PREPARE IN PROPORTION: DELETION-HAZARD-AWARE TRAINING FOR UNLEARNING-READY FEDERATED LANGUAGE MODELS

Anonymous authors
Paper under double-blind review

ABSTRACT

Federated unlearning (FU) removes a withdrawing client's contribution from a model trained across clients, and preparing at training time lowers the cost of later requests. Preparation faces a distinctive obstacle: it is bought with utility, and one budget must be divided across clients whose withdrawal likelihood differs and is predictable from signals the deployer holds. Current methods isolate every client's contribution equally, and they spend its utility cost on clients who will never leave. To address these, we first formulate preparation as a budgeted allocation over deletion hazards, whose uniform-preparation excess has a closed form in the hazard variance. We then design **PREMIUM**, a framework that prepares each client in proportion to its hazard of deletion. Within this framework, Hazard Estimation fits a discrete-time survival model on operating signals and returns a deletion probability per client. Budgeted Readiness Allocation then solves the budgeted program by water-filling with one dual price. In addition, Readiness-Proportional Training routes each client's gradient between a shared and a private adapter in the assigned share, so a request detaches the private part and removes the shared part in the priced steps. Across three hazard profiles on Reddit and six published baselines at a common utility budget, **PREMIUM** needs fewer deletion steps than every baseline and less than half the steps of uniform preparation when hazards are dispersed, with comparable forgetting and membership inference. For example, with fitted hazards, **PREMIUM** serves a request in 110 steps, against 243 under uniform preparation and 272 for MemSinks, the strongest published baseline.

1 INTRODUCTION

A federation that fine-tunes a language model will receive deletion requests after the model is released, because some of the clients that contributed data will withdraw and ask that their contribution be unlearned (Ginart et al., 2019). The answer is machine unlearning: a procedure edits the released weights until the withdrawn data no longer shapes what the model produces, without retraining from scratch (Cao & Yang, 2015; Guo et al., 2020; Zhang et al., 2024). Serving such a request costs unlearning steps, and the count grows with how deeply the client's data entered the shared weights, so recent work pays part of the cost in advance: each contributor's learning is kept in a component detachable at request time, at the price of utility the others no longer receive (Bourtoule et al., 2021; Gururangan et al., 2022; Ghosal et al., 2025). Uniform isolation gives every contributor the same share; request-aware designs also exploit deletion priors. We review related work in Appendix A.

As illustrated in Figure 1, who will withdraw is neither uniform nor unknown. A client's participation history, jurisdiction and content type predict whether it files a request within a compliance period, exactly as those signals predict churn in any service the deployer operates (Ascarza, 2018), and the predicted probability varies across clients by orders of magnitude. Section 2 names this probability the hazard of deletion and the isolated share the readiness of a client. Uniform preparation prepares a client that will stay as much as one that will leave, so most of the utility it spends goes where no request will arrive, and the records that would redirect it are already the deployer's. This raises the first question: ***I) What does a deployer already know about which clients will ask for deletion, and what does an allocation of preparation need from that knowledge?***

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.

Once the hazards are known, preparation becomes a budget to divide. The deployer declares how much utility it will forgo, and every unit spent on one client's readiness is a unit denied another; what it buys down is the number of unlearning steps the period's requests will take, weighted by their likelihood. Prior designs fix the division in advance, isolating every contributor equally (Bourtoule et al., 2021; Ghosal et al., 2025) or placing one entirely in or out of the shared weights (Min et al., 2024; Sim et al., 2024), each right for one distribution of hazards. The division that minimizes expected cost under one budget, and the distance of the fixed ones from it, is what settles whether estimating hazards is worth doing. This raises the second question: ***II) Given the hazards and one budget, how much readiness should each client receive, and what does uniform preparation cost in the expected deletion work?***

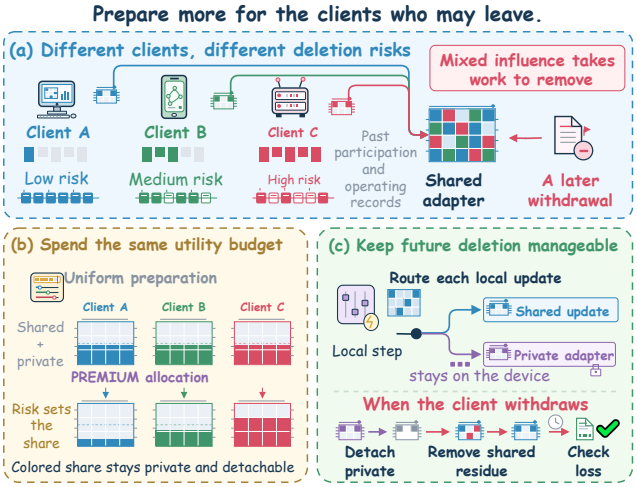


Figure 1: **Problem Illustration.** Clients differ in withdrawal risk, and every contribution mixed into the shared adapter takes work to remove. **I)** Different deletion risks: participation and operating records predict which clients will leave. **II)** One utility budget: uniform preparation spends it evenly, while a risk-set share stays private and detachable. **III)** Manageable deletion: each local update is routed, so a withdrawal detaches the private part and removes the shared residue.

An allocation is a promise about the cost of a request, and the training loop has to make it true: route a different share into the shared weights than the allocation priced, and the steps a request takes bear no relation to the steps it reasoned about. Mechanisms that keep one part of a client's adapter private and aggregate the other are in use for personalization (Guo et al., 2025; Yang et al., 2024), and what turns them into preparation is an identity between the share they route and a quantity a cost model prices, leaving the private part detached at no step and the shared part finished in the steps priced in advance. This raises the third question: ***III) How does training realize the assigned readiness exactly, so that a request costs what the allocation priced, and how is that realization audited across later hazard refitting periods?***

We address these challenges with **`PREMIUM`**, which allocates preparation according to each client's withdrawal risk under one shared utility budget. **Hazard Estimation (HE)** supplies the probabilities and their ordering; the budget determines which information the allocation needs. **Budgeted Readiness Allocation (BRA)** converts those predictions into private shares through one convex program, with a common price balancing preparation across clients. **Readiness-Proportional Training (RPT)** routes the corresponding learning and records its shared exposure, so the declared cost model prices the shared residue and the training ledger records what a later request must remove. Our main contributions are:

❶ ***Budgeted Preparation.*** We identify and formalize a budgeted preparation problem for federated unlearning: each client's learning is divided between shared parameters that require later removal and private parameters that can be detached, while one utility budget constrains the federation's total preparation. The optimal readiness follows a water-filling rule with one dual price, and the unsaturated quadratic case gives the excess cost of uniform preparation in closed form. The analysis separates probability information from ordering information: convex utility budgets depend on hazard values, while capacity budgets depend on ranks. Prediction-error bounds then connect inaccurate hazards to the extra deletion cost paid by the resulting allocation.

❷ ***Hazard-Guided Training Protocol.*** We introduce **`PREMIUM`**, realizing the allocation through three coordinated mechanisms: HE estimates future withdrawal probabilities from censored operating histories and updates them for each declared prediction period, distinguishing deletion events from temporary absence; BRA solves a common utility-constrained allocation and assigns each client a private share according to its hazard and the shared budget price; and RPT routes local learning between private and shared adapters, records the shared exposure and detaches the private component when a request arrives. We back the protocol with an exact SGD exposure identity, its

momentum-weighted counterpart and a stopping-time bound under the declared descent condition. A persistent exposure ledger preserves earlier contributions when hazards are refitted and future readiness changes.

❸ ***Evaluation Protocol.*** We organize evaluation on Reddit, Stack Overflow and GitHub around expected deletion work at a common utility budget, comparing preparation rules with one common request-time removal procedure and matched leave-group-out references. Hazard-spread and budget sweeps examine where selective preparation changes cost; component ablations isolate estimation, allocation and routing. Independent audits combine retained utility, extraction-based forgetting, membership inference and exposure accounting across training seeds and paired request sets. This design tests whether predicted readiness becomes detachable learning and whether the remaining shared contribution costs what the allocation prices.

2 PRELIMINARIES

Federated fine-tuning. We consider a server and a set $\mathcal{C}$ of clients, where client c holds a private dataset $\mathcal{D}_c$ of n_c samples and n denotes the total sample count. A frozen base model W_0 is adapted with low-rank modules, and training proceeds in rounds. In round t the server broadcasts the shared adapter $w[t]$, client c computes an update $\Delta_c[t]$ on $\mathcal{D}_c$, and the server aggregates the uploads,

$$w[t+1] \;=\; w[t] \;+\; \sum_{c\in\mathcal{C}} \frac{n_c}{n}\,\Delta_c[t], \qquad \Delta_c[t] \;=\; -\,\alpha \sum_{k=1}^{K} \nabla_w \mathcal{L}_c\big(w_c[t,k]\big), \tag{1}$$

where $\mathcal{L}_c$ is the token-level negative log-likelihood of $\mathcal{D}_c$ under W_0 carrying the adapter, α the local learning rate, K the local steps a round runs, and $w_c[t,k]$ the client's iterate after k of them; after T rounds the released model is W_0 with $w[T]$. The aggregation weight n_c/n is the whole channel through which client c enters the model, and is the same for every sample it contributes.

Deletion requests. After training, a client may withdraw and ask that its data be removed from the released model, within a compliance period the deployer fixes in advance. Withdrawal is distinct from dropping out of a round: a client that misses rounds contributes nothing while absent, whereas a client that withdraws asks that what it already contributed be undone. Write h_c for the probability that client c files such a request within the period. The deployer observes signals that predict this probability, such as the client's participation history, its jurisdiction, and the type of content it holds, and Section 3 fits h_c from them; what matters for the setting is that h_c varies across clients by orders of magnitude and is known, up to estimation error, before the model is trained.

Unlearning problem. Write $\mathcal{L}_c$ for the loss of Eq. (1) evaluated at the released model, and let the deployer fix a threshold γ at the value a reference model that never saw $\mathcal{D}_c$ attains, so the model answers as forgotten on client c once $\mathcal{L}_c$ stands above γ. Serving a request means driving the model past that threshold while preserving performance on the remaining clients, and its cost is the number of unlearning steps the deployer spends. Two ways of paying that cost exist. The deployer may pay at request time, by ascending $\mathcal{L}_c$ past the threshold, and the number of steps grows with how deeply the client's data entered the shared adapter. Or the deployer may pay at training time, by keeping part of each client's contribution in a component detachable at no step, which is what prior preparation methods do by assigning every client a component of the same size (Bourtoule et al., 2021; Xiong et al., 2023; Wen et al., 2026).

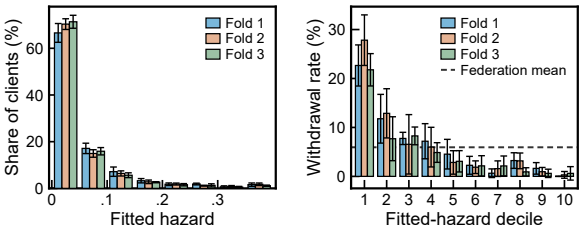
Preparation. The object of this paper is the quantity those methods fix. Write r_c in $[0,1]$ for the readiness of client c, the share of its learning routed into a private adapter the client keeps and the server never aggregates, so the remaining share $1-r_c$ enters the shared adapter through Eq. (1). A client at readiness one is detached in full and costs no unlearning step; a client at readiness zero is served entirely at request time. Uniform preparation assigns every client one readiness. Preparation trades utility for readiness, since learning routed into a private adapter is learning the others do not receive, and the deployer declares a budget η on the utility it is willing to forgo across the federation.

Desiderata. We require of a preparation rule three properties. (i) Hazard enters the rule: the readiness assigned to a client is a function of its probability of withdrawal, so clients more likely to leave are prepared more. (ii) The budget binds once: the utility the rule forgoes across all clients is at most η, and the rule is optimal within that budget rather than uniform. (iii) Preparation is realized in training: the readiness the rule assigns is the share the training dynamics actually route, so that the cost the rule

prices is the cost the deployer pays when a request arrives. A rule with none of the three is uniform preparation, which fixes one readiness for every client without reference to who will leave.

Observation 1

The probability that a client files a deletion request within a compliance period varies across a federation's clients, and the records a deployer already keeps order the clients by it.



(a) Hazard spread. (b) Ranking quality.

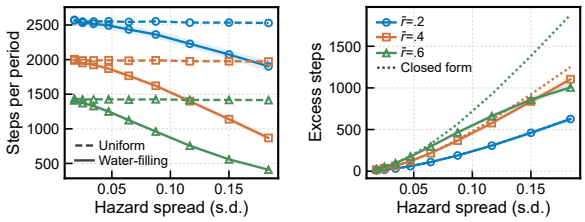
Figure 2: Fitted withdrawal hazards across a federation and the withdrawals they predict on held-out periods.

We fit a discrete-time survival model on the withdrawal events of earlier compliance periods, from the participation, jurisdiction and content signals the deployer keeps per client, and read it on three later periods held out of the fit. Two quantities are read per client: the probability the model assigns to a request arriving within the period, and the rank the model gives that client among all clients. Figure 2a plots the distribution of those probabilities across the federation, with one bar per interval of hazard. Figure 2b plots the withdrawal rate recorded on the held-out period within each decile of the fitted order, so the two readings answer whether the signals separate the clients at all and whether they separate them in the right order.

Observation 2

Uniform preparation assigns one readiness to clients that differ in their probability of withdrawal, so at a fixed budget its expected cost stands above a hazard-proportional allocation by a margin that grows with the dispersion of the probabilities and vanishes only when they coincide.

We fix the budget η at three levels, vary the spread of the withdrawal probabilities across the federation at a fixed mean, and record the number of unlearning steps the period's requests take under two rules: one readiness for every client, and a readiness that follows each client's probability within the same budget. Figure 3a plots both counts against the spread, and Figure 3b plots their difference on the same axis beside its closed form, which is the quantity a deployer weighs against the work of estimating the probabilities at all.



(a) Expected cost. (b) Excess cost.

Figure 3: Expected and excess cost of uniform preparation at three budgets as the hazard spread widens.

3 Methodology

3.1 Framework Overview

Motivated by the preparation problem of Section 2, **PREMIUM** allocates preparation for future deletion in proportion to the probability that each client will ask for it, in three stages (Figure 4): **I)** Hazard Estimation (HE, Section 3.2) fits a survival model on signals the deployer already holds and returns a deletion probability for each client together with the ordering of the clients by that probability; **II)** Budgeted Readiness Allocation (BRA, Section 3.3) solves one program over the readiness vector under the deployer's utility budget, whose solution is a water-filling function (Boyd & Vandenberghe, 2004) of the hazard with a single dual price; **III)** Readiness-Proportional Training (RPT, Section 3.4) realizes the allocation in the training loop by routing each client's gradient between a shared adapter and a private one in the assigned proportion, so that a request is served by detaching the private adapter and finishing the shared part in the number of steps the allocation priced. Algorithm 1 in Appendix B summarizes the procedure, and the proofs are in Appendix C.

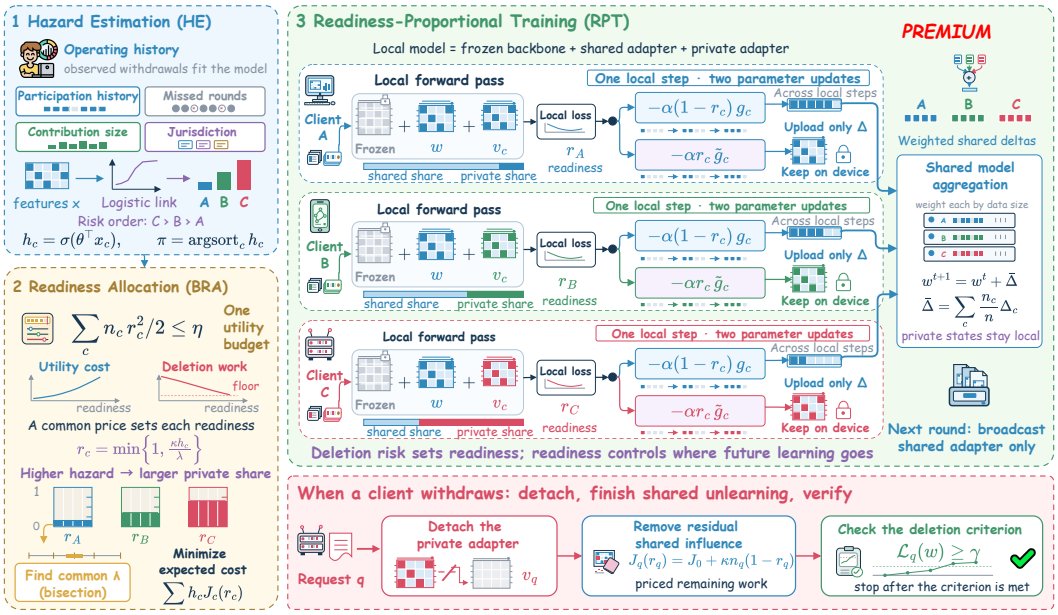


Figure 4: **Overview of PREMIUM.** HE fits a logistic hazard on participation history, missed rounds, contribution size and jurisdiction, and ranks clients. BRA sets each readiness from one price under a single utility budget, so higher hazard buys a larger private share. RPT splits every local step into an uploaded shared update and a private update kept on the device; a withdrawal detaches the private adapter, removes the priced residue and checks the criterion.

3.2 Hazard Estimation (HE)

Motivation. The allocation of the next stage consumes a probability per client, and uniform preparation is what one gets by declining to estimate it. The signals that predict withdrawal are ones the deployer already holds to operate the federation, and churn prediction is a task operators already solve, so HE asks for nothing new.

The hazard model. Write x_c for the feature vector of client c, collecting its participation history, rate of missed rounds, contribution size, jurisdiction and content type. HE fits a discrete-time survival model (Cox, 1972) over the compliance period,

$$h_c \;=\; \frac{1}{1+\exp\!\big(-\theta^\top x_c\big)}, \qquad \pi \;=\; \operatorname*{arg\,sort}_{c\in\mathcal{C}}\; h_c, \tag{2}$$

where the logistic link turns the score $\theta^\top x_c$ into a probability, θ is fitted by maximum likelihood on observed withdrawal events, and π orders the clients from most to least likely to withdraw. A hazard near one half says withdrawal is as likely as not within the period; one near zero says the data can be treated as permanent. The rate of missed rounds enters x_c as a feature and nowhere else: absence contributes nothing while it lasts, whereas withdrawal asks that a contribution be undone, and only the second is the label the model fits.

What the allocation consumes. The next stage reads Eq. (2) in two ways, and the appendix bounds each. Under a capacity budget the allocation depends on the hazards only through the ordering π, so a correct ranking loses nothing to error in the values, and a ranking that departs by k adjacent swaps loses at most k times the largest true-hazard gap among the swapped pairs, scaled by the cost of one unit of shared exposure; a two-client construction attains the single-swap loss. Under a strictly convex budget the allocation is monotone in the hazard and the regret is bounded by the weighted absolute error, so an estimator close in value is close in cost. Order suffices under a capacity budget, while a quadratic budget also uses calibration. On Reddit-2024 the fitted logistic hazard reaches an AUC of 0.71 and a Kendall correlation of 0.47, and the allocation it feeds stays 9 steps above the program value, against 142 for a constant hazard (Table 5).

3.3 BUDGETED READINESS ALLOCATION (BRA)

Motivation. Preparation is bought with utility, and the deployer has one budget for all clients. Spending it uniformly prepares the clients who will never leave as much as the ones who will; this stage divides it once the hazards are known.

The program. Serving a request for client c detaches its private adapter at no step, then ascends $\mathcal{L}_c$ on the shared adapter until the threshold γ is crossed, in a number of steps proportional to the shared exposure $n_c(1 - r_c)$. Write κ for the number of unlearning steps one unit of shared exposure costs, J_0 for the fixed cost of a detachment, and ϕ for a convex increasing function with $\phi(0)$ zero that prices the utility forgone by routing a share of one client's learning away from the federation. The expected cost of a request and the utility forgone by preparation are

$$J_c(r_c) \;=\; J_0 \;+\; \kappa\, n_c\,(1 - r_c), \qquad U_c(r_c) \;=\; n_c\,\phi(r_c), \tag{3}$$

and BRA solves the stochastic program whose outer expectation runs over the deletion requests the period will bring,

$$\min_{\substack{r\in[0,1]^{|\mathcal{C}|} \\ \sum_{c\in\mathcal{C}} n_c\,\phi(r_c)\,\le\,\eta}} \;\sum_{c\in\mathcal{C}} h_c\,\big(J_0 \;+\; \kappa\, n_c\,(1 - r_c)\big), \tag{4}$$

where the budget η bounds the utility forgone across the federation. An objective value of one hundred steps is one hundred unlearning steps expected over the period, and the minimizer is the preparation the budget buys. Eq. (3) counts what the deployer pays to serve the request; what the remaining clients lose to the correction is the object of separate work (Wen et al., 2026). We instantiate $\phi(r)$ as $r^2/2$ and calibrate κ once from the steps a held-out request takes.

The water-filling rule. Eq. (4) is convex with a strictly feasible point, and on an interior positive-budget regime its stationarity condition returns each readiness as a function of the hazard and one shared dual price λ,

$$r_c(\lambda) \;=\; \min\!\left\{1,\; (\phi')^{-1}\!\left(\frac{\kappa\, h_c}{\lambda}\right)\right\}, \qquad \sum_{c\in\mathcal{C}} n_c\,\phi\big(r_c(\lambda)\big) \;=\; \eta, \tag{5}$$

where the first rule fills each client to a level set by its hazard and the second fixes the level at which the budget is exactly spent; its left side decreases in λ, so a bisection recovers the price, which at the quadratic ϕ is κ times the square root of $\sum_c n_c h_c^2/(2\eta)$ once no client saturates. A dual price of two is the exchange rate at which one unit of utility buys two expected unlearning steps. Two properties organize the stage. The contribution size n_c cancels between cost and utility, so readiness depends on the hazard alone and data size acts only through the shared price. And the rule is monotone in the hazard, which gives the previous stage its rank robustness.

The price of uniform preparation. The appendix gives the value of Eq. (4) as a bound pair, and its gap to uniform preparation is the quantity the paper is about. No preparation rule attains a cost below the program's value, which is concave in the hazard vector, and the water-filling rule attains it. Uniform preparation at the same budget sets one readiness $\bar{r}$ for every client, and under the quadratic instance its excess cost has the closed form

$$\Gamma \;=\; \kappa\sqrt{2\eta n}\,\Big(\sqrt{\mathbb{E}_\nu[h^2]} \;-\; \mathbb{E}_\nu[h]\Big) \;=\; \kappa\sqrt{2\eta n}\;\frac{\mathrm{Var}_\nu(h)}{\sqrt{\mathbb{E}_\nu[h^2]} + \mathbb{E}_\nu[h]}, \tag{6}$$

where ν weights client c by n_c/n, the bracket holds the costs the water-filling and uniform rules attain in units of $\kappa\sqrt{2\eta n}$, and $\bar{r}$ is the one readiness uniform preparation affords at $\sqrt{2\eta/n}$. A variance of zero says every client is equally likely to leave and uniform preparation is already optimal; a large variance says the budget reaches clients who will not leave, and Eq. (6) counts the steps that costs. The gap vanishes exactly at equal hazards, grows with the variance, and is bounded by $\kappa\sqrt{2\eta n}$ times the standard deviation. It is also the prediction the experiments test, since hazard variance is a knob and the uniform baseline's excess cost is a measurement.

3.4 READINESS-PROPORTIONAL TRAINING (RPT)

Motivation. An allocation prices a cost the training loop has to make true, and RPT is the routing that makes the shared exposure the program reasons about equal to the exposure the aggregation receives.

Routing and aggregation. Client c holds the shared adapter w it received and a private adapter v_c that never leaves the device, with local model W_0 carrying both. Write g_c and $\tilde{g}_c$ for the gradients of its local loss with respect to the shared and private adapters. A local step of learning rate α routes the two gradients in the proportion the readiness sets,

$$w \;\leftarrow\; w \;-\; \alpha\,(1 - r_c)\,g_c, \qquad v_c \;\leftarrow\; v_c \;-\; \alpha\,r_c\,\tilde{g}_c, \tag{7}$$

where the client uploads only the change in w as its update Δ_c to Eq. (1). A readiness of one routes the whole step into the private adapter and the upload is zero; a readiness of zero recovers ordinary federated fine-tuning. Summing the routed steps over a local epoch of K steps gives the exposure identity the cost model rests on,

$$\Delta_c \;=\; -\,\alpha\,(1 - r_c)\sum_{k=1}^{K} g_c[k], \qquad v_c[K] - v_c[0] \;=\; -\,\alpha\,r_c\sum_{k=1}^{K}\tilde{g}_c[k], \tag{8}$$

where $g_c[k]$ and $\tilde{g}_c[k]$ are the gradients along the routed trajectory. Momentum preserves the same readiness factor on its filtered gradient sequence under consistent buffer initialization (Appendix C.6), adaptive optimizers are audited through realized update scaling, and a discrete variant rounds the split into private and shared ranks with its realized budget recorded separately.

Deletion execution. When client c withdraws, the deployer detaches v_c, which is exact and costs no step because the released model never held it, then ascends $\mathcal{L}_c$ on the shared adapter until the loss stands above γ. The recorded step count is the realized cost against which Eq. (3) is read. Under a per-step ascent condition stated in the appendix, that count obeys

$$m_c \;\le\; \left\lceil \frac{[\gamma - \mathcal{L}_c[0]]_+}{\delta} \right\rceil \;\le\; \left\lceil J_0 + \kappa n_c(1 - r_c) \right\rceil, \tag{9}$$

where $\mathcal{L}_c[0]$ is the loss at detachment, $\delta > 0$ lower-bounds the ascent before crossing, $[z]_+ = \max\{z, 0\}$, and m_c is the first count reaching γ; the ceiling is needed because the affine cost need not be an integer. Each compliance period refits Eq. (2), resolves Eq. (5), and applies the new readiness to later rounds; exposure already in the shared adapter is a constant in the program, so a rise in a hazard routes future learning differently rather than reopening the past.

Degeneracy. The rule collapses onto three published designs, and each collapse tells a deployer when uniform preparation suffices. When every hazard is equal, Eq. (5) assigns one readiness to every client and Eq. (6) vanishes, which recovers the uniform isolation of memorization sinks and per-owner experts (Ghosal et al., 2025; Gururangan et al., 2022). When every hazard is zero or one and the capacity budget covers all positive-hazard clients, readiness is one for the clients who will leave and zero for the rest, recovering the sharding of prior exact-unlearning and legal-risk designs (Bourtoule et al., 2021; Min et al., 2024). When every hazard is zero, readiness is zero throughout and Eq. (7) is ordinary federated fine-tuning. For positive budgets and unequal hazards in the unsaturated quadratic regime, the rule improves on uniform preparation by Eq. (6), and the appendix gives each collapse as a proposition.

4 Experiments

4.1 Experimental Setup

Data and protocol. The design uses user-partitioned Reddit (Arctic Shift, 2023), Stack Overflow and GitHub (Kocetkov et al., 2023) corpora, with earlier periods training a withdrawal predictor and later periods evaluating its event probabilities. The federated setting selects 100 clients per corpus and trains shared and private rank-16 adapters (Hu et al., 2021) on a frozen Llama-3.2-1B backbone (Grattafiori et al., 2024) with momentum SGD, and a matched leave-group-out reference fixes each client's deletion threshold. Every preparation rule receives the same utility budget, participation trace and paired requests, over three training seeds and twenty request sets per seed. Dataset construction, censoring, reference groups and implementation details are in Appendix B.

Comparisons and metrics. The comparison includes six preparation or reconstruction designs, uniform readiness at the same budget, and no preparation, with a common request-time deletion procedure separating allocation from removal. We read deletion steps jointly with retained utility,

Table 1: Main comparison and component ablation on Reddit-2024 with fitted hazards; three seeds and 20 requests per seed. Steps: deletion updates; Util.: retained accuracy; FQ: one minus extraction recall; MIA: membership AUC; Gap: utility gap across extreme hazard quintiles; Regret: cost above the program value. HE, BRA and RPT are prediction, allocation and routing; band (b) accumulates them onto the baseline above its rule and removes one at a time below it. Red and blue mark best and runner-up within each band; the two reference rows are excluded.

Method	Steps↓	Util.↑	FQ↑	MIA↓	Gap↓	Regret↓
(a) Fitted hazards (Gini 0.57), held-out withdrawals						
SISA [S&P'21] (Bourtoule et al., 2021)	422(±12)	0.606(±0.037)	0.72(±0.04)	0.53(±0.01)	0.032(±0.001)	321(±12)
Exact-Fun [ICDM'23] (Xiong et al., 2023)	362(±1)	0.649(±0.012)	**0.82**(±0.05)	**0.52**(±0.01)	0.029(±0.001)	261(±2)
FedShard [AAAI'26] (Wen et al., 2026)	349(±12)	0.677(±0.020)	0.77(±0.01)	**0.52**(±0.01)	0.028(±0.002)	248(±12)
DADS [ICML'24] (Sim et al., 2024)	492(±21)	0.637(±0.012)	0.68(±0.02)	0.57(±0.02)	0.044(±0.003)	391(±20)
MemSinks [ICML'25] (Ghosal et al., 2025)	272(±11)	**0.688**(±0.033)	0.73(±0.01)	0.54(±0.01)	0.029(±0.0003)	171(±10)
System-Aware [ICML'25] (Lu et al., 2025)	315(±11)	0.653(±0.026)	0.71(±0.03)	0.56(±0.02)	0.035(±0.001)	214(±12)
Uniform preparation	243(±13)	0.659(±0.056)	0.71(±0.04)	0.59(±0.04)	0.025(±0.0005)	142(±13)
PREMIUM	**110**(±4)	0.649(±0.016)	0.74(±0.07)	0.53(±0.01)	**0.020**(±0.001)	**9**(±4)
No preparation	631(±14)	0.672(±0.089)	0.69(±0.01)	0.54(±0.02)	0.000(±0.000)	530(±13)
(b) Component ablation of PREMIUM						
Baseline: no component	589(±21)	0.668(±0.017)	0.70(±0.01)	0.54(±0.01)	0.000(±0.000)	488(±22)
+ HE	589(±21)	**0.668**(±0.017)	0.70(±0.01)	0.54(±0.01)	**0.000**(±0.000)	488(±22)
+ BRA	589(±21)	**0.668**(±0.017)	0.70(±0.01)	0.54(±0.01)	**0.000**(±0.000)	488(±22)
+ RPT (PREMIUM)	**110**(±4)	0.649(±0.016)	**0.74**(±0.07)	**0.53**(±0.01)	0.020(±0.001)	**9**(±4)
w/o HE	243(±5)	0.666(±0.018)	0.72(±0.01)	0.56(±0.02)	0.025(±0.001)	142(±5)
w/o BRA	243(±5)	0.666(±0.018)	0.72(±0.01)	0.56(±0.02)	0.025(±0.001)	142(±5)
w/o RPT	589(±21)	**0.668**(±0.017)	0.70(±0.01)	0.54(±0.01)	**0.000**(±0.000)	488(±22)

Table 2: Component ablation at fitted hazards, one configuration per row and the three corpora reading across. $\mathrm{Var}(h)$ and λ are the hazard variance and the dual price on Reddit-2024, read at the realized readiness. Parentheses give changes from the complete configuration; components and metrics follow Table 1.

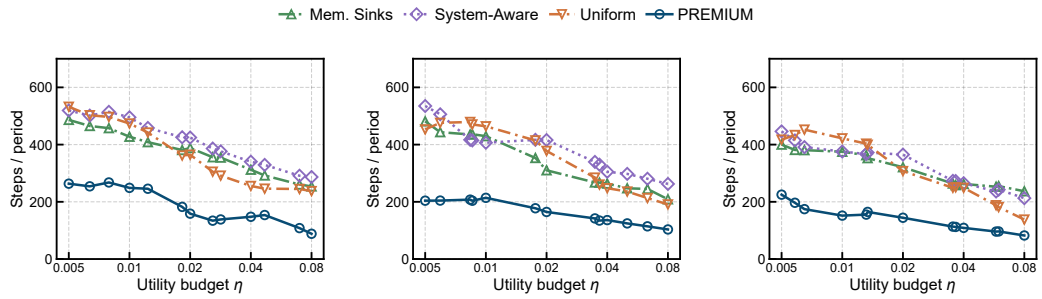
Components			**Program**		**Reddit-2024**			**StackOverflow-2024**			**GitHub · Stack opt-out**		
HE	BRA	RPT	$\mathrm{Var}(h)$	λ	Steps↓	Util.↑	Regret↓	Steps↓	Util.↑	Regret↓	Steps↓	Util.↑	Regret↓
✓	✓	✓	0.041	2.1	**110**(0)	0.649(0)	**9**(0)	**156**(0)	0.592(0)	**17**(0)	**90**(0)	**0.420**(0)	**4**(0)
✗	✓	✓	0.000	2.1	243(↑133)	0.666(↑0.017)	142(↑133)	296(↑140)	**0.618**(↑0.026)	157(↑140)	210(↑120)	0.409(↓0.011)	124(↑120)
✓	✗	✓	0.041	–	243(↑133)	0.666(↑0.017)	142(↑133)	296(↑140)	**0.618**(↑0.026)	157(↑140)	210(↑120)	0.409(↓0.011)	124(↑120)
✓	✓	✗	0.041	2.1	589(↑479)	**0.668**(↑0.019)	488(↑479)	676(↑520)	0.606(↑0.014)	537(↑520)	638(↑548)	0.413(↓0.007)	552(↑548)
✓	✗	✗	0.041	–	589(↑479)	**0.668**(↑0.019)	488(↑479)	676(↑520)	0.606(↑0.014)	537(↑520)	638(↑548)	0.413(↓0.007)	552(↑548)
✗	✓	✗	0.000	2.1	589(↑479)	**0.668**(↑0.019)	488(↑479)	676(↑520)	0.606(↑0.014)	537(↑520)	638(↑548)	0.413(↓0.007)	552(↑548)
✗	✗	✗	0.000	–	589(↑479)	**0.668**(↑0.019)	488(↑479)	676(↑520)	0.606(↑0.014)	537(↑520)	638(↑548)	0.413(↓0.007)	552(↑548)

extraction-based forget quality (Carlini et al., 2021) and membership inference (Shokri et al., 2017). Regret compares cost with the allocation program under the same declared inputs, and the utility gap compares retained clients in the extreme hazard quintiles. Appendix B.4 defines aggregation and uncertainty, and Appendix D specifies the independent audit.

4.2 Deletion Cost at a Fixed Utility Budget

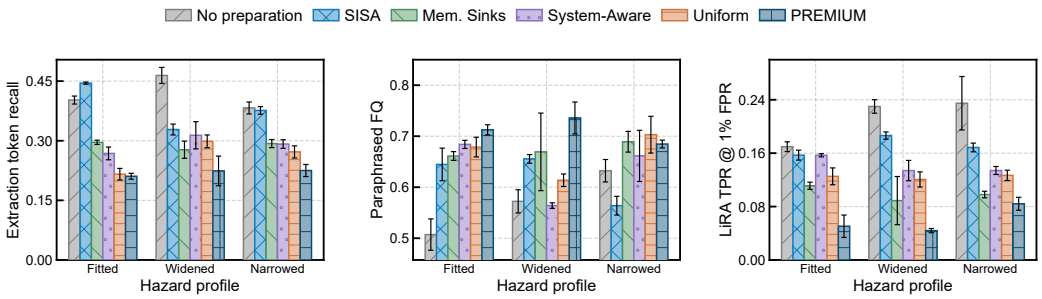
Table 1 separates deletion cost from release utility under the fitted hazard profile. PREMIUM reduces the request-time work relative to uniform preparation while their forgetting and membership readings stay close. Cost and utility order differently, because learning kept in shared parameters both improves retained prediction and leaves more shared influence to remove. The comparison that reads this correctly is preparation at a common utility budget with deletion quality read alongside the saved work, and uniform preparation isolates the benefit of spending that budget by withdrawal probability.

Figure 5 shows the same cost–budget relationship across the three corpora. More preparation lowers the shared residue and hence the deletion work, and the curves flatten as additional budget reaches clients with less expected withdrawal cost. Table 4 changes hazard spread at a fixed mean: the uniform gap widens when demand is concentrated and contracts when hazards become similar. Equal request volume thus leaves allocation, rather than the number of deletions, as the source of the separation. Under widened hazards PREMIUM needs 82 steps against 230 for uniform preparation, and under narrowed hazards the two rules nearly meet at 238 and 240 steps.



(a) Reddit-2024. (b) Stack Overflow. (c) GitHub.

Figure 5: Unlearning cost versus the utility budget across three corpora.



(a) Extraction recall. (b) Paraphrased probes. (c) Membership inference.

Figure 6: Deletion quality under extraction, paraphrase and membership-inference probes; mean and SD over three runs.

4.3 Contribution of the Three Stages

Band (b) of Table 1 shows that the allocation has to reach the training loop. The largest cost increase accompanies removal of RPT: a predicted readiness takes effect only once the corresponding learning stops entering the shared adapter. Removing HE or BRA separates the cost as well, through a less selective estimate of demand or a different use of the budget.

Reading the band in both directions distinguishes allocation from execution: removing HE replaces the hazards by their mean, removing BRA assigns uniform readiness, and removing RPT aggregates both adapters, so a stored readiness vector no longer preserves its private share. Steps and Regret track the realized routing rule and Util. the retained prediction quality under it, while a configuration that never differentiates readiness leaves the quintile gap at zero by construction, so the rows above BRA hold its smallest entries at the highest deletion cost in the band.

Table 2 repeats the ablation on all three corpora and adds the hazard variance a configuration realizes and the dual price it pays. Figure 6 reads deletion completeness alongside that cost, which separates a useful allocation from an early stop. Appendix E checks the exposure-cost relation, the uniform-gap formula, hazard estimation, limiting cases, refitting, privacy and model scale.

5 Conclusion

We formulated preparation for federated unlearning as an allocation of one utility budget across clients with different withdrawal probabilities. **PREMIUM** estimates those probabilities, assigns readiness through a shared dual price, and routes learning between shared and private adapters. Under the declared cost model the allocation is optimal, a closed form prices uniform preparation, and a regret bound ties hazard error to cost. Preparation thus becomes a budgeted part of the deletion process.

AI USE STATEMENT

For this work, we used generative AI tools for proposing and refining hypotheses, for help with writing the proofs, for designing experiments and reviewing the research methodology, for cleaning and reformatting the datasets, and for interpreting the results. No generative AI tool was used to create synthetic data sets, to build theoretical models or conceptual frameworks, to state mathematical claims, to contribute critical steps to their proofs, to implement methods, or to translate text, and support for qualitative and thematic data analysis is not applicable here. We further used generative AI tools to explore possible research topics, to produce the plots of our experimental results, to retrieve and summarize relevant literature, to draft the notation table of the appendix, to edit the paper for clarity, and to format the bibliography. We have reviewed all AI-assisted work. The authors checked each step of every AI-assisted proof against the stated assumptions, matched the values in every AI-produced plot to the experiment logs, ran a literature survey to check the selected topic for overlap with existing work, and verified each reference against its publisher page, CrossRef or DBLP. We take responsibility for the final content of this work, including text, claims or artifacts produced with the aid of generative AI.

REFERENCES

Arctic Shift. Arctic Shift: Making Reddit data accessible to researchers, moderators and everyone else. https://github.com/ArthurHeitmann/arctic_shift, 2023.

Eva Ascarza. Retention futility: Targeting high-risk customers might be ineffective. *Journal of Marketing Research*, 55(1):80–98, 2018. doi: 10.1509/jmr.16.0163. URL https://doi.org/10.1509/jmr.16.0163.

Dimitris Bertsimas and Nathan Kallus. From predictive to prescriptive analytics. *Management Science*, 66(3):1025–1044, 2020. doi: 10.1287/mnsc.2018.3253.

Lucas Bourtoule, Varun Chandrasekaran, Christopher A. Choquette-Choo, Hengrui Jia, Adelin Travers, Baiwu Zhang, David Lie, and Nicolas Papernot. Machine unlearning. In *2021 IEEE Symposium on Security and Privacy (SP)*, pp. 141–159, 2021. doi: 10.1109/sp40001.2021.00019.

Stephen Boyd and Lieven Vandenberghe. *Convex Optimization*. Cambridge University Press, 2004.

Xiaoyu Cao, Jinyuan Jia, Zaixi Zhang, and Neil Zhenqiang Gong. FedRecover: Recovering from poisoning attacks in federated learning using historical information. In *2023 IEEE Symposium on Security and Privacy (SP)*, pp. 1366–1383, 2023. doi: 10.1109/sp46215.2023.10179336.

Yinzhi Cao and Junfeng Yang. Towards making systems forget with machine unlearning. In *2015 IEEE Symposium on Security and Privacy*, pp. 463–480, 2015. doi: 10.1109/sp.2015.35.

Nicholas Carlini, Florian Tramèr, Eric Wallace, Matthew Jagielski, Ariel Herbert-Voss, Katherine Lee, Adam Roberts, Tom Brown, Dawn Song, Úlfar Erlingsson, Alina Oprea, and Colin Raffel. Extracting training data from large language models. In *30th USENIX Security Symposium (USENIX Security 21)*, pp. 2633–2650, 2021.

Nicholas Carlini, Steve Chien, Milad Nasr, Shuang Song, Andreas Terzis, and Florian Tramèr. Membership inference attacks from first principles. In *2022 IEEE Symposium on Security and Privacy (SP)*, pp. 1897–1914, 2022. doi: 10.1109/sp46214.2022.9833649.

Liam Collins, Hamed Hassani, Aryan Mokhtari, and Sanjay Shakkottai. Exploiting shared representations for personalized federated learning. In *Proceedings of the 38th International Conference on Machine Learning*, volume 139 of *Proceedings of Machine Learning Research*, pp. 2089–2099. PMLR, 2021.

D. R. Cox. Regression models and life-tables. *Journal of the Royal Statistical Society: Series B (Methodological)*, 34(2):187–202, 01 1972. ISSN 0035-9246. doi: 10.1111/j.2517-6161.1972.tb00899.x. URL https://doi.org/10.1111/j.2517-6161.1972.tb00899.x.

Yonatan Dukler, Benjamin Bowman, Alessandro Achille, Aditya Golatkar, Ashwin Swaminathan, and Stefano Soatto. Safe: Machine unlearning with shard graphs. In *2023 IEEE/CVF International Conference on Computer Vision (ICCV)*, pp. 17062–17072, 2023. doi: 10.1109/ICCV51070.2023.01569.

Gaurav Rohit Ghosal, Pratyush Maini, and Aditi Raghunathan. Memorization sinks: Isolating memorization during LLM training. In *Proceedings of the 42nd International Conference on Machine Learning*, volume 267 of *Proceedings of Machine Learning Research*, pp. 19307–19326. PMLR, 2025.

Antonio Ginart, Melody Guan, Gregory Valiant, and James Zou. Making AI forget you: Data deletion in machine learning. In *Advances in Neural Information Processing Systems*, volume 32, 2019.

Aaron Grattafiori, Abhimanyu Dubey, Abhinav Jauhri, Abhinav Pandey, Abhishek Kadian, Ahmad Al-Dahle, Aiesha Letman, Akhil Mathur, Alan Schelten, Alex Vaughan, Amy Yang, Angela Fan, Anirudh Goyal, Anthony Hartshorn, Aobo Yang, Archi Mitra, Archie Sravankumar, Artem Korenev, Arthur Hinsvark, Arun Rao, Aston Zhang, Aurelien Rodriguez, Austen Gregerson, Ava Spataru, Baptiste Roziere, Bethany Biron, Binh Tang, Bobbie Chern, Charlotte Caucheteux, Chaya Nayak, Chloe Bi, Chris Marra, Chris McConnell, Christian Keller, Christophe Touret, Chunyang Wu, Corinne Wong, Cristian Canton Ferrer, Cyrus Nikolaidis, Damien Allonsius, Daniel Song, Danielle Pintz, Danny Livshits, Danny Wyatt, David Esiobu, Dhruv Choudhary, Dhruv Mahajan, Diego Garcia-Olano, Diego Perino, Dieuwke Hupkes, Egor Lakomkin, Ehab AlBadawy, Elina Lobanova, Emily Dinan, Eric Michael Smith, Filip Radenovic, Francisco Guzmán, Frank Zhang, Gabriel Synnaeve, Gabrielle Lee, Georgia Lewis Anderson, Govind Thattai, Graeme Nail, Gregoire Mialon, Guan Pang, Guillem Cucurell, Hailey Nguyen, Hannah Korevaar, Hu Xu, Hugo Touvron, Iliyan Zarov, Imanol Arrieta Ibarra, Isabel Kloumann, Ishan Misra, Ivan Evtimov, Jack Zhang, Jade Copet, Jaewon Lee, Jan Geffert, Jana Vranes, Jason Park, Jay Mahadeokar, Jeet Shah, Jelmer van der Linde, Jennifer Billock, Jenny Hong, Jenya Lee, Jeremy Fu, Jianfeng Chi, Jianyu Huang, Jiawen Liu, Jie Wang, Jiecao Yu, Joanna Bitton, Joe Spisak, Jongsoo Park, Joseph Rocca, Joshua Johnstun, Joshua Saxe, Junteng Jia, Kalyan Vasuden Alwala, Karthik Prasad, Kartikeya Upasani, Kate Plawiak, Ke Li, Kenneth Heafield, Kevin Stone, Khalid El-Arini, Krithika Iyer, Kshitiz Malik, Kuenley Chiu, Kunal Bhalla, Kushal Lakhotia, Lauren Rantala-Yeary, Laurens van der Maaten, Lawrence Chen, Liang Tan, Liz Jenkins, Louis Martin, Lovish Madaan, Lubo Malo, Lukas Blecher, Lukas Landzaat, Luke de Oliveira, Madeline Muzzi, Mahesh Pasupuleti, Mannat Singh, Manohar Paluri, Marcin Kardas, Maria Tsimpoukelli, Mathew Oldham, Mathieu Rita, Maya Pavlova, Melanie Kambadur, Mike Lewis, Min Si, Mitesh Kumar Singh, Mona Hassan, Naman Goyal, Narjes Torabi, Nikolay Bashlykov, Nikolay Bogoychev, Niladri Chatterji, Ning Zhang, Olivier Duchenne, Onur Çelebi, Patrick Alrassy, Pengchuan Zhang, Pengwei Li, Petar Vasic, Peter Weng, Prajjwal Bhargava, Pratik Dubal, Praveen Krishnan, Punit Singh Koura, Puxin Xu, Qing He, Qingxiao Dong, Ragavan Srinivasan, Raj Ganapathy, Ramon Calderer, Ricardo Silveira Cabral, Robert Stojnic, Roberta Raileanu, Rohan Maheswari, Rohit Girdhar, Rohit Patel, Romain Sauvestre, Ronnie Polidoro, Roshan Sumbaly, Ross Taylor, Ruan Silva, Rui Hou, Rui Wang, Saghar Hosseini, Sahana Chennabasappa, Sanjay Singh, Sean Bell, Seohyun Sonia Kim, Sergey Edunov, Shaoliang Nie, Sharan Narang, Sharath Raparthy, Sheng Shen, Shengye Wan, Shruti Bhosale, Shun Zhang, Simon Vandenhende, Soumya Batra, Spencer Whitman, Sten Sootla, Stephane Collot, Suchin Gururangan, Sydney Borodinsky, Tamar Herman, Tara Fowler, Tarek Sheasha, Thomas Georgiou, Thomas Scialom, Tobias Speckbacher, Todor Mihaylov, Tong Xiao, Ujjwal Karn, Vedanuj Goswami, Vibhor Gupta, Vignesh Ramanathan, Viktor Kerkez, Vincent Gonguet, Virginie Do, Vish Vogeti, Vítor Albiero, Vladan Petrovic, Weiwei Chu, Wenhan Xiong, Wenyin Fu, Whitney Meers, Xavier Martinet, Xiaodong Wang, Xiaofang Wang, Xiaoqing Ellen Tan, Xide Xia, Xinfeng Xie, Xuchao Jia, Xuewei Wang, Yaelle Goldschlag, Yashesh Gaur, Yasmine Babaei, Yi Wen, Yiwen Song, Yuchen Zhang, Yue Li, Yuning Mao, Zacharie Delpierre Coudert, Zheng Yan, Zhengxing Chen, Zoe Papakipos, Aaditya Singh, Aayushi Srivastava, Abha Jain, Adam Kelsey, Adam Shajnfeld, Adithya Gangidi, Adolfo Victoria, Ahuva Goldstand, Ajay Menon, Ajay Sharma, Alex Boesenberg, Alexei Baevski, Allie Feinstein, Amanda Kallet, Amit Sangani, Amos Teo, Anam Yunus, Andrei Lupu, Andres Alvarado, Andrew Caples, Andrew Gu, Andrew Ho, Andrew Poulton, Andrew Ryan, Ankit Ramchandani, Annie Dong, Annie Franco, Anuj Goyal, Aparajita Saraf, Arkabandhu Chowdhury, Ashley Gabriel, Ashwin Bharambe, Assaf Eisenman, Azadeh Yazdan, Beau James, Ben Maurer, Benjamin Leonhardi, Bernie Huang, Beth Loyd, Beto De Paola, Bhargavi Paranjape, Bing Liu, Bo Wu, Boyu Ni, Braden Hancock, Bram Wasti, Brandon

Spence, Brani Stojkovic, Brian Gamido, Britt Montalvo, Carl Parker, Carly Burton, Catalina Mejia, Ce Liu, Changhan Wang, Changkyu Kim, Chao Zhou, Chester Hu, Ching-Hsiang Chu, Chris Cai, Chris Tindal, Christoph Feichtenhofer, Cynthia Gao, Damon Civin, Dana Beaty, Daniel Kreymer, Daniel Li, David Adkins, David Xu, Davide Testuggine, Delia David, Devi Parikh, Diana Liskovich, Didem Foss, Dingkang Wang, Duc Le, Dustin Holland, Edward Dowling, Eissa Jamil, Elaine Montgomery, Eleonora Presani, Emily Hahn, Emily Wood, Eric-Tuan Le, Erik Brinkman, Esteban Arcaute, Evan Dunbar, Evan Smothers, Fei Sun, Felix Kreuk, Feng Tian, Filippos Kokkinos, Firat Ozgenel, Francesco Caggioni, Frank Kanayet, Frank Seide, Gabriela Medina Florez, Gabriella Schwarz, Gada Badeer, Georgia Swee, Gil Halpern, Grant Herman, Grigory Sizov, Guangyi (Jack) Zhang, Guna Lakshminarayanan, Hakan Inan, Hamid Shojanazeri, Han Zou, Hannah Wang, Hanwen Zha, Haroun Habeeb, Harrison Rudolph, Helen Suk, Henry Aspegren, Hunter Goldman, Hongyuan Zhan, Ibrahim Damlaj, Igor Molybog, Igor Tufanov, Ilias Leontiadis, Irina-Elena Veliche, Itai Gat, Jake Weissman, James Geboski, James Kohli, Janice Lam, Japhet Asher, Jean-Baptiste Gaya, Jeff Marcus, Jeff Tang, Jennifer Chan, Jenny Zhen, Jeremy Reizenstein, Jeremy Teboul, Jessica Zhong, Jian Jin, Jingyi Yang, Joe Cummings, Jon Carvill, Jon Shepard, Jonathan McPhie, Jonathan Torres, Josh Ginsburg, Junjie Wang, Kai Wu, Kam Hou U, Karan Saxena, Kartikay Khandelwal, Katayoun Zand, Kathy Matosich, Kaushik Veeraraghavan, Kelly Michelena, Keqian Li, Kiran Jagadeesh, Kun Huang, Kunal Chawla, Kyle Huang, Lailin Chen, Lakshya Garg, Lavender A, Leandro Silva, Lee Bell, Lei Zhang, Liangpeng Guo, Licheng Yu, Liron Moshkovich, Luca Wehrstedt, Madian Khabsa, Manav Avalani, Manish Bhatt, Martynas Mankus, Matan Hasson, Matthew Lennie, Matthias Reso, Maxim Groshev, Maxim Naumov, Maya Lathi, Meghan Keneally, Miao Liu, Michael L. Seltzer, Michal Valko, Michelle Restrepo, Mihir Patel, Mik Vyatskov, Mikayel Samvelyan, Mike Clark, Mike Macey, Mike Wang, Miquel Jubert Hermoso, Mo Metanat, Mohammad Rastegari, Munish Bansal, Nandhini Santhanam, Natascha Parks, Natasha White, Navyata Bawa, Nayan Singhal, Nick Egebo, Nicolas Usunier, Nikhil Mehta, Nikolay Pavlovich Laptev, Ning Dong, Norman Cheng, Oleg Chernoguz, Olivia Hart, Omkar Salpekar, Ozlem Kalinli, Parkin Kent, Parth Parekh, Paul Saab, Pavan Balaji, Pedro Rittner, Philip Bontrager, Pierre Roux, Piotr Dollar, Polina Zvyagina, Prashant Ratanchandani, Pritish Yuvraj, Qian Liang, Rachad Alao, Rachel Rodriguez, Rafi Ayub, Raghotham Murthy, Raghu Nayani, Rahul Mitra, Rangaprabhu Parthasarathy, Raymond Li, Rebekkah Hogan, Robin Battey, Rocky Wang, Russ Howes, Ruty Rinott, Sachin Mehta, Sachin Siby, Sai Jayesh Bondu, Samyak Datta, Sara Chugh, Sara Hunt, Sargun Dhillon, Sasha Sidorov, Satadru Pan, Saurabh Mahajan, Saurabh Verma, Seiji Yamamoto, Sharadh Ramaswamy, Shaun Lindsay, Shaun Lindsay, Sheng Feng, Shenghao Lin, Shengxin Cindy Zha, Shishir Patil, Shiva Shankar, Shuqiang Zhang, Shuqiang Zhang, Sinong Wang, Sneha Agarwal, Soji Sajuyigbe, Soumith Chintala, Stephanie Max, Stephen Chen, Steve Kehoe, Steve Satterfield, Sudarshan Govindaprasad, Sumit Gupta, Summer Deng, Sungmin Cho, Sunny Virk, Suraj Subramanian, Sy Choudhury, Sydney Goldman, Tal Remez, Tamar Glaser, Tamara Best, Thilo Koehler, Thomas Robinson, Tianhe Li, Tianjun Zhang, Tim Matthews, Timothy Chou, Tzook Shaked, Varun Vontimitta, Victoria Ajayi, Victoria Montanez, Vijai Mohan, Vinay Satish Kumar, Vishal Mangla, Vlad Ionescu, Vlad Poenaru, Vlad Tiberiu Mihailescu, Vladimir Ivanov, Wei Li, Wenchen Wang, Wenwen Jiang, Wes Bouaziz, Will Constable, Xiaocheng Tang, Xiaojian Wu, Xiaolan Wang, Xilun Wu, Xinbo Gao, Yaniv Kleinman, Yanjun Chen, Ye Hu, Ye Jia, Ye Qi, Yenda Li, Yilin Zhang, Ying Zhang, Yossi Adi, Youngjin Nam, Yu (Sid) Wang, Yu Zhao, Yuchen Hao, Yundi Qian, Yunlu Li, Yuzi He, Zach Rait, Zachary DeVito, Zef Rosnbrick, Zhaoduo Wen, Zhenyu Yang, Zhiwei Zhao, and Zhiyu Ma. The llama 3 herd of models, 2024. URL https://arxiv.org/abs/2407.21783.

Laura Graves, Vineel Nagisetty, and Vijay Ganesh. Amnesiac machine learning. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 35, pp. 11516–11524, 2021. doi: 10.1609/aaai.v35i13.17371.

Chuan Guo, Tom Goldstein, Awni Hannun, and Laurens Van Der Maaten. Certified data removal from machine learning models. In *Proceedings of the 37th International Conference on Machine Learning*, volume 119 of *Proceedings of Machine Learning Research*, pp. 3832–3842. PMLR, 2020.

Pengxin Guo, Shuang Zeng, Yanran Wang, Huijie Fan, Feifei Wang, and Liangqiong Qu. Selective aggregation for low-rank adaptation in federated learning. In *International Conference on Learning Representations*, pp. 99003–99027, 2025.

Suchin Gururangan, Mike Lewis, Ari Holtzman, Noah Smith, and Luke Zettlemoyer. DEMix layers: Disentangling domains for modular language modeling. In *Proceedings of the 2022 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pp. 5557–5576, 2022. doi: 10.18653/v1/2022.naacl-main.407.

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, 2021. URL https://arxiv.org/abs/2106.09685.

Denis Kocetkov, Raymond Li, Loubna Ben Allal, Jia Li, Chenghao Mou, Yacine Jernite, Margaret Mitchell, Carlos Muñoz Ferrandis, Sean Hughes, Thomas Wolf, Dzmitry Bahdanau, Leandro von Werra, and Harm de Vries. The Stack: 3 TB of permissively licensed source code. *Transactions on Machine Learning Research*, 2023. URL https://openreview.net/forum?id=pxpbTdUEpD.

Tian Li, Anit Kumar Sahu, Manzil Zaheer, Maziar Sanjabi, Ameet Talwalkar, and Virginia Smith. Federated optimization in heterogeneous networks, 2020. URL https://arxiv.org/abs/1812.06127.

Gaoyang Liu, Xiaoqiang Ma, Yang Yang, Chen Wang, and Jiangchuan Liu. FedEraser: Enabling efficient client-level data removal from federated learning models. In *2021 IEEE/ACM 29th International Symposium on Quality of Service (IWQOS)*, pp. 1–10, 2021. doi: 10.1109/iwqos52092.2021.9521274.

Linda Lu, Ayush Sekhari, and Karthik Sridharan. System-aware unlearning algorithms: Use lesser, forget faster. In Aarti Singh, Maryam Fazel, Daniel Hsu, Simon Lacoste-Julien, Felix Berkenkamp, Tegan Maharaj, Kiri Wagstaff, and Jerry Zhu (eds.), *Forty-second International Conference on Machine Learning, ICML 2025, Vancouver, BC, Canada, July 13-19, 2025*, volume 267 of *Proceedings of Machine Learning Research*. PMLR / OpenReview.net, 2025. URL https://proceedings.mlr.press/v267/lu25h.html.

Brendan McMahan, Eider Moore, Daniel Ramage, Seth Hampson, and Blaise Aguera y Arcas. Communication-Efficient Learning of Deep Networks from Decentralized Data. In *Proceedings of the 20th International Conference on Artificial Intelligence and Statistics*, volume 54 of *Proceedings of Machine Learning Research*, pp. 1273–1282. PMLR, 2017.

Sewon Min, Suchin Gururangan, Eric Wallace, Weijia Shi, Hannaneh Hajishirzi, Noah Smith, and Luke Zettlemoyer. SILO language models: Isolating legal risk in a nonparametric datastore. In *International Conference on Learning Representations*, pp. 25132–25159, 2024.

Zibin Pan, Zhichao Wang, Chi Li, Kaiyan Zheng, Boqi Wang, Xiaoying Tang, and Junhua Zhao. Federated unlearning with gradient descent and conflict mitigation. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 39, pp. 19804–19812, 2025. doi: 10.1609/aaai.v39i19.34181.

Michael Patriksson. A survey on the continuous nonlinear resource allocation problem. *European Journal of Operational Research*, 185(1):1–46, 2008. doi: 10.1016/j.ejor.2006.12.006.

Krishna Pillutla, Kshitiz Malik, Abdelrahman Mohamed, Michael G. Rabbat, Maziar Sanjabi, and Lin Xiao. Federated learning with partial model personalization. In Kamalika Chaudhuri, Stefanie Jegelka, Le Song, Csaba Szepesvári, Gang Niu, and Sivan Sabato (eds.), *International Conference on Machine Learning, ICML 2022, 17-23 July 2022, Baltimore, Maryland, USA*, volume 162 of *Proceedings of Machine Learning Research*, pp. 17716–17758. PMLR, 2022. URL https://proceedings.mlr.press/v162/pillutla22a.html.

Reza Shokri, Marco Stronati, Congzheng Song, and Vitaly Shmatikov. Membership inference attacks against machine learning models. In *2017 IEEE Symposium on Security and Privacy (SP)*, pp. 3–18, 2017. doi: 10.1109/sp.2017.41.

Rachael Hwee Ling Sim, Jue Fan, Xiao Tian, Patrick Jaillet, and Bryan Kian Hsiang Low. Deletion-anticipative data selection with a limited budget. In Ruslan Salakhutdinov, Zico Kolter, Katherine A. Heller, Adrian Weller, Nuria Oliver, Jonathan Scarlett, and Felix Berkenkamp (eds.), *Forty-first*

International Conference on Machine Learning, ICML 2024, Vienna, Austria, July 21-27, 2024, volume 235 of *Proceedings of Machine Learning Research*, pp. 45468–45507. PMLR / OpenReview.net, 2024. URL https://proceedings.mlr.press/v235/sim24a.html.

Enayat Ullah, Tung Mai, Anup Rao, Ryan A. Rossi, and Raman Arora. Machine unlearning via algorithmic stability. In *Proceedings of Thirty Fourth Conference on Learning Theory*, volume 134 of *Proceedings of Machine Learning Research*, pp. 4126–4142. PMLR, 2021.

Siyuan Wen, Meng Zhang, Yang Yang, and Ningning Ding. FedShard: Federated unlearning with efficiency fairness and performance fairness. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 40, pp. 26841–26848, 2026. doi: 10.1609/aaai.v40i32.39895.

Zuobin Xiong, Wei Li, Yingshu Li, and Zhipeng Cai. Exact-Fun: An exact and efficient federated unlearning approach. In *2023 IEEE International Conference on Data Mining (ICDM)*, pp. 1439–1444, 2023. doi: 10.1109/icdm58522.2023.00188.

Yiyuan Yang, Guodong Long, Tao Shen, Jing Jiang, and Michael Blumenstein. Dual-personalizing adapter for federated foundation models. In *Advances in Neural Information Processing Systems 37*, pp. 39409–39433, 2024. doi: 10.52202/079017-1245.

Fuyao Zhang, Xinyu Yan, Tiantong Wu, Wenjie Li, Tianxiang Chen, Yang Cao, Ran Yan, Longtao Huang, Wei Yang Bryan Lim, and Qiang Yang. Oblivionis: A lightweight learning and unlearning framework for federated large language models. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 40, pp. 28185–28193, 2026. doi: 10.1609/aaai.v40i33.40045.

Lefeng Zhang, Tianqing Zhu, Haibin Zhang, Ping Xiong, and Wanlei Zhou. FedRecovery: Differentially private machine unlearning for federated learning frameworks. *IEEE Transactions on Information Forensics and Security*, 18:4732–4746, 2023. doi: 10.1109/tifs.2023.3297905.

Ruiqi Zhang, Licong Lin, Yu Bai, and Song Mei. Negative preference optimization: From catastrophic collapse to effective unlearning. *CoRR*, abs/2404.05868, 2024. doi: 10.48550/ARXIV.2404.05868. URL https://doi.org/10.48550/arXiv.2404.05868.

APPENDIX CONTENTS

A Related Work **16**

B Technical Appendix **16**

B.1 Notation Table 16
B.2 Dataset Details 16
B.3 Baseline Descriptions 17
B.4 Evaluation Metrics 18
B.5 Implementation Details 18
B.6 Alternative Designs 19
B.7 Algorithm 19
B.8 Broader Impacts 19

C Proofs of Theoretical Results **20**

C.1 Assumptions and the Allocation Program 20
C.2 The Price of Preparation 21
C.3 The Price of Uniform Preparation 22
C.4 Rank and Value Robustness 23
C.5 Limiting Hazard Profiles 23
C.6 Realization in Training 24

D Audit Protocol **25**

D.1 Temporal and Data Commitments 25
D.2 Budget and Allocation Audit 25
D.3 Training and Deletion Audit 25
D.4 Uncertainty and Resource Accounting 26

E Additional Experiments **26**

E.1 Cost Calibration and Theoretical Diagnostics 26
E.2 Complete Comparison across Hazard Profiles 27
E.3 Hazard Prediction and Estimation Error 28
E.4 Limiting Cases and Full Component Ablation 28
E.5 Budget, Hazard and Rebalancing Sensitivity 29
E.6 Feature, Fairness and Privacy Comparisons 30
E.7 Model Scale, Routing and Deletion Quality 31

A Related Work

Preparing models for deletion. Preparation methods include sharded retraining (Bourtoule et al., 2021), checkpoint reuse (Graves et al., 2021), modular isolation (Gururangan et al., 2022; Dukler et al., 2023) and memorization sinks (Ghosal et al., 2025). Request-aware sharding uses deletion priors when arranging data (Bourtoule et al., 2021), while system-aware unlearning relates retained training information to removal cost (Lu et al., 2025; Ullah et al., 2021). These approaches establish preparation and request probabilities as useful design choices. **PREMIUM** studies their allocation across a federation: each client receives a continuous private share under one aggregate utility budget. The resulting water-filling rule prices that share, and the quadratic allocation gap quantifies the excess cost of preparing every client equally.

Federated unlearning and parameter isolation. Federated deletion methods include history reconstruction (Cao et al., 2023; Zhang et al., 2023), retained-update calibration (Liu et al., 2021), loss-based editing (Pan et al., 2025; Zhang et al., 2026) and selective retraining (Xiong et al., 2023). Personalized federated learning additionally divides parameters into shared and private components (Collins et al., 2021; Pillutla et al., 2022), with adapters providing a compact implementation (Guo et al., 2025). **PREMIUM** uses this division before requests arrive: predicted withdrawal risk determines how much learning remains private, and a common deletion procedure removes the shared residue afterward. Exposure accounting links the assigned share to routed training, allowing preparation cost, deletion quality and retained utility to be read together.

Prediction-guided resource allocation. Survival models estimate event probabilities from censored histories (Cox, 1972), while prediction-guided optimization allocates budgets using forecasts of demand (Bertsimas & Kallus, 2020). Capacity-constrained selection depends on ordering; convex resource allocation also uses probability values (Patriksson, 2008; Boyd & Vandenberghe, 2004). **PREMIUM** connects these regimes to federated withdrawal through separate ranking and calibration bounds. Its exposure ledger carries past shared contributions across refitting periods, so updated predictions alter future readiness while preserving the cost already committed by earlier training.

B Technical Appendix

B.1 Notation Table

We summarize the commonly used notation throughout this paper in Table 3.

B.2 Dataset Details

Corpora and client identity. The evaluation design uses Reddit comments (Arctic Shift, 2023), Stack Overflow posts and GitHub code activity, with one author or account defining a client. The dated snapshots and record identifiers determine the exact training population. A temporal split keeps earlier records for hazard fitting and later periods for withdrawal evaluation. User grouping precedes tokenization and train–test splitting, preventing records from the same account from crossing client boundaries.

Withdrawal labels. Reddit account deletion and comment removal are recorded as separate events; removing one comment is not automatically a whole-client withdrawal. Stack Overflow labels use account-deletion indicators between snapshots. GitHub activity is joined to The Stack opt-out records (Kocetkov et al., 2023) only when the opt-out scope can be mapped to the same training owner. Unmatched or ambiguous identifiers are excluded from withdrawal evaluation and their counts remain in the data manifest. A client absent from later snapshots is treated as censored unless a withdrawal indicator is available.

Hazard populations and training populations. HE is fitted on the eligible user population before selecting the 100 clients per corpus used in the federated training design. Selection is stratified by fitted hazard. Report population and sampled-client hazard summaries separately; stratification can change the sample mean and Gini coefficient. Availability traces determine round participation and may supply a historical feature, but they do not define withdrawal events.

Table 3: Notation and Definitions.

Notation	Definition
$\mathcal{C}$, $\mathcal{D}_c$	The client set and client-local dataset.
n_c, n	The client sample count and total sample count.
W_0, w, v_c	The frozen backbone, shared adapter and private adapter.
$\mathcal{L}_c$, γ	The client loss and its matched-reference deletion threshold.
x_c, θ	The client features and hazard-model parameters.
h_c, $\widehat{h}_c$	The withdrawal probability and its fitted estimate.
π	The ordering of clients by decreasing deletion hazard.
r_c, $\bar{r}$	The client readiness and uniform readiness.
r^*, $\widehat{r}$	The optimal and estimated-hazard readiness allocations.
s_c	The shared exposure recorded for client c.
ϕ, U_c	The utility-cost shape and client preparation cost.
η, λ	The aggregate utility budget and its common dual price.
κ, J_0	The exposure-cost slope and fixed request-time cost.
J_c, J_h	The client request cost and hazard-weighted total cost.
V, V^{u}	The optimal expected cost and uniform-preparation cost.
ν_c	The client weighting n_c/n used for hazard moments.
$\mathbb{E}_\nu[h]$, $\mathrm{Var}_\nu(h)$	The weighted mean and variance of deletion hazards.
Γ	The excess cost of uniform over optimal preparation.
Reg	The true-cost excess of an estimated-hazard allocation.
B	The capacity budget in the rank-robustness result.
α, K, T	The learning rate, local steps and rounds per period.
g_c, $\widetilde{g}_c$	The shared and private adapter gradients.
$\Delta_c[t]$	The shared update uploaded by client c in round t.
δ, m_c	The loss-ascent lower bound and deletion stopping step.

Three hazard profiles. The fitted profile preserves the estimated probabilities. Widened and narrowed profiles change dispersion at a fixed mean using $h'_c = \sigma(a\,\mathrm{logit}(\widehat{h}_c) + b)$, where b is solved to restore the original weighted mean and a controls spread. Requests drawn from these profiles test the allocation mechanism under a controlled change in demand. Their provenance is distinct from withdrawals observed in later snapshots. Client partitions and retained evaluation records stay fixed across profiles.

B.3 Baseline Descriptions

Sharding and selective reconstruction. SISA and the sharding configurations named Exact-Fun and FedShard localize retraining to the components affected by a request (Bourtoule et al., 2021; Xiong et al., 2023; Wen et al., 2026). The comparison records shard membership, saved checkpoints, reconstructed components and the aggregation used at inference. An adaptation to shared adapters is evaluated as that adaptation rather than inheriting an exactness claim from a different training pipeline.

Training-time preparation. DADS supplies a deletion-aware selection comparison; Memorization Sinks isolates memorized content during training; the system-aware comparison controls information retained by the learning system (Sim et al., 2024; Ghosal et al., 2025; Lu et al., 2025). Their preparation, storage and request-time computation are counted separately. The same model size, retained-data evaluation and stopping criterion are used wherever the mechanisms permit a matched comparison.

Allocation controls. Uniform preparation solves $n\phi(\bar{r}) = \eta$ and gives every client the same readiness. No preparation fixes $r_c = 0$. An oracle uses the declared true hazards only in the controlled allocation studies and provides the program reference. HE, BRA and RPT ablations change prediction, allocation and routing respectively; all three keep the budget and request set fixed. Removing an allocator and removing its input estimator can yield different experimental controls, so their resulting readiness vectors are saved explicitly.

B.4 Evaluation Metrics

Deletion cost. One step is one full-withdrawn-set gradient update at the declared request-time learning rate. Count steps to the client-specific reference threshold and record the maximum audit budget for censored requests. Report both steps per served request and total cost per compliance period: Eq. (4) prices the latter, $\sum_c h_c J_c$, without dividing by the expected number of requests. The cost fit uses held-out requests and freezes J_0 and κ before comparison.

Retained utility and forgetting. Util. is next-token accuracy on retained-client evaluation records. FQ is one minus extraction recall on withdrawn records under a fixed extraction protocol. MIA is the area under a membership-inference ROC curve; the low-false-positive audit additionally reports its true-positive rate at 1% false positives (Carlini et al., 2022). The forget threshold is set by the matched leave-group-out reference used in the design, and all methods use the same reference grouping and probes.

Hazard quality. ROC AUC evaluates event discrimination, Brier score evaluates probability error, Kendall correlation evaluates ordering, and top-decile lift compares the event rate in the highest predicted decile with the population rate. These metrics use a temporal holdout and observed eligible labels. Ordering-swap experiments use explicitly constructed hazard permutations; a swap count is not inferred from AUC. Calibration and allocation are evaluated together because quadratic budgets consume probability values as well as ranks.

Allocation diagnostics. Regret is $J_h(\widehat{r}) - V(h, \eta)$ for a feasible allocation and the same cost model. A realized cost minus a fitted program value additionally includes cost-model error, so both quantities are retained in the audit. Gap is the retained-utility difference between the highest and lowest hazard quintiles of non-withdrawing clients. The limiting-case diagnostic D_∞ is the maximum readiness difference from the specified reference allocation. Time and communication count training, refitting and request processing separately.

Aggregation. The evaluation design uses three independent training seeds with twenty request sets per seed. Requests are paired across preparation rules and summarized within seed before computing a between-seed standard deviation. Multiple temporal splits for hazard estimation are a separate replication unit. Quantities marked as program values have no empirical sampling error unless an explicitly identified fitted input is varied. In Table 1 the $\pm$ entries give this deviation, which on Reddit-2024 is 4 steps for PREMIUM around a mean of 110 and 13 for uniform preparation around 243; Table 5 computes it over the five temporal splits, where the logistic hazard reaches an AUC of 0.71 with a deviation of 0.02 and a Kendall correlation of 0.47 with a deviation of 0.04.

B.5 Implementation Details

Training configuration. The current design instantiates a frozen Llama-3.2-1B backbone with rank-16 shared and private adapters, momentum SGD and a fixed participation trace. Tokenization, context length, local passes, learning-rate schedule, adapter target modules and the request-time optimizer belong to the run manifest. Scale and optimizer variants keep the data split, withdrawal sequence and utility-budget definition fixed. A pretraining cutoff date by itself does not establish absence of overlap; the snapshot manifest and base-model diagnostics are retained with the run.

Hazard fitting. Construct features using records available before the prediction period. Fit the logistic model on earlier event/censoring intervals and choose regularization on a later validation interval. Evaluate once on the held-out period. If intervals are shorter than the compliance period, convert their conditional event probabilities to a period probability by $1 - \prod_j (1 - p_{cj})$. A one-period logistic model directly estimates h_c in Eq. (2). The oracle, privacy and cold-start variants reuse the same temporal boundary.

Dual solve. Set zero-hazard clients to zero readiness. If η can fully prepare every positive-hazard client, set those readiness values to one and leave the unused budget unspent. Otherwise bracket the positive dual price and bisect the monotone budget residual. Return the feasible side of the bracket, and save the budget slack, price and readiness vector. Quadratic preparation uses $r_c = \min\{1, \kappa h_c/\lambda\}$; the closed-form uniform-gap expression is evaluated only before saturation.

Routing and exposure. Keep readiness fixed within each local segment. Scale the shared and private gradient channels before a linear optimizer, with consistently initialized momentum buffers;

sum their resulting updates to audit the routing identity. Adaptive optimizers require a separate realized-to-priced exposure diagnostic. When readiness changes, preserve the previous exposure ledger and account for optimizer-state carryover. Private-rank rounding is recorded explicitly and is compared at the realized, rather than nominal, preparation budget.

Deletion and repeated periods. Detach the requested private component, evaluate the shared-model forget loss, and execute the common removal procedure until the declared threshold is reached or the audit budget is exhausted. Refit hazards only at the chosen boundary and apply new readiness to future contributions. Past exposure is an additive cost term. The ledger therefore supports a direct audit of whether a lower priced cost is realized by the trained shared model.

B.6 Alternative Designs

This section records why the three stages of Section 3 take the form they do, by ruling out three designs that are simpler to implement. **(i) Allocating by data volume instead of hazard.** Client size is already known to the server and needs no survival model, so spending the budget in proportion to n_c is the cheapest selective rule available (McMahan et al., 2017). Size and withdrawal probability are different quantities: the program of Section 3.3 prices a client by the work its request will cost, which is the product of the two, so a size-proportional rule prepares large clients that never leave and leaves small frequent leavers in the shared adapter. Uniform preparation is the limiting case the comparison in Section 4 isolates. **(ii) A shared-exposure penalty instead of explicit routing.** Adding a penalty on the shared component to the local objective keeps the training loop unchanged and needs no second adapter (Li et al., 2020). A penalty biases the optimizer without fixing the share it realizes, so the readiness the program priced and the readiness the run delivers come apart, and the cost the deployer budgeted is not the cost the request pays. Proposition 7 is the property routing supplies and a penalty does not. **(iii) Detaching after training instead of during it.** Splitting a trained adapter at request time would avoid any preparation cost. Nothing is detachable that was not routed: once a client's learning has entered the shared parameters through Eq. (1), removing it is the request-time work the preparation was meant to avoid, which is the no-preparation reference of Section 4.

B.7 Algorithm

The three stages of Section 3 run as the single pipeline of Algorithm 1, once per compliance period. A deployment changes only the features the hazard model is fitted on and the budget η the deployer declares; the allocation program, the routing rule and the exposure ledger are the same in every corpus compared in Section 4.

The procedure runs once per compliance period and adds two computations to the training loop: a survival fit over the deployer's own records, and a bisection over one scalar. Phase III changes what a client does locally by one scaling per gradient and changes nothing about what the server aggregates, so an existing federated pipeline carries the rule by handing each client its readiness at the start of the period. The exposure ledger s_c is the quantity the cost of Eq. (3) is read against, and it replaces $n_c(1 - r_c)$ in Eq. (4) whenever the readiness of a client has changed across periods, so exposure that has already entered the shared adapter is carried as a constant rather than re-priced. Table 10 reports the resulting round cost: on Llama-3.2-1B with rank-16 adapters a training round takes 40 seconds and uploads 53 megabytes, and the realized exposure matches the priced exposure at a ratio of 1.00. The ratio stays at 1.00 on Llama-3.2-3B and Llama-3.1-8B, whose rounds take 116 and 280 seconds, and rounding the private share to a discrete rank gives a ratio of 0.85 at 115 steps against 110 for continuous routing. At rank 4 and rank 64 the ratio also stays at 1.00, with training rounds of 36 and 69 seconds and uploads of 13 and 209 megabytes per round.

B.8 Broader Impacts

PREMIUM lowers the cost of honoring deletion requests in federated fine-tuning, supporting compliance with regulations that grant a right to withdraw contributed data. Two indirect concerns remain: hazard estimation reads operating records that describe clients, so a deployer should restrict the features it fits to the ones it is authorized to use and disclose that a client's predicted likelihood of withdrawal shapes how its learning is routed; and a cost priced in advance invites a deployer to treat

Algorithm 1 PREMIUM: Hazard-Proportional Preparation for Unlearning

Input: clients $\mathcal{C}$ with features $\{x_c\}$ and sizes $\{n_c\}$; utility budget η; exposure cost κ; detachment cost J_0; utility shape ϕ; threshold γ; rounds per period T; learning rate α.

Initialize the shared adapter w at the server and a private adapter v_c at every client; open the exposure ledger s_c at zero
▷ *Phase I: Hazard Estimation (HE)*
Fit θ on the withdrawal events of earlier periods against their features
Return the hazard h_c for every client and the ordering π ▷ Eq. (2)
Record AUC and calibration error on a held-out period
▷ *Phase II: Budgeted Readiness Allocation (BRA)*
Bisect the dual price λ until the budget condition holds ▷ Eq. (5)
Assign every client its readiness $r_c(\lambda)$ from the readiness rule ▷ Eq. (5)
Report the expected cost and the gap Γ against uniform preparation ▷ Eq. (4), Eq. (6)
▷ *Phase III: Readiness-Proportional Training (RPT)*
for round $t = 1, 2, \ldots, T$ **do**
  for each participating client c **in parallel do**
    Take local steps by the readiness-proportional routing and upload the change in w ▷ Eq. (7)
  end for
  Aggregate the uploads and advance s_c by $(1 - r_c)$ times the samples client c used ▷ Eq. (1)
end for
if a request arrives from client c **then**
  Detach v_c; ascend $\mathcal{L}_c$ on w until it stands above γ; log the steps as the realized cost
end if
At the next period refit θ, resolve λ, and apply the new readiness to the rounds that follow
return released model (W_0, w); readiness $\{r_c\}$; dual price λ; exposure ledger $\{s_c\}$

preparation as a settled account, which logging the unlearning steps each served request takes against the priced cost keeps honest. Across hazard groups, Table 1 gives a retained-utility gap between the highest and lowest hazard quintiles of 0.020 under PREMIUM, against 0.025 under uniform preparation and 0.028 to 0.044 for the published baselines, and Figure 11 reads the deletion cost of a tighter fairness bound.

C Proofs of Theoretical Results

We expand the allocation program and its implementation identities using the notation of Section 3. The cost statements refer to the affine exposure model $J_c(r) = J_0 + \kappa n_c(1 - r)$ and the utility model $U_c(r) = n_c\phi(r)$. The final subsection states the separate conditions linking that model to request-time updates.

C.1 Assumptions and the Allocation Program

Let $n_c > 0$, $n = \sum_c n_c$, $\nu_c = n_c/n$, $h_c \geq 0$ and $\kappa > 0$. Define

$$\begin{aligned} \mathcal{R}_\eta &= \left\{ r \in [0,1]^{|\mathcal{C}|} : \sum_c n_c \phi(r_c) \leq \eta \right\}, \\ C_h(r) &= \sum_c h_c J_c(r_c) = J_0 \sum_c h_c + \kappa \sum_c n_c h_c - \kappa \sum_c n_c h_c r_c, \\ V(h, \eta) &= \min_{r \in \mathcal{R}_\eta} C_h(r). \end{aligned} \tag{10}$$

Thus minimizing cost is equivalent to maximizing the preparation benefit $F_h(r) = \kappa \sum_c n_c h_c r_c$ on the same feasible set.

Assumption 1 (Utility shape)**.** *The function ϕ is continuous, convex and increasing on $[0, 1]$, with $\phi(0) = 0$. For the differentiable water-filling rule, ϕ' is continuous and strictly increasing, and*

$\phi'(0) = 0$. *The budget satisfies* $\eta > 0$. *Clients with zero hazard use the minimum-preparation tie-break* $r_c = 0$. *The linear capacity budget is treated separately.*

Assumption 2 (Request-time ascent)**.** *Until removal is reached, each shared-adapter deletion step increases* $\mathcal{L}_c$ *by at least* $\delta > 0$. *At request time,* $[\gamma - \mathcal{L}_c[0]]_+ \le \delta(J_0 + \kappa s_c)$, *where* s_c *is the declared shared exposure.*

C.2 The Price of Preparation

Theorem 3 (Optimal allocation and hazard concavity)**.** *Under Assumption 1, every feasible readiness vector has model cost at least* $V(h, \eta)$. *The clipped water-filling rule in Eq.* (5) *attains this minimum. For fixed* η, $V(h, \eta)$ *is concave in* h; *hence* $\mathbb{E}[V(H, \eta)] \le V(\mathbb{E}[H], \eta)$ *for integrable random hazards.*

Proof. The feasible set is nonempty and compact and the objective is continuous, so a minimizer exists. Write its Lagrangian with budget multiplier $\lambda \ge 0$ and box multipliers $\mu_c, \zeta_c \ge 0$:

$$\begin{aligned}\mathcal{L}(r, \lambda, \mu, \zeta) &= J_0 \sum_c h_c + \kappa \sum_c n_c h_c (1 - r_c) \\ &\quad + \lambda \left(\sum_c n_c \phi(r_c) - \eta \right) + \sum_c \mu_c (r_c - 1) - \sum_c \zeta_c r_c, \\ \frac{\partial \mathcal{L}}{\partial r_c} &= -\kappa n_c h_c + \lambda n_c \phi'(r_c) + \mu_c - \zeta_c.\end{aligned} \tag{11}$$

A sufficiently small vector in $(0, 1)^{|\mathcal{C}|}$ satisfies the budget strictly, by continuity at zero and $\eta > 0$. Slater's condition therefore holds, and the KKT conditions are sufficient as well as necessary (Boyd & Vandenberghe, 2004).

When $0 < r_c < 1$, complementary slackness sets $\mu_c = \zeta_c = 0$. At the two endpoints the stationarity inequalities yield

$$\begin{aligned} 0 < r_c < 1 &\implies \phi'(r_c) = \frac{\kappa h_c}{\lambda}, \\ r_c = 1 &\implies \mu_c = n_c(\kappa h_c - \lambda \phi'(1)) \ge 0, \\ r_c = 0,\ h_c > 0 &\implies \zeta_c = -\kappa n_c h_c < 0, \end{aligned} \tag{12}$$

so the last case is impossible for a finite positive price. Define the inverse derivative with saturation at 1. The optimal readiness is

$$r_c(\lambda) = \begin{cases} 0, & h_c = 0, \\ (\phi')^{-1}(\kappa h_c / \lambda), & 0 < \kappa h_c < \lambda \phi'(1), \\ 1, & \kappa h_c \ge \lambda \phi'(1). \end{cases} \tag{13}$$

If $0 < \eta < \sum_{h_c > 0} n_c \phi(1)$, some positive-hazard client is unsaturated and complementary slackness gives

$$\sum_c n_c \phi(r_c(\lambda)) = \eta. \tag{14}$$

Its left-hand side is continuous and decreases from full positive-hazard preparation to zero as λ increases; it is strictly decreasing wherever an active client is unsaturated. Hence the positive price is determined by the budget. If the budget covers every positive-hazard client, set their readiness to 1, leave zero-hazard clients at 0, and use $\lambda = 0$. If all hazards vanish, $r = 0$ attains value zero.

For concavity, let r_s minimize cost at $sh' + (1 - s)h''$, with $0 \le s \le 1$. Linearity in the hazards gives

$$\begin{aligned} V(sh' + (1-s)h'', \eta) &= \sum_c \big(sh'_c + (1-s)h''_c\big) J_c(r_{s,c}) \\ &= sC_{h'}(r_s) + (1-s)C_{h''}(r_s) \\ &\ge sV(h', \eta) + (1-s)V(h'', \eta). \end{aligned} \tag{15}$$

The same feasible set is used on both sides. Jensen's inequality completes the expectation claim. It compares reoptimized allocations after the hazard vector is known; it does not assign that benefit to a fixed allocation facing unobserved uncertainty. □

C.3 The Price of Uniform Preparation

Set $S_1 = \sum_c n_c h_c$, $S_2 = \sum_c n_c h_c^2$, and suppose $\phi(r) = r^2/2$ with $S_2 > 0$.

Theorem 4 (Unsaturated uniform gap)**.** *If $h_{\max}\sqrt{2\eta/S_2} < 1$, the optimal allocation is unsaturated and*

$$\begin{aligned} \lambda &= \kappa\sqrt{S_2/(2\eta)}, & r_c^* &= h_c\sqrt{2\eta/S_2}, \\ V(h,\eta) &= J_0 \sum_c h_c + \kappa S_1 - \kappa\sqrt{2\eta S_2}, \\ V^{\mathrm{u}}(h,\eta) &= J_0 \sum_c h_c + \kappa S_1 - \kappa\sqrt{2\eta/n}\, S_1. \end{aligned} \tag{16}$$

The gap is zero exactly at constant hazards, increases with weighted hazard variance at fixed weighted mean, and is at most $\kappa\sqrt{2\eta n\ \mathrm{Var}_\nu(h)}$.

Proof. Since $\phi'(r) = r$, stationarity and the tight budget imply

$$\begin{aligned} \eta &= \frac{1}{2}\sum_c n_c (r_c^*)^2 = \frac{1}{2}\sum_c n_c \left(\frac{\kappa h_c}{\lambda}\right)^2 = \frac{\kappa^2 S_2}{2\lambda^2}, \\ \lambda &= \kappa\sqrt{\frac{S_2}{2\eta}}, \qquad r_c^* = \frac{\kappa h_c}{\lambda} = h_c\sqrt{\frac{2\eta}{S_2}}. \end{aligned} \tag{17}$$

The hypothesis is precisely $\max_c r_c^* < 1$. Substitution into the full objective gives

$$\begin{aligned} V(h,\eta) &= \sum_c h_c \left[J_0 + \kappa n_c \left(1 - \frac{\kappa h_c}{\lambda}\right)\right] \\ &= J_0 \sum_c h_c + \kappa S_1 - \frac{\kappa^2}{\lambda}\sum_c n_c h_c^2 \\ &= J_0 \sum_c h_c + \kappa S_1 - \kappa\sqrt{2\eta S_2}. \end{aligned} \tag{18}$$

Uniform preparation spends the same budget at $\bar{r} = \sqrt{2\eta/n} < 1$. Therefore

$$\begin{aligned} V^{\mathrm{u}}(h,\eta) &= \sum_c h_c[J_0 + \kappa n_c(1-\bar{r})] \\ &= J_0 \sum_c h_c + \kappa S_1 - \kappa\bar{r} S_1 \\ &= J_0 \sum_c h_c + \kappa S_1 - \kappa\sqrt{2\eta/n}\, S_1. \end{aligned} \tag{19}$$

Write $m = \mathbb{E}_\nu[h]$, $q = \mathbb{E}_\nu[h^2]$ and $\sigma^2 = q - m^2$. Using $S_1 = nm$ and $S_2 = nq$, subtract and rationalize:

$$\begin{aligned} \Gamma = V^{\mathrm{u}}(h,\eta) - V(h,\eta) &= \kappa\sqrt{2\eta S_2} - \kappa\sqrt{2\eta/n}\, S_1 \\ &= \kappa\sqrt{2\eta n}\,(\sqrt{q} - m) = \kappa\sqrt{2\eta n}\,\frac{q - m^2}{\sqrt{q} + m} \\ &= \kappa\sqrt{2\eta n}\,\frac{\sigma^2}{\sqrt{m^2+\sigma^2} + m}. \end{aligned} \tag{20}$$

The denominator is positive when $S_2 > 0$. Thus $\Gamma = 0$ exactly when $\sigma^2 = 0$, which means all hazards are equal because every $\nu_c > 0$. At fixed m,

$$\frac{\partial\Gamma}{\partial\sigma^2} = \frac{\kappa\sqrt{2\eta n}}{2\sqrt{m^2+\sigma^2}} > 0$$

within the unsaturated regime. Finally,

$$\Gamma = \kappa\sqrt{2\eta n}\frac{\sigma^2}{\sqrt{m^2+\sigma^2}+m} \le \kappa\sqrt{2\eta n}\frac{\sigma^2}{\sigma} = \kappa\sqrt{2\eta n}\,\sigma \tag{21}$$

for $\sigma > 0$; the zero-variance case follows directly. All-zero hazards give zero cost and gap without using the rationalized expression. □

C.4 Rank and Value Robustness

Let r^* and $\widehat{r}$ minimize cost under h and $\widehat{h}$ on a common feasible set. Define

$$\begin{aligned}\mathrm{Reg} &= C_h(\widehat{r}) - C_h(r^*) = F_h(r^*) - F_h(\widehat{r}) \\ &= \kappa \sum_c n_c h_c (r_c^* - \widehat{r}_c) \geq 0.\end{aligned} \tag{22}$$

Theorem 5 (Rank robustness)**.** *Under the capacity constraint $\sum_c n_c r_c \leq B$, an optimal allocation depends on hazards only through their order, up to ties. If k adjacent transpositions connect the true and estimated orders, then*

$$\mathrm{Reg} \leq \kappa n_{\max} \sum_{j=1}^{k} |h_{a_j} - h_{b_j}| \leq k\kappa n_{\max}(h_{\max} - h_{\min}).$$

For any common feasible readiness set,

$$\mathrm{Reg} \leq \kappa \sum_c n_c |h_c - \widehat{h}_c|. \tag{23}$$

Proof. Put $x_c = n_c r_c$. The capacity problem becomes

$$\max \left\{ \kappa \sum_c h_c x_c : 0 \leq x_c \leq n_c,\ \sum_c x_c \leq B \right\}.$$

If $h_a > h_b$ but $x_a < n_a$ and $x_b > 0$, move $\epsilon = \min(n_a - x_a, x_b)$ units from b to a. The benefit difference expands as

$$F_h(x + \epsilon e_a - \epsilon e_b) - F_h(x) = \kappa[h_a \epsilon - h_b \epsilon] = \kappa\epsilon(h_a - h_b) \geq 0. \tag{24}$$

Repeated exchanges fill clients in descending hazard order, with at most one partial client. An adjacent swap changes allocation only within its two-client block. At most $\min(n_a, n_b) \leq n_{\max}$ units move between them, so its absolute cost change is at most $\kappa n_{\max}|h_a - h_b|$. Telescoping along the sequence of permutations and applying the triangle inequality proves the rank bound. The pair gaps are those of the intermediate swaps, not necessarily neighboring gaps in the final sorted order.

For two equal-sized clients of size m and capacity m, reversing $h_1 > h_2$ exchanges allocations $(1, 0)$ and $(0, 1)$:

$$\mathrm{Reg} = \kappa m[h_1(1-0) + h_2(0-1)] = \kappa m(h_1 - h_2). \tag{25}$$

For the value-error claim, add and subtract the estimated benefit:

$$\begin{aligned}\mathrm{Reg} &= F_h(r^*) - F_h(\widehat{r}) \\ &= [F_h(r^*) - F_{\widehat{h}}(r^*)] + [F_{\widehat{h}}(r^*) - F_{\widehat{h}}(\widehat{r})] + [F_{\widehat{h}}(\widehat{r}) - F_h(\widehat{r})] \\ &\leq \kappa \sum_c n_c (h_c - \widehat{h}_c)(r_c^* - \widehat{r}_c) \\ &\leq \kappa \sum_c n_c |h_c - \widehat{h}_c|\, |r_c^* - \widehat{r}_c| \leq \kappa \sum_c n_c |h_c - \widehat{h}_c|.\end{aligned} \tag{26}$$

The middle estimated-benefit difference is nonpositive by optimality of $\widehat{r}$. The last step uses $r_c^*, \widehat{r}_c \in [0, 1]$. Recomputing the dual price at $\widehat{h}$ preserves feasibility and is fully allowed in this argument. □

C.5 Limiting Hazard Profiles

Proposition 6 (Degeneracy)**.** *Constant positive hazards give uniform readiness under the common utility shape. Binary hazards under a capacity budget covering every positive-hazard client give full preparation to those clients and none to the rest. All-zero hazards give zero readiness under the tie-break.*

Proof. For $h_c = h > 0$, the common price gives

$$r_c^* = \min\{1, (\phi')^{-1}(\kappa h/\lambda)\} = \bar{r} \quad \text{for every } c, \qquad \sum_c n_c \phi(\bar{r}) = n\phi(\bar{r}).$$

If the full-preparation budget is available, $\bar{r} = 1$. In the unsaturated quadratic case the variance is zero, so Eq. (20) gives $\Gamma = 0$.

For $h_c \in \{0, 1\}$, let $\mathcal{H} = \{c : h_c = 1\}$. If $B \geq \sum_{c\in\mathcal{H}} n_c$, the readiness $r_c = \mathbb{I}[c \in \mathcal{H}]$ is feasible and

$$F_h(r) = \kappa \sum_{c\in\mathcal{H}} n_c = \max_{0\leq z_c\leq 1} \kappa \sum_{c\in\mathcal{H}} n_c z_c.$$

Zero-hazard readiness adds no benefit, so the tie-break sets it to zero. If $\mathcal{H} = \varnothing$, then $C_h(r) = 0$ for every feasible r and the same tie-break gives $r = 0$. □

C.6 Realization in Training

Proposition 7 (Exposure identity)**.** *At fixed readiness, plain SGD preserves the factors $1 - r_c$ and r_c in the accumulated shared and private updates. Momentum preserves them on its filtered gradient sequence when its initial buffers are zero or scaled consistently.*

Proof. For local gradients $g_c[k]$ and $\widetilde{g}_c[k]$ evaluated along the routed trajectory, sum the actual updates:

$$\begin{aligned} \Delta_c &= \sum_{k=1}^{K} (w_c[k] - w_c[k-1]) = \sum_{k=1}^{K} [-\alpha(1 - r_c) g_c[k]] = -\alpha(1 - r_c) \sum_{k=1}^{K} g_c[k], \\ v_c[K] - v_c[0] &= \sum_{k=1}^{K} [-\alpha r_c \widetilde{g}_c[k]] = -\alpha r_c \sum_{k=1}^{K} \widetilde{g}_c[k]. \end{aligned} \tag{27}$$

For momentum coefficient $0 \leq \beta < 1$, the shared buffer satisfies $b_k = \beta b_{k-1} + (1 - r_c) g_c[k]$. Unrolling and reversing the finite sums gives

$$\begin{aligned} b_k &= \beta^k b_0 + (1 - r_c) \sum_{j=1}^{k} \beta^{k-j} g_c[j], \\ \Delta_c &= -\alpha \sum_{k=1}^{K} b_k \\ &= -\alpha \frac{\beta(1 - \beta^K)}{1 - \beta} b_0 - \alpha(1 - r_c) \sum_{j=1}^{K} \left(\sum_{k=j}^{K} \beta^{k-j} \right) g_c[j] \\ &= -\alpha \frac{\beta(1 - \beta^K)}{1 - \beta} b_0 - \alpha(1 - r_c) \sum_{j=1}^{K} \frac{1 - \beta^{K-j+1}}{1 - \beta} g_c[j]. \end{aligned} \tag{28}$$

With $b_0 = 0$ the factor is explicit. With $b_0 = (1 - r_c)\bar{b}_0$ it factors from the initial-buffer term as well. The private-buffer derivation replaces $1 - r_c$ by r_c. If readiness is refreshed, this identity applies within each fixed-readiness segment; an existing buffer must be rescaled or reset consistently at the boundary. Gradient-dependent normalization does not obey this linear-filter identity. □

Proposition 8 (Deletion completeness)**.** *Removing a private adapter changes no parameter of the released shared model. Under Assumption 2, the first shared-adapter step meeting the loss threshold is at most $\lceil J_0 + \kappa s_c \rceil$.*

Proof. The released parameters consist of W_0 and w; v_c is excluded from their aggregation. Detaching v_c therefore leaves (W_0, w) unchanged. Let m_c be the first deletion step with $\mathcal{L}_c[m_c] \geq \gamma$.

Before that step,

$$\begin{aligned}\mathcal{L}_c[m] - \mathcal{L}_c[0] &= \sum_{j=0}^{m-1} (\mathcal{L}_c[j+1] - \mathcal{L}_c[j]) \\ &\geq \sum_{j=0}^{m-1} \delta = m\delta.\end{aligned} \tag{29}$$

If the starting loss already meets the threshold, $m_c = 0$. Otherwise taking the first integer with $m\delta \geq \gamma - \mathcal{L}_c[0]$ yields

$$\begin{aligned}m_c &\leq \left\lceil \frac{[\gamma - \mathcal{L}_c[0]]_+}{\delta} \right\rceil \\ &\leq \left\lceil \frac{\delta(J_0 + \kappa s_c)}{\delta} \right\rceil = \lceil J_0 + \kappa s_c \rceil.\end{aligned} \tag{30}$$

For $s_c = n_c(1 - r_c)$ this is the integer envelope of the affine cost model. Since $0 \leq \lceil x \rceil - x < 1$, rounding adds less than one step per request, or less than $\sum_c h_c$ to the hazard-weighted total. The ascent condition, rather than the routing identity alone, supplies the link to threshold completion. □

D Audit Protocol

D.1 Temporal and Data Commitments

Freeze snapshot identifiers, client grouping, event definitions and temporal boundaries before fitting hazards. Store the population eligible at the start of each interval, the observed withdrawal event and any censoring time. A missed round remains an availability event. Feature extraction uses only the history preceding the prediction boundary. The audit records unmatched opt-out identifiers and the number of clients excluded by each eligibility rule.

For every training seed, save the selected clients, local sample identifiers, participation schedule and matched reference groups. The same request sets are used across preparation rules. An observed withdrawal sequence and a sequence drawn from a controlled hazard profile have distinct identifiers, so the data-driven prediction test remains separate from the allocation stress test.

D.2 Budget and Allocation Audit

For fitted hazards $\widehat{h}$, recompute the returned readiness vector and verify

$$0 \leq r_c \leq 1, \qquad \sum_c n_c \phi(r_c) \leq \eta, \qquad \lambda \left(\sum_c n_c \phi(r_c) - \eta \right) = 0. \tag{31}$$

On interior coordinates the stationarity residual is $|\lambda \phi'(r_c) - \kappa \widehat{h}_c|$. At zero and one use the corresponding one-sided KKT inequalities. Report numerical residuals and any unused budget. Zero-hazard and fully saturated cases are solved directly, without dividing by a zero dual price.

Compute the program value from the same hazards, sample weights and exposure-cost calibration. In controlled studies the true hazards define regret; fitted-hazard cost predictions are evaluated separately against observed held-out events. The unsaturated quadratic gap is checked only when every optimal readiness is below one. Saturated cases use the solved program value and the feasible uniform allocation.

D.3 Training and Deletion Audit

At each local segment, record readiness, processed sample count, shared and private update norms, and the optimizer state used to produce them. The cumulative exposure ledger is the sum of segment-specific routed contributions. A readiness change applies to future segments; previously uploaded contributions remain in the ledger. Compare the realized update with the identity for the declared optimizer, rather than equating a gradient multiplier with an adaptive optimizer's final parameter displacement.

At a request, record the private-component detachment, initial shared-model forget loss, threshold, update count and post-deletion retained utility. Audit extraction and membership inference on fixed reserved records. If the threshold is not reached within the limit, report a censored request and its final margin. A smaller update count is interpreted alongside deletion quality and retention, preventing an early stop from being counted as a cost improvement.

D.4 Uncertainty and Resource Accounting

For each method, first aggregate its paired request costs within an independently trained seed. Compute uncertainty across those seed-level summaries. Temporal hazard splits are summarized separately, and repeated requests on one checkpoint do not become additional training seeds. Report interval estimates for the paired cost difference, fairness gap and realized-minus-priced cost. The program optimum is deterministic given its declared inputs.

Charge hazard fitting, readiness allocation, shared/private training, stored checkpoints and request execution to separate ledger entries. Communication counts uploaded and downloaded bytes, including metadata. Runtime distinguishes wall-clock latency from total accelerator time. These records let an auditor reconstruct the cost–utility comparison at the same budget and identify whether a saving comes from allocation, training realization or a changed deletion protocol.

E Additional Experiments

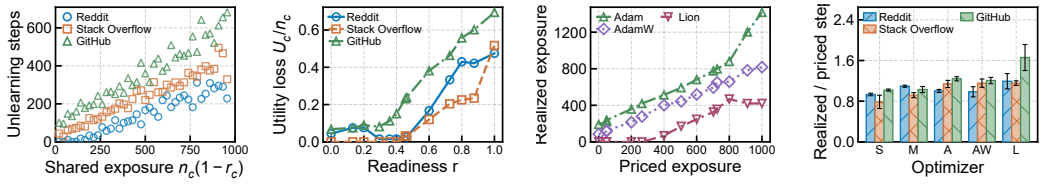
Each extended comparison is placed beside the allocation, prediction or routing question it examines.

E.1 Cost Calibration and Theoretical Diagnostics

Figure 7 separates the assumed exposure-cost relation, the utility-cost calibration and the realized optimizer updates. Figure 8 compares uniform excess cost and allocation regret across the declared error and dispersion sweeps. The closed-form quadratic gap applies in the unsaturated regime; the solved program supplies the reference after any client reaches readiness one.

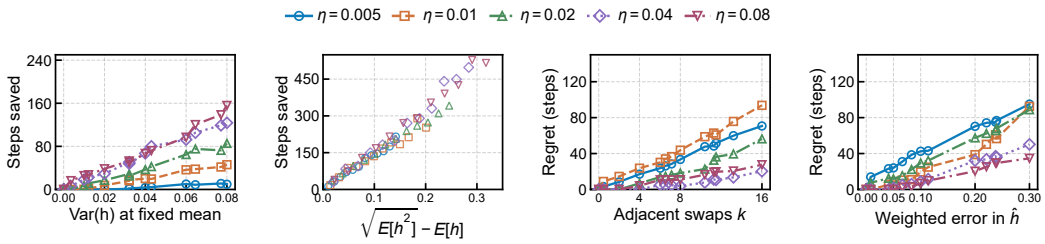
The panels expose two links in the argument. Shared exposure increases deletion work, and stronger private routing consumes more of the utility budget. The optimizer panels then ask whether a nominal readiness produces the intended update scaling. Momentum SGD follows a linear gradient filter with consistent state initialization, whereas adaptive normalization can alter the relationship between the scaled gradient and the final update. A good allocation under the nominal model consequently requires a realization audit at the optimizer level. In Figure 7(a), GitHub needs the most unlearning steps and Reddit the fewest across the shared-exposure range. In panel (b), utility loss stays near zero at low readiness, and GitHub is the first corpus to rise and the highest at full readiness.

The gap curves spread as hazard variance grows, and the transformed horizontal axis aligns that change with the expression in Eq. (6). The two regret panels distinguish ordering from value error. A capacity budget changes only when an ordering error alters which clients receive capacity; a quadratic budget also changes continuously with estimated probability. These different responses explain why one predictor summary cannot characterize both regimes.



(a) Exposure cost. (b) Utility cost. (c) Exposure identity. (d) Deletion-cost ratio.

Figure 7: Cost-model diagnostics: deletion cost, utility cost and optimizer exposure; bars show mean and SD across three runs. S, M, A, AW and L denote SGD, momentum SGD, Adam, AdamW and Lion.



(a) Hazard variance. (b) Gap transformation. (c) Ordering error. (d) Value error.

Figure 8: Uniform-preparation gap and estimator regret as hazard dispersion, ordering error and value error change.

E.2 Complete Comparison across Hazard Profiles

Table 4 retains all three hazard profiles. Widening or narrowing changes the dispersion at a fixed weighted mean; the comparison therefore separates allocation from the number of expected requests. Read Steps with FQ, MIA and retained utility for every profile. The fitted-profile rows also appear in the main text to keep the central comparison self-contained.

Table 4: Main comparison on Reddit-2024 across fitted, widened and narrowed hazards; 20 requests per seed, three seeds. Steps: unlearning updates; Util.: retained next-token accuracy; FQ: one minus extraction recall; MIA: LiRA AUC; Gap: retained-utility difference between the highest and lowest hazard quintiles; Regret: expected cost above the program value. Best in red, runner-up in blue; the reference is excluded from ranking.

Method	Steps↓	Util.↑	FQ↑	MIA↓	Gap↓	Regret↓
Fitted hazards (Gini 0.57), held-out withdrawals						
SISA [S&P'21] (Bourtoule et al., 2021)	422(±12)	0.606(±0.037)	0.72(±0.04)	0.53(±0.01)	0.032(±0.001)	321(±12)
Exact-Fun [ICDM'23] (Xiong et al., 2023)	362(±1)	0.649(±0.012)	**0.82**(±0.05)	**0.52**(±0.01)	0.029(±0.001)	261(±2)
FedShard [AAAI'26] (Wen et al., 2026)	349(±12)	0.677(±0.020)	0.77(±0.01)	**0.52**(±0.01)	0.028(±0.002)	248(±12)
DADS [ICML'24] (Sim et al., 2024)	492(±21)	0.637(±0.012)	0.68(±0.02)	0.57(±0.02)	0.044(±0.003)	391(±20)
MemSinks [ICML'25] (Ghosal et al., 2025)	272(±11)	**0.688**(±0.033)	0.73(±0.01)	0.54(±0.01)	0.029(±0.0003)	171(±10)
System-Aware [ICML'25] (Lu et al., 2025)	315(±11)	0.653(±0.026)	0.71(±0.03)	0.56(±0.02)	0.035(±0.001)	214(±12)
Uniform preparation	243(±13)	0.659(±0.056)	0.71(±0.04)	0.59(±0.04)	0.025(±0.0005)	142(±13)
PREMIUM	**110**(±4)	0.649(±0.016)	0.74(±0.07)	0.53(±0.01)	**0.020**(±0.001)	**9**(±4)
No preparation	631(±14)	0.672(±0.089)	0.69(±0.01)	0.54(±0.02)	0.000(±0.000)	530(±13)
Widened (Gini 0.71), requests drawn						
SISA [S&P'21] (Bourtoule et al., 2021)	504(±21)	0.623(±0.013)	0.69(±0.01)	**0.53**(±0.03)	0.040(±0.003)	430(±21)
Exact-Fun [ICDM'23] (Xiong et al., 2023)	349(±5)	0.640(±0.019)	0.75(±0.07)	0.55(±0.02)	0.037(±0.001)	275(±5)
FedShard [AAAI'26] (Wen et al., 2026)	356(±5)	0.649(±0.032)	0.75(±0.03)	0.54(±0.02)	0.032(±0.001)	282(±6)
DADS [ICML'24] (Sim et al., 2024)	540(±10)	0.675(±0.027)	0.64(±0.03)	0.58(±0.04)	0.053(±0.002)	466(±11)
MemSinks [ICML'25] (Ghosal et al., 2025)	309(±4)	**0.683**(±0.011)	0.68(±0.05)	0.57(±0.04)	0.037(±0.0003)	235(±5)
System-Aware [ICML'25] (Lu et al., 2025)	383(±16)	0.605(±0.005)	0.71(±0.04)	0.54(±0.02)	0.042(±0.002)	309(±17)
Uniform preparation	230(±2)	0.618(±0.027)	0.75(±0.06)	0.54(±0.01)	0.031(±0.001)	156(±1)
PREMIUM	**82**(±2)	0.654(±0.019)	**0.76**(±0.01)	**0.53**(±0.01)	**0.024**(±0.0005)	**8**(±2)
No preparation	570(±29)	0.686(±0.017)	0.67(±0.02)	0.60(±0.01)	0.000(±0.000)	496(±30)
Narrowed (Gini 0.12), requests drawn						
SISA [S&P'21] (Bourtoule et al., 2021)	445(±26)	0.632(±0.033)	0.70(±0.02)	0.53(±0.03)	0.003(±0.002)	209(±31)
Exact-Fun [ICDM'23] (Xiong et al., 2023)	400(±7)	0.635(±0.032)	**0.79**(±0.02)	0.54(±0.02)	0.004(±0.002)	164(±3)
FedShard [AAAI'26] (Wen et al., 2026)	303(±5)	0.623(±0.019)	0.77(±0.05)	0.54(±0.04)	0.004(±0.001)	67(±13)
DADS [ICML'24] (Sim et al., 2024)	585(±14)	**0.654**(±0.021)	0.66(±0.04)	0.55(±0.01)	0.005(±0.001)	349(±17)
MemSinks [ICML'25] (Ghosal et al., 2025)	306(±24)	0.595(±0.072)	0.73(±0.01)	0.53(±0.02)	0.003(±0.0005)	70(±34)
System-Aware [ICML'25] (Lu et al., 2025)	337(±8)	0.621(±0.028)	0.76(±0.02)	0.53(±0.02)	0.004(±0.001)	101(±15)
Uniform preparation	240(±10)	0.632(±0.026)	0.75(±0.07)	0.53(±0.01)	0.003(±0.001)	4(±0.1)
PREMIUM	**238**(±11)	0.619(±0.016)	0.78(±0.02)	**0.51**(±0.004)	**0.001**(±0.0004)	**2**(±1)
No preparation	688(±35)	0.682(±0.049)	0.67(±0.01)	0.58(±0.02)	0.000(±0.000)	452(±32)

The three profiles separate the effect of preparation from its allocation. The no-preparation reference keeps all learning in shared parameters, while every preparation rule pays some utility cost to reduce the residue. Within that prepared group, the advantage of hazard weighting is most visible when a small part of the federation accounts for much of the expected demand. When hazards narrow, similar clients receive similar readiness and the uniform rule approaches the optimized allocation. The comparison therefore identifies the mechanism precisely: heterogeneous demand is what creates a reason to allocate unequally. The retained-utility and forgetting columns carry their own ordering

and are read alongside the cost. Under widened hazards PREMIUM serves a request in 82 steps with Regret 8, against 230 steps and Regret 156 for uniform preparation, and its FQ of 0.76 and MIA of 0.53 are the best readings in that band. Under narrowed hazards PREMIUM and uniform preparation reach Regret 2 and 4, and PREMIUM holds the best-ranked Gap, 0.001.

E.3 Hazard Prediction and Estimation Error

Table 5 pairs prediction metrics with the cost of the resulting allocation. AUC and rank correlation describe ordering, while Brier score tests the probability values consumed by a quadratic budget. Population sizes describe the eligible hazard-fitting users, distinct from the selected training clients. An oracle supplies a reference for the controlled profile rather than a fitted competitor. The logistic model reaches an AUC of 0.71 on Reddit-2024, 0.67 on StackOverflow-2024 and 0.81 on GitHub, with Kendall correlations of 0.47, 0.43 and 0.52 and top-decile lifts of 3.0, 2.4 and 3.0. Gradient-boosted trees attain the highest fitted AUC on every corpus, at 0.77, 0.72 and 0.89. The oracle allocation needs 101 steps on Reddit-2024, 139 on StackOverflow-2024 and 86 on GitHub, each at zero Regret, and the constant hazard sits at an AUC of 0.50 with 50 adjacent swaps.

Predictor quality and deletion cost need not have identical rankings. A predictor can improve discrimination while changing few readiness values near the budget's active region. Conversely, a probability error concentrated among high-exposure clients can move the allocation even when aggregate AUC barely changes. The table pairs prediction and cost for this reason. The constant-hazard row removes selectivity altogether, while the fitted models test whether available signals place preparation on clients whose withdrawals are more likely. The difference from the oracle includes both imperfect probability estimates and the constraints of the realized training procedure. Reddit-2024 shows this directly: gradient-boosted trees lead or tie on AUC, Kendall correlation, adjacent swaps, Brier score and lift, yet their allocation takes 143 steps at a Regret of 42, while the logistic model takes 110 steps at a Regret of 9 and the discrete-time Cox model 124 steps at a Regret of 23. GitHub shows the same split, with the logistic model at 90 steps and Regret 4 against 104 steps and Regret 18 for gradient-boosted trees, whose AUC of 0.89 is the highest fitted value in the table. On StackOverflow-2024 the cost order of the fitted models matches their AUC order, and gradient-boosted trees reach the lowest fitted cost there, 143 steps with Regret 4, followed by the discrete-time Cox model at 145 steps with Regret 6.

Table 5: Hazard prediction over five temporal splits, with 100 stratified clients per corpus used for allocation. N: users used to fit the hazard model; Gini: hazard concentration; AUC: withdrawal-prediction ROC area; τ: Kendall rank correlation; k: adjacent ordering swaps; Brier: probability error; Lift: lift in the top 10%. Steps and Regret follow Table 1.

Model	N	Gini↑	AUC↑	τ↑	k ↓	Brier↓	Lift↑	Steps↓	Regret↓
Reddit-2024 (account / comment deletion between snapshots)									
Logistic (Eq. hazard)	48,200	0.56	0.71(±0.02)	0.47(±0.04)	12(±1)	0.172(±0.011)	3.0(±0.2)	**110**(±4)	**9**(±4)
Discrete-time Cox		0.57	0.74(±0.05)	0.46(±0.03)	**10**(±0.4)	**0.164**(±0.008)	2.7(±0.1)	124(±7)	23(±7)
Gradient-boosted trees		**0.62**	**0.77**(±0.02)	**0.50**(±0.05)	**10**(±0.5)	**0.164**(±0.010)	**3.2**(±0.1)	143(±7)	42(±8)
Constant hazard		0.00	0.50	0.00	50	0.207	1.0	243(±13)	142
Oracle hazards		0.59	1.00	1.00	0	0.000	3.4	101(±1)	0
StackOverflow-2024 (user deletion between dumps)									
Logistic (Eq. hazard)	31,600	**0.50**	0.67(±0.04)	0.43(±0.03)	16(±1)	0.192(±0.009)	2.4(±0.04)	156(±6)	17(±6)
Discrete-time Cox		0.49	0.69(±0.05)	0.43(±0.02)	16(±1)	**0.178**(±0.017)	2.1(±0.04)	145(±4)	6(±3)
Gradient-boosted trees		**0.50**	**0.72**(±0.02)	**0.48**(±0.02)	**15**(±1)	0.195(±0.009)	**2.9**(±0.1)	**143**(±1)	**4**(±2)
Constant hazard		0.00	0.50	0.00	50	0.223	1.0	249(±13)	110
Oracle hazards		0.47	1.00	1.00	0	0.000	2.9	139(±1)	0
GitHub · The Stack opt-out (public opt-out requests)									
Logistic (Eq. hazard)	22,900	0.65	0.81(±0.03)	0.52(±0.02)	9(±1)	**0.153**(±0.002)	3.0(±0.1)	**90**(±5)	**4**(±1)
Discrete-time Cox		**0.69**	0.76(±0.11)	0.50(±0.01)	9(±0.2)	**0.153**(±0.008)	2.9(±0.1)	116(±5)	30(±7)
Gradient-boosted trees		0.66	**0.89**(±0.04)	**0.59**(±0.04)	**7**(±2)	0.156(±0.013)	**3.7**(±0.2)	104(±2)	18(±7)
Constant hazard		0.00	0.50	0.00	50	0.198	1.0	219(±22)	133
Oracle hazards		0.68	1.00	1.00	0	0.000	3.9	86(±6)	0

E.4 Limiting Cases and Full Component Ablation

Table 6 evaluates equal, binary, zero, two-level and fitted hazard profiles. Binary allocation uses a capacity budget; the continuous profiles use the declared convex utility cost. Table 7 exposes the corresponding representative allocations. Table 2 in the main text removes HE, BRA and RPT across all three corpora, retaining both the outcome metrics and the allocation diagnostics.

Table 6: Limiting hazards in a 20-client federation. Each row is one hazard profile: P is PREMIUM and R the comparison rule named beside the profile. D_∞ is the largest readiness difference between PREMIUM and the reference rule; Γ is the cost gap that uniform preparation leaves at the same budget. SISA family includes Exact-Fun and FedShard. Binary hazards use a capacity budget, and two-level hazards use a quadratic budget. Readiness vectors appear in Table 7.

Profile	Rule	$D_\infty \downarrow$	Steps↓		Util.↑		Γ
			P	R	P	R	
Equal ($h = 0.3$)	Mem. Sinks	0.000	**211**(±7)	**211**(±7)	**0.628**(±0.028)	**0.628**(±0.028)	0
Binary ($h \in \{0, 1\}$)	SISA [S&P'21] (Bourtoule et al., 2021) family	0.000	**34**(±1)	**34**(±1)	**0.754**(±0.019)	**0.754**(±0.019)	260
Zero ($h = 0$)	No preparation	0.000	**652**(±8)	**652**(±8)	**0.571**(±0.015)	**0.571**(±0.015)	0
Two-level ($h \in \{0.05, 0.6\}$)	Binary split	0.072(±0.001)	**135**(±5)	146(±7)	0.589(±0.019)	**0.645**(±0.018)	96
Fitted Reddit-2024	Reference	0.142(±0.003)	**110**(±4)	243(±13)	0.649(±0.016)	**0.659**(±0.056)	133

Equal hazards collapse the allocation onto uniform readiness, while a zero-hazard population gives no reason to spend a preparation budget. The binary case is different: with sufficient capacity, readiness can fully isolate the clients known to withdraw. Between these limits, a two-level or fitted profile produces intermediate readiness values. This progression is the role of the limiting-case comparison; it tests where the continuous rule reduces to a simpler design and where selective preparation remains useful.

Table 7: Client readiness under the five profiles in Table 6. Each r_i is the private preparation share assigned to representative client i.

Profile	r_1	r_2	r_3	r_4	r_5	r_6
Equal ($h = 0.3$)	0.42	0.42	0.42	0.42	0.42	0.42
Binary ($h \in \{0, 1\}$)	1.00	0.00	1.00	0.00	0.00	1.00
Zero ($h = 0$)	0.00	0.00	0.00	0.00	0.00	0.00
Two-level	0.07	0.84	0.07	0.84	0.07	0.84
Fitted	0.03	0.11	0.28	0.46	0.77	1.00

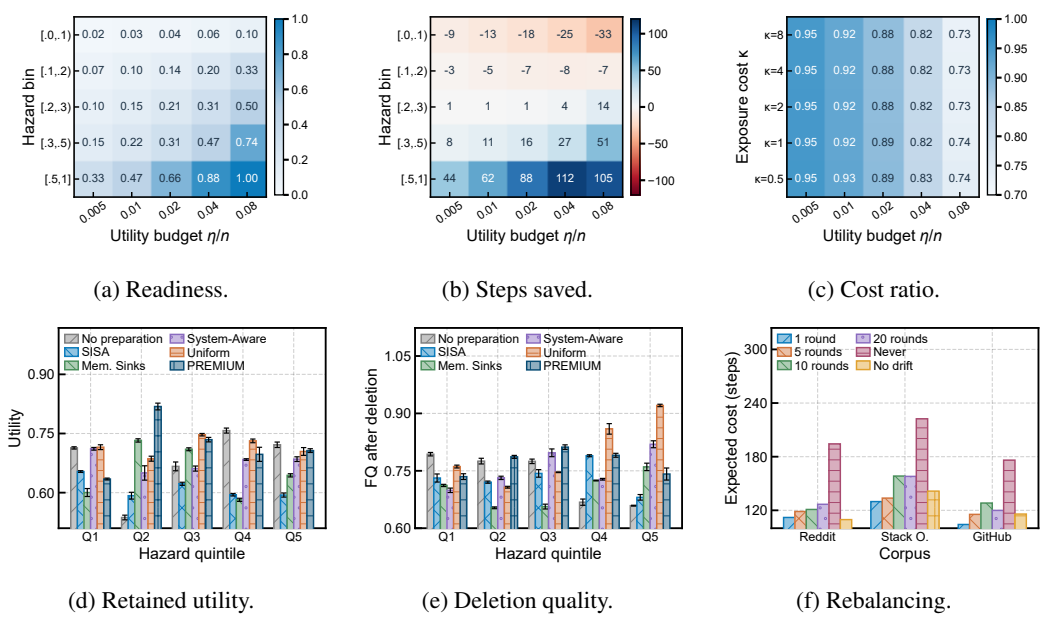
The full ablation separates a good prediction from an executed allocation. A stage can improve its immediate intermediate quantity without improving request cost if the next stage ignores it. Removing RPT is the clearest version of this effect: the saved readiness vector may still look selective even while training writes the corresponding information into the shared model. With RPT disabled, both adapters enter ordinary aggregation, including when HE is also disabled. Reading one configuration across the three corpora keeps these interactions visible instead of imposing one universal ablation order. In Table 2, every configuration with RPT disabled reads 589 steps and Regret 488 on Reddit-2024, 676 steps and Regret 537 on StackOverflow-2024 and 638 steps and Regret 552 on GitHub, whichever of HE and BRA remains active, against 110, 156 and 90 steps for the complete configuration. Retained utility moves in the other direction on the first two corpora, from 0.649 to 0.668 and from 0.592 to 0.606, and from 0.420 to 0.413 on GitHub.

E.5 Budget, Hazard and Rebalancing Sensitivity

Figure 9 reads readiness and cost over the budget and hazard axes, then checks outcomes across hazard quintiles and under drift. Its allocation heatmaps are program diagnostics; empirical utility and deletion-quality panels require the corresponding training runs. The temporal comparison in Figure 10 distinguishes refitting frequency from cold-start policies. Previously accumulated exposure is fixed while the new readiness governs future rounds.

The readiness heatmap concentrates preparation in high-hazard clients first. More budget expands that prepared group, and saturation limits the additional gain at the top end. The cost-ratio panel reads the same allocation against uniform preparation, making the useful part of the budget range visible. Quintile-level utility and deletion quality then check whether aggregate savings are concentrated at the expense of one group.

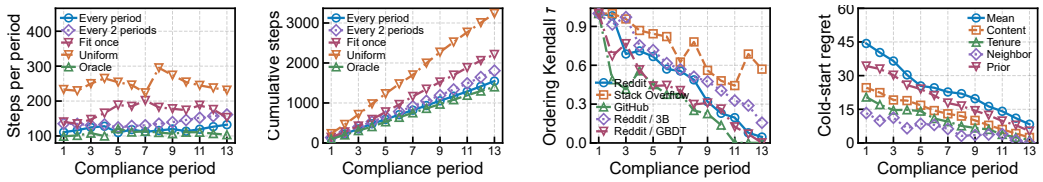
After hazard drift, an infrequently refreshed allocation continues to prepare the previous high-risk clients. More frequent refitting reduces this lag, and the exposure already accumulated under the old routing is carried forward as a constant, so the cumulative-cost separation is larger than a single-period difference shows. Cold-start policies act on the same information boundary: historical individual signals are missing, so category, tenure or population summaries stand in until a client develops its own record.



(a) Readiness. (b) Steps saved. (c) Cost ratio.

(d) Retained utility. (e) Deletion quality. (f) Rebalancing.

Figure 9: Sensitivity of allocation and deployment. (a–c) A shared cohort and normalized budget η/n determine readiness, savings versus uniform and the cost ratio; negative savings mean extra cost. (d–f) Quality and rebalancing comparisons.



(a) Period cost. (b) Cumulative cost. (c) Ordering drift. (d) Cold start.

Figure 10: Hazard refitting across compliance periods: deletion cost, cumulative cost, ordering stability and cold-start regret.

E.6 Feature, Fairness and Privacy Comparisons

Table 8 removes one feature group from HE at a time and follows the change through allocation. Figure 11 varies request volume and the fairness constraint; cost is compared at the achieved fairness level. Table 9 fits private hazard estimators and recomputes readiness at every privacy budget, so it measures the downstream cost of prediction error as well as predictor quality.

The feature ablation distinguishes activity-based information from content and tenure signals. The useful feature is the one whose removal changes the allocation-relevant prediction, which is a different question from which feature discriminates best in isolation. Removing the activity history moves the allocation most: Kendall correlation falls from 0.47 to 0.37, adjacent swaps rise from 12 to 23, and Steps rise from 110 to 143 at a Regret of 42. Removing account tenure gives the lowest AUC, 0.63, yet its allocation stays at 119 steps, and removing contribution size raises AUC to 0.74 while Steps rise to 132. Among model families the two-layer MLP reaches 105 steps at a Regret of 4, while gradient-boosted trees attain the highest fitted AUC, 0.77, at 143 steps. The privacy comparison adds uncertainty to that prediction channel. Stronger privacy constraints bring the resulting allocation closer to less selective preparation, and the ordering that matters for cost is the one the allocation reads. On Reddit-2024 the private estimator holds an AUC of 0.71 at privacy budgets ϵ of 8 and 2, where Steps rise from 110 to 123 and Regret from 9 to 22, and at a budget of 0.1 its AUC of 0.56 gives 217 steps and Regret 116, against 243 and 142 for the uniform rule.

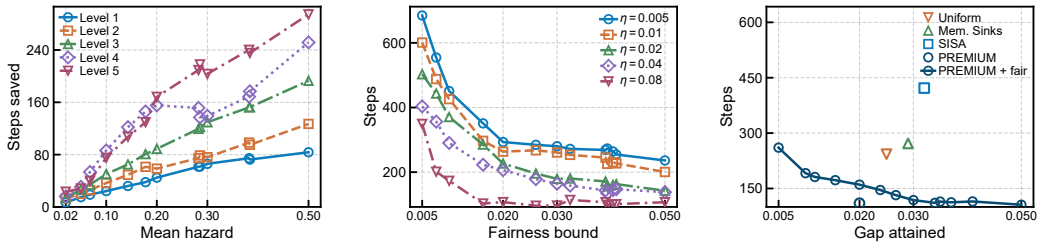
Table 8: Hazard-estimator ablation on Reddit-2024 over five temporal splits. τ: Kendall rank correlation; k: adjacent ordering swaps; Err.: weighted absolute hazard error; AUC follows Table 5; Steps, Util. and Regret follow Table 1.

Model	AUC↑	τ↑	k ↓	Err.↓	Steps↓	Util.↑	Regret↓
(a) Signal families removed (logistic link)							
All signals	0.71(±0.02)	**0.47**(±0.04)	**12**(±1)	0.057(±0.003)	**110**(±4)	0.649(±0.016)	**9**(±4)
− activity history (posts per period)	0.68(±0.02)	0.37(±0.01)	23(±1)	0.101(±0.006)	143(±5)	**0.687**(±0.014)	42(±6)
− account tenure	0.63(±0.04)	0.42(±0.004)	13(±1)	0.086(±0.002)	119(±4)	0.643(±0.064)	18(±4)
− content category (subreddit class)	0.69(±0.03)	0.39(±0.01)	17(±1)	0.085(±0.005)	130(±8)	0.661(±0.035)	29(±7)
− contribution size	**0.74**(±0.02)	0.44(±0.01)	13(±1)	0.067(±0.004)	132(±1)	0.640(±0.038)	31(±2)
− availability trace (dropout)	0.65(±0.02)	**0.47**(±0.03)	13(±1)	**0.053**(±0.001)	130(±14)	0.655(±0.021)	29(±15)
Constant hazard (no model)	0.50	0.00	50	0.207	243	0.659	142
(b) Model family (all signals)							
Logistic (Eq. hazard)	0.71(±0.02)	0.47(±0.04)	12(±1)	0.057(±0.003)	110(±4)	0.649(±0.016)	9(±4)
Discrete-time Cox	0.74(±0.05)	0.46(±0.03)	**10**(±0.4)	0.064(±0.003)	124(±7)	0.645(±0.023)	23(±7)
Gradient-boosted trees	**0.77**(±0.02)	**0.50**(±0.05)	**10**(±0.5)	**0.043**(±0.003)	143(±7)	0.649(±0.034)	42(±8)
Two-layer MLP	0.75(±0.03)	0.46(±0.02)	**10**(±1)	0.057(±0.003)	**105**(±2)	**0.650**(±0.006)	**4**(±3)
Oracle hazards (upper reference)	1.00	1.00	0	0.000	101	0.647	0

Table 9: Private hazard estimation, one privacy level per row and the three corpora reading across. DP-SGD fits each model on the full user population; ϵ is its privacy budget, and the uniform rule fits no hazard model at all. Allocation is recomputed at each privacy level. AUC follows Table 5; Steps and Regret follow Table 1.

Privacy		Reddit-2024			StackOverflow-2024			GitHub · Stack opt-out		
Estimator	ϵ	AUC↑	Steps↓	Regret↓	AUC↑	Steps↓	Regret↓	AUC↑	Steps↓	Regret↓
No privacy	∞	**0.71**(±0.02)	**110**(±4)	**9**(±4)	0.67(±0.04)	156(±6)	17(±6)	**0.81**(±0.03)	**90**(±5)	**4**(±1)
DP-SGD	8	**0.71**(±0.03)	123(±4)	22(±5)	**0.68**(±0.07)	162(±9)	23(±8)	0.75(±0.02)	106(±7)	20(±6)
DP-SGD	2	**0.71**(±0.03)	123(±3)	22(±2)	0.66(±0.05)	**145**(±6)	**6**(±6)	0.72(±0.02)	116(±3)	30(±3)
DP-SGD	0.5	0.62(±0.04)	154(±3)	53(±3)	0.60(±0.03)	171(±1)	32(±2)	0.68(±0.04)	128(±2)	42(±6)
DP-SGD	0.1	0.56(±0.004)	217(±11)	116(±10)	0.51(±0.04)	226(±5)	87(±6)	0.54(±0.01)	183(±12)	97(±8)
Uniform (no hazards)	0	0.50	243	142	0.50	249	110	0.50	219	133

The fairness panels show the price of restricting unequal treatment across hazard groups. Relaxing the fairness constraint gives the allocator more freedom to reduce request cost; tightening it restricts how much readiness can differ. Reporting cost against the achieved utility gap exposes that tradeoff directly. A fixed aggregate budget alone does not force the group-level outcomes to coincide.



(a) Request volume. (b) Fairness constraint. (c) Cost–fairness tradeoff.

Figure 11: Request volume and the deletion-cost–fairness frontier. Levels 1–5 in (a) increase hazard dispersion; η in (b) is the utility budget; (c) pairs the default operating points from Table 1 with the fairness-constrained PREMIUM curve.

E.7 Model Scale, Routing and Deletion Quality

Table 10 changes model scale, adapter rank, optimizer and discrete allocation. The realized-to-priced exposure ratio checks whether the training implementation preserves the allocation's accounting. Time and communication are read alongside preparation and deletion cost. Figure 6 in the main text reads three complementary deletion audits: extraction, paraphrased prompts and membership inference at low false-positive rate. All methods use the same probes and stopping criterion within a comparison.

The scale table keeps resource growth separate from deletion effectiveness. Larger models and wider adapters change training time, communication and the space in which private learning can be stored. The optimizer and discrete-rank rows additionally change how faithfully a readiness value is realized. Those changes are therefore read through both the exposure ratio and the final cost–quality pair.

The three deletion-quality panels check different manifestations of retained information. Extraction probes exact continuations, paraphrased prompts vary the query surface, and membership inference tests distinguishability of withdrawn records. A cost advantage accompanied by similar audit outcomes has a different interpretation from an advantage obtained by stopping before removal. Their joint presentation gives the deletion-cost comparison its operational meaning.

Table 10: Scale, adapter rank, optimizer and discrete-rank variants on Reddit-2024. Ratio: realized divided by priced exposure; Time: wall-clock seconds per training round; Comm.: uploaded megabytes per round. Other metrics follow Table 1.

Setting	Steps↓	Util.↑	FQ↑	MIA↓	Regret↓	Ratio	Time↓	Comm.↓
Model size (rank 16, SGD + momentum, continuous routing)								
Llama-3.2-1B	**110**(±4)	0.649(±0.016)	0.74(±0.07)	0.53(±0.01)	9(±4)	**1.00**	**40**(±5)	**53**(±1)
Llama-3.2-3B	142(±11)	0.703(±0.034)	**0.81**(±0.03)	**0.52**(±0.003)	**7**(±1)	**1.00**	116(±6)	94(±2)
Llama-3.1-8B	166(±7)	**0.750**(±0.027)	0.80(±0.01)	0.53(±0.02)	8(±0.2)	**1.00**	280(±15)	156(±2)
Adapter rank (Llama-3.2-1B)								
Rank 4	**105**(±2)	0.644(±0.022)	0.74(±0.03)	**0.52**(±0.01)	**6**(±0.5)	**1.00**	**36**(±1)	**13**(±1)
Rank 16	110(±4)	0.649(±0.016)	0.74(±0.07)	0.53(±0.01)	9(±4)	**1.00**	40(±5)	53(±1)
Rank 64	150(±1)	**0.750**(±0.056)	**0.80**(±0.01)	**0.52**(±0.01)	8(±1)	**1.00**	69(±2)	209(±17)
Optimizer (Llama-3.2-1B, rank 16)								
SGD + momentum (main)	**110**(±4)	0.649(±0.016)	0.74(±0.07)	0.53(±0.01)	**9**(±4)	**1.00**	**40**(±5)	53(±1)
AdamW, priced as SGD	138(±4)	0.643(±0.044)	**0.78**(±0.04)	0.56(±0.03)	31(±2)	1.15(±0.04)	42(±2)	**41**(±2)
AdamW, price corrected by the measured ratio	131(±1)	**0.699**(±0.014)	**0.78**(±0.02)	**0.51**(±0.0003)	**9**(±1)	1.39(±0.02)	43(±3)	49(±2)
Realization (Llama-3.2-1B, rank 16)								
Continuous routing (Eq. routing)	**110**(±4)	0.649(±0.016)	0.74(±0.07)	**0.53**(±0.01)	**9**(±4)	1.00	40(±5)	53(±1)
Discrete private rank $\lceil r_c \cdot 16 \rceil$	115(±0.3)	**0.669**(±0.017)	0.72(±0.04)	**0.53**(±0.01)	14(±3)	**0.85**(±0.04)	**36**(±2)	**42**(±3)
Discrete private rank, rank 64	165(±5)	0.659(±0.021)	**0.80**(±0.05)	**0.53**(±0.03)	11(±1)	0.97(±0.04)	73(±3)	171(±3)