khanduri, and Jia Liu. Anarchic Federated Learning. In *International Conference on Machine Learning (ICML)*, volume 162, pp. 25331–25363, 2022.

Rui Ye, Wenhao Wang, Jingyi Chai, Dihan Li, Zexi Li, Yinda Xu, Yaxin Du, Yanfeng Wang, and Siheng Chen. OpenFedLLM: Training Large Language Models on Decentralized Private Data via Federated Learning. In *Proceedings of the 30th ACM SIGKDD Conference on Knowledge Discovery and Data Mining*, pp. 6137–6147. Association for Computing Machinery, 2024.

Shahryar Zehtabi, Dong-Jun Han, Rohit Parasnis, Seyyedali Hosseinalipour, and Christopher G. Brinton. Decentralized Sporadic Federated Learning: A Unified Algorithmic Framework with Convergence Guarantees. In *International Conference on Learning Representations (ICLR)*, 2025.

Jianyi Zhang, Saeed Vahidian, Martin Kuo, Chunyuan Li, Ruiyi Zhang, Tong Yu, Guoyin Wang, and Yiran Chen. Towards Building the FederatedGPT: Federated Instruction Tuning. In *IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pp. 6915–6919. IEEE, 2024.

A Extended related work

Federated-learning methods address intermittent availability (Yan et al., 2024; Gu et al., 2021; Yang et al., 2022), unequal local work (Wang et al., 2020), biased selection (Cho et al., 2022) and client sampling (Fraboni et al., 2023). FedVARP stores clients' latest updates at a server to reduce partial-participation variance under uniform sampling (Jhunjhunwala et al., 2022). Sec. 2 and Appendix C describe the gossip adaptations used in the comparisons.

Loss can guide either the training objective or the size of an update. Fairness and distributionally robust objectives give greater emphasis to poorly served clients (Mohri et al., 2019; Li et al., 2020; Ben Issaid et al., 2022), as does tilted empirical risk minimization for larger losses when its tilt parameter is positive (Li et al., 2021). Adaptive learning rate clipping uses loss to control update size, reducing an update when loss exceeds a running reference (Ede & Beanland, 2020). RUMOR also uses loss to control update size while retaining the uniform node objective. Its cap limits participation-history weighting using the lowest smoothed local loss as the reference.

Local averaging underlies standard gossip mixing (Xiao & Boyd, 2004). Stochastic Gradient Push couples local SGD with push-sum communication over sparse directed links (Assran et al., 2019), while Epidemic Learning exchanges models with randomly selected peers and averages received models (de Vos et al., 2023). Other analyses allow changing topology under expected consensus contraction (Koloskova et al., 2020) or intermittent computation and exchanges (Zehtabi et al., 2025). Random-walk learning offers a related communication pattern in which a global model moves between nodes. Random Walking Snakes extends this pattern by having selected neighbors cooperate on a partitioned update using the visited node's data (Ardic et al., 2023). This cooperative update of one moving model differs from gossip's mixing of separately trained local models.

Foundation-model work spans low-communication pretraining (Jaghouar et al., 2024b;a), server-coordinated fine-tuning (Zhang et al., 2024; Ye et al., 2024) and decentralized LoRA aggregation (Ghiasvand et al., 2025; Saadati et al., 2026). Appendix B.2 explains the difference between averaging LoRA factors and averaging their products. Participation weighting before product aggregation remains an unevaluated extension.

B Implementation details

This section specifies initialization, timing and constants for RUMOR and the comparison methods in Appendix C. Round indices are omitted for stored state.

B.1 RUMOR initialization and cap behavior

Algorithm 1 stores an initially empty participation window, loss statistics and the previous active-neighbor observation. Both loss averages start at $\log 32$ on the small transformer, using the cross-entropy of a uniform prediction over its 32-token vocabulary as the initial reference. On pretrained testbeds, $\ell_i = 5.0$ and $\bar{\ell}_i$ is undefined until the first batch loss. Initially $b_i = +\infty$ and $s_i, v_i, \theta_i^{\text{last}}, t_i^{\text{last}}$ are undefined. On the small transformer, the reference minimum is updated for every node each round, so $b_i = \log 32$ after the first round for a node that has not participated. The pretrained reference first becomes finite after a batch loss. For defined state, $\mathrm{EMA}_\alpha(x, y) = (1-\alpha)x + \alpha y$, otherwise it takes y. Table 1 collects RUMOR's constants and the additional window length used by RUMOR, SHORT WINDOW. The cap endpoints were retained from tight- and loose-cap IPW comparisons during method development. For example, Equation 5 requires participation probabilities at least $1/3.33 \approx 0.30$ and 0.05 at caps 3.33 and 20, respectively, to match the full-participation multiplier.

At probability 0.2, cap 3.33 limits the average multiplier to 0.67. The fast-loss rates 0.1 and 0.2 assign more weight to each new batch than the slow rate 0.02, so recent loss responds faster than its reference. The drift averages retain rate 0.1 to smooth successive observations. The threshold $\gamma = 1.5$ enables drift correction only when history already assigns more than 1.5 times the unweighted step.

Table 1: RUMOR constants. Paired values give small-transformer / pretrained settings. The short window applies only to RUMOR, SHORT WINDOW. The clipping bound is shared with the comparison methods and applied before weighting.

symbol	meaning	value
α_f	recent-loss EMA rate	0.1 / 0.2
α_s	slow-loss EMA rate	0.02
α_d	EMA rate of s_i and v_i	0.1
C	base history cap and drift-weight cap	3.33
C_{hi}	maximum history cap	20
δ	loss-excess response scale	0.08 nats
γ	history-weight threshold for drift correction	1.5
H	participation-history length in rounds	200 / 100
ϵ_{num}	numerical denominator floor	10^{-12}
H_{short}	short-window length in rounds	20 / 12
–	multiplier of s_i in the drift denominator	$1/2$
–	raw-gradient Euclidean norm limit	5

Loss is measured in nats, using natural logarithms. Substituting the design constants into Equation 3 gives cap 20 at zero excess, 9.5 at excess $\delta = 0.08$ nats and 3.7 at 0.3 nats. These values illustrate how the chosen loss scale changes the cap. Initially $D_i = 0$, so the cap is large. After m initial absences, the first weight can reach $\min(m + 2, C_{\mathrm{hi}})$. All runs include this transient.

From the second pretrained participation event, ℓ_i generally exceeds local loss and keeps the cap small. The recent-loss average ℓ_i takes about fifteen events to decay to within δ of local loss. GPT-2 medium's Turkish/Swahili nodes are exceptions because their losses are near the initial value 5.0. On the small transformer, the cap stays large while loss falls.

Figure 6 checks whether the loss-dependent cap actually changes during training. Over 3000 rounds, low-participation-node caps become smaller under feedback than under static participation, allowing the rule to retain stronger history correction in the latter setting. The nonlinear cap formula makes the average cap differ from its value at average loss excess. From the first 500 rounds to rounds after 1500, the share of low-participation caps above 10 falls from 97 to 73 percent at $\beta = 4$, and 94 to 44 percent at $\beta = 6$. Static shares remain between 89 and 92 percent. To verify that these trajectories follow the intended updates, we replay Algorithm 1 from recorded batch losses and recover every logged cap and loss excess. The component ablations then test how these updates affect performance.

B.2 COMMUNICATION TIMING AND ADAPTER REPRESENTATION

As specified in Algorithm 1, pretrained drift rules read neighbor parameters before local updates, then exchange updated parameters for mixing. The extra exchange doubles model traffic in a message implementation, although the simulation provides the pre-update models directly.

On the small transformer, RUMOR, RUMOR, FIXED CAP, RUMOR, SHORT WINDOW and RUMOR, NO HISTORY can reuse the previous disagreement and observation interval for weighting and rate updates when an active node has no active neighbor. Across fifteen diagnostic RUMOR runs, five per setting, reuse could directly change fewer than 0.05% of active-event weights at $\beta = 4$ and 6, and none under static participation. This count does not bound later effects through the disagreement rate. The pretrained implementation follows Algorithm 1.

Pretrained nodes concatenate rank-8 adapter factors from the same attention projections into one gossip state. This vector determines the size of each model exchange and of any additional model-sized state. It contains 786,432 entries on GPT-2 medium and Pythia 410M, 540,672 on Qwen 0.5B, 2,523,136 on Qwen 7B and 4,915,200 on Mistral-Nemo 12B. Mixing and evaluation use factor averages, whose products differ from averages of factor products. Disagreement is the Euclidean

Algorithm 1 One round of RUMOR at node i, as used by the pretrained implementation.

Read $\theta_i = \theta_i^t$, indicator ξ_i^t (active if 1), and the state initialized above
$\hat{\pi}_i \leftarrow (c_i + 1)/(m_i + 2)$, using c_i active entries in the preceding window of length m_i
$D_i \leftarrow \max(\ell_i - b_i, 0), \quad C_i \leftarrow C + (C_{\mathrm{hi}} - C)e^{-D_i/\delta}$
$h_i \leftarrow \min(1/\hat{\pi}_i, C_i), \quad w_i \leftarrow h_i$
if i is active and has an active neighbor **then**
  $\bar{\theta} \leftarrow$ mean of the current active-neighbor models before their updates
  $d \leftarrow \|\theta_i - \bar{\theta}\|_2$
  if t_i^{last} is defined **then**
    $k_{\mathrm{obs}} \leftarrow t - t_i^{\mathrm{last}}$
    $s_i \leftarrow \mathrm{EMA}_{\alpha_d}\left(s_i, \|\bar{\theta} - \bar{\theta}_i^{\mathrm{last}}\|_2 / k_{\mathrm{obs}}\right)$
    if $h_i > \gamma$ **then**
      $r_i \leftarrow \min(1 + d/\max(v_i, s_i/2, \epsilon_{\mathrm{num}}), C)$, omitting undefined v_i
      $w_i \leftarrow \max(h_i, r_i)$
    end if
    $v_i \leftarrow \mathrm{EMA}_{\alpha_d}(v_i, d/k_{\mathrm{obs}})$
  end if
  $\bar{\theta}_i^{\mathrm{last}} \leftarrow \bar{\theta}, \quad t_i^{\mathrm{last}} \leftarrow t$
end if
$\widetilde{\theta}_i \leftarrow \theta_i$
if i is active **then**
  Evaluate batch loss L_{batch} and its gradient g at θ_i
  Clip g to Euclidean norm at most 5, then set $\widetilde{\theta}_i \leftarrow \theta_i - \eta_t w_i g$
  $\ell_i \leftarrow \mathrm{EMA}_{\alpha_f}(\ell_i, L_{\mathrm{batch}})$
  $\bar{\ell}_i \leftarrow \mathrm{EMA}_{\alpha_s}(\bar{\ell}_i, L_{\mathrm{batch}})$
  $b_i \leftarrow \min(b_i, \bar{\ell}_i)$
end if
Append ξ_i^t to the window and retain at most H entries
Active nodes exchange $\widetilde{\theta}$ and set θ_i^{t+1} by mixing in Equation 1
Inactive nodes retain $\theta_i^{t+1} = \theta_i^t$

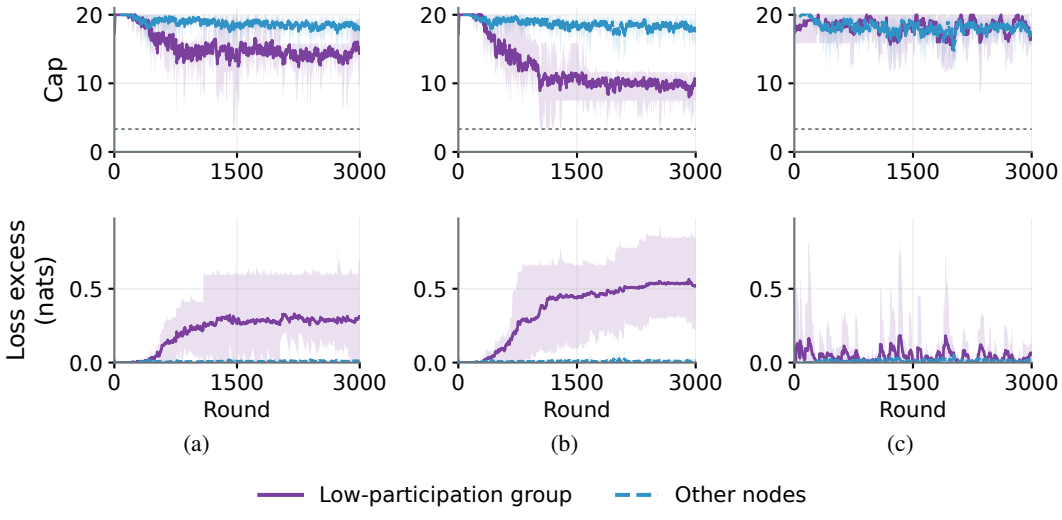


Figure 6: RUMOR cap C_i and loss excess D_i over training rounds on the small transformer, five seeds per setting. Columns (a), (b) and (c) show loss-adaptive participation at $\beta = 4$ and $\beta = 6$, then static participation. Purple solid lines average the four low-participation nodes and blue dashed lines the other twelve, with bands spanning seeds. The dotted line marks the small cap 3.33.

distance between factor vectors. TDW and FPW store two additional vectors of this adapter size and the scalars in Table 2, while history-only rules store scalars.

B.3 State and updates of comparison methods

TDW records the neighbor mean when a node participates with at least one active neighbor. It stores a slow mean μ_i of these observations, the previous neighbor mean $\bar{\theta}_i^{\text{last}}$ and its observation round t_i^{last}, and the scalars in Table 2. B_i, σ_i^2, V_i initialize from their first observation, $\bar{e}_i$ from zero and μ_i from the first neighbor mean. σ_i^2 averages squared Euclidean distances between neighbor means observed in consecutive rounds. V_i averages the neighbor mean's squared Euclidean distance from μ_i at each subsequent observation.

Table 2: TDW and windowed IPW constants.

symbol	meaning	value
α_B	EMA rate of B_i, σ_i^2, V_i	0.05
α_μ	per-round rate of the slow neighbor mean	1/60
α_e	EMA rate of $\bar{e}_i$	0.02
S_i^t	estimated saturation scale	$2V_i$
T_i^t	estimated time constant	$\max(S_i^t/\sigma_i^2, 1)$
z_i^t	normalized excess clipping range	$[-0.5,\ 0.95]$
C	default TDW cap, unless labeled 20	3.33
ϵ_{num}	numerical floor for denominators	10^{-12}
H	windowed IPW history, small / pretrained	200 / 100
$\hat{\pi}_i^t$	windowed estimate, $m_i^t \le H$	$(c_i^t + 1)/(m_i^t + 2)$
τ	IPW floor, cap 20 / cap 3.33	0.05 / 0.3

Windowed IPW estimates participation from the most recent H indicators. The uniform-prior posterior mean in Table 2 adds one active and one inactive pseudocount to avoid a zero estimate when the observed window contains no participation.

AH-IPW adapts its history after detected changes in participation, using the estimate $\hat{\pi}_{i,\text{AH}}^t = (c_i^{\text{AH}} + 1)/(m_i^{\text{AH}} + 2)$. Its effective window grows to H, after which the count is discounted through

$$c_i^{\text{AH}} \leftarrow c_i^{\text{AH}} + \xi_i^t - c_i^{\text{AH}}/m_i^{\text{AH}}.$$

Each round, a two-sided CUSUM-style detector (Xie et al., 2021) standardizes the new indicator and accumulates positive and negative deviations,

$$\zeta_i^t = \frac{\xi_i^t - \hat{\pi}_{i,\text{AH}}^t}{\sqrt{\max\Big(\hat{\pi}_{i,\text{AH}}^t(1 - \hat{\pi}_{i,\text{AH}}^t),\ \tau(1-\tau)\Big)}},$$
$$J_i^{\pm} \leftarrow \max(0, J_i^{\pm} \pm \zeta_i^t - 0.5),$$

where τ is the run's innovation floor. If either accumulator exceeds 5, both reset. The mean of the current indicator and eight preceding indicators restarts the participation estimate with effective count eight.

The weight $\min(1/\max(\hat{\pi}_{i,\text{AH}}^t, \phi), C)$ uses the history-weight floor ϕ and is computed before the current indicator enters the count. Fixed-cap runs use $(\phi, C) = (0.3, 3.33)$ or $(0.05, 20)$. Adaptive-cap runs replace C in this weight by C_i^t and set $\phi = 1/C_i^t$.

FPW estimates participation probability from indicators and the model-based drift observations in Appendix M. Its approximate Kalman-style filter (Kalman, 1960) stores mean λ_i and variance P_i for the participation logit, the logarithm of participation odds. Each drift observation is a logit with variance from the delta method, accepted only when its squared standardized residual from λ_i is at most χ. Process noise increases the filter's uncertainty so that its participation estimate can respond to changes. Let Q_i denote the variance added each round. The filter starts with $\lambda_i = 0$, $P_i = P_0 = \pi^2/3$ and $Q_i = 0$. Before processing the round's observations, it sets

$$P_i \leftarrow \min(P_i + Q_i, P_0).$$

FPW sets this added variance from changes in local loss. Whenever a participating node has an active neighbor, it records its loss EMA before the current batch. From the second such observation, it sets

$$Q_i = \frac{[g_{\mathrm{proc}}(\ell_i - \ell_i^{\mathrm{last}})]^2}{k},$$

where ℓ_i^{last} is the previously recorded loss EMA and k is the number of elapsed rounds. Larger loss changes give larger process variance. The new value applies from the next round and is retained until another observation updates it.

FPWI uses errors in predicting participation. After choosing the current weight, it computes $p_i = \sigma(\lambda_i)$ and the scaled prediction error

$$e_i = \frac{(\xi_i^t - p_i)^2}{\max(p_i(1-p_i),\, 0.05 \cdot 0.95)}.$$

A running average ν_i starts at one and updates as

$$\nu_i \leftarrow 0.98\nu_i + 0.02e_i, \qquad Q_i \leftarrow 0.5\max(\nu_i - 1, 0).$$

Thus FPWI adds process variance when the average prediction error exceeds one. These updates precede the indicator update below, and the new Q_i applies from the next round.

The variance-moderated probability $\tilde{\pi}_i = \sigma(\lambda_i/\sqrt{1+(\pi/8)P_i})$ gives weight $\min(1/\max(\tilde{\pi}_i, \tau), C)$. After choosing the current weight, the indicator updates the filter for the next round. With $p_i = \sigma(\lambda_i)$ and $P_i^+ = P_i/(1 + P_i p_i(1-p_i))$, set

$$\lambda_i \leftarrow \mathrm{clip}_{[-\Lambda,\Lambda]}\left(\lambda_i + P_i^+(\xi_i^t - p_i)\right), \qquad P_i \leftarrow P_i^+.$$

The uniform probability prior gives $P_0 = \pi^2/3$. We use the following settings:

- The process-noise gain is $g_{\mathrm{proc}} = 4$. Each drift observation uses noise floor $R_0 = 1.5$, and the z-noise EMA has floor 0.02.
- The inversion validity band is $\epsilon = 0.01$. We exclude events with $d^2 \le 10^{-12}$, accept squared standardized residuals up to $\chi = 9$, and clip logits to $[-\Lambda, \Lambda]$ with $\Lambda = 6$.
- The probability floor is $\tau = 0.05$ and the update-weight cap is $C = 3.33$.

Table 3 compares FPW and FPWI, while additional process-noise gains 2, 4 and 8 yield small-transformer reductions 77.2, 80.1 and 85.0 at $\beta = 4$, and 89.4, 87.8 and 88.6 at $\beta = 6$. These comparisons vary how the filter responds to changes in its observations.

RUMOR, FIXED CAP and **RUMOR, SHORT WINDOW** follow Algorithm 1 with floor 0.3 and fixed cap C. Comparing RUMOR with RUMOR, FIXED CAP tests the loss-dependent cap in Figures 2 and 15. Relative to RUMOR, FIXED CAP, RUMOR adds the constants C_{hi} and δ and three loss statistics per node. The second active-neighbor event initializes s_i before weighting and v_i afterward. RUMOR, SHORT WINDOW also takes the smaller of long- and short-window history weights, with a short window of 20 rounds on the small transformer and 12 on pretrained testbeds. Its comparison with RUMOR, FIXED CAP in Table 3 tests that additional history window.

RUMOR, NO HISTORY removes history correction and its activation threshold, testing the drift-based rule alone in Figure 4 and Table 3. It replaces Equation 4 by $w_i^t = r_i^t$ with the same update order and fixed drift cap $C = 3.33$, using weight one until the neighbor-mean change rate is defined. On pretrained testbeds it also uses weight one without an active neighbor. The complementary ablation in Figure 10 retains the loss-dependent history cap and removes the drift-based weight.

Reweighted mixing gives more weight to models received from infrequent neighbors, adapting reception-probability weighting (Hewa Kaluannakkage & Buyya, 2026). Each node counts its own participation events and those shared with each neighbor over 200 rounds on the small transformer and 100 rounds on pretrained testbeds. Dividing the joint count plus one by the own-participation count plus two estimates how often the neighbor participates when the node itself is active. The inverse estimate, using probability floor 0.3 and weight cap 3.33, multiplies the neighbor's full-graph Metropolis weight. The node then makes the mixing weights sum to one. If the received weights exceed one, it rescales them and gives its own model zero weight. Otherwise, its own model receives the remaining weight. This preserves a weighted average at each node, although the resulting matrix can change the network-average model because its columns may no longer sum to one.

The **variance-based cap** tracks errors in predicting whether the node participates. It smooths the standardized squared error $(\xi_i^t - \hat{\pi}_i^t)^2 / \max\{\hat{\pi}_i^t(1-\hat{\pi}_i^t), 0.05 \cdot 0.95\}$ with EMA rate 0.02. When this average is at most ν, it uses probability floor 0.05 and history cap 20. Otherwise, it uses floor 0.3 and cap 3.33, while the drift cap always stays at 3.33. The **alarm-based cap** uses the same two-sided detector as AH-IPW, applied to the windowed participation estimate. It allows the larger cap only after the specified number of rounds without an alarm and when both detector accumulators are below half the alarm level. Table 3 gives ν and the quiet-period lengths for these comparisons.

C Additional methods and controls

The comparisons below vary participation estimates, update weights and model exchange. Appendix B specifies their initialization and timing, and Appendix B.3 gives the state updates and constants.

C.1 Participation estimates and caps

RUMOR with AH-IPW estimator replaces the windowed history estimate by AH-IPW's discounted-count estimate to examine the effect of that choice. The drift-based weight can increase the update weight only when both AH-IPW and windowed weights exceed γ and Algorithm 1's active-neighbor and defined-rate conditions hold. Table 3 reports this variant and **AH-IPW with a loss-dependent cap**, which uses Equation 3 without the drift-based weight. **Windowed IPW with the same loss-dependent cap** is the complementary drift-removal ablation in Figure 10.

IPW with an adaptive floor is a comparison variant that limits large weights after a drop in participation. It compares participation estimates from the preceding 20 and 200 rounds, denoted $\hat{\pi}_{i,f}$ and $\hat{\pi}_i$. Both use the same pseudocount estimator.

Each node starts with probability floor $\tau_i = 0.05$. The **participation-only** rule raises this floor to 0.3 when

$$\hat{\pi}_i - \hat{\pi}_{i,f} > 3\sqrt{\max(\hat{\pi}_i(1-\hat{\pi}_i), 10^{-4})(1/20 - 1/200)}.$$

Otherwise, the floor moves gradually toward 0.05 according to $\tau_i \leftarrow 0.05 + 0.995(\tau_i - 0.05)$. The update weight is $w_i = 1/\max(\hat{\pi}_i, \tau_i)$.

The **participation-and-drift** rule uses the same trigger and pauses this relaxation while disagreement remains large. Each node starts by allowing the floor to relax. When it participates and its stored disagreement d_i and neighbor-change rate s_i are defined, it pauses relaxation if $d_i/\max(s_i, 10^{-8}) > 5$ and allows it otherwise. This decision is retained until the next such observation. A participation-drop trigger always raises the floor to 0.3. Figure 8 compares these two floor rules. Table 3 instead compares the **variance-based** and **alarm-based** cap rules defined in Appendix B.3 with the loss-dependent cap.

C.2 Disagreement and update-scale controls

Momentum-projected weighting (MPW) uses the direction of past disagreement to scale local updates. It stores an average disagreement vector u_i, which starts at zero. At a participating node with active neighbors, let $x_i = \theta_i - \bar{\theta}_i$ be the current disagreement vector and $q_i = \|u_i\|_2$. When $q_i > 10^{-12}$, its projection onto the stored direction is $z_i = \langle x_i, u_i \rangle / q_i$. Otherwise, the projection is zero.

MPW compares this projection with the speed of the neighbor mean. Let $\Delta m_i = \bar{\theta}_i - \bar{\theta}_i^{\text{last}}$ be the change since its previous observation, separated by k rounds. It maintains averages F_i and S_i of

$$\frac{\|\Delta m_i\|_2}{k} \quad \text{and} \quad \frac{\max(-\langle \Delta m_i, u_i \rangle / q_i, 0)}{k},$$

respectively. The first measures total movement. The second measures movement opposite to the stored disagreement direction. Each average starts from its first valid observation and then gives each new observation weight 0.1. Updating S_i requires $q_i > 10^{-12}$ and $\|x_i\|_2^2 > 10^{-12}$.

A baseline B_i averages the projection z_i when neighbor observations occur in consecutive rounds and $\|x_i\|_2^2 > 10^{-12}$. It uses the same averaging rule and is updated before selecting the weight. Once

both speed averages are available, MPW uses

$$w_i = \min\left(1 + \frac{\max(z_i - B_i, 0)}{\max(S_i, 0.2F_i, 10^{-8})}, C\right).$$

An undefined baseline contributes zero to this calculation. Before both speed averages are available, the weight is one.

When $\|x_i\|_2^2 \le 10^{-12}$, MPW uses the stored magnitude $z_i = q_i$ and leaves S_i and B_i unchanged. After weighting, it updates $u_i \leftarrow 0.8u_i + 0.2x_i$. At these near-consensus observations, it instead uses $u_i \leftarrow 0.8u_i$. It then stores the current neighbor mean and observation time. Nodes without active neighbors use weight one and retain their state. Table 3 and Figure 4 report this directional correction alongside the norm-based rules derived in Appendix M.

The **constant weight** is 1.96, matching DW's measured update-norm ratio to full participation. Figure 12 uses this control to examine how much of the correction comes from larger updates. **Matched total weight** gives every active node the same weight so that their weights sum to N. The idealized **self-normalized IPW** fixes the same total by globally rescaling inverse-propensity weights, retaining their relative differences (Swaminathan & Joachims, 2015). Figure 8 reports both controls.

Uncorrected training uses unit update weights under the specified participation rule. **Full participation** uses unit weights and updates and mixes every node in every round. These matched reference runs define the normalized reductions and absolute loss differences throughout the evaluation.

C.3 Update reuse and communication

Reusing and scaling updates Figure 8 compares gossip adaptations of replay, catch-up, credit accumulation and loss weighting. Appendix A explains their relation to the published methods.

- **Previous-gradient replay** lets inactive nodes apply their last stored gradients, retaining updates from nodes that do not currently train (Gu et al., 2021).
- **Catch-up weighting** compensates for missed updates by multiplying a returning gradient by rounds since participation, including the current round, without a cap (Xiang et al., 2024).
- The **Credit accumulation** comparison adapts compensation for missed participation to gossip updates (Xiang et al., 2024). It initializes $q_i = 0$. Each round it sets $\tilde{q}_i = \min(q_i + 1, 3K)$, uses $w_i = \min(\tilde{q}_i, K)$, and stores $q_i \leftarrow \tilde{q}_i - \xi_i^t w_i$, where K is the update-weight cap.
- **Loss weighting** uses $w_i = (\ell_i / \max(|\mathcal{A}_t|^{-1} \sum_{j \in \mathcal{A}_t} \ell_j, 10^{-12}))^q$ for active nodes $\mathcal{A}_t$, using the stored recent-loss EMAs before the current batch, with default exponent $q = 1$ (Li et al., 2020).
- **Interval-average weighting** averages intervals completed by participation or cutoff, including continued inactivity. Appendix L.2 defines the estimator, and Table 8 reports both cutoffs.

Model exchange

- **Model relay** lets every node exchange and mix models each round, while only participating nodes take local gradient steps. Inactive models can therefore change through averaging. Table 3 reports **IPW with cap 3.33 plus relay**. Appendix H.2 gives the all-method small-transformer relay tests, while Tables 4 and 5 report the separate 12B relay comparison.
- **Earlier-round RUMOR** computes disagreement using neighbor models received during previous gossip exchanges, allowing one model exchange per round. Appendix J.1 tests the effect of removing the separate exchange for fresh disagreement.

Mixing corrections

- **Reweighted mixing** scales received mixing weights by inverse estimated conditional participation probabilities to account for infrequent model reception without changing local updates. Table 9 evaluates it alone and with RUMOR, using the implementation in Appendix B.3.
- **Gradient tracking** mixes an auxiliary vector updated from changes in local gradients alongside the model, approximately doubling model-sized traffic (Nedić et al., 2017). Figure 8 reports this comparison.

D EXPERIMENT DETAILS

The small transformer has two layers, hidden width 128 and a vocabulary of 32 tokens, and trains across sixteen nodes. For each of four first-order Markov data groups and each current token, we draw eight successor-token indices uniformly with replacement from the 32-token vocabulary and draw their probabilities from $\mathrm{Dirichlet}(0.5\mathbf{1}_8)$. Initial tokens are uniform on the vocabulary. Each training batch selects group d_0 with probability 60% at four nodes and 2% at the other twelve; otherwise it selects one of the other three groups. We call d_0 the low-participation data group because it is concentrated on the four nodes assigned lower participation rates in the static setting. The same data assignment is used under loss-adaptive participation.

Each seed samples node positions independently and uniformly in $[0,1]^2$, joins pairs at Euclidean distance below $0.45\sqrt{16/N}$, and resamples until connected. Batches contain 16 sequences of context 64. Plain SGD uses rate 0.3 and clips the gradient's Euclidean norm to 5.0 before weighting. Training lasts 3000 rounds with evaluation every 250 rounds. In Equation 2, we set $c_0 = 1.5$, giving participation probability $\sigma(1.5) \approx 0.82$ when a node's loss EMA equals the network mean. We use $\beta \in \{2, 4, 6\}$ on the small transformer and $\beta \in \{2, 6\}$ on pretrained testbeds to compare weaker and stronger feedback. The broad comparison includes nineteen methods and controls, with twenty seeds per method and feedback setting. The constant multiplier 1.96 matches DW's measured update-norm ratio to full participation and is retained in the twenty-seed $\beta = 6$ control. DW and TDW use cap 3.33 unless stated otherwise.

The main pretrained experiments use rank-8 LoRA (Hu et al., 2022) on GPT-2 small and medium (Radford et al., 2019), Pythia 410M (Biderman et al., 2023), Qwen2.5 0.5B, 1.5B and 7B (Qwen Team, 2024), and Mistral-Nemo 12B (Mistral AI team, 2024). GPT-2 small and Qwen2.5 1.5B are evaluated only under static participation. These experiments use eight nodes on a random geometric graph in $[0,1]^2$ at radius $0.6\sqrt{8/N}$, resampled until connected. Each node trains on one language, with two nodes per language. Static participation probabilities are 0.2 for Turkish/Swahili nodes and 0.85 for English/German nodes. Loss-adaptive participation uses the same $c_0 = 1.5$ setting described above. Plain SGD uses rate 0.02 and the same clipping, with batches of 4 sequences of context 128 over 1000 rounds. Every 100 rounds, we evaluate the network-average adapter on text reserved for evaluation in each language. This text is divided into 128-token sequences, evaluated in batches of four. We use up to 24 batches per language, or all available evaluation sequences when there are fewer than 96. Backbones use FP32 for GPT-2 small and medium, FP16 for Pythia 410M and BF16 for Qwen and Mistral-Nemo. LoRA factors use FP32.

The integrated group loss gap measures both the size and duration of the disparity during training. At each evaluation, we take the part of the gap above 0.1 nats, sum these values and multiply by the 250-round evaluation interval: $\Delta t \sum_{t \in \mathcal{E}} \max(\mathrm{gap}_t - 0.1, 0)$, with $\mathcal{E} = \{250, 500, \ldots, 3000\}$ and $\Delta t = 250$ on the small transformer.

Paired comparisons start from matching seeds, graphs and data assignments and run for the same number of rounds. On the small transformer, methods share the random numbers used to decide participation and the candidate training batch for each node and round. Different losses can still give different participation probabilities, so the same random numbers can select different active nodes. On pretrained testbeds, a batch is sampled only when its node trains. Batch sampling and participation use the same random-number generator, so different active nodes can change both later batches and later participation draws. The numbers of local updates and model exchanges can therefore differ across methods.

Windowed IPW with cap 20 has flagged small-transformer runs with final group gaps of 0.30 to 1.00 nats on the original seeds and 0.31 to 2.04 on independent seeds. The flagged RUMOR and RUMOR, FIXED CAP gaps are 0.32 and 0.35 nats, both on independent seeds at $\beta = 6$. Under static participation at 12B, the flagged IPW run ends 6.0 nats above full participation.

For the small transformer's twenty original seeds at $\beta = 6$, counts above 0.2 nats are 8, 1 and 7 for RUMOR, RUMOR, FIXED CAP and IPW with cap 20. Above 0.5 nats they are 0, 0 and 2. The independent-seed counts are respectively 4, 2 and 11 at 0.2 nats, and 0, 0 and 6 at 0.5 nats.

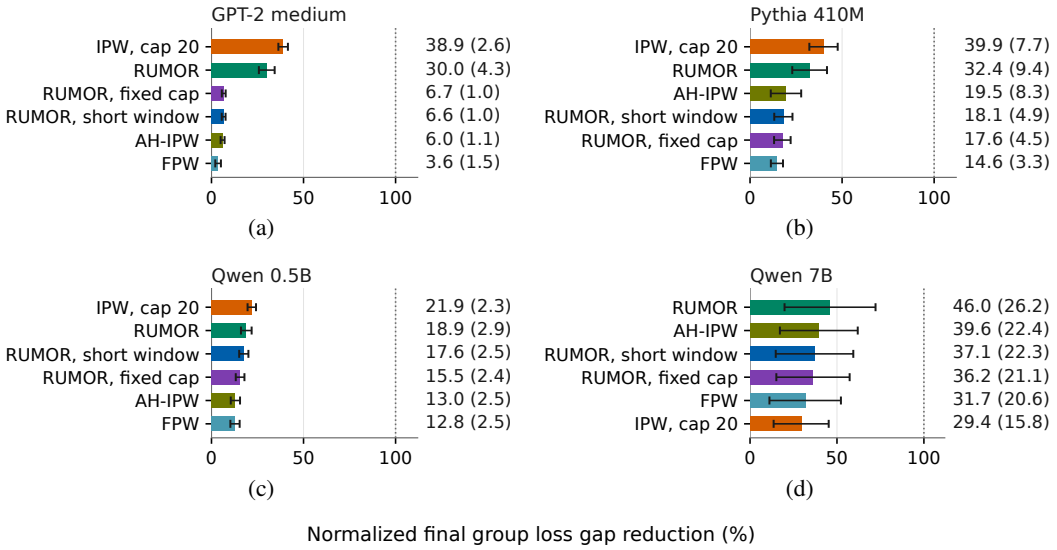


Figure 7: Methods using participation history under loss-adaptive participation at $\beta = 6$, ten matched seeds per model. Bars, SEs, references and cap labels follow Figure 4. No run is flagged. The 12B comparison uses absolute losses in Figure 3 because its normalization denominator is non-positive.

E RUMOR WITH MODIFIED ADAMW

We also evaluate RUMOR with a modified version of AdamW (Loshchilov & Hutter, 2019). In our variant, unweighted gradients update node-local moments, and the RUMOR weight scales the full optimizer step, including weight decay. On return after an absence, the optimizer decays the second moment across missed rounds and advances the per-node counter used for both moment bias corrections. The first moment is retained during absence.

At $\beta = 6$, five Qwen 7B seeds and three Mistral-Nemo 12B seeds use learning rate 10^{-5}, moment rates 0.9 and 0.999, weight decay 0.01 and 1000 rounds. Relative to matched uncorrected runs with the same optimizer, RUMOR changes the final group loss gap by -0.0177 nats (paired SE 0.0025) on Qwen 7B and 0.0099 (paired SE 0.0118) on Mistral-Nemo 12B. No RUMOR run is flagged in either setting.

F ADDITIONAL BASELINE COMPARISONS

F.1 PRETRAINED MODELS UNDER STRONGER FEEDBACK

Figure 7 compares the history-based methods at $\beta = 6$ on the matched pretrained runs. Windowed IPW with cap 20 has greater gap reduction on the three smaller models, while RUMOR has the largest reduction on Qwen 7B. The paired losses in Figure 5 put these rankings in perspective: on the three smaller models, the two methods' mean uniform node-average losses differ by less than 0.003 nats. Neither method has a flagged run in these four settings.

F.2 SMALL-TRANSFORMER WEIGHTING AND UPDATE RULES

Figure 8 compares the weighting and update rules defined in Appendix C. RUMOR and RUMOR, FIXED CAP have the two largest point estimates at both feedback strengths, with no flagged runs. The broad comparison combines repeated runs of the same nominal setup and seed set, each normalized to its own matched references. The negative reductions and flagged runs shown for other rules are results for the gossip adaptations defined in Appendix C.

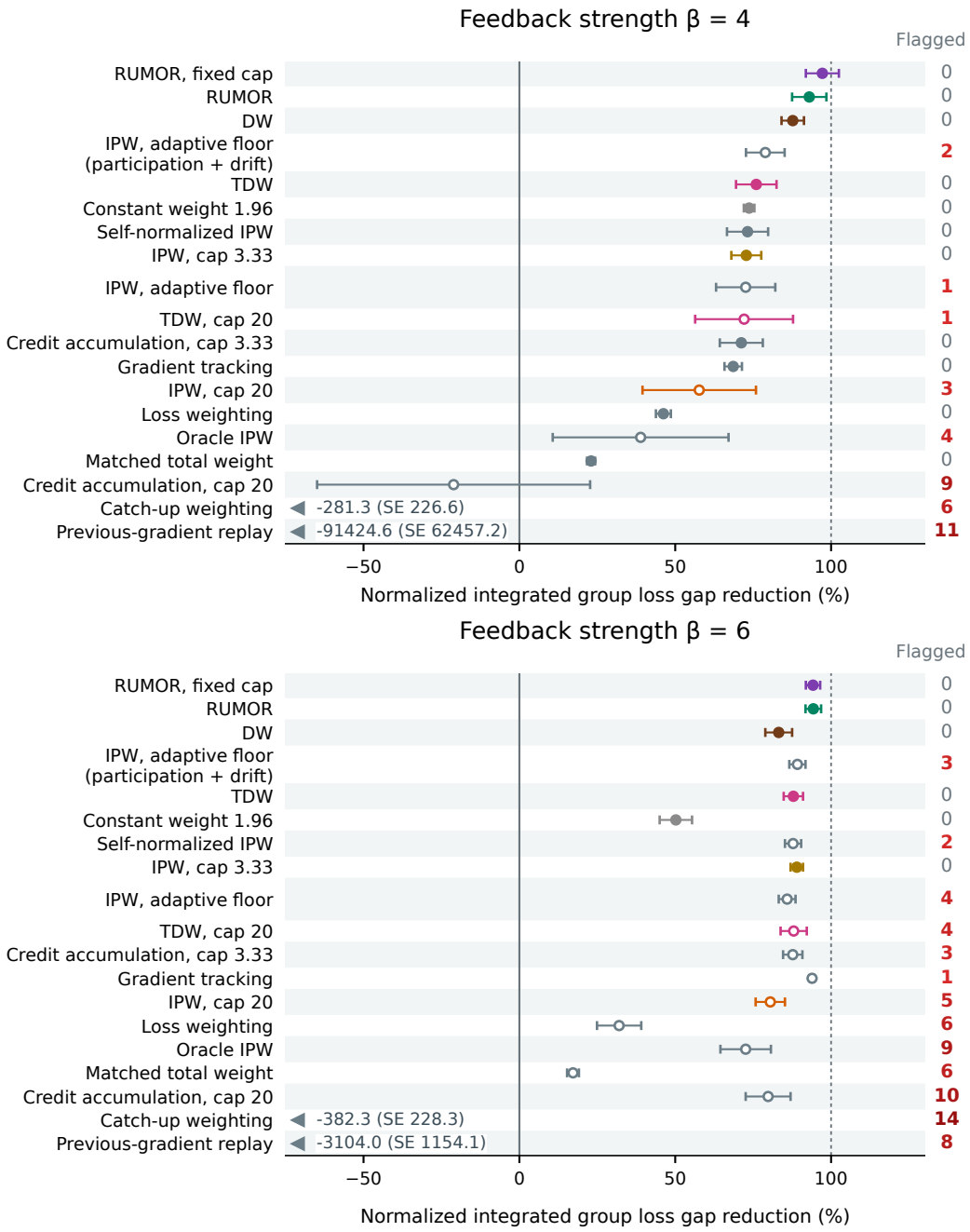


Figure 8: Nineteen methods and controls on the small transformer at $\beta = 4$ and $\beta = 6$, twenty seeds per method and setting. Points show normalized integrated group loss gap reduction with one leave-one-seed-out SE. Each panel's right-hand column gives flagged count k out of twenty. Hollow points denote $k > 0$. Left arrows mark off-scale estimates, whose values and SE remain printed. The dotted line marks full participation. Rows share the $\beta = 4$ ordering. Adaptive-floor IPW uses participation history alone unless labeled participation + drift. Appendix C defines the adaptations and cites their sources.

G REPLICATION ON INDEPENDENT SEEDS

The original seeds were used during method development. The independent seeds were reserved for later evaluation. As Figure 9 shows, RUMOR retains strong correction on twenty seeds excluded from method design and selection of the loss-excess scale δ. Each cell has uncorrected and full-participation references from the same batch of runs and seeds. At $\beta = 4$ and 6, both RUMOR and RUMOR, FIXED CAP retain high integrated group loss gap reduction, with paired leads of 1.1 (SE 0.5) and 1.2 (SE 0.9) points for RUMOR, FIXED CAP. Each method has one flagged run at $\beta = 6$. Under static participation, RUMOR's normalized final low-participation-group loss reduction exceeds RUMOR, FIXED CAP's by 12.5 points, compared with 13.8 on the original seeds.

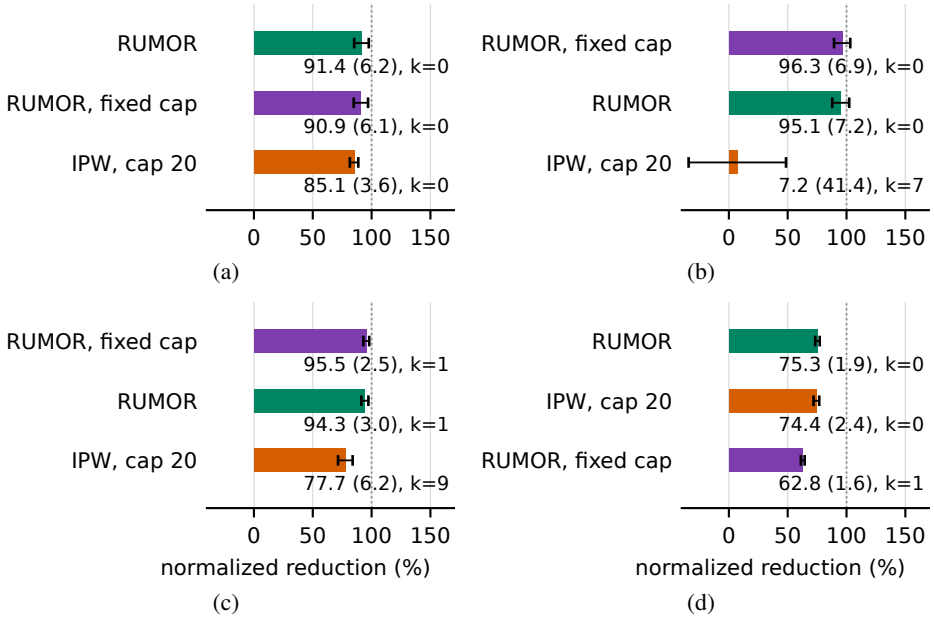


Figure 9: Replication on twenty seeds excluded from method selection. Normalized reductions use integrated group loss gap at $\beta = 2$, 4 and 6 in (a)–(c), and final low-participation-group loss under static participation in (d). Labels give estimate (leave-one-seed-out SE) and flags k. Zero is uncorrected and the dotted line full participation.

H ABLATIONS AND CONTROLS

H.1 HISTORY AND DISAGREEMENT ABLATIONS

Figure 10 removes only RUMOR's drift-based weight, while the complementary RUMOR, NO HISTORY ablation removes the history weight, its cap and threshold condition. The RUMOR, NO HISTORY comparisons below use shared seeds within each testbed but separate runs from the drift-removal comparison. On the small transformer, RUMOR, NO HISTORY exceeds RUMOR at $\beta = 2$ with normalized integrated group loss gap reduction 123.0 (SE 5.7), but loses correction at stronger feedback. It has no feedback flags and four static residual-deficit flags in Table 3. On Qwen 0.5B and 7B, its pretrained gap reductions in Figure 4 are negative, without flagged runs.

The losses in Table 7 and Figure 12 show that some gains from drift-only weights reflect update scale. Their uniform node-average losses can be lower despite worse gap reduction.

The recent-history ablation RUMOR, SHORT WINDOW also improves on IPW with cap 20 at $\beta = 2$, with paired gap-reduction gains of 50.8 points (SE 2.6), 46.6 (SE 4.2) and 62.7 (SE 14.8) on GPT-2 medium, Pythia 410M and Qwen 7B. At 7B, the full-participation reference improvement is 0.0039 nats, so the 62.7-point gain corresponds to a 0.0025-nat smaller gap. On Qwen 0.5B, IPW leads by 1.2 points (SE 2.4). DW and RUMOR, NO HISTORY resemble the constant control's weak gap reduction on Qwen 0.5B and 7B.

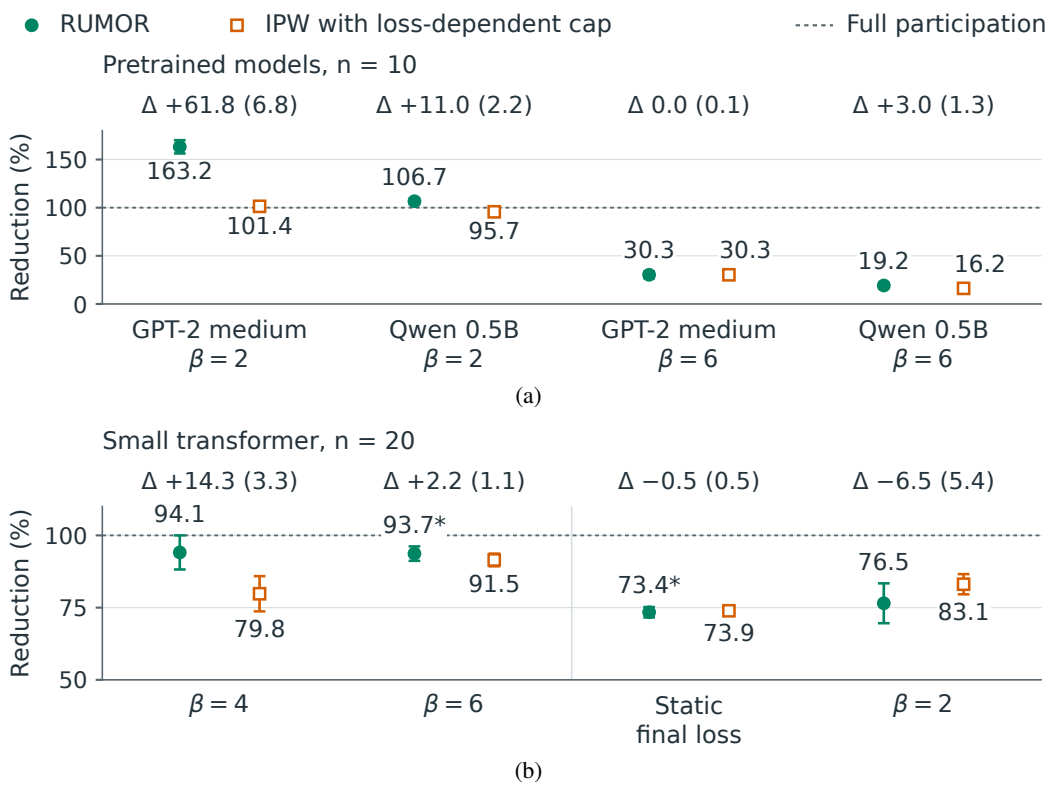


Figure 10: Drift-removal ablation with matched seeds. The reduced rule is windowed IPW with the same loss-dependent cap as RUMOR. Points show normalized reductions with one leave-one-seed-out SE, and Δ is RUMOR minus the reduced variant in points (paired SE). An asterisk marks one flagged run. Panel (a) uses final group loss gap and ten seeds. Panel (b) uses integrated group loss gap and twenty seeds, except static uses final low-participation-group loss. The reduction axis begins at 50 in panel (b) and includes lower values in panel (a). These runs are separate from the main comparisons and the independent-seed replication in Figure 9.

H.2 Participation assignments and update scale

Swapping GPT-2 medium's low-rate languages from Turkish/Swahili to English/German preserves similar deficit reductions, as shown in Figure 11. History-based methods approach full participation when the cap rises. Their similar behavior across language assignments supports the role of participation frequency and the cap in the static deficit, consistent with Equation 5 and Proposition 3.

Table 3 compares the variance-based and alarm-based caps defined in Appendix B.3. The variance-based cap responds to errors in predicting participation, while the alarm-based cap increases only after a sufficiently long period without a detected change. The variance-based cap has $\beta = 4$ flags and the alarm-based cap leaves a larger static deficit. RUMOR's loss-dependent cap combines stronger static correction with fewer flags. RUMOR, FIXED CAP was selected using the small-transformer settings and the assignment-swap experiment before evaluation on pretrained models under stronger feedback.

Reference deficits for Turkish (tr), Swahili (sw) and German (de) are respectively 0.026, 0.053 and 0.039 nats before the swap, and 0.021, 0.037 and 0.069 afterward. Full participation does not lower English loss relative to uncorrected training in either assignment, leaving no positive reference improvement for its normalized score. Figure 11 therefore omits English. DW's reductions above 100 mean that it lowers the displayed language's loss below full participation. Larger updates can

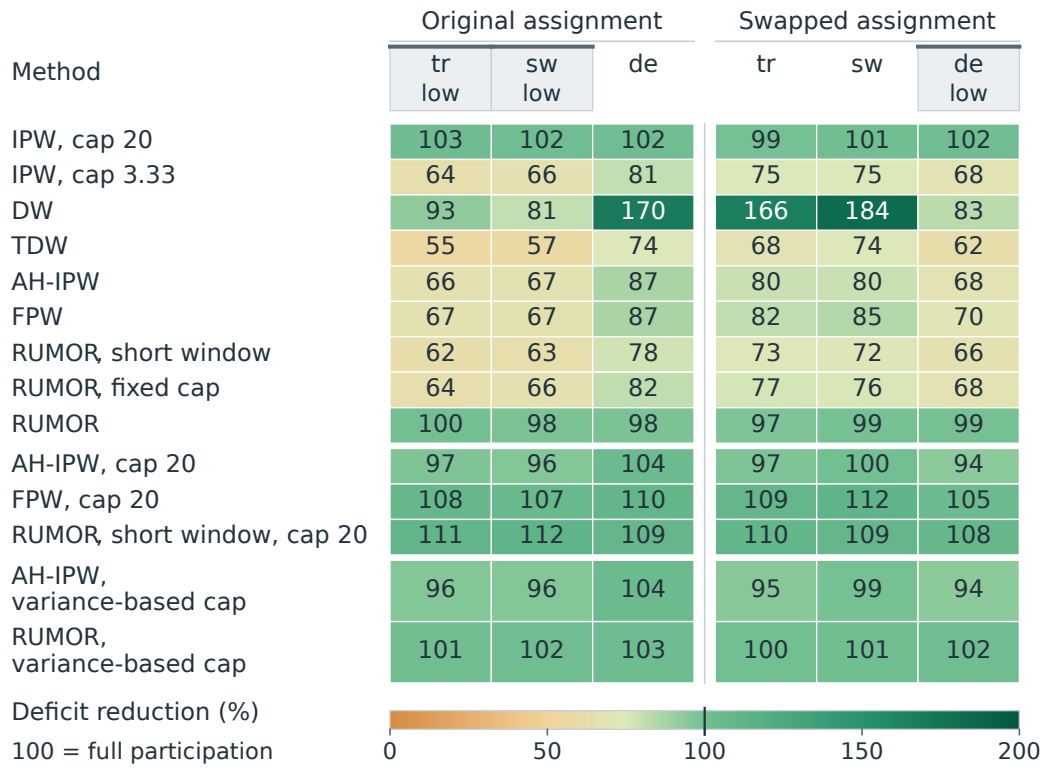


Figure 11: Per-language loss deficit reduction on static GPT-2 medium, ten seeds per assignment. Values are point estimates without SE, and no run is flagged. Zero matches the uncorrected run and 100 full participation. All rows use the same color scale, with greener cells indicating larger reductions. Headers mark low-participation languages.

improve both high- and low-participation groups by similar amounts, leaving their loss difference largely unchanged. Figure 12 separates these groupwise gains.

Learning-rate controls We repeat selected comparisons at half and twice the usual learning rate to check whether changing the overall update size explains the correction.

On the small transformer at $\beta = 4$, twenty seeds per setting use rates 0.15 and 0.6. At half the usual rate, normalized integrated group loss gap reductions are 101.6 for RUMOR, SHORT WINDOW, 92.0 for TDW and 70.7 for the constant-weight control. The first two each have three flagged runs. At twice the rate, the control reaches 74.3, compared with 54.5 and 42.3 for RUMOR, SHORT WINDOW and TDW, with large variation across runs.

On GPT-2 medium at $\beta = 2$, five seeds per setting use rates 0.01 and 0.04. At twice the usual rate, normalized final group loss gap reductions are 144.2 for RUMOR, SHORT WINDOW, 110.9 for TDW and 98.0 for IPW with cap 20, compared with -20.9 for the control. At half the rate, full participation has a 0.003-nat larger final gap than uncorrected training after 1000 rounds, so it provides no positive gap improvement against which to normalize these methods.

Figure 12 shows that the gains from RUMOR and the history-based variants are more concentrated on low-participation data than those from constant or drift-only weights. To check whether larger updates alone explain this gain, we replace each method's changing weights by its mean weight. On GPT-2 medium, the constant weight 1.55 gives normalized final group loss gap reduction 25, compared with 149 for RUMOR, SHORT WINDOW. Likewise, constant weight 1.41 gives 14, compared with 103 for TDW. The larger reductions from the changing weights show that their allocation across nodes and rounds matters beyond their average size.

RUMOR, FIXED CAP's paired leads over IPW at the same floor and cap are 16.7 points (SE 3.1) at $\beta = 4$ and 3.8 (SE 1.7) at $\beta = 6$. The corresponding leads over RUMOR, SHORT WINDOW are

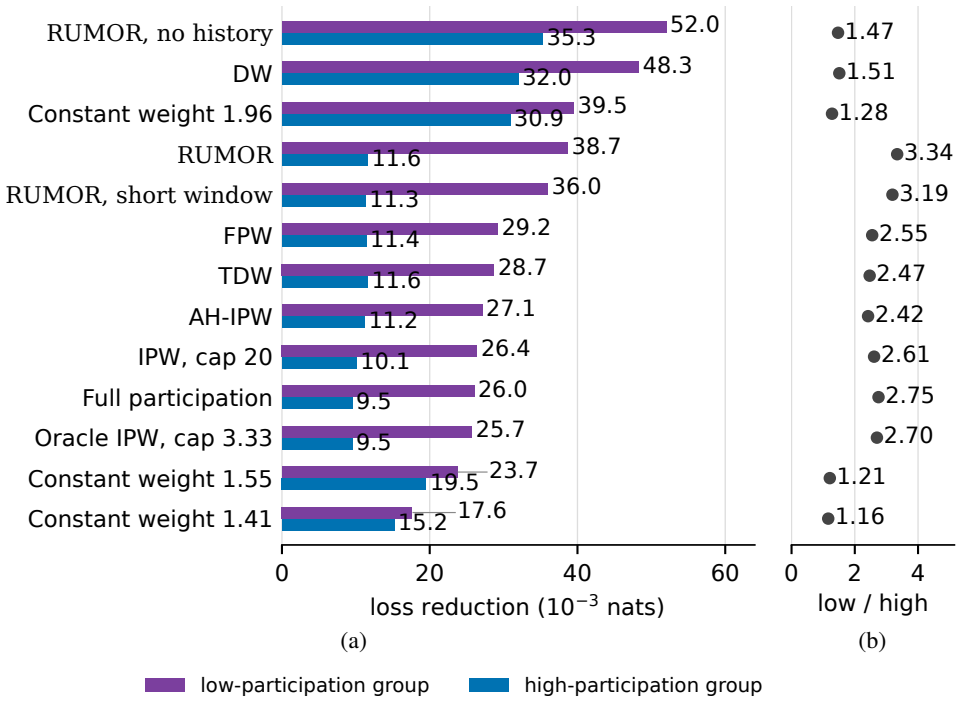


Figure 12: Final-loss reduction from the uncorrected run on GPT-2 medium at $\beta = 2$ after 1000 rounds, ten shared seeds without SE. Panel (a) separates the low-participation and high-participation groups in 10^{-3} nats. Panel (b) gives their ratio.

Table 3: Small-transformer normalized reductions with one leave-one-seed-out SE and flagged count k. Cells use twenty seeds, except † marks ten. Feedback uses integrated group loss gap and static uses final low-participation-group loss. Methods are grouped by family. Caps are 3.33 unless labeled 20, while RUMOR uses 3.33 to 20. Variance caps use $\nu = 1.5$ and alarm labels give the quiet period in rounds. Bold marks the largest reduction in each column.

method	$\beta = 4$	k	$\beta = 6$	k	static	k
RUMOR	93.0 (5.5)	0	94.3 (2.5)	0	73.8 (1.7)	1
RUMOR, FIXED CAP	**97.2 (5.3)**	0	94.2 (2.3)	0	60.0 (1.5)	3
RUMOR, SHORT WINDOW	86.3 (5.1)	0	91.5 (2.7)	0	59.4 (1.5)	5
RUMOR, NO HISTORY	90.5 (2.9)	0	73.7 (6.1)	0	72.5 (1.1)	4
RUMOR, VARIANCE-BASED CAP	66.3 (15.4)	3	84.5 (6.2)	7	70.3 (2.2)	1
RUMOR, ALARM-BASED CAP, 200 ROUNDS	93.8 (5.0)	0	**94.4 (2.4)**	1	61.4 (2.0)	3
RUMOR, ALARM-BASED CAP, 400 ROUNDS	95.9 (4.9)	0	94.2 (2.5)	0	59.7 (1.7)	4
RUMOR with AH-IPW estimator	85.3 (6.3)	0	91.6 (3.0)	1	71.2 (1.7)	1
IPW, cap 3.33	80.5 (5.3)	0	90.4 (2.0)	0	60.7 (2.3)†	3
IPW, cap 20	54.8 (12.7)	5	82.9 (4.5)	6	71.2 (2.2)	1
Oracle IPW, cap 20	3.8 (39.6)	5	69.8 (8.9)	8	73.3 (2.8)†	1
IPW, cap 3.33, relay	93.4 (3.4)	0	90.2 (6.6)	1	**74.4 (1.6)**	1
AH-IPW	80.8 (4.5)	0	89.6 (2.8)	0	60.3 (1.5)	4
AH-IPW, variance-based cap	69.9 (10.4)	2	88.6 (2.7)	3	69.8 (1.7)	1
AH-IPW, alarm-based cap, 200 rounds	80.4 (5.3)	0	89.3 (2.9)	1	63.1 (1.7)	4
AH-IPW, loss-dependent cap	80.3 (4.9)	0	88.3 (2.6)	2	71.2 (1.6)	1
FPW	80.1 (6.4)	0	87.8 (2.7)	2	60.2 (1.4)	4
FPWI	80.0 (5.4)	0	89.5 (2.6)	1	61.6 (1.5)	5
DW	87.0 (4.1)	0	83.9 (4.2)	0	68.0 (1.2)	5
TDW	76.9 (6.4)	0	87.9 (3.1)	0	50.1 (1.8)	10
MPW	61.1 (4.8)	0	69.7 (4.8)	0	43.4 (1.7)	15
Constant weight	73.7 (1.7)	0	50.2 (5.2)	0	60.7 (1.1)	9

10.9 (SE 3.4) and 2.7 (SE 1.0). *Oracle IPW* uses the simulator's true current participation probability under each result's specified cap. Even with these probabilities, the cap-20 runs retain the flags shown in Table 3 and Figure 2.

Two ten-seed stress tests change the ranking. Inactive-node relay raises history-based reductions to 89 to 101 percent but leaves TDW at 2.0 and 0.6 at $\beta = 4$ and 6, because disagreement no longer encodes missed training. Raising every cap to 20 flags nearly every rule at $\beta = 6$.

I ABSOLUTE LOSSES AND PAIRED COMPARISONS

Figure 13 extends the primary feedback comparisons in Figure 5 with static participation, relay and fixed-cap variants. Tables 4–7 retain the absolute losses. The signed differences distinguish small IPW advantages from larger RUMOR gains without treating every change in rank as equally large. Both RUMOR, FIXED CAP and RUMOR, SHORT WINDOW use a fixed history cap, and RUMOR, SHORT WINDOW also combines long- and short-window history weights. All flagged runs remain in the means, including non-relay 12B IPW runs at $\beta = 6$ and under static participation.

For pretrained models, the low- and high-participation groups are Turkish/Swahili and English/German. On the small transformer, low denotes d_0 and others denotes the mean loss over the remaining three data groups. Mean denotes the uniform node-average loss, with absolute losses rounded to three decimals. Normalized reductions are undefined for the additional 12B $\beta = 2$ and relay comparisons.

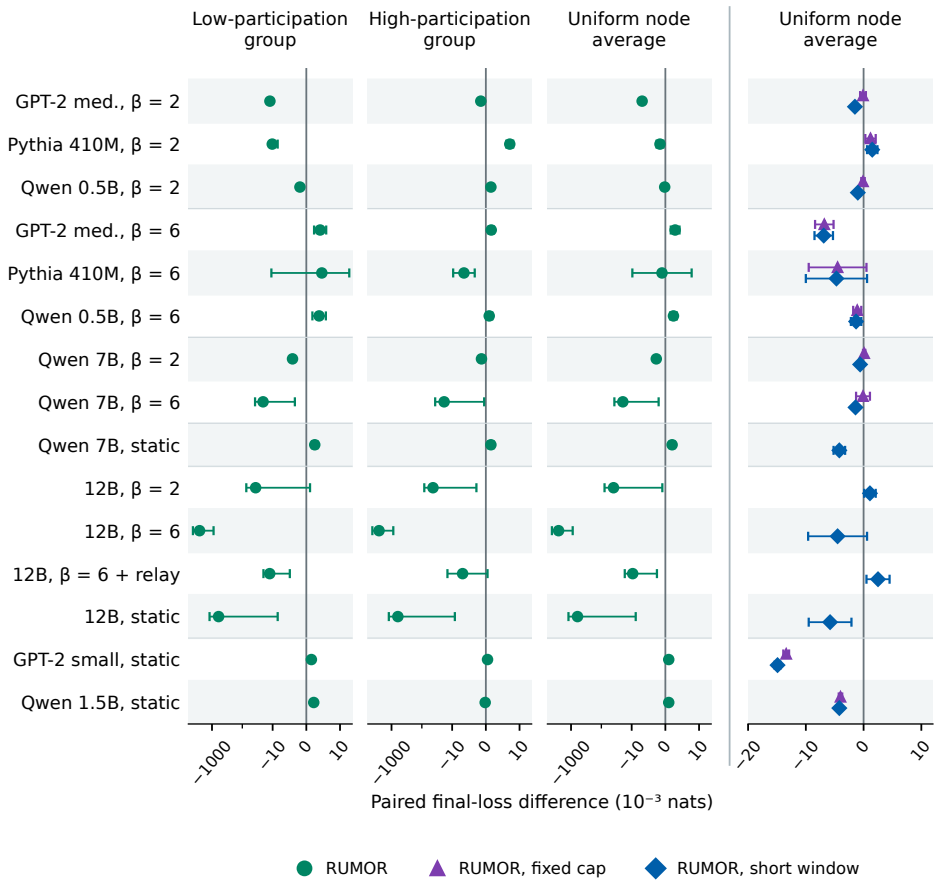


Figure 13: Paired final-loss differences in 10^{-3} nats, with mean and one paired SE over ten matched seeds, except five for 12B at $\beta = 2$. The first three panels subtract IPW with cap 20 from RUMOR, and the last shows RUMOR minus each labeled variant. Negative values favor RUMOR. The first three axes are linear between -10 and 10 and logarithmic beyond. The last axis is linear.

Table 4: Final group losses in nats on matched pretrained runs, n seeds per cell. Bold marks the lowest correction-rule loss, including ties.

testbed and regime	group	Uncorrected	Full	IPW cap 20	RUMOR, FIXED CAP	RUMOR, SHORT WINDOW	RUMOR	n
GPT-2 medium, $\beta = 2$	low	5.034	5.008	5.007	**4.995**	4.998	**4.995**	10
	high	3.298	3.289	3.288	**3.287**	**3.287**	**3.287**	
Pythia 410M, $\beta = 2$	low	3.686	3.653	3.651	**3.639**	**3.639**	3.641	10
	high	2.921	2.929	**2.930**	2.937	2.937	2.937	
Qwen 0.5B, $\beta = 2$	low	4.326	4.290	4.291	4.290	4.292	**4.289**	10
	high	2.571	2.574	**2.574**	2.575	2.575	2.576	
GPT-2 medium, $\beta = 6$	low	5.049	5.007	**5.031**	5.045	5.045	5.035	10
	high	3.303	3.289	**3.295**	3.300	3.300	3.296	
Pythia 410M, $\beta = 6$	low	3.781	3.653	**3.736**	3.756	3.756	3.741	10
	high	2.912	2.929	2.925	**2.913**	**2.913**	2.919	
Qwen 0.5B, $\beta = 6$	low	4.383	4.291	**4.365**	4.372	4.371	4.369	10
	high	2.573	2.574	**2.575**	2.576	2.577	2.576	
Qwen 7B, $\beta = 2$	low	3.056	3.046	3.046	**3.041**	3.042	3.042	10
	high	1.937	1.930	1.930	1.929	1.929	**1.928**	
Qwen 7B, $\beta = 6$	low	3.072	3.046	3.084	3.064	3.065	**3.063**	10
	high	1.943	1.930	1.958	**1.939**	1.940	1.940	
GPT-2 small, static	low	5.389	5.326	**5.325**	5.347	5.348	5.327	10
	high	3.612	3.571	**3.570**	3.578	3.579	3.571	
Qwen 1.5B, static	low	3.708	3.656	**3.656**	3.666	3.666	3.658	10
	high	2.268	2.259	**2.258**	**2.258**	2.259	**2.258**	

testbed and regime	group	Uncorrected	Full	IPW cap 20	RUMOR, SHORT WINDOW	RUMOR	n
Qwen 7B, static	low	3.095	3.046	**3.047**	3.056	3.049	10
	high	1.956	1.930	**1.932**	1.934	1.933	
Mistral-Nemo 12B, $\beta = 2$	low	2.515	2.510	2.547	**2.507**	2.511	5
	high	1.649	1.643	1.685	1.643	**1.642**	
Mistral-Nemo 12B, $\beta = 6$	low	2.512	2.510	5.083	2.514	**2.511**	10
	high	1.669	1.643	4.248	1.675	**1.668**	
Mistral-Nemo 12B $\beta = 6$, relay	low	2.520	2.510	2.535	**2.518**	2.522	10
	high	1.658	1.643	1.664	**1.656**	1.657	
Mistral-Nemo 12B, static	low	2.546	2.510	3.115	2.519	**2.509**	10
	high	1.658	1.643	2.261	1.644	**1.643**	

Evaluating each node's own adapter checks whether the network-average model obscures local outcomes. Over five GPT-2 medium seeds, both evaluations give the same method ranking, with own-adapter group losses within 0.004 nats of the network-average losses. For RUMOR, RUMOR, FIXED CAP and windowed IPW with cap 20, own-adapter normalized final group loss gap reductions are 151, 151 and 93 at $\beta = 2$, and 30, 6 and 37 at $\beta = 6$.

The $\beta = 2$ gap reductions in Figure 4 show gains over windowed IPW with cap 20 for both RUMOR and RUMOR, FIXED CAP. On Pythia 410M, their paired gains are 42.7 points (SE 6.7) and 45.5 (SE 5.4), respectively. On Qwen 7B, the gains are 69.4 points (SE 12.3) and 85.7 (SE 10.3). Thus RUMOR, FIXED CAP adds 2.8 points (SE 4.2) on Pythia and 16.3 (SE 6.6) on 7B to RUMOR's gain.

Figure 13 places these scores alongside absolute losses. At 7B, full participation improves the gap by only about 0.004 nats, which makes a small absolute difference large on the normalized scale. The two variants' uniform node-average losses differ by 0.0001 nats (SE 0.0002). At 12B and $\beta = 2$, the low- and high-participation nodes train in 71 and 89 percent of rounds on average, and no run is flagged. The paired absolute losses in Figure 13 include their uncertainty.

At $\beta = 6$, RUMOR has higher final group loss gap reduction than RUMOR, FIXED CAP on GPT-2 medium, Pythia 410M and Qwen 0.5B, and the largest reduction at 7B. Its paired 7B lead over RUMOR, FIXED CAP is 9.8 points (SE 9.3). IPW with cap 20 leads on the three smaller models, with neither rule flagged. On Pythia 410M, IPW's gap-reduction lead is 7.5 points (SE 10.0), while RUMOR minus IPW uniform node-average loss is -0.0010 nats (SE 0.0088), showing no clear separation. On Qwen 0.5B, IPW leads by 3.0 points (SE 2.5) and 0.0024 nats in uniform node-average loss (SE 0.0010).

At $\beta = 6$, full participation reduces the group loss gap relative to uncorrected training by 0.0283, 0.1458, 0.0940 and 0.0138 nats on GPT-2 medium, Pythia 410M, Qwen 0.5B and Qwen 7B. These reference improvements set the scale of normalized reduction, so the same absolute gain produces more reduction points when the reference improvement is small. At 12B, full participation instead has a 0.0239-nat larger gap than uncorrected training. We therefore compare absolute losses and flagged runs directly.

On independent small-transformer seeds, RUMOR's static gain over RUMOR, FIXED CAP again transfers loss between groups. Low-participation loss falls 0.070 nats while the other groups' mean rises 0.018. On both seed sets, the uniform node-average loss difference is smaller than its SE in magnitude.

Table 5: Uniform node-average loss (nats) and RUMOR low-group participation rate for the runs in Table 4. Bold marks the lowest correction-rule loss, including ties.

testbed and regime	Uncorrected	Full	IPW cap 20	RUMOR, FIXED CAP	RUMOR, SHORT WINDOW	RUMOR	rate
GPT-2 medium, $\beta = 2$	4.166	4.148	4.148	**4.141**	4.142	**4.141**	0.52
Pythia 410M, $\beta = 2$	3.304	3.291	3.291	**3.288**	**3.288**	3.289	0.66
Qwen 0.5B, $\beta = 2$	3.448	3.432	3.433	3.433	3.433	**3.432**	0.51
GPT-2 medium, $\beta = 6$	4.176	4.148	**4.163**	4.172	4.172	4.166	0.08
Pythia 410M, $\beta = 6$	3.346	3.291	3.331	3.334	3.334	**3.330**	0.47
Qwen 0.5B, $\beta = 6$	3.478	3.432	**3.470**	3.474	3.474	3.473	0.28
Qwen 7B, $\beta = 2$	2.496	2.488	2.488	**2.485**	2.486	**2.485**	0.65
Qwen 7B, $\beta = 6$	2.508	2.488	2.521	**2.501**	2.503	**2.501**	0.57
GPT-2 small, static	4.501	4.449	**4.448**	4.462	4.464	4.449	0.21
Qwen 1.5B, static	2.988	2.957	**2.957**	2.962	2.963	2.958	0.21
Qwen 7B, static	2.525	2.488	**2.489**	2.495	2.491	0.21
Mistral-Nemo 12B, $\beta = 2$	2.082	2.076	2.116	**2.075**	2.076	0.71
Mistral-Nemo 12B, $\beta = 6$	2.091	2.077	4.666	2.094	**2.090**	0.70
Mistral-Nemo 12B, $\beta = 6$, relay	2.089	2.077	2.099	**2.087**	2.089	0.68
Mistral-Nemo 12B, static	2.102	2.077	2.688	2.082	**2.076**	0.21

All flagged runs remain in these means.

Table 6: Small-transformer final losses in nats, twenty seeds per row. Bold marks the lowest displayed loss for each group or uniform average, including ties.

	IPW, cap 20			RUMOR, FIXED CAP			RUMOR		
Setting and seeds	low	others	mean	low	others	mean	low	others	mean
static, original	1.675	1.542	1.575	1.738	**1.512**	**1.568**	**1.660**	1.541	1.571
$\beta = 2$, original	1.572	1.516	1.530	**1.546**	1.482	1.498	**1.546**	**1.480**	**1.497**
$\beta = 4$, original	2.020	1.871	1.908	**1.567**	1.501	**1.518**	1.578	**1.499**	1.519
$\beta = 6$, original	2.254	2.053	2.103	**1.611**	**1.515**	**1.539**	1.639	1.521	1.551
static, held-out	1.648	1.540	1.567	1.713	**1.513**	1.563	**1.643**	1.532	**1.559**
$\beta = 2$, held-out	1.539	**1.499**	**1.509**	**1.538**	**1.499**	**1.509**	1.539	1.500	1.510
$\beta = 4$, held-out	2.318	1.982	2.066	**1.571**	**1.520**	**1.533**	1.577	1.521	1.535
$\beta = 6$, held-out	2.687	2.373	2.452	**1.634**	1.545	1.568	1.648	**1.539**	**1.566**

Table 7: Final group and uniform node-average losses in nats for pretrained methods at $\beta = 2$, ten seeds each. Bold marks the lowest correction-rule loss in each column, including ties at displayed precision.

	GPT-2 medium			Pythia 410M			Qwen 0.5B		
Method	Low	High	Mean	Low	High	Mean	Low	High	Mean
RUMOR	4.995	3.287	4.141	3.641	2.937	3.289	**4.289**	2.576	3.432
RUMOR, FIXED CAP	4.995	3.287	4.141	**3.639**	2.937	3.288	4.290	2.575	3.433
Uncorrected	5.034	3.298	4.166	3.686	2.921	3.304	4.326	2.571	3.448
Full participation	5.008	3.289	4.148	3.653	2.929	3.291	4.290	2.574	3.432
IPW, cap 20	5.007	3.288	4.148	3.651	2.930	3.291	4.291	2.574	3.433
TDW	5.005	3.287	4.146	3.652	2.927	3.289	4.294	2.572	3.433
DW	4.985	3.266	4.126	3.647	2.908	3.278	4.302	2.545	3.423
RUMOR, NO HISTORY	**4.982**	**3.263**	**4.122**	3.641	**2.896**	**3.268**	4.302	2.542	**3.422**
AH-IPW	5.007	3.287	4.147	3.650	2.929	3.290	4.293	2.573	3.433
FPW	5.004	3.287	4.146	3.647	2.931	3.289	4.294	2.572	3.433
RUMOR, SHORT WINDOW	4.998	3.287	4.142	**3.639**	2.937	3.288	4.292	2.575	3.433
Constant weight 1.96	4.994	3.267	4.131	3.665	2.900	3.282	4.315	**2.541**	3.428

J WEIGHT USE AND ADDITIONAL DIAGNOSTICS

Figure 14 summarizes RUMOR's weights at nodes holding the low-participation data. These are the Turkish and Swahili nodes in the original static assignment and the pretrained feedback settings. The swapped static assignment instead gives English and German nodes the lower participation probability.

Because the drift weight cannot exceed 3.33, a weight above this threshold confirms a larger loss-dependent cap. A larger cap allows stronger weighting, while the selected weight also depends on participation history and disagreement. Figure 6 shows how the cap changes during training. In Figure 14, such weights predominate among the logged static events but are uncommon in most feedback settings. On GPT-2 medium at $\beta = 6$, the low-participation group improves slowly despite infrequent training, keeping loss excess near zero and the cap large (Sec. 5). RUMOR's gap reduction lies between RUMOR, FIXED CAP and IPW with cap 20, without a flagged run over 1000 rounds.

J.1 EVALUATION SCOPE AND LIMITATIONS

Our main experiments use SGD and controlled participation on fixed random-geometric graphs, with sixteen nodes for the small transformer and eight for pretrained models. Appendix E reports additional adaptive-optimizer experiments. These tests leave open how the method behaves in volunteer systems with changing topology, bandwidth limits or strategic behavior. Loss-adaptive participation also depends on how losses compare across languages. Constant language-specific loss offsets can change participation probabilities even when the gradients are unchanged. The static assignment swap checks whether deficits follow the participation assignment. The running-minimum loss reference introduces another consideration. An unusually low past loss or a lasting data shift can keep current loss above that reference and hold the cap down. These cases were not evaluated.

Reusing models from previous gossip exchanges removes the separate exchange for fresh disagreement described in Appendix B.2. On ten GPT-2 medium seeds, this one-exchange variant has normalized final group loss gap reduction 150.6, compared with 164.1 for RUMOR at $\beta = 2$, a decrease of 13.5 points (SE 3.2). At $\beta = 6$, its reduction is 34.7, compared with 30.0 for RUMOR, an increase of 4.7 points (SE 2.8). Uniform node-average losses differ by less than 0.002 nats, while the second exchange contributes to the weaker-feedback gap gain. Across 120 small-transformer feedback runs per method on the original and independent seeds, RUMOR and RUMOR, FIXED CAP each have one flagged run, compared with 27 for IPW with cap 20. In the separate drift-removal comparison, RUMOR has one flag among 60 feedback runs and one residual-deficit flag among 20 static runs, while IPW with the same loss-dependent cap has none. Neither method is flagged among 40 pretrained runs per method. These counts describe the evaluated settings and training duration.

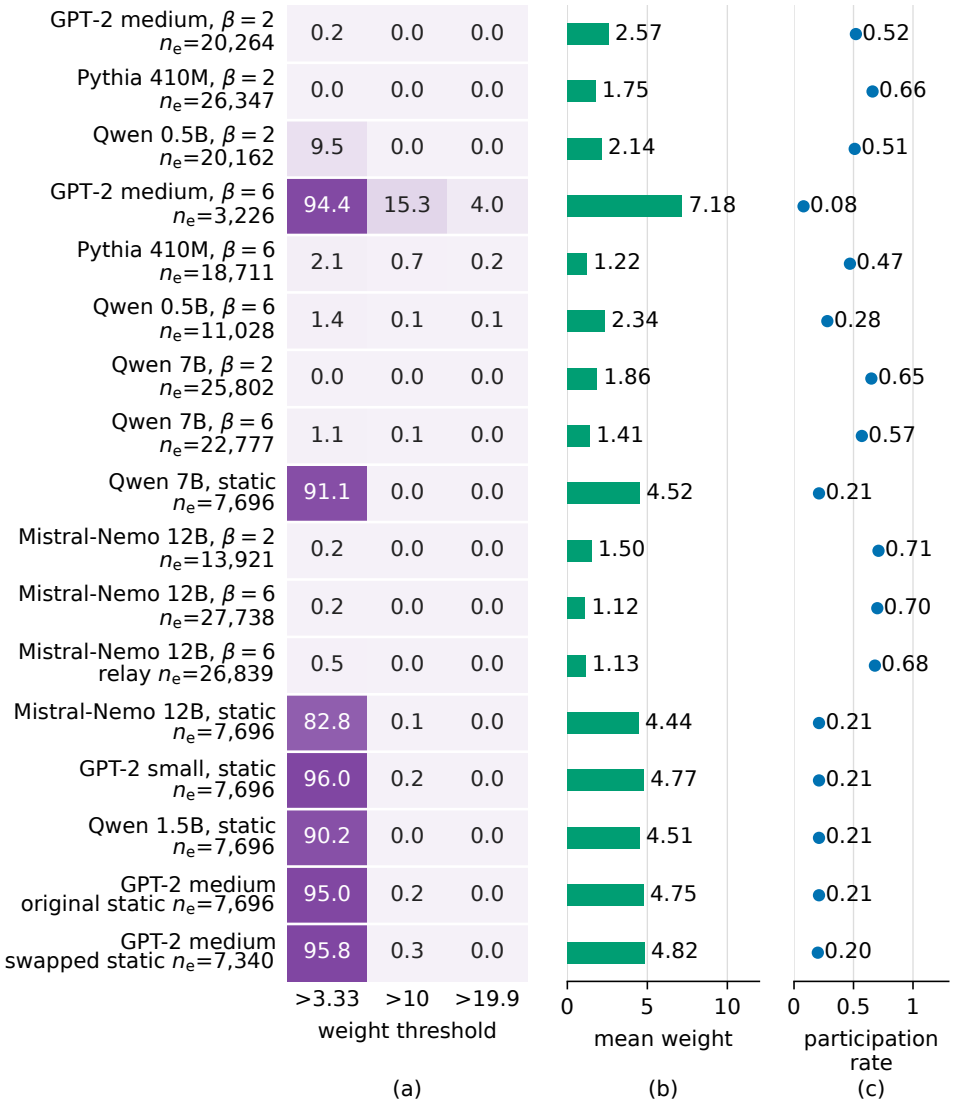


Figure 14: RUMOR weights on low-participation nodes. Panel (a) gives percentages above each threshold, (b) mean weight and (c) participation rate. The count n_e is the number of node-round observations in which the node and at least one neighbor participate.

J.1.1 What disagreement records at 12B

We record model disagreement whenever a node participates with at least one active neighbor. Across four 12B DW runs, 23,026 of 25,030 such observations occur at a node that also had an observation in the previous round, excluding each node's first observation. Thus disagreement often remains even with consecutive participation and model exchange. TDW estimates this baseline separately so that it can weight the additional disagreement associated with missed rounds.

To examine Proposition 2, we call an observation *near consensus* when the node's model is within 10^{-5} of its active-neighbor mean before the local update. A round is *fully near consensus* when every recorded node meets this threshold. This occurs when the preceding round's recorded nodes are all connected to one another and include every node recorded in the current round. Mixing has given those nodes the same model. This condition correctly predicts all 3996 evaluated consecutive-round pairs, with precision and recall 1.000.

From round 100 onward, the fraction of near-consensus observations is 0.145 to 0.162 in the three unflagged runs. In each run, two or three nodes exceed a near-consensus fraction of 0.2. These nodes

have 6.0 to 7.0 neighbors on average, compared with 4.0 to 4.4 for the other nodes. Seed 0's node with only one neighbor has no near-consensus observation in this period.

Over the full run, near consensus accounts for 15.3% of observations in unflagged seed 0 and 77.5% in later-flagged seed 2. The unflagged run has four fully near-consensus rounds, with at most two in a row. The flagged run has 758, including 74 in a row. During these stretches, disagreement provides little information about missed training because neighboring models are almost identical. The flagged DW run shows that prolonged model agreement can coexist with a large final loss deficit.

K Proofs and additional analysis

K.1 Proofs of the main-text propositions

Proof of Proposition 1. The claim concerns what can be learned from participation indicators alone. The indicator ξ^t equals one when the node participates and zero otherwise. Index the window as rounds 1 to H and let $\mathcal{G}_{t-1}$ denote the node's indicator history before round t, including rounds before the window. For either node, the hypothesis gives $\Pr(\xi^t = 1 \mid \mathcal{G}_{t-1}) = p$ after every history of positive probability. For the loss-adaptive node, this probability is $\mathbb{E}[\pi_j^t \mid \mathcal{G}_{t-1}]$: we average over network states consistent with its indicator history. Multiplying the successive conditional probabilities gives the probability of any binary sequence $(x_1, \ldots, x_H)$,

$$\Pr(\xi^1 = x_1, \ldots, \xi^H = x_H \mid \mathcal{G}_0) = \prod_{t=1}^{H} p^{x_t}(1-p)^{1-x_t}.$$

This expression is the same for both nodes and does not depend on the pre-window history $\mathcal{G}_0$. Thus any shared rule applied to these sequences, such as counting participations and choosing a cap from that count, also has the same output distribution. A sufficient condition is $c_0 - \beta(\ell_j^t - \ell_{\text{net}}^t) = \log(p/(1-p))$ throughout the window with probability one: the logistic rule then returns p in every round. Equal average participation rates alone are not sufficient; the hypothesis requires the same conditional probability after every indicator history.

Derivation of Equation 5. An inactive round contributes multiplier zero, and an active round contributes at most C. Therefore $w_i\xi_i \le C\xi_i$ for every outcome, even if the weight depends on participation. Taking expectations gives $\mathbb{E}[w_i\xi_i] \le C\mathbb{E}[\xi_i] = C\pi_i$. To match multiplier one from an unweighted step every round, it is necessary that $C\pi_i \ge 1$, or $\pi_i \ge 1/C$. This is a bound on the scalar multiplier, not on the gradient direction or loss improvement.

Proof of Proposition 2. Let $\bar{\theta}_{i,\text{post}}^t$ be the mean of node i's active neighbors after their local updates but before mixing. The degree condition says that node i has at least as many active neighbors as each of those neighbors. Hence each neighbor receives Metropolis weight $1/(1+\deg_i^t)$, leaving the same weight for node i itself. Subtracting the fixed neighbor mean gives

$$\theta_i^{t+1} - \bar{\theta}_{i,\text{post}}^t = \frac{\widetilde{\theta}_i^t - \bar{\theta}_{i,\text{post}}^t}{1+\deg_i^t}.$$

Taking Euclidean norms shows that the distance to this fixed mean shrinks by the stated factor. This does not imply that next-round disagreement shrinks: the neighbors also mix, so their mean changes. For example, let all four nodes on the path b–i–a–c participate, with edges only between consecutive nodes. In that order, post-update scalar models $(0, 0, 0, 9)$ mix to $(0, 0, 3, 6)$. If the same nodes participate next round, node i is still at zero but its neighbor mean is 1.5, rather than zero.

In a complete connected component with $r \ge 2$ active nodes, every node has $r-1$ neighbors and every mixing weight is $1/r$. All nodes therefore receive the same average of the post-update models. At the next pre-update observation, disagreement is zero for a node with an active neighbor if it and all its active neighbors belong to that previously mixed set. Their earlier participation frequencies can differ: averaging has erased the model differences, not the participation histories.

K.2 Additional analysis of update weights

Corollary 1 (Bounds on the RUMOR update weight)**.** *At every node and round, $h_i^t \le w_i^t \le C_i^t \le C_{\text{hi}}$. Let $\mathcal{F}_{t-1}$ denote the network history before round t, including stored loss and participation*

statistics. Given this history, node i participates with probability π_i^t, and

$$\mathbb{E}[w_i^t \xi_i^t \mid \mathcal{F}_{t-1}] \le C_i^t \pi_i^t.$$

Thus, even at the cap, the expected multiplier for this round is below one if $\pi_i^t < 1/C_i^t$. For finite loss excess, $C < C_i^t \le C_{\mathrm{hi}}$ places this probability threshold in $[1/C_{\mathrm{hi}}, 1/C)$.

Proof of Corollary 1. The history weight is at most C_i^t, while the drift weight is at most $C < C_i^t$. Taking their maximum when drift correction is enabled therefore preserves both $w_i^t \ge h_i^t$ and $w_i^t \le C_i^t$. These bounds also hold when the small transformer reuses an earlier disagreement observation. Algorithm 1 computes the cap from stored losses, before the current update. The cap is therefore fixed once $\mathcal{F}_{t-1}$ is given, although the weight can still depend on current active neighbors. Consequently,

$$\mathbb{E}[w_i^t \xi_i^t \mid \mathcal{F}_{t-1}] \le C_i^t \mathbb{E}[\xi_i^t \mid \mathcal{F}_{t-1}] = C_i^t \pi_i^t.$$

If $C_i^t \pi_i^t < 1$, even this upper bound is below the full-participation multiplier.

For example, at participation probability 0.2, matching multiplier one requires a cap of at least 5. With the constants in Table 1, Equation 3 gives

$$C_i^t \ge 5 \quad \Longleftrightarrow \quad D_i^t \le \delta \log\left(\frac{C_{\mathrm{hi}} - C}{5 - C}\right) \approx 0.18 \text{ nats}.$$

This is a necessary condition, not a guarantee of matching: the selected weight can be smaller than the cap.

Corollary 2 (When disagreement can increase the update weight)**.** *The weight in Equation 4 differs from h_i^t only if $\gamma < h_i^t < C$, and then $w_i^t \le C$. For* RUMOR*, this necessary condition is equivalent to*

$$1/C < \hat{\pi}_i^t < 1/\gamma.$$

The boundaries are approximately 0.30 and 0.67 for the constants in Table 1. Outside this interval RUMOR *uses h_i^t alone, and the same interval applies to* RUMOR, FIXED CAP*. Inside the interval, an increase is possible but still requires the drift weight to exceed the history weight and the drift correction to be enabled.*

Proof of Corollary 2. If $h_i^t \le \gamma$, the rule does not enable drift correction. If $h_i^t \ge C$, the drift weight cannot exceed the history weight, because $r_i^t \le C$. Only $\gamma < h_i^t < C$ can therefore allow an increase, and both weights are then at most C. Since $h_i^t = \min(1/\hat{\pi}_i^t, C_i^t)$ and $C_i^t \ge C$, a history weight below C must equal $1/\hat{\pi}_i^t$. Taking reciprocals of $\gamma < 1/\hat{\pi}_i^t < C$ gives the stated interval. For RUMOR, FIXED CAP, $h_i^t = \min(1/\max(\hat{\pi}_i^t, 0.3), C)$. Because $0.3 \le 1/C$, the probability floor is inactive inside the interval, giving the same result.

The next result asks whether a finite participation history yields an expected multiplier of one before any cap is applied.

Proposition 3 (Expected update weight of windowed IPW under static participation)**.** *Let node i participate independently in each round with fixed probability $0 < \pi_i < 1$. Let $m = m_i^t$ be the number of preceding rounds in its window and c its participation count in those rounds, so $\hat{\pi}_i^t = (c+1)/(m+2)$. Then $\mathbb{E}[\hat{\pi}_i^t] = (m\pi_i + 1)/(m+2)$, and uncapped windowed IPW satisfies*

$$\mathbb{E}\left[\frac{\xi_i^t}{\hat{\pi}_i^t}\right] = \frac{m+2}{m+1}\Big(1 - (1-\pi_i)^{m+1}\Big).$$

The multiplier inside this expectation is zero on inactive rounds. Its expected value lies between $1 - (1-\pi_i)^{m+1}$ and $1 + 1/(m+1)$. A probability floor or a cap can only lower this expected value.

Proof of Proposition 3. The preceding count c is Binomial(m, π_i), with mean $m\pi_i$, giving the stated mean estimate. Because the window excludes the current round and participation is independent across rounds, c is independent of ξ_i^t. To average the reciprocal count, use $1/(c+1) = \int_0^1 x^c \, dx$ and the binomial expansion:

$$\mathbb{E}\left[\frac{1}{c+1}\right] = \int_0^1 (1 - \pi_i + \pi_i x)^m \, dx = \frac{1 - (1-\pi_i)^{m+1}}{(m+1)\pi_i}.$$

Independence then gives $\mathbb{E}[\xi_i^t/\hat{\pi}_i^t] = \pi_i(m+2)\mathbb{E}[1/(c+1)]$, which is the claimed formula. The factor $(m+2)/(m+1)$ is at least one, and the bracketed term is at most one, giving the lower and upper bounds, respectively. A probability floor or cap reduces or preserves each realized weight, so it cannot increase the expectation.

With $\pi_i = 0.2$, the uncapped expected multipliers are 1.010 for a full window of $m = 100$ rounds and 1.005 for $m = 200$. Thus finite-window estimation alone does not cause an expected-multiplier shortfall in these cases. For a new node with no preceding observations ($m = 0$), the estimate is $1/2$ and the expected multiplier is $2\pi_i$; it is only 0.4 when $\pi_i = 0.2$. The formula therefore also explains why a shortfall can occur during initialization, even without a cap.

L Additional results

L.1 Static participation

As shown in Figure 15, RUMOR's loss adaptation removes much of RUMOR, FIXED CAP's static deficit on GPT-2 small and Qwen 1.5B. IPW with cap 20 has slightly lower final low-participation-group loss, by 0.0015 (paired SE 0.0006) and 0.0022 (paired SE 0.0004) nats, respectively.

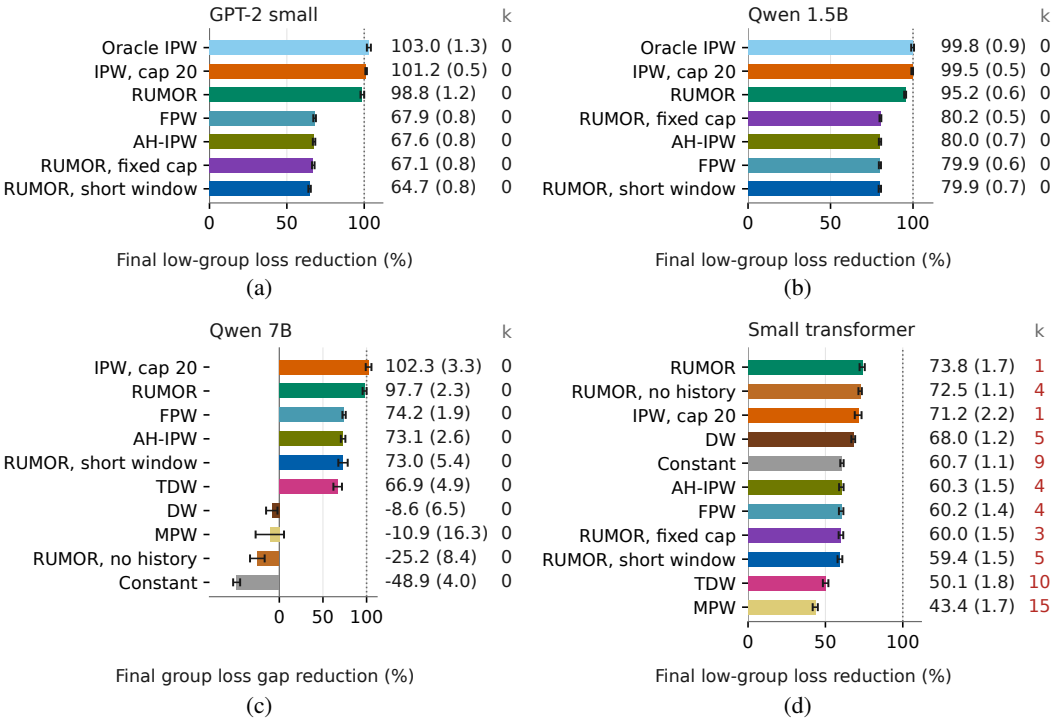


Figure 15: Static participation on (a) GPT-2 small, (b) Qwen 1.5B, (c) Qwen 7B and (d) the small transformer. Panels (a), (b) and (d) show normalized final low-participation-group loss reduction, while (c) shows normalized final group loss gap reduction. Labels give estimate (leave-one-seed-out SE), with ten matched seeds for pretrained models and twenty original seeds for the small transformer. Larger values indicate greater reduction in the named metric. Zero matches uncorrected training and the dotted line marks full participation. The k column counts flagged runs. All runs remain in the means. Figure 3 shows the 12B absolute losses. IPW caps are 20 unless labeled, fixed-cap methods use 3.33, and RUMOR uses a loss-dependent cap from 3.33 to 20.

L.2 Interval-average weighting

Interval-average weighting adapts FedAU's interval estimator (Wang & Ji, 2024) to gossip by averaging completed interval lengths. An interval closes when the node participates or reaches the cutoff K, even if it remains inactive, and the next interval starts from zero. The weight averages

intervals completed in earlier rounds and is one before the first interval closes, so K is also the maximum weight. Table 8 compares cutoffs $K = 3$ and $K = 50$ with RUMOR and windowed IPW using the matched runs from the main comparisons, with ten seeds per pretrained setting and twenty per small-transformer setting. RUMOR exceeds interval-average weighting with cutoff 50 on GPT-2 medium and Qwen 0.5B final-gap reduction at $\beta = 2$. That baseline leads on GPT-2 medium at $\beta = 6$ and small-transformer integrated-gap reduction at $\beta = 2$ and 6, but has flagged runs at 7B and 12B where RUMOR has none.

The feedback comparisons measure final group loss gap on pretrained models and integrated group loss gap on the small transformer, while static comparisons measure final low-participation-group loss. In the matched pretrained runs reported in Table 8, RUMOR has no flagged run. Interval-average weighting with cutoff 50 has one on Qwen 0.5B, one on Qwen 7B and three at 12B under $\beta = 6$. The Qwen 0.5B loss increase occurs between rounds 700 and 800, ends 1.75 nats above full participation and reaches weight 50. The 12B increases are 8.1, 11.0 and 12.6 nats. Relative to cutoff 50, RUMOR's uniform node-average loss is lower by 0.0078 (SE 0.0004) and 0.0011 (SE 0.0003) nats on GPT-2 medium and Qwen 0.5B at $\beta = 2$, higher by 0.0032 (SE 0.0012) on GPT-2 medium at $\beta = 6$, and lower by 3.01 (SE 1.58) at 12B.

Cutoff 3 avoids these flags on Qwen 7B and 12B. On Qwen 7B, RUMOR leads by 23.5 reduction points (paired SE 15.7) and has 0.0021 nats lower uniform loss (paired SE 0.0013).

At 12B, cutoff 3 and RUMOR have low-participation-group losses of 2.5131 and 2.5115 nats, and uniform losses of 2.0892 and 2.0895. Thus RUMOR's low-group loss is 0.0016 nats lower but its uniform loss is 0.0003 nats higher (paired SE 0.0015). The negative reference denominator precludes normalized reduction.

On the small transformer, cutoff 50 has no flag across sixty feedback runs, versus eleven for IPW with cap 20. Its uniform losses are lower than RUMOR's by 0.011 (SE 0.004), 0.024 (SE 0.007) and 0.037 (SE 0.011) nats at $\beta = 2$, 4 and 6, respectively.

Table 8: Interval-average weighting with cutoff K, the maximum interval length and weight. Reduction is 0 for uncorrected training and 100 for full participation. Negative reductions are worse than uncorrected. The last columns give $K = 50$ minus the named method in reduction points. Positive differences favor $K = 50$ and negative differences favor the named method. Parentheses give one SE, paired for differences. The count k gives flagged runs, and n.d. marks undefined reduction. The 0.0138-nat reference gap improvement on Qwen 7B makes its normalized reductions sensitive to variation across seeds.

	cutoff $K = 50$		cutoff $K = 3$		$K = 50$ minus method (points)	
Setting	Reduction (SE)	k	Reduction (SE)	k	RUMOR	IPW, cap 20
GPT-2 medium, $\beta = 2$	97.2 (3.7)	0	63.8 (3.0)	0	−66.9 (3.1)	−1.4 (2.5)
Qwen 0.5B, $\beta = 2$	98.6 (2.6)	0	64.1 (2.0)	0	−8.6 (2.8)	+0.0 (2.4)
GPT-2 medium, $\beta = 6$	40.5 (3.4)	0	4.7 (1.0)	0	**+10.4 (3.9)**	**+1.5 (1.2)**
Qwen 0.5B, $\beta = 6$	−27.2 (53.9)	1	8.1 (2.5)	0	−46.1 (53.0)	−49.1 (52.9)
Qwen 7B, $\beta = 6$	−1140.3 (1250.1)	1	22.5 (13.6)	0	−1186.3 (1257.9)	−1169.7 (1254.3)
Mistral-Nemo 12B, $\beta = 6$	n.d.	3	n.d.	0	n.d.	n.d.
Small transformer, $\beta = 2$	86.2 (3.5)	0	81.0 (3.2)	0	**+13.4 (6.2)**	**+4.8 (1.8)**
Small transformer, $\beta = 4$	89.7 (3.9)	0	83.9 (3.3)	0	−3.2 (2.9)	**+34.9 (11.7)**
Small transformer, $\beta = 6$	96.1 (1.3)	0	81.7 (5.5)	0	**+1.8 (1.6)**	**+13.2 (4.2)**
Small transformer, static	72.9 (1.8)	1	51.9 (1.5)	9	−0.8 (0.6)	**+1.7 (1.1)**

L.3 Reweighted mixing

Table 9 tests adding reweighted mixing to RUMOR. The two corrections act at different steps. RUMOR weights a node's local training update, while reweighted mixing gives more weight to models received from infrequently encountered neighbors. Windowed IPW also weights local updates, so it differs from reweighted mixing in what is weighted and whose participation is estimated. Appendix B.3 specifies how each receiving node estimates its neighbors' participation and normalizes the mixing weights.

Table 9: Reweighted mixing using the runs and metrics in Table 8. Entries give normalized reduction (leave-one-seed-out SE), flagged count k and paired changes (SE). Reduction gains are in points. Uniform-loss changes are in 10^{-3} nats, with decreases bold. The last column compares mixing alone with no correction.

	reweighted mixing alone		RUMOR with reweighted mixing		Effect of adding mixing to RUMOR		Mixing alone vs. uncorrected
Setting	Reduction (SE)	k	Reduction (SE)	k	Reduction gain	Loss change	Loss change
GPT-2 medium, $\beta = 2$	50.5 (1.1)	0	278.4 (8.1)	0	+114.3 (5.2)	**−3.9 (0.4)**	+3.0 (0.3)
Pythia 410M, $\beta = 2$	−1.5 (3.0)	0	150.4 (9.9)	0	+2.8 (3.2)	+0.9 (0.9)	**−0.3 (0.7)**
Qwen 0.5B, $\beta = 2$	101.9 (2.8)	0	216.8 (4.6)	0	+109.6 (3.3)	+4.1 (0.6)	+1.7 (0.2)
GPT-2 medium, $\beta = 6$	71.4 (2.0)	0	88.0 (6.6)	0	+57.9 (3.9)	+6.2 (1.3)	+14.0 (0.4)
Pythia 410M, $\beta = 6$	0.7 (2.1)	0	31.4 (7.8)	0	−0.9 (8.9)	+1.1 (5.8)	**−0.9 (0.9)**
Qwen 0.5B, $\beta = 6$	22.9 (1.8)	0	32.9 (2.8)	0	+14.0 (1.9)	+6.4 (0.9)	+4.2 (0.4)
small transformer, $\beta = 2$	51.2 (2.1)	0	114.0 (5.5)	0	+41.1 (6.1)	**−8.6 (2.4)**	**−1.3 (0.4)**
small transformer, $\beta = 4$	78.3 (1.7)	0	109.4 (5.0)	0	+16.4 (3.0)	**−4.9 (5.3)**	**−4.3 (1.0)**
small transformer, $\beta = 6$	92.8 (1.9)	0	100.4 (1.8)	1	+6.1 (1.7)	+19.7 (14.9)	**−93.5 (19.8)**
small transformer, static	22.8 (1.4)	12	87.7 (1.2)	0	+14.0 (0.8)	+12.2 (7.3)	+23.0 (3.8)

Under feedback, adding reweighted mixing improves gap reduction on GPT-2 medium, Qwen 0.5B and the small transformer, as reported in Table 9. Uniform node-average loss also falls on GPT-2 medium at $\beta = 2$ and the small transformer at $\beta = 2$ and 4. However, on Qwen 0.5B and GPT-2 medium at $\beta = 6$, higher high-participation-group loss drives much of the gap gain, while uniform loss rises by 0.004 to 0.006 nats. One combined small-transformer run is flagged at $\beta = 6$. Used alone, reweighted mixing raises uniform node-average loss by 0.002 to 0.014 nats relative to uncorrected training in the affected pretrained settings and leaves twelve static residual-deficit flags on the small transformer.

M Derivation of the drift-based comparison weights

Drift weighting (DW) DW compares a node's distance from its active-neighbor mean with the rate at which that mean changes. The small-transformer and pretrained implementations update an exponential moving average u_i^t of $\|\bar{\theta}_i^t - \bar{\theta}_i^{t'}\|_2/(t - t')$ at rate α_u, where t' is the preceding participation round with an active neighbor. With the active-neighbor mean defined in Sec. 4, the weight is

$$w_i^t = \min(1 + d_i^t/u_i^t,\, C) \tag{6}$$

The ratio compares disagreement with typical neighborhood movement per round. The weight equals one at zero disagreement and has maximum $C \geq 1$.

Topological drift weighting (TDW) TDW accounts for disagreement that remains even when a node participates in consecutive rounds. It uses the stationary saturation model

$$\mathbb{E}[(d_i^t)^2 \mid k_{i,\mathrm{part}}^t = k] = a_i + S_i(1 - q_i^k), \qquad q_i = e^{-1/T_i},$$

where $k_{i,\mathrm{part}}^t$ counts rounds since the node's last participation. The topology offset a_i represents persistent disagreement associated with the mixing pattern. The scale $S_i > 0$ sets how much additional squared disagreement can accumulate, and the time constant $T_i > 0$ controls how many rounds this accumulation takes. An EMA B_i^t of squared disagreement at $k_{i,\mathrm{part}}^t = 1$ estimates the no-missed-round baseline $B_i = a_i + S_i(1 - q_i)$, removing both the topology offset and one-round drift. Appendix J.1.1 examines the relation between disagreement and mixing in the 12B diagnostics.

TDW smooths $(d_i^t)^2 - B_i^t$ into $\bar{e}_i^t$ and divides by the estimated scale to form $z_i^t = \bar{e}_i^t/S_i^t$, clipped to $[-0.5, 0.95]$. It estimates S_i^t from the neighbor mean's spread and T_i^t from this spread relative to its one-round change, as specified in Table 2. Under the stationary model with independent Bernoulli participation at fixed probability $\pi_i > 0$ and an active neighbor at every participation event, the gap K has probability $\Pr(K = k) = \pi_i(1 - \pi_i)^{k-1}$ for $k \geq 1$. At population values, write $q = q_i$,

$\pi = \pi_i$, $z = \mathbb{E}[(d_i)^2 - B_i]/S_i$ and $\rho = (1-\pi)q$. Summing over this gap distribution gives

$$\mathbb{E}[q^K] = \frac{\pi q}{1-(1-\pi)q}, \qquad z = q - \mathbb{E}[q^K] = \frac{\rho(1-q)}{1-\rho}, \qquad \frac{1}{\pi} - 1 = \frac{z}{(1-q)(q-z)}.$$

The last identity converts normalized excess disagreement to the excess weight $1/\pi - 1$.

To reduce sensitivity near $z_i = q_i$, TDW uses $q_i^t = \exp(-1/T_i^t)$ and replaces $q_i - z_i$ by $1 - z_i$,

$$\kappa_i^t = \frac{z_i^t}{(1-z_i^t)(1-q_i^t)}, \qquad w_i^t = \min\big(1 + \max(\kappa_i^t, 0),\, C\big). \tag{7}$$

For $0 < z_i < q_i$, the replacement enlarges the denominator and lowers the uncapped population weight, with equality at $z_i = 0$. The cap can make the resulting weights equal, while the floor keeps online weights at least one. Implementation uses the active-neighbor observation gap $k_{i,\mathrm{obs}}^t$. Without an active neighbor, it uses weight one and leaves its statistics unchanged.

Filtered participation weighting (FPW) Under the same model and participation assumptions, $\pi_{\mathrm{drift}} = 1 - z/[q(z+1-q)]$. FPW combines $\log(\pi_{\mathrm{drift}}/(1-\pi_{\mathrm{drift}}))$ with participation indicators. Its observation variance accounts for excess-disagreement variability and logit sensitivity, reducing the weight of noisier observations. Probability and logit-residual checks follow Appendix B.3.