Exact Unlearning via Quantized Sufficient Statistics

Ami Tavory
Meta Platforms, Inc.

Shripad Gade
Meta Platforms, Inc.

Tal Sarig
Meta Platforms, Inc.

Noam Touitou
Meta Platforms, Inc.

Ido Guy
Technion - Israel Institute of Technology

Abstract

Exact unlearning requires a deployed predictor to match one rebuilt without the information named by a deletion request. Existing general-purpose exact methods localize retraining through disjoint shards, but every request still invalidates a model, and smaller shards reduce the data available to each constituent predictor. We introduce *Quantized Sufficient Statistics* (QSS), which separates a small frozen *schema* from mutable, sum-decomposable *content*. The schema learns global structure; the content stores local prediction corrections as additive statistics indexed by quantized regions. Deleting content is therefore exact subtraction rather than optimization. We distinguish two guarantees: QSS-L exactly removes a label while retaining the unlabelled input, whereas QSS-E exactly removes both input and label by learning the schema without deletable examples. A deletion takes the arithmetic fast path with probability $1 - \rho$ and triggers a full rebuild with probability ρ; all reported expected latencies include both events. Across 15 vision, text, and tabular datasets at $\rho = 0.5\%$, QSS-L is within 2 percentage points of SISA on 11 tasks and provides 4–483$\times$ lower expected deletion latency on the low-class-count tasks where a compact schema is effective. QSS-E quantifies the additional accuracy cost of removing every trace of an input.

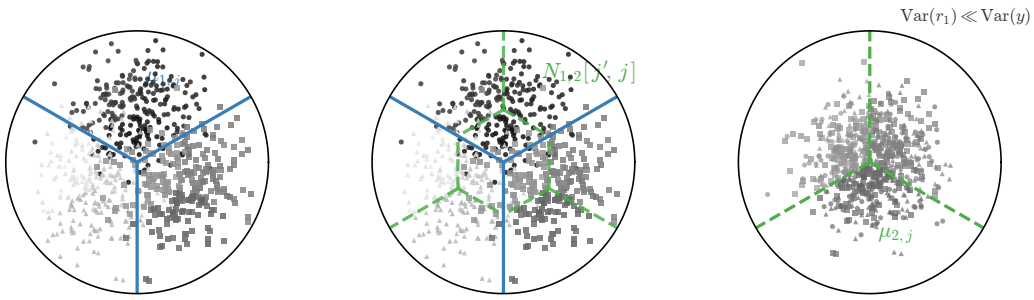
1 Introduction

Machine unlearning has become a legal and operational requirement for deployed models: privacy regulations such as the EU's GDPR [19] and California's CCPA [5] mandate the ability to delete the effect of specific training instances, while security and safety concerns (removing poisoned [25], biased or outdated data points [18]) provide independent motivation. Exactness is defined relative to the information named by the request. We study two neighboring-dataset relations. *Label deletion* retains an input x but removes its label y and all supervised influence; this models retracted annotations, outcomes, or responses when retaining the unlabelled input is permitted. *Instance deletion* removes both x and y, so the result must match training as if the example had never been present. We write $\mathcal{D}^{-y_j}$ for the former neighboring dataset and $\mathcal{D}^{-j}$ for the latter. The fundamental obstacle in either case is *entanglement*: standard training weaves examples into shared parameters, leaving retraining as the generic way to produce the appropriate neighboring-dataset model.

The standard exact baseline, SISA [1], attacks this problem by *disjoint partitioning*: it partitions the dataset (size n) into S shards, and trains a sub-model on each, simultaneously bounding the retraining cost to $O(n/S)$, but also ensuring that every deletion *deterministically* invalidates a single shard and forces an eventual retrain. Under a batch of B deletions, the expected number of affected shards is $S\big(1-(1-1/S)^B\big)$, which approaches S once $B \gtrsim S \log S$. SISA cannot lower its invalidation rate

[0]Code: `https://github.com/atavory/rqu`

40th Conference on Neural Information Processing Systems (NeurIPS 2026).



(a) **Absolute space** (b) **Nested joint-count regions** (c) **Pooled residual space**

Figure 1: **QSS translates Residual Quantization on embeddings to residual boosting on labels.** Marker shape ($\circ$, $\triangle$, $\square$) encodes cell assignment; intensity encodes the target y_i (dark=high, light=low). **(a)** Level 1 Voronoi cells (b=3) partition the embedding space; each cell has a distinct intensity level with visible internal spread. **(b)** A nested Level 2 codebook subdivides each cell; the intersection regions define the co-occurrence counts $N_{1,2}[j', j]$ stored as sum-decomposable statistics, enabling exact unlearning via simple count decrements. **(c)** After subtracting Level 1 cell means, all points collapse into the pooled residual space. The wide intensity spread collapses to a narrow, uniform band, visualizing that label residuals carry far less variance than the original labels, since each RQ level corrects the previous, analogous to gradient boosting [9] with quantization centroids as weak learners.

without proportionally reducing the data available to each shard; redundancy multiplies compute overhead without improving availability.

We propose **Quantized Sufficient Statistics (QSS)**, which breaks this coupling by changing the model representation. The intuition is a wall of labelled bins. A frozen quantizer assigns similar inputs to bins, and each bin stores only sums and counts of prediction errors. Prediction consults the relevant bins; deleting an example subtracts its contribution. Recomputing the same bin totals from the remaining information gives exactly the same state, without rerunning an optimizer or scanning the dataset.

QSS realizes this idea by separating *schema* from *content*. A small random subset $\mathcal{D}^\circ$ (size n°, $\rho = n^\circ/n$, typically $\leq 1\%$) trains a frozen base model g and a Residual Quantization (RQ) [7] codebook, the *schema* (see Table 1 for symbols). The remaining data $\mathcal{D}^\bullet = \mathcal{D} \setminus \mathcal{D}^\circ$ is pure *content*: stored as sum-decomposable accumulators indexed by RQ codes over frozen embeddings (Figure 1). Because the data structure *is* the model, ordinary unlearning is subtraction: deleting content is a constant-time arithmetic decrement, independent of n. QSS-L trains the label-blind codebook on all inputs and supports exact label deletion. QSS-E trains the codebook only on the retained schema and supports exact instance deletion for content examples. In either mode, deleting a retained schema example invokes a full rebuild. Thus QSS has a Las Vegas-style runtime guarantee: the result is always exact for the stated neighbor relation, while the randomized schema determines whether the request takes the arithmetic or rebuild path. SISA cannot exploit partition redundancy, as overlapping shards would simply require multiple partition rebuilds per deletion. The probabilistic schema–content separation does not suffer the same way. We therefore maintain p independent instances, each with its own random $\mathcal{D}_i^\circ$, base model (g_i trained on $\mathcal{D}_i^\circ$), and codebook. A deletion triggers the $O(1)$ fast path for all active instances with probability $(1-\rho)^p$; a synchronous rebuild must occur only when *all* p instances are invalidated, an event with probability ρ^p. Every instance aggregates all of $\mathcal{D}$ in its accumulators, so while a pair rebuilds, the remaining pairs continue serving at full accuracy. In SISA, taking a shard offline for retraining removes $1/S$ of the data from the ensemble; in QSS, every pair independently sees all data, so no accuracy is lost during rebuilds.

Contributions.

1. **Schema–content separation for exact unlearning** (§2): The schema–content separation proposed in this work reduces ordinary label or instance deletion to constant-time arithmetic updates, independent of n and identical to rebuilding for the corresponding neighbor relation. Co-occurrence counts avoid exponential fragmentation across quantization levels (Equation 6), leading to efficient polynomial-time re-materialization.

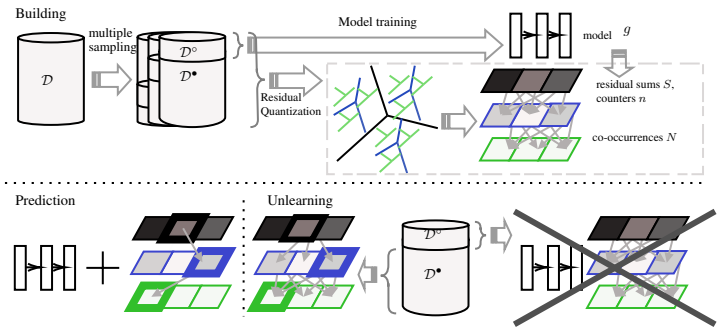


Figure 2: QSS architecture across three phases. **Building** (top): $\mathcal{D}$ is split into retained $\mathcal{D}^\circ$ (which trains the base model g and the RQ codebook) and deletable $\mathcal{D}^\bullet$; all data is encoded and accumulated into mutable sufficient statistics (residual sums S, counters n, co-occurrence counts N). **Prediction** (bottom-left): a query is encoded through the frozen RQ levels and its cell means are summed; bold outlines trace the lookup path through embedding, codebook, and accumulators; the base model g adds its prediction ($+$). **Unlearning** (bottom-right): deleting a point from $\mathcal{D}^\bullet$ decrements accumulators only. In the low-probability event that a deletion affects $\mathcal{D}^\circ$, the pair is invalidated ($\times$) and must be rebuilt.

2. **Redundant schemas for amortized $O(1)$ deletion** (§3.2): p independent copies reduce the synchronous-rebuild probability to ρ^p, with zero accuracy degradation. Under batch deletions of size B, QSS forces a rebuild with probability $(1-(1-\rho)^B)^p$, negligible for typical ρ and p.

Related work. *Exact retraining methods.* SISA [1] partitions data into shards and retrains the affected shard per deletion at $O(n/S)$ cost. Exact k-means deletion [11] and exact random forest unlearning via sufficient statistics [2] achieve n-independent deletion for specific model families, but rely on fixed, single-level partitions. [22] freeze a pretrained LLM and isolate mutable state in k-means clusters for $O(1)$ exact unlearning, but require a full LLM forward pass per prediction; their clusters suffer the same codebook instability we formalize in Appendix B. QSS extends sufficient-statistics unlearning to a learned, multi-level quantized partition: the co-occurrence correction in RQA.MATERIALIZE (Equation 6) is what preserves decomposability across residual levels; without it, level interactions would require joint b^ℓ-cell bookkeeping.

Analytic sufficient-statistic updates. Linear and ridge heads admit exact deletion by maintaining $X^\top X$ and $X^\top Y$ and applying a rank-one downdate. ACU [14] and the centralized form of Quan et al. [24] develop this analytic approach for frozen representations. These methods are natural complements to QSS: analytic updates are preferable when a modest-dimensional linear head is accurate, whereas QSS stores nonlinear local residual structure in quantized cells. The schema mechanism and the choice of analytic versus quantized residual statistics are composable; Appendix D.2 gives matched comparisons and their different state/update costs.

Approximate methods. Certified removal [13], influence functions [17], Fisher scrubbing [12], and gradient ascent [20] provide varying degrees of approximation; see surveys [23, 8].

Quantization and unlearning. QUAIL [21], LUNAR [26], and OEU [27] treat quantization as a complication for unlearning, treating quantization artifacts as problems to be mitigated. QSS inverts this relationship: quantization *is* the architecture, and the quantized representation is what makes exact deletion tractable.

2 Method

2.1 Instances: Schema & Content

QSS maintains $p \geq 1$ independent *instances* (Figure 2, top). Each instance consists of a frozen *schema* (base model g_i and RQ codebook, trained on a retained subset $\mathcal{D}_i^\circ$ of size $n^\circ < n$) and mutable *content* (accumulators aggregating all of $\mathcal{D}$). The content stores only sufficient statistics

Table 1: Notation. Superscript $\circ$ marks the retained subset; $\bullet$ marks the deletable portion.

Symbol	Meaning
$X, Y; x_i, y_i$	Inputs, labels; single instance, single label
n, d, k	Dataset size, embedding dimensionality, label dimensionality
f	Frozen upstream encoder (CLIP, SBERT, or identity); distinct from RQ.encode
g	Base model trained on $\mathcal{D}^\circ$ (frozen)
α	Blending weight (cross-validated); $\alpha{=}0$ recovers g
$\varphi(x)$	Feature map: $L_2(f(x))$
b, ℓ	Centroids per RQ level; number of levels
$\mathbf{C}$; c_m	Code matrix $\in [b]^{n\times\ell}$; code at level m
y_i^r	RQ target for point i; $y_i^r = y_i - g(x_i)$
$\mathbf{T}_{m,j}$	RQ target sum $\sum_{i:\, c_{i,m}=j} y_i^r$ per cell
$N_{k,m}$	Co-occurrence count matrix $\in \mathbb{N}^{b\times b}$ $(k < m)$
$\mathbf{S}_{m,j}$; $n_{m,j}$	Materialized accumulator / count (level m, cell j)

($\mathbf{T}$, n, N), not the data itself; at our operating point, all accumulators occupy ≤ 51 MB even for ImageNet. The deletable portion is $\mathcal{D}_i^\bullet = \mathcal{D} \setminus \mathcal{D}_i^\circ$. Because every instance's accumulators aggregate *all* of $\mathcal{D}$ (using $\mathcal{D}_i^\circ$ only to train the schema), decommissioning an instance incurs zero accuracy loss.

Within each instance, the base model g and the RQ codebook are trained on $\mathcal{D}^\circ$; all of $\mathcal{D}$ is then encoded through the frozen codebook and accumulated into mutable sufficient statistics (the residual sums $\mathbf{T}$, counters n, and co-occurrence counts N) visible at the right of the building panel. Prediction (Figure 2, bottom-left) sums the cell means along the query's code path and adds the base model's output:

$$\hat{y}(x) \;=\; g(x) \;+\; \alpha \cdot \text{RQA.}\textsc{Predict}(x)\,, \tag{1}$$

where α is cross-validated on a held-out fold along with other hyperparameters. Unlearning (Figure 2, bottom-right) decrements the affected accumulators and re-materializes; the schema (codebook, g) is untouched. If the deleted point lies in $\mathcal{D}^\circ$, the pair is invalidated and a replacement is needed.

Guarantee-specific constructions. The neighboring relations were defined in Section 1; the construction must match the requested relation. Under label deletion, the codebook and assignments may still depend on x_j, but the released predictor must equal one whose supervised state was built on $\mathcal{D}^{-y_j}$. Under instance deletion, every model component must equal one built on $\mathcal{D}^{-j}$. Three modes trade off these guarantees against accuracy (Table 2):

- **QSS-E** (instance): The codebook is trained on $\mathcal{D}^\circ$ only. Exact label and instance deletion for points in $\mathcal{D}^\bullet$.
- **QSS-L** (label): The codebook is trained on all $\mathcal{D}$. This leads to better codebook resolution and improved model performance, but only allows for exact label deletion.
- **QSS-DP**: QSS-L plus Gaussian noise on published cell means. Provides (ε, δ)-label differential privacy (Proposition 3).

2.2 RQA Data Structure

Residual Quantization (RQ) [7] encodes each embedding as a sum of ℓ centroid vectors, one per level, each chosen from b candidates; level m quantizes the residual from levels $1, \ldots, m{-}1$. We treat the RQ codebook and its encoder as a frozen black box (trained via Faiss [15]); the *RQA* (RQ-Accumulator) data structure defined below stores sufficient statistics on top of these codes. The codebook may be trained on $\mathcal{D}^\circ$ only (QSS-E, exact instance and label deletion) or all of $\mathcal{D}$ (QSS-L, better resolution, exact label deletion only; Table 2).

The data structure stores three sum-decomposable quantities per code tuple (each a sum over points, so adding or removing a point costs $O(\ell^2)$ decrements):

$$\mathbf{T}_{m,j} = \textstyle\sum_{i:\, c_{i,m}=j} y_i^r \qquad \text{(target sums)}\,, \tag{2}$$

$$n_{m,j} = |\{i : c_{i,m}{=}j\}| \qquad \text{(cell counts)}\,, \tag{3}$$

$$N_{k,m}[j', j] = |\{i : c_{i,k}{=}j',\ c_{i,m}{=}j\}| \qquad \text{(co-occurrence,}\ \ k < m)\,. \tag{4}$$

Table 2: QSS complexity and deletion guarantees. QSS-E, QSS-L, and QSS-DP differ in codebook training set and noise; all use the RQA data structure (Algorithm 1). When a deletion hits $\mathcal{D}^\circ$ (probability n°/n), g and the codebook must be rebuilt.

	QSS-E	QSS-L	QSS-DP	SISA
Codebook	$\mathcal{D}^\circ$	All $\mathcal{D}$	All $\mathcal{D}$	—
Train	$t_g(n^\circ) + O(n\ell bd)$			$O(n)$
Predict	$t_g + O(\ell bd)$			$O(S{\cdot}dk)$
Unlearn from $\mathcal{D}^\bullet$	$O(\ell^2 b^2 k)$			$O(n/S)$
Unlearn from $\mathcal{D}^\circ$	t_{rebuild}			$O(n/S)$
$\mathbb{E}$[unlearn]	$(1-\frac{n^\circ}{n})\,O(\ell^2 b^2 k) + \frac{n^\circ}{n}\,t_{\text{rebuild}}$			$O(n/S)$
Label del.	Exact	Exact	ε-DP†	Exact
Instance del.	Exact	—	Empirical DP	Exact

t_g: base-model cost (linear probe or MLP, model-dependent). QSS-E: codebook on $\mathcal{D}^\circ$ only (instance guarantee). QSS-L: codebook on all data (better accuracy). QSS-DP: + Gaussian noise on cell means guaranteeing (ε, δ)-label DP (Proposition 3).

Prediction recovers per-level cell means $\boldsymbol{\mu}_{m,j} = \mathbf{S}_{m,j}/n_{m,j}$ via RQA.MATERIALIZE (Equation 6), which subtracts earlier-level contributions using co-occurrence counts, analogous to gradient boosting [9] but avoiding exponential fragmentation across levels. For query x:

$$\text{RQA.PREDICT}(x) = \textstyle\sum_{m=1}^{\ell} \boldsymbol{\mu}_{m,\,c_m(x)}\,. \tag{5}$$

Unlearning is exact: decrement and re-materialize (Figure 1). The complete RQA pseudocode appears in Algorithm 1 (Appendix A.9).

3 Analysis

Table 2 summarizes the operating modes, their complexity, and deletion guarantees (proofs in Appendix A.6).

3.1 Unlearning

Proposition 1 (Exact Instance Unlearning in QSS-E)**.** *In the QSS-E method* $\hat{y}(x) = g(x) + \alpha \cdot \text{RQA.PREDICT}(x)$*, where the schema (g, φ, and the codebook) is trained on the retained portion* $\mathcal{D}^\circ$ *and the RQA accumulators are filled on all data, deleting any point* $(x_j, y_j) \in \mathcal{D}^\bullet$ *via* RQA.UNLEARN *produces predictions bitwise identical to retraining the full system from scratch on* $\mathcal{D}^{-j}$.

Proof. Since the schema (φ, codebook, g) is trained on $\mathcal{D}^\circ$ and $(x_j, y_j) \in \mathcal{D}^\bullet$, deleting point j changes neither code assignments nor residual targets. The accumulators $(\mathbf{T}, n, N)$ are additive, so RQA.UNLEARN (subtract j's contributions) produces exactly the values from RQA.BUILD on $\mathcal{D} \setminus \{j\}$. It remains to show that RQA.MATERIALIZE is a deterministic function of $(\mathbf{T}, n, N)$.

By definition, $\mathbf{S}_{m,j} = \sum_{i:\,c_{i,m}=j} r_m(x_i)$ where $r_m(x_i) = y_i - \sum_{k=1}^{m-1} \boldsymbol{\mu}_{k,c_{i,k}}$. Distributing the outer sum and grouping points in cell j at level m by their cell at level k:

$$\mathbf{S}_{m,j} = \mathbf{T}_{m,j} - \sum_{k=1}^{m-1} \sum_{j'=1}^{b} \underbrace{\left|\{i : c_{i,k}{=}j',\, c_{i,m}{=}j\}\right|}_{N_{k,m}[j',j]} \cdot \boldsymbol{\mu}_{k,j'}, \tag{6}$$

with $\boldsymbol{\mu}_{m,j} = \mathbf{S}_{m,j}/n_{m,j}$ ($\mathbf{0}$ if $n_{m,j} = 0$). Levels are computed sequentially ($m = 1, 2, \ldots$) because $\boldsymbol{\mu}_{k,j'}$ depends on accumulators at levels $< k$. Since RQA.MATERIALIZE is thus deterministic in $(\mathbf{T}, n, N)$, the predictions after unlearning are bitwise identical to retraining from scratch on $\mathcal{D} \setminus \{j\}$. □

Exact *instance* deletion (no trace of the point in any model component) requires the codebook to be independent of deletable data. Refitting k-means after a single-point removal can flip every code (Appendix B, [22]); QSS-E avoids this by training the codebook on $\mathcal{D}^\circ$ only.

Remark 2 (Exact Label Unlearning in QSS-L)**.** *Proposition 1 is stated for QSS-E (codebook on $\mathcal{D}^\circ$), but exact* label *deletion also holds in QSS-L (codebook on all $\mathcal{D}$). The codebook is trained by k-means on feature vectors $\varphi(x_i)$ without access to labels, so it is label-blind: labels enter only through the accumulators $\mathbf{T}_{m,j}$ and $n_{m,j}$. Subtracting a point's label contributions from these accumulators and re-materializing produces predictions identical to rebuilding the supervised accumulators from scratch on $\mathcal{D}^{-y_j}$ with the same codebook. What QSS-L does* not *guarantee is instance deletion: the codebook centroids depend on the deleted point's features, so traces of the point's input remain in the schema (Table 2).*

Proposition 3 (Label DP in QSS-DP)**.** *Adding $\mathcal{N}(0, \sigma^2)$ noise to each published cell mean provides (ε, δ)-label differential privacy with*

$$\varepsilon = \frac{\|\Delta y^r\|^2}{2\, n_{\min}\, \sigma^2} + \sqrt{\frac{2\, \|\Delta y^r\|^2\, \ln(1/\delta)}{n_{\min}\, \sigma^2}}, \tag{7}$$

where $\|\Delta y^r\|$ is the maximum residual-label change from a single point's deletion and $n_{\min} = \min_{m,j:\, n_{m,j}>0} n_{m,j}$ is the smallest occupied cell count across all levels and centroids (Equation 3; empty cells are excluded from prediction). The bound is independent of both depth ℓ and width b.

Detailed proof in Appendix A.4. Each centering step in RQA.MATERIALIZE is an orthogonal projection, so the L_2 perturbation energy from a single label change is non-increasing across levels. Under zero-concentrated DP [3], the per-level privacy costs $\rho_m = \|\boldsymbol{\delta}_m\|_2^2/(2\sigma^2)$ telescope: $\sum_m \rho_m \le \|\Delta y^r\|^2/(2\, n_{\min}\, \sigma^2)$, eliminating the depth factor entirely. Note that QSS-DP provides *label*-DP only: the noise masks the influence of individual labels on cell means, but the codebook and code assignments may still depend on the point's features (unless QSS-E is used). We perform empirical studies using membership inference attacks to measure residual membership signal (§4).

3.2 Operational guarantees

When a deletion invalidates a instance, it begins rebuilding immediately while the remaining instances continue serving.

Proposition 4 (Rebuild Scheduling)**.** *Under immediate rebuilds with $p-1$ instances serving during each rebuild window T_r, and deletion requests arriving as a Poisson process with rate λ_d:*

1. *Outage probability per window: $P_{\text{out}} \approx (1 - e^{-\rho\lambda_d T_r})^{p-1}$, exponentially small in p.*
2. *Rebuild overhead: at most $p\, H_p$ times the offline optimal, where $H_p = \sum_{i=1}^{p} 1/i$ ($5.5\times$ for p=3).*

Proofs in Appendices A.6 and A.8. Under the eager policy, the expected inter-rebuild gap is $1/(p\rho)$ deletions, so the expected cost per deletion is $O(\ell^2 b^2 k)$ (fast-path re-materialization) plus $p\rho\, t_{\text{rebuild}}$ (amortized rebuild). The offline optimal achieves gap H_p/ρ (coupon-collector lower bound), giving the $p\, H_p$ competitive ratio. Given that $\mathcal{D}^\circ$ membership is secret, an adversary controls only the *rate* of deletions, not which instances are invalidated; a batch of B simultaneous deletions invalidates all p instances with probability $(1-(1-\rho)^B)^p$ ($B > 1{,}300$ to exceed 1% at ρ=0.5%, p=3). Steady-state cost rates for deployment sizing appear in Appendix A.8.

3.3 Prediction quality

Proposition 5 (Variance Decay)**.** *Within the RQ cascade, the residual variance after m levels satisfies $\mathrm{Var}(r_m) = \mathrm{Var}(y^r) \prod_{j=1}^{m}(1 - \eta_j)$, where $\eta_m = \mathrm{Var}(\mathbb{E}[r_{m-1} \mid c_m])/\mathrm{Var}(r_{m-1})$ is the variance explained at level m. If $\eta_m \ge \gamma > 0$, MSE decays as $(1-\gamma)^\ell$.*

Proof in Appendix A.3. The cascade is analogous to gradient boosting [9] with quantization centroids as weak learners. Excess risk scales as b/n, not b^ℓ/n: each level estimates cell means independently with $\sim n/b$ points (Appendix A.5).

Table 3: Architectural comparison with unlearning methods.

Method	Exact?	Unlearn	Predict	Indep. of n?
Retrain	Yes	$O(n)$	0	No
SISA [1]	Yes	$O(n/S)$	$\times S$	No
Certified [13]	ε	$O(d^3)$	0	Yes
SLUG [4]	No	$O(d^2)$	0	Yes
[22]	Yes	$O(1)$	LLM fwd	Yes
Ridge/ACU [14, 24]	Yes	$O(d^2{+}dk)$	$O(dk)$	Yes
QSS (ours)	Yes	$O(\ell^2 b^2 k)$	$O(\ell b d)$	Yes

Remark 6 (Depth Collapse)**.** *Because residual norms shrink geometrically (Proposition 5), deeper levels add estimation noise without reducing approximation error. The practical rule is* width $>$ depth *(Appendix A.5).*

4 Experimental Results

4.1 Setup

We evaluate on 15 datasets with $n \geq 50$K (where deletion speed is a genuine bottleneck), spanning three modalities and k from 2 to 1,000 (Table 4). A frozen encoder maps inputs to L_2-normalized embeddings: CLIP ViT-L/14 (768-d) for 5 vision tasks, all-MiniLM-L6-v2 (384-d) for 4 text tasks, and LightGBM leaf contributions (trained on $\mathcal{D}^\circ$) for 6 tabular tasks. All 27 viable datasets are reported in Appendix C; no dataset with $n \geq 50$K was excluded based on QSS performance.

Data splits and operating point. We set $\rho = n^\circ/n = 0.5\%$ throughout, so that 0.5% of the data is retained and 99.5% is deletable. This gives n° ranging from 256 (SST-2) to 41K (HIGGS). A base model g (linear probe when $n^\circ < 5{,}000$; one-hidden-layer MLP otherwise) is trained on $\mathcal{D}^\circ$ and frozen. QSS corrects g's residuals using accumulators built on all of $\mathcal{D}$. The blending weight α and codebook hyperparameters (b, ℓ) are cross-validated on $\mathcal{D}^\circ$; $\alpha = 0$ recovers g exactly, guaranteeing QSS $\geq \mathcal{D}^\circ$-only.

Baselines. We compare three baselines. $\mathcal{D}^\circ$*-only* (floor) trains the base model g on $\mathcal{D}^\circ$ alone; QSS is guaranteed to match or exceed this by construction (α=0 fallback). *SISA* [1] (ceiling) partitions $\mathcal{D}^\bullet$ into S=5 shards, trains the same model architecture as g on $\mathcal{D}^\circ \cup \text{shard}_k$, and majority-votes predictions; unlearning retrains the affected shard at cost $O(n/S)$. We use S=5 following [1], the most favorable accuracy setting for SISA since larger S reduces per-shard accuracy. *Exact ridge* [24] fits a ridge head using all training labels and maintains $X^\top X$ and $X^\top Y$ for rank-one exact deletion. We report matched submitted-split comparisons on AG News and SST-2, the two tasks for which the submitted frozen-SBERT pipeline can be reconstructed exactly (Appendix D.2).

Table 3 places QSS among existing methods: QSS is exact and n-independent, unlike SISA, and avoids the full LLM forward pass of [22].

4.2 The Accuracy–Speed Tradeoff

The retention ratio $\rho = n^\circ/n$ controls the accuracy–speed tradeoff. A larger ρ yields a stronger base model g and a higher-fidelity codebook, but increases the rebuild probability. The expected per-deletion cost is

$$\mathbb{E}[t] \;=\; (1{-}\rho)\, t_{\text{RQ}} \;+\; \rho\, t_{\text{rebuild}}\,, \tag{8}$$

where $t_{\text{RQ}} = O(\ell^2 b^2 k)$ is the fast-path accumulator update (independent of n) and t_{rebuild} is the full pipeline cost (retrain g, re-extract embeddings, retrain codebook, re-encode and re-accumulate). At ρ=0.5%, the $\rho\, t_{\text{rebuild}}$ term is small even when t_{rebuild} itself is large: for HIGGS (n=11M), the expected cost is ≈ 18 s vs. SISA's 114 s (Table 6). Figure 3 traces the tradeoff as ρ varies from 0.5% to 5%; most low-k datasets cluster near the break-even line, while high-k outliers narrow their gap as ρ increases.

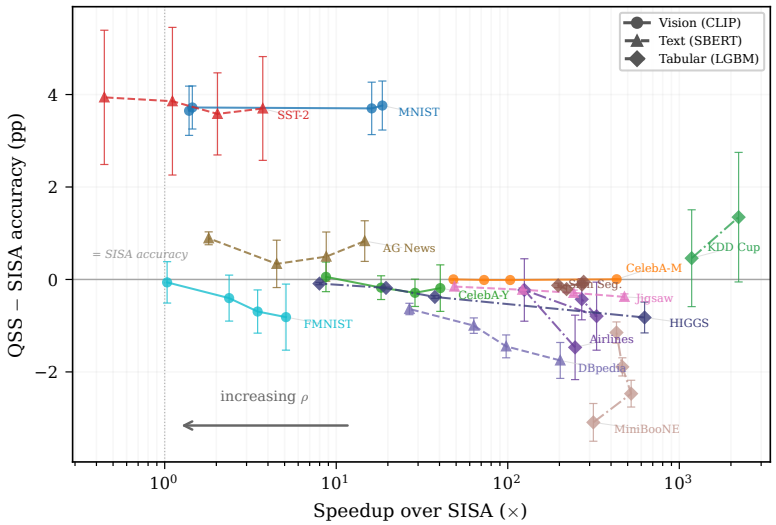


Figure 3: QSS-L accuracy–speedup Pareto as ρ varies from 0.5% to 5% on the 13 low-k datasets. Both axes are relative to SISA (x=1, y=0). Each trajectory traces one dataset; moving *left* corresponds to increasing ρ (lower speedup, higher accuracy). Most trajectories cluster near the break-even line, trading modest accuracy for large speedups. Full version with all 15 datasets in Figure 8 (appendix).

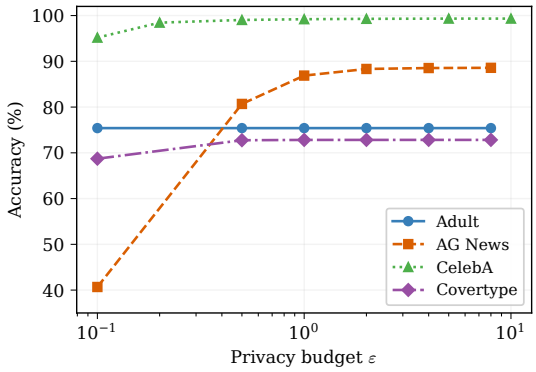
Table 4: Guarantee-aligned accuracy at ρ=0.5% on low-k datasets (mean $\pm$ std over 5 random splits; ‡2–3 seeds). g-only: base model on $\mathcal{D}^{\circ}$ (floor). QSS-L: one fixed label-deletion construction per feature pipeline; modes are not selected using test accuracy. QSS-E is reported separately in Appendix D.1. SISA: S=5 shard retraining [1]. Δ: QSS-L − SISA accuracy (pp). High-k and QSS-E results appear in Table 11 (appendix).

Task	k	n	g-only	QSS-L	SISA	Δ
Vision (CLIP ViT-L/14)						
MNIST	10	53K	.814$_{\pm.009}$	.944$_{\pm.003}$	.921$_{\pm.001}$	+2.4
FMNIST	10	53K	.749$_{\pm.015}$	.825$_{\pm.014}$	.854$_{\pm.003}$	−2.9
CelebA Young	2	137K	.886$_{\pm.002}$	.893$_{\pm.005}$	.902$_{\pm.001}$	−0.9
CelebA Male	2	137K	.992$_{\pm.000}$	.993$_{\pm.001}$	.993$_{\pm.001}$	+0.0
Text (all-MiniLM-L6-v2)						
SST-2	2	51K	.785$_{\pm.008}$	.854$_{\pm.020}$	.835$_{\pm.003}$	+1.8
AG News	4	96K	.861$_{\pm.003}$	.899$_{\pm.004}$	.895$_{\pm.001}$	+0.4
DBpedia	14	473K	.952$_{\pm.002}$	.958$_{\pm.004}$	.977$_{\pm.001}$	−1.9
Jigsaw	2	1.4M	.923$_{\pm.000}$	.924$_{\pm.001}$	.927$_{\pm.000}$	−0.4
Tabular (LightGBM leaf contributions)						
Skin Seg.	2	245K	.986$_{\pm.002}$	.997$_{\pm.001}$	.999$_{\pm.000}$	−0.2
Airlines	2	500K	.627$_{\pm.004}$	.640$_{\pm.002}$	.655$_{\pm.007}$	−1.5
MiniBooNE	2	130K	.903$_{\pm.004}$	.912$_{\pm.004}$	.943$_{\pm.002}$	−3.1
HIGGS‡	2	11M	.728$_{\pm.000}$	.730$_{\pm.001}$	.740$_{\pm.003}$	−1.0
KDD Cup	17	494K	.996$_{\pm.001}$	.999$_{\pm.000}$	.994$_{\pm.011}$	+0.5

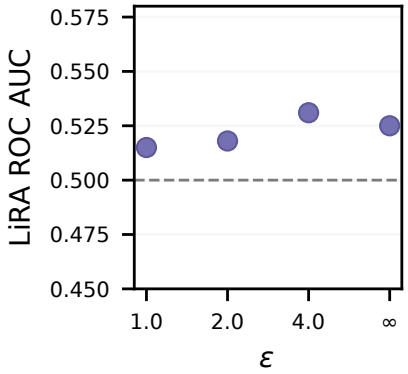
4.3 Accuracy and Timing

Table 4 reports accuracy and expected deletion latency at ρ=0.5% on 13 low-k datasets (mean over 5 random splits; full results including high-k tasks in Table 11, appendix). QSS matches or exceeds $\mathcal{D}^{\circ}$-only on all datasets, confirming that cross-validated α reliably extracts value from deletable data without risk.

Fixed QSS-L exceeds SISA accuracy on 4 of 13 datasets (MNIST, SST-2, AG News, and KDD Cup) and trails by ≤ 2 pp on 7 more. The separately measured QSS-L operating points achieve 4–483$\times$ lower expected deletion cost (Figure 3 and Table 6); every latency includes the rare rebuild event.



(a) ε-label-DP accuracy. Adult (binary) is insensitive; Covertype (7-class) and AG News (4-class) degrade gracefully.



(b) LiRA membership inference on CelebA. AUC ≈ 0.52 at all ε, including $\varepsilon{=}\infty$.

Figure 4: Privacy validation. **(a)** DP noise preserves most accuracy at $\varepsilon \geq 1$. **(b)** Even without DP, QSS leaks no membership information (64 shadow models, 1000 audit points).

Deletion timing. The fast path dominates: at $\rho{=}0.5\%$, the $\rho\, t_{\text{rebuild}}$ term contributes < 1 s for most datasets. On Jigsaw ($n{=}1.4$M), QSS-L deletes in 0.2 s expected vs. SISA's 109 s ($483\times$). The exceptions are HIGGS and KDD Cup, where large t_{rebuild} limits the speedup to $6\times$ (Table 6, appendix).

Aggregate analysis (Figure 7, appendix) confirms that k is the primary factor: the gap clusters near zero for low-k (median -0.4 pp) but separates at $k{=}1{,}000$ (§4.5).

4.4 Correctness and Privacy

We verify the additive exactness mechanism empirically by deleting 100 points and comparing accumulator-subtracted predictions to rebuilt statistics. Across nine datasets spanning all three modalities, *all differences are exactly 0.0*: bitwise identical for the relevant fixed-schema relation. A targeted QSS-L audit on AG News obtains the same result for batches of 1, 10, 100, and 1,000 label deletions. Deleting up to 50% of ImageNet (640K points) likewise produces bitwise-identical predictions, with only 0.25 pp accuracy loss from the reduced labels (Table 7, appendix).

QSS predictions depend on cell-level aggregates ($n_{m,j} \gg 1$), not individual points, limiting membership leakage by construction (Figure 4). Adding (ε, δ)-label DP (Proposition 3) preserves most accuracy at $\varepsilon \geq 1$ (panel a). A LiRA attack [6] (64 shadow models, 1000 audit points) achieves AUC ≈ 0.52, even at $\varepsilon{=}\infty$ (panel b; Table 8, appendix).

4.5 Discussion and Limitations

High-k gap. On ImageNet ($k{=}1{,}000$, -15.6 pp) and Covertype ($k{=}7$, -10.7 pp), QSS lags SISA at $\rho{=}0.5\%$. The codebook partitions embedding space without regard to class boundaries, diluting cell-mean signal when k is large. Level-0 cell purity predicts this: MNIST achieves 96.6% purity ($+3.8$ pp over SISA), Covertype only 63.1% (Figure 20, appendix). The gap narrows with ρ: Covertype shrinks to -5 pp at $\rho{=}5\%$ (Table 6).

Rebuild cost and practical guidance. The $\rho\, t_{\text{rebuild}}$ term scales with n; for HIGGS ($n{=}11$M), this limits the speedup to $67\times$. On tabular datasets, accumulator re-fill converges early (1% of n for Skin Segmentation, 10% for HIGGS; Table 9, appendix). At $\rho{=}0.5\%$ with $k{\leq}17$, QSS trails SISA by at most 3 pp while deleting 4–$483\times$ faster; SISA remains preferable when k is large.

Schema capacity. The submitted linear and one-hidden-layer heads are operating points, not method restrictions. A richer schema may improve difficult tasks, but a larger retained set raises both rebuild probability and rebuild cost. On Covertype, a three-layer head gains little over linear at $\rho = 0.5\%$ but

gains 10.74 pp at $\rho = 5\%$, while its training time rises from 1.29 to 14.65 seconds. Appendix D.5 reports the joint capacity–schema sweep and shows that the benefit is dataset-dependent.

4.6 Summary

QSS separates *schema* from *content*, reducing ordinary deletion to constant-time arithmetic independent of n; redundant schemas drive rebuild probability to ρ^p with zero accuracy loss. QSS-L provides exact label deletion with the competitive accuracy reported above, while QSS-E exposes the additional cost of exact instance deletion. Unlike SISA's S pipelines, QSS is a single model with a sidecar data structure.

Acknowledgments and Disclosure of Funding

This work received no external funding. The authors declare that they have no competing interests.

References

[1] Lucas Bourtoule, Varun Chandrasekaran, Christopher A Choquette-Choo, Hengrui Jia, Adelin Travers, Baoyuan Zhang, David Lie, and Nicolas Papernot. Machine unlearning. In *IEEE Symposium on Security and Privacy (SP)*, pages 141–159, 2021.

[2] Jonathan Brophy and Daniel Lowd. Machine unlearning for random forests. In *International Conference on Machine Learning*, pages 1092–1104. PMLR, 2021.

[3] Mark Bun and Thomas Steinke. Concentrated differential privacy: Simplifications, extensions, and lower bounds. In *Theory of Cryptography*, pages 635–658, 2016.

[4] Zikui Cai, Yaoteng Tan, and M. Salman Asif. Targeted unlearning with single layer unlearning gradient. In *International Conference on Machine Learning (ICML)*, 2025. arXiv:2407.11867.

[5] California Department of Justice. California consumer privacy act (CCPA), 2024. Accessed: 2024-05-22.

[6] Nicholas Carlini, Steve Chien, Milad Nasr, Shuang Song, Andreas Terzis, and Florian Tramer. Membership inference attacks from first principles. In *IEEE Symposium on Security and Privacy*, 2022.

[7] Yongjian Chen, Tao Guan, and Cheng Wang. Approximate nearest neighbor search by residual vector quantization. *Sensors*, 10(12):11259–11273, 2010.

[8] Jiali Cheng and Hadi Amiri. MU-Bench: A multitask multimodal benchmark for machine unlearning. *arXiv preprint arXiv:2406.14796*, 2024.

[9] Jerome H. Friedman. Greedy function approximation: A gradient boosting machine. *Annals of Statistics*, 29(5):1189–1232, 2001.

[10] Tiezheng Ge, Kaiming He, Qifa Ke, and Jian Sun. Optimized product quantization. In *IEEE Transactions on Pattern Analysis and Machine Intelligence*, volume 36, pages 744–755, 2014.

[11] Antonio Ginart, Melody Y. Guan, Gregory Valiant, and James Zou. Making AI forget you: Data deletion in machine learning. In *Advances in Neural Information Processing Systems (NeurIPS)*, volume 32, pages 3513–3526, 2019.

[12] Aditya Golatkar, Alessandro Achille, and Stefano Soatto. Eternal sunshine of the spotless net: Selective forgetting in deep networks. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 9301–9309, 2020.

[13] Chuan Guo, Tom Goldstein, Awni Y. Hannun, and Laurens van der Maaten. Certified data removal from machine learning models. In *International Conference on Machine Learning (ICML)*, pages 3832–3842. PMLR, 2020.

[14] Yajiang Huang, Jianheng Tang, Kejia Fan, Huiping Zhuang, Anfeng Liu, Tian Wang, Yunhuai Liu, Mianxiong Dong, and Houbing Herbert Song. Towards efficient and exact forgetting services in pre-trained-model-based continual learning. *arXiv preprint arXiv:2505.12239*, 2025.

[15] Jeff Johnson, Matthijs Douze, and Hervé Jégou. Billion-scale similarity search with GPUs. *IEEE Transactions on Big Data*, 7(3):535–547, 2021.

[16] Guolin Ke, Qi Meng, Thomas Finley, Taifeng Wang, Wei Chen, Weidong Ma, Qiwei Ye, and Tie-Yan Liu. Lightgbm: A highly efficient gradient boosting decision tree. In *Advances in Neural Information Processing Systems*, volume 30, pages 3146–3154, 2017.

[17] Pang Wei Koh and Percy Liang. Understanding black-box predictions via influence functions. In *ICML*, 2017.

[18] Dianqing Liu, Yi Liu, Guoqing Jin, and Zhendong Mao. Mitigating biases in language models via bias unlearning. In *Proceedings of the 2025 Conference on Empirical Methods in Natural Language Processing*, pages 4160–4178, 2025.

[19] Alessandro Mantelero. The eu proposal for a general data protection regulation and the roots of the 'right to be forgotten'. *Computer Law & Security Review*, 29(3):229–235, 2013.

[20] Odelia Melamed, Gilad Yehudai, and Gal Vardi. Provable unlearning with gradient ascent on two-layer ReLU neural networks. *arXiv preprint arXiv:2510.14844*, 2025.

[21] Himanshu Mishra and Kanwal Mehreen. QUAIL: Quantization aware unlearning for mitigating misinformation in LLMs. *arXiv preprint arXiv:2601.15538*, 2026.

[22] Andrei Ioan Muresanu, Anvith Thudi, Michael R. Zhang, and Nicolas Papernot. Fast exact unlearning for in-context learning data for LLMs. In *ICML*, 2025. arXiv:2402.00751.

[23] Thanh Tam Nguyen, Thanh Trung Huynh, Zhao Ren, Phi Le Nguyen, Alan Wee-Chung Liew, Hongzhi Yin, and Quoc Viet Hung Nguyen. A survey of machine unlearning. *ACM Transactions on Intelligent Systems and Technology*, 16(5):108:1–108:46, 2025.

[24] Yijun Quan, Wentai Wu, and Giovanni Montana. Exact federated continual unlearning for ridge heads on frozen foundation models. *arXiv preprint arXiv:2603.12977*, 2026.

[25] Stefan Schoepf, Jack Foster, and Alexandra Brintrup. Potion: Towards poison unlearning. *Journal of Data-centric Machine Learning Research*, 2024.

[26] William F. Shen, Xinchi Qiu, Meghdad Kurmanji, Alex Iacob, Lorenzo Sani, Yihong Chen, Nicola Cancedda, and Nicholas D. Lane. LLM unlearning via neural activation redirection. *arXiv preprint arXiv:2502.07218*, 2025.

[27] Tian Zhang, Yujia Tong, Junhao Dong, Ke Xu, Yuze Wang, and Jingling Yuan. Forget by uncertainty: Orthogonal entropy unlearning for quantized neural networks. *arXiv preprint arXiv:2602.00567*, 2026.

A Proofs

A.1 Proof of Proposition 1 (Exact Label Unlearning)

The proof appears in the main text (§3). We expand Step 2 here with explicit subtraction formulas.

Let $(x_j, y_j) \in \mathcal{D}^\bullet$. RQA.UNLEARN subtracts point j's contributions:

$$\begin{aligned} \mathbf{T}'_{m,c} &= \mathbf{T}_{m,c} - y_j^r \, \mathbf{1}[c_{j,m} = c], & n'_{m,c} &= n_{m,c} - \mathbf{1}[c_{j,m} = c], \\ N'_{k,m}[c', c] &= N_{k,m}[c', c] - \mathbf{1}[c_{j,k} = c',\, c_{j,m} = c]. \end{aligned} \tag{9}$$

These are exactly the values obtained by running RQA.BUILD on $\mathcal{D} \setminus \{j\}$.

A.2 Materialization Lemma

The materialization recurrence (Equation 6) and its derivation appear in the proof of Proposition 1 (§3).

Complexity. Each (k, m) pair requires a matrix–matrix product $N_{k,m}^{\top}\boldsymbol{\mu}_k$ of dimensions $b \times b$ times $b \times k$, costing $O(b^2k)$. Summing over all $\binom{\ell}{2}$ pairs gives $O(\ell^2b^2k)$ total. The computation has ℓ sequential rounds (level m depends on $\boldsymbol{\mu}_1, \ldots, \boldsymbol{\mu}_{m-1}$); within each round, the $m-1$ matmuls are independent and parallelizable.

Storage. The stored state is $\binom{\ell}{2}b^2$ integers (co-occurrence counts) plus ℓbk floats (raw label sums) plus ℓb integers (cell counts). The materialized cache adds ℓbk floats (residual accumulators for prediction). At the operating points used in our experiments ($b \leq 256$, $\ell \leq 8$): Adult 1 MB, Covertype 3.5 MB, AG News 5.1 MB, CelebA 20 MB, ImageNet 51 MB. Co-occurrence counts dominate for $k \ll b$; label sums dominate for large k (ImageNet: 31 MB of 51 MB). This storage trades static memory for eliminating $O(n)$ retraining.

Batch unlearning. Deleting B points requires $O(B\ell^2)$ count decrements followed by a single $O(\ell^2b^2k)$ re-materialization. The per-deletion amortized cost is $O(\ell^2 + \ell^2b^2k/B)$.

A.3 Proof of Proposition 5 (Variance Decay)

Proof. By the law of total variance:

$$\mathrm{Var}(r_{m-1}) = \mathrm{Var}\big(\mathbb{E}[r_{m-1} \mid c_m]\big) + \mathbb{E}\big[\mathrm{Var}(r_{m-1} \mid c_m)\big]. \tag{10}$$

Since $r_m = r_{m-1} - \mathbb{E}[r_{m-1} \mid c_m]$ and the conditional expectation is the minimum-variance predictor, $\mathbb{E}[r_m \mid c_m] = 0$. Therefore:

$$\begin{aligned}\mathrm{Var}(r_m) &= \mathbb{E}\big[\mathrm{Var}(r_{m-1} \mid c_m)\big] \\ &= \mathrm{Var}(r_{m-1}) - \mathrm{Var}\big(\mathbb{E}[r_{m-1} \mid c_m]\big) \\ &= \mathrm{Var}(r_{m-1})(1 - \eta_m). \end{aligned} \tag{11}$$

Unrolling the recursion from $m = 1$ to $m = \ell$ with $r_0 = y$:

$$\mathrm{Var}(r_\ell) = \mathrm{Var}(y) \prod_{j=1}^{\ell} (1 - \eta_j). \tag{12}$$

□

Remark 7. *Because the materialized views $\mathbf{S}_{m,j}$ are algebraically equivalent to sequentially accumulated residuals (Equation 6), the variance decay properties of the cascade remain identical under the joint-count formulation.*

A.4 Label Differential Privacy Analysis

Scope. This bound applies to *label* DP: it bounds the effect of changing a single label on the published cell means. The codebook is trained via k-means on feature vectors without access to labels, so it is label-blind. This mechanism does *not* provide instance DP for the data vectors themselves.

Proof of Proposition 3. Under the joint-count formulation, the materialized accumulator at level m depends on all earlier cell means: $S_{m,j} = T_{m,j} - \sum_{k<m} \sum_{j'} N_{k,m}[j', j]\, \mu_{k,j'}$. When a single label y_i changes by Δy, define the perturbation vector $\mathbf{x}_m \in \mathbb{R}^N$ where $(\mathbf{x}_m)_i$ is the change in point i's level-m residual. Each centering step in RQA.MATERIALIZE is an orthogonal projection (subtract the cell mean from each entry), so $\|\mathbf{x}_m\|_2 \leq \|\mathbf{x}_0\|_2 = \|\Delta y\|$.

Define the per-level L_2 sensitivity $\|\boldsymbol{\delta}_m\|_2^2 = \sum_j \delta_{m,j}^2$. By Cauchy–Schwarz on each cell, $\delta_{m,j}^2 \leq \|(\mathbf{x}_{m-1})_{[\text{cell } j]}\|_2^2 / n_{m,j}$. Summing over cells at level m:

$$\|\boldsymbol{\delta}_m\|_2^2 \leq \frac{1}{n_{\min}} \sum_j \|(\mathbf{x}_{m-1})_{[\text{cell } j]}\|_2^2 = \frac{\|\mathbf{x}_{m-1}\|_2^2}{n_{\min}}. \tag{13}$$

Under zero-concentrated DP [3], adding $\mathcal{N}(0, \sigma^2)$ noise to each cell mean at level m provides ρ_m-zCDP with $\rho_m = \|\boldsymbol{\delta}_m\|_2^2/(2\sigma^2)$. Since orthogonal projections are contractive, $\|\mathbf{x}_{m-1}\|_2^2 - \|\mathbf{x}_m\|_2^2 \geq 0$ and

$$\sum_{m=1}^{\ell} \rho_m \leq \frac{1}{2\, n_{\min}\, \sigma^2} \sum_{m=1}^{\ell} \|\mathbf{x}_{m-1}\|_2^2 \leq \frac{\ell\, \|\Delta y\|^2}{2\, n_{\min}\, \sigma^2}. \tag{14}$$

The key observation is that the L_2 energies telescope: $\sum_m \|\mathbf{x}_{m-1}\|_2^2 \leq \ell\, \|\mathbf{x}_0\|_2^2 = \ell\, \|\Delta y\|^2$. In fact, since $\|\mathbf{x}_m\|_2 \leq \|\mathbf{x}_{m-1}\|_2$, we can tighten this to $\sum_m \|\mathbf{x}_{m-1}\|_2^2 \leq \ell\, \|\Delta y\|^2$, giving $\rho_{\text{total}} \leq \ell\, \|\Delta y\|^2/(2\, n_{\min}\, \sigma^2)$. Converting to (ε, δ)-DP via $\varepsilon = \rho + \sqrt{2\rho \ln(1/\delta)}$ yields the stated bound. □

Remark 8. *When the residual energy decays rapidly (empirically $\eta_m \geq 0.3$), the sum $\sum_m \|\mathbf{x}_{m-1}\|_2^2$ is dominated by the first few levels and the effective ℓ factor is much smaller than the nominal depth.*

Adversarial simulations provide strong evidence that the true bound is even tighter. We tested all code assignments exhaustively for $N \leq 6$, and ran random (50,000 trials) and greedy adversarial search (200 restarts) for $N \leq 500$, $b \leq 16$, $\ell \leq 12$ (millions of configurations total). We measured the per-level ratio $|\delta_{m,j}|/(\|\Delta y\|/n_{\min})$ at each level independently.

Depth-independent sensitivity. Across all configurations, the maximum per-level ratio remains between 0.95 and 1.25 at every level from 1 to 12, with no upward trend. Table 5 summarizes the per-level maxima for representative configurations. This suggests a per-level bound of the form $|\delta_{m,j}| \leq C\, \|\Delta y\|/n_{m,j}$ with $C \approx 1.25$, independent of depth. If this bound holds, the total privacy cost under the Gaussian mechanism would be $\rho_{\text{total}} \leq C^2\, \ell\, \|\Delta y\|^2/(2\, n_{\min}^2\, \sigma^2)$, which is linear in depth and inversely proportional to $n_{\min}^2$.

Table 5: Maximum per-level sensitivity ratio $|\delta_{m,j}|/(\|\Delta y\|/n_{\min})$ across 30,000 random code assignments per configuration. The ratio stays flat across levels.

n	b	ℓ	L1	L2	L4	L6	L8	L10
50	4	6	1.00	0.95	1.12	1.20	—	—
100	5	8	1.00	0.97	1.09	1.12	1.16	—
200	8	10	1.00	0.97	1.08	1.09	1.10	1.10
500	16	12	1.00	0.98	1.01	1.00	1.02	0.96

Figure 5 visualizes the per-level ratio across configurations: the empirical sensitivity stays flat near 1, far below both the OGMD bound ($O(m)$, dotted) and the Cauchy–Schwarz ceiling ($O(\sqrt{n_{\min}})$, dashed). Proving the $O(\ell/n_{\min})$ bound rigorously (or constructing a counterexample) remains open.

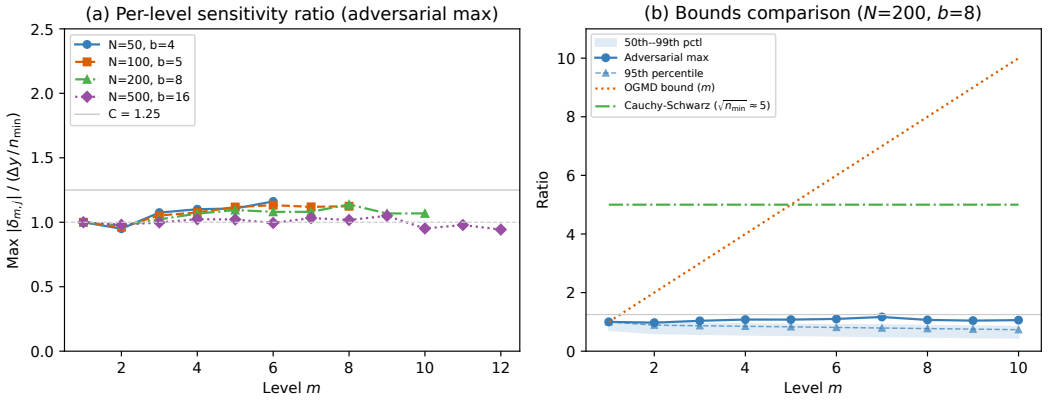


Figure 5: Per-level sensitivity ratio $|\delta_{m,j}|/(\|\Delta y\|/n_{\min})$. **(a)** Adversarial maximum across 20,000 random code assignments for four configurations. The ratio stays flat near 1 at all levels. **(b)** Comparison with theoretical bounds for N=200, b=8: the empirical sensitivity (solid) is far below both the naïve recurrence bound (dotted, grows linearly) and the Cauchy–Schwarz ceiling (dashed, $\sqrt{n_{\min}} \approx 5$).

Empirically, membership inference (LiRA [6]) achieves AUC ≈ 0.52 across all noise levels, consistent with the hypothesis that the cascade sensitivity is far below the $2^{\ell-1}$ bound.

A.5 Excess Risk Bound

Proposition 9 (MSE Decomposition)**.** *Let $\eta_m \geq \gamma > 0$ for all levels, and assume balanced clusters ($n_{m,j} \approx n/b$). The expected MSE of the finite-sample QSS estimator $\hat{y}_\ell$ decomposes as:*

$$\mathbb{E}[(\hat{y}_\ell - y)^2] \;\leq\; \underbrace{\operatorname{Var}(y) \prod_{m=1}^{\ell} (1-\eta_m)}_{\textit{approximation}} \;+\; \underbrace{\frac{b \operatorname{Var}(y)(1-\gamma)}{n\gamma}}_{\textit{estimation}}. \tag{15}$$

Under a Lipschitz condition on the target function, the approximation term is $O(L^2 R^2 \rho^{2\ell})$.

Proof. Decompose the error through the oracle (infinite-data) predictor $\bar{y}_\ell = \sum_m \mathbb{E}[r_{m-1} \mid c_m]$. The approximation term is $\operatorname{Var}(r_\ell)$ (Proposition 5).

For the estimation term, each level's estimator $\hat{g}_m$ estimates the cell mean with $\tilde{n} \approx n/b$ samples:

$$\begin{aligned} \operatorname{Var}(\hat{g}_m) &= \frac{1}{\tilde{n}} \mathbb{E}\big[\operatorname{Var}(r_{m-1} \mid c_m)\big] \\ &= \frac{b}{n}(1-\eta_m) \operatorname{Var}(r_{m-1}). \end{aligned} \tag{16}$$

Summing across levels and expanding $\operatorname{Var}(r_{m-1})$:

$$\begin{aligned} \sum_{m=1}^{\ell} \operatorname{Var}(\hat{g}_m) &= \frac{b}{n} \sum_{m=1}^{\ell} (1-\eta_m) \operatorname{Var}(y) \prod_{j=1}^{m-1} (1-\eta_j) \\ &= \frac{b \operatorname{Var}(y)}{n} \sum_{m=1}^{\ell} \prod_{j=1}^{m} (1-\eta_j) \\ &\leq \frac{b \operatorname{Var}(y)}{n} \sum_{m=1}^{\ell} (1-\gamma)^m \\ &\leq \frac{b \operatorname{Var}(y)}{n} \cdot \frac{1-\gamma}{\gamma}. \end{aligned} \tag{17}$$

The estimation term scales as b/n, *not* b^ℓ/n: the label-residual structure uses each level independently (b cells per level, n/b samples each), avoiding the curse of dimensionality that would arise from joint b^ℓ-cell estimation. □

Interpretation and codebook collapse. The bound reveals two collapse modes. *Width collapse:* the optimal b^* balances approximation (decreasing in b) against estimation (increasing as b/n). When $b \to n$, cells receive too few points and the per-cell means become noisy; at the extreme, empty cells ($n_{m,j} = 0$) require smoothing but cannot recover signal. *Depth collapse:* if preconditioning concentrates between-class variance into the first few dimensions, $\eta_m \to 0$ for later levels and those levels are wasted, adding estimation error without reducing approximation error. The practical rule is $b \ll n$ and increase ℓ only while η_m remains appreciable.

A.6 Complexity of the QSS Method (Table 2)

Proposition 10 (QSS Training Cost)**.** *Training the full QSS method costs $t_g(n^\circ) + O(n\ell bd)$.*

Proof. Training proceeds in two stages. *Stage 1:* train the base model g on $\mathcal{D}^\circ$ ($|\mathcal{D}^\circ| = n^\circ$), costing $t_g(n^\circ)$ (model-dependent). *Stage 2:* build the RQA data structure. Encoding each of the n points requires ℓ sequential levels; at each level, the nearest of b centroids is found by computing b inner products in $\mathbb{R}^d$, costing $O(bd)$ per level. Total encoding: $O(n\ell bd)$. The scatter-add pass that fills $\mathbf{T}_{m,j}$, $n_{m,j}$, and $N_{k,m}$ costs $O(n\ell^2)$ (each point updates ℓ target sums and $\binom{\ell}{2}$ co-occurrence counts).

Materialization costs $O(\ell^2 b^2 k)$ (Appendix A.2). Since $bd \gg \ell$ in practice, the encoding term dominates and the total is $t_g(n^\circ) + O(n\ell bd)$. □

Proposition 11 (QSS Prediction Cost)**.** *Predicting with the full QSS method costs $t_g + O(\ell bd)$ per query.*

Proof. Prediction (Equation 1) evaluates $\hat{y}(x) = g(x) + \alpha \cdot \text{RQA.PREDICT}(x)$. The base-model inference $g(x)$ costs t_g. RQA.PREDICT encodes the query ($O(\ell bd)$, same argument as above) and sums ℓ cached cell means ($O(\ell k)$). The total is $t_g + O(\ell bd)$. □

Proposition 12 (Expected Unlearning Cost)**.** *For a uniformly random deletion, the expected cost is $(1 - n^\circ/n)\, O(\ell^2 b^2 k) + (n^\circ/n)\, t_{\text{rebuild}}$.*

Proof. A deletion request targets a uniformly random point in $\mathcal{D}$. With probability $1 - n^\circ/n$, the point lies in $\mathcal{D}^\bullet$: RQA.UNLEARN decrements $O(\ell^2)$ counters and re-materializes in $O(\ell^2 b^2 k)$ (Appendix A.2). With probability n°/n, the point lies in $\mathcal{D}^\circ$: g and the codebook must be retrained, costing t_{rebuild}. The expectation follows by linearity. □

A.7 Amortized Deletion and CTMC Analysis

Proof of Proposition 3.2. Each deletion hits instance i's $\mathcal{D}_i^\circ$ independently with probability ρ. A synchronous rebuild occurs when all p instances are invalidated. The expected number of deletions until all p are hit follows the coupon collector with non-uniform probabilities; for uniform ρ, it reduces to H_p/ρ by a standard coupon-collector argument. Dividing the rebuild cost by this expected inter-rebuild count gives the amortized term. □

CTMC steady state. Model the system as a continuous-time Markov chain on the number of active instances $\{0, 1, \ldots, p\}$, with invalidation rate $\lambda\rho$ per active instance and repair rate μ per inactive instance (background rebuild). The steady-state probability of full outage (state 0) is

$$\pi_0 = \left(\sum_{j=0}^{p} \binom{p}{j} \frac{(\lambda\rho)^{p-j}\, \mu^j}{\prod_{i=1}^{p-j} (i\lambda\rho) \prod_{i=1}^{j} (i\mu)} \right)^{-1} = O\left(\left(\frac{\lambda\rho}{\mu} \right)^p \right). \tag{18}$$

For $\rho \ll 1$ and $\mu \gg \lambda\rho$ (rebuilds faster than invalidations), full outages are exponentially rare in p. The optimal rebuild policy is analyzed in Appendix A.8.

A.8 Rebuild Scheduling (Proposition 4)

We analyze the non-blocking rebuild model of Proposition 4. The system maintains p independent schemas. A deletion request independently invalidates each active instance with probability ρ (since $\mathcal{D}^\circ$ membership is secret and uniformly random). A rebuild costs t_{rebuild} in CPU, takes wall-clock time T_r, restores all p instances, and does not interrupt serving by surviving instances. At most one rebuild is in flight at a time. The *eager* policy starts a background rebuild whenever any instance is invalidated, if none is already in progress.

Proof of Proposition 4(i): Outage Bound. When eager triggers a rebuild, exactly $p-1$ instances remain active (the one just invalidated plus the in-progress rebuild). During the rebuild window T_r, the number of arriving deletions is $D \sim \text{Poisson}(\lambda_d T_r)$. For each deletion, each surviving instance is independently invalidated with probability ρ. Conditional on D, instance i survives all D deletions with probability $(1-\rho)^D$, and these events are independent across instances.

Marginal survival probability. The unconditional survival probability of a single instance is:

$$\alpha \;=\; \mathbb{E}\big[(1-\rho)^D\big] \;=\; e^{-\lambda_d T_r\,(1-(1-\rho))} \;=\; e^{-\rho\lambda_d T_r}. \tag{19}$$

Joint outage probability. A full outage requires all $p-1$ surviving instances to die. By inclusion-exclusion:

$$P_{\text{out}} = \sum_{k=0}^{p-1}(-1)^k \binom{p-1}{k} \mathbb{E}\big[(1-\rho)^{kD}\big] = \sum_{k=0}^{p-1}(-1)^k \binom{p-1}{k} e^{-\lambda_d T_r(1-(1-\rho)^k)}. \tag{20}$$

Lower bound. Since $f(x) = x^{p-1}$ is convex for $p \geq 2$ and $q(D) = 1-(1-\rho)^D \in [0,1]$:

$$P_{\text{out}} = \mathbb{E}\big[q(D)^{p-1}\big] \geq \big(\mathbb{E}[q(D)]\big)^{p-1} = (1-\alpha)^{p-1}. \tag{21}$$

Upper bound and tightness. The gap between (20) and (21) arises from the positive correlation: a large D realization simultaneously increases all instances' death probabilities. To quantify this, note that the joint survival probability of k instances is:

$$\mathbb{E}[(1-\rho)^{kD}] = e^{-\lambda_d T_r(1-(1-\rho)^k)} = \alpha^k \cdot e^{\lambda_d T_r\,(k\rho-(1-(1-\rho)^k))}. \tag{22}$$

The correction exponent satisfies $k\rho - (1-(1-\rho)^k) = \binom{k}{2}\rho^2 + O(k^3\rho^3)$. Substituting into (20) and comparing with the binomial expansion of $(1-\alpha)^{p-1}$, the multiplicative gap is:

$$\frac{P_{\text{out}}}{(1-\alpha)^{p-1}} = 1 + O\big(\tbinom{p-1}{2}\,\rho^2\lambda_d T_r\big)\,. \tag{23}$$

For $\rho = 0.5\%$, $p = 3$, $\lambda_d T_r = 10$: the correction is $1 + O(2.5 \times 10^{-4}) \approx 1.0003$. In the practical regime ($\rho \ll 1$), the lower bound (21) is therefore tight, and

$$P_{\text{out}} \approx (1 - e^{-\rho\lambda_d T_r})^{p-1}. \tag{24}$$

For ρ=0.5% and $\lambda_d T_r$=10: $P_{\text{out}} \approx 0.049^{p-1}$. At p=3: $P_{\text{out}} \approx 0.24\%$. □

Proof of Proposition 4(ii): Rebuild Overhead. We compare the number of rebuilds performed by the eager policy against the offline optimal (which knows the full deletion sequence in advance).

Eager rebuild rate. Starting from $k = p$ (all instances active), the next invalidation event has probability $1-(1-\rho)^p$ per deletion. For small ρ, this is approximately $p\rho$. Under eager, every invalidation triggers a rebuild, so the expected inter-rebuild gap is:

$$N_{\text{eager}} = \frac{1}{1-(1-\rho)^p} \approx \frac{1}{p\rho} \quad \text{deletions.} \tag{25}$$

Lower bound on any policy's rebuild rate. Without a rebuild, the instance count k can only decrease. Starting from $k = p$, the expected number of deletions until all p instances are invalidated is a coupon-collector problem: at state k, the next death has probability $1-(1-\rho)^k \approx k\rho$, giving expected waiting time $\approx 1/(k\rho)$. Summing:

$$N_{p\to 0} = \sum_{k=1}^{p} \frac{1}{1-(1-\rho)^k} \approx \frac{1}{\rho}\sum_{k=1}^{p}\frac{1}{k} = \frac{H_p}{\rho}. \tag{26}$$

Any policy, including the offline optimal, must rebuild at least once every H_p/ρ deletions on average, since without a rebuild the system inevitably reaches $k = 0$.

Competitive ratio. The ratio of rebuild rates is:

$$\frac{N_{p\to 0}}{N_{\text{eager}}} = \frac{H_p/\rho}{1/(p\rho)} = p\,H_p. \tag{27}$$

This bound is tight: an offline-optimal policy with instantaneous rebuilds ($T_r = 0$) can wait until $k = 0$ and rebuild without any outage, achieving inter-rebuild gap H_p/ρ. For $p = 3$: $p\,H_3 = 3 \cdot (1 + \frac{1}{2} + \frac{1}{3}) = 5.5$; for $p = 2$: $p\,H_2 = 3$. □

Algorithm 1 The RQA data structure. RQ.encode is the frozen Faiss encoder; the four RQA procedures operate on target values $\mathbf{Y}^r$; in QSS, $y_i^r = y_i - g(x_i)$ (§2.1).

Input: Feature map φ, RQ with b centroids/level, ℓ levels
State: $\mathbf{T}_{m,j} \in \mathbb{R}^k$, $n_{m,j} \in \mathbb{N}$, $N_{k,m} \in \mathbb{N}^{b \times b}$. Cache $\mathbf{S}_{m,j} \in \mathbb{R}^k$.

procedure RQA.BUILD($\mathbf{X}, \mathbf{Y}^r$)
  $\mathbf{C} \leftarrow$ RQ.encode($\varphi(\mathbf{X})$) ▷ $n \times \ell$
  for $m = 1$ **to** ℓ **do**
    $\mathbf{T}_{m,j} \leftarrow \sum_{i:\, C_{i,m}=j} y_i^r$ ▷ scatter-add
    $n_{m,j} \leftarrow |\{i : C_{i,m} = j\}|$
  end for
  for $k = 1$ **to** ℓ; $m = k{+}1$ **to** ℓ **do**
    $N_{k,m}[j', j] \leftarrow |\{i : C_{i,k}{=}j',\, C_{i,m}{=}j\}|$
  end for
  RQA.MATERIALIZE
end procedure

procedure RQA.PREDICT(x)
  $\mathbf{c} \leftarrow$ RQ.encode($\varphi(x)$)
  return $\sum_{m=1}^{\ell} \mathbf{S}_{m,c_m} / n_{m,c_m}$
end procedure

procedure RQA.UNLEARN(x, y^r)
  $\mathbf{c} \leftarrow$ RQ.encode($\varphi(x)$)
  for $m = 1$ **to** ℓ **do**
    $\mathbf{T}_{m,c_m} \mathrel{-}= y^r$; $n_{m,c_m} \mathrel{-}= 1$
    for $k = 1$ **to** $m{-}1$ **do**
      $N_{k,m}[c_k, c_m] \mathrel{-}= 1$
    end for
  end for
  RQA.MATERIALIZE
end procedure

procedure RQA.MATERIALIZE ▷ $O(\ell^2 b^2 k)$
  for $m = 1$ **to** ℓ **do**
    $\mathbf{S}_{m,j} \leftarrow \mathbf{T}_{m,j} - \sum_{k<m} \sum_{j'} N_{k,m}[j', j] \cdot \boldsymbol{\mu}_{k,j'}$
    $\boldsymbol{\mu}_{m,j} \leftarrow \mathbf{S}_{m,j} / n_{m,j}$ (**0** if $n_{m,j}{=}0$)
  end for
end procedure

Steady-state cost rate. To choose among rebuild thresholds, we generalize to the threshold-k^* policy: trigger a rebuild when the active instance count drops to k^*. The long-run average cost per deletion follows from renewal theory (ratio of expected cost per cycle to expected cycle length):

$$\mathcal{C}(k^*) = \frac{t_{\text{rebuild}} + P_{\text{out}}(k^*) \cdot C \cdot \lambda_p \cdot T_r}{N_{p \to k^*} + \lambda_d \cdot T_r}, \tag{28}$$

where $N_{p \to k^*} = \sum_{k=k^*+1}^{p} [1-(1-\rho)^k]^{-1}$ is the expected number of deletions from $k{=}p$ to $k{=}k^*{+}1$, and $P_{\text{out}}(k^*) \approx (1-e^{-\rho \lambda_d T_r})^{k^*}$ is the outage probability with k^* instances surviving during the rebuild window.

In the regime $C \gg t_{\text{rebuild}}$ (outage is catastrophic), the second term in the numerator dominates and $k^* = p{-}1$ (eager) minimizes (28). When C is moderate, the optimal threshold balances rebuild frequency against outage risk; for small p, this can be evaluated numerically.

A.9 RQA Pseudocode

B Codebook Fragility: Extended Analysis

We train RQ ($b{=}1024$, $\ell{=}8$) on 162,770 CelebA CLIP embeddings, remove 100 random points (0.06%), retrain, and re-encode. Result: 100% of remaining codes change (per-level flip rate 99.8–99.9%, 3 repeats). Minimum Voronoi margin is ≈ 0 at every level; cross-level dilution is 1.0, confirming cascades propagate at full strength.

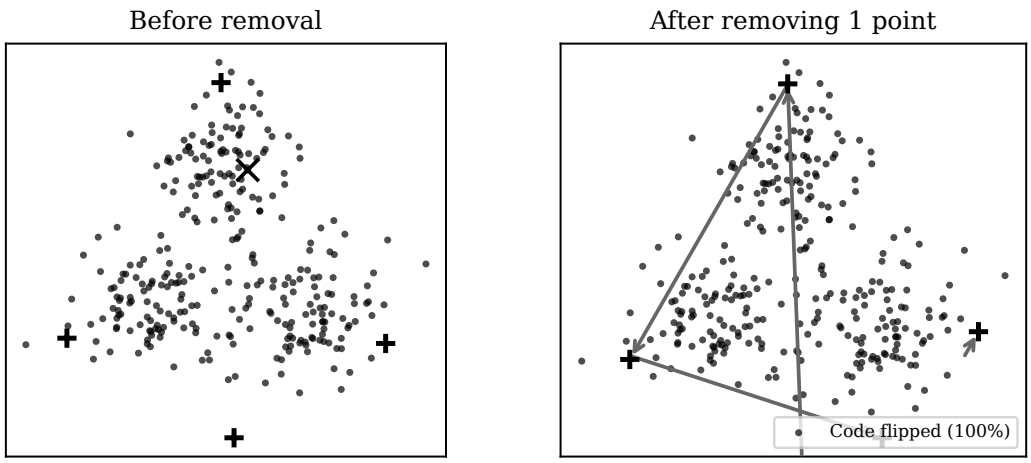


Figure 6: Codebook instability (§B) on 2D synthetic data (b=20, 300 points). **Left:** original k-means centroids (+) and Voronoi boundaries. **Right:** removing one point ($\times$) and retraining k-means shifts centroids (arrows), flipping 30% of codes (dark points). Flips occur far from the removed point, demonstrating that the perturbation is global. In 768-d CLIP embeddings, the flip rate is 100%.

C Additional Experiments

C.1 Aggregate Accuracy Patterns

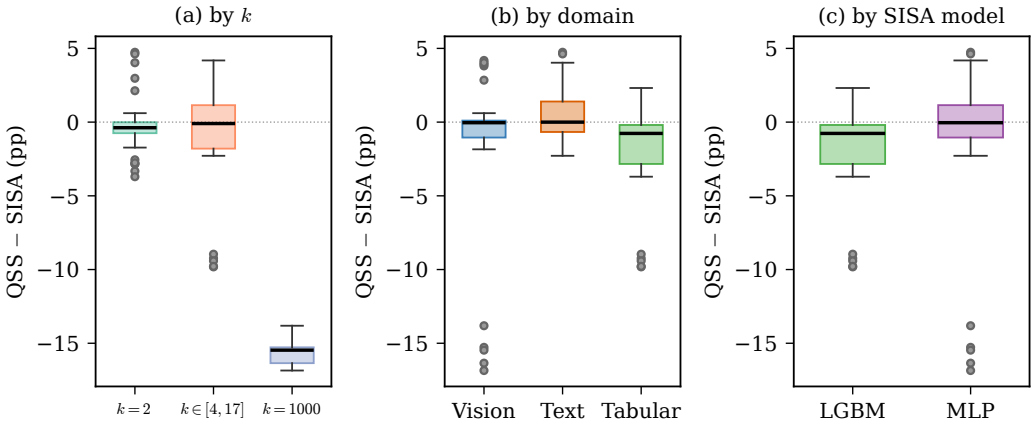


Figure 7: Aggregate view across all 15 datasets (ρ=0.5%, per-seed). **(a)** QSS-L–SISA accuracy gap by k: binary and low-k tasks cluster near zero; high-k (ImageNet) is a clear outlier (§4.5). **(b)** Gap by domain: all three modalities behave similarly. **(c)** Gap by SISA model type (LGBM, MLP): QSS is competitive regardless of baseline architecture.

C.2 Full Accuracy and Speedup

Fixed QSS-L and QSS-E accuracy for all 15 datasets appears in Table 11; the low-k QSS-L/SISA comparison appears in Table 4. We do not report a best-over-guarantees aggregate.

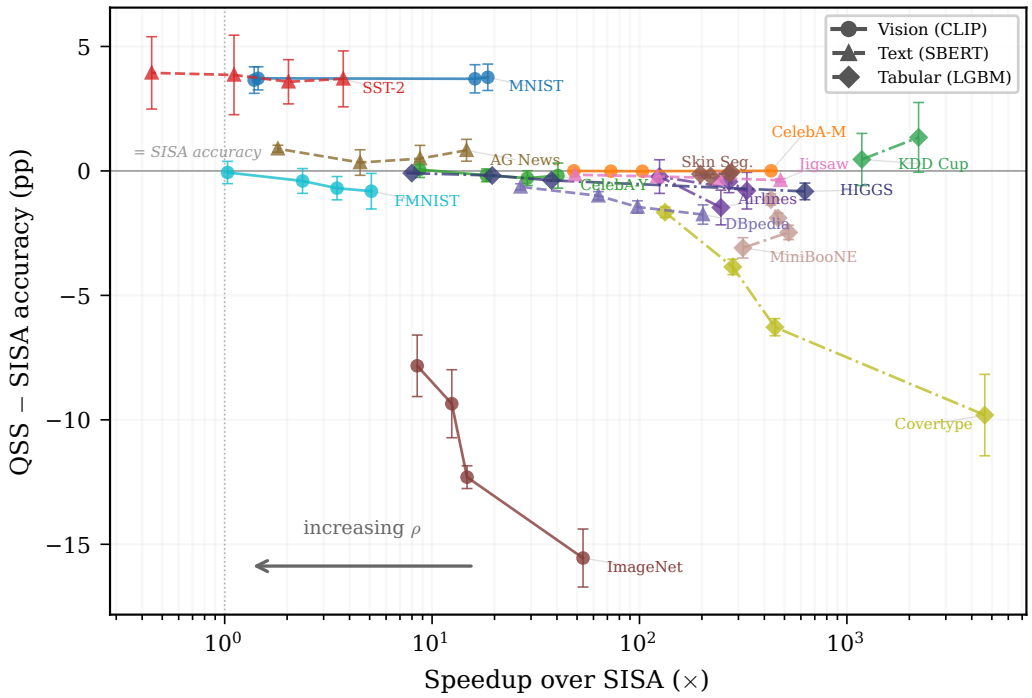


Figure 8: QSS-L accuracy–speedup Pareto as ρ varies from 0.5% to 5%, all 15 datasets. Both axes are relative to SISA (x=1, y=0). Each trajectory traces one dataset; moving *left* corresponds to increasing ρ (lower speedup, higher accuracy). Gaps on ImageNet and Covertype (high-k tasks) narrow with larger ρ. Full timing breakdown in Table 6.

C.3 Expected Unlearning Latency

Table 6: Expected unlearning latency at ρ=0.5%. $\mathbb{E}[\text{QSS-L}] = (1-\rho)\, t_{\text{RQ}} + \rho\, t_{\text{rebuild}}$ (Equation 8); SISA (S=5) retrains one shard. t_{rebuild} includes retraining g, re-extracting embeddings (for LGBM: leaf contributions), re-fitting LDA and codebook, re-encoding all n points, and re-accumulating. Per-component RQ timings appear in Table 10. †High-k limitation (§4.5).

Dataset	n	ρ	t_{reb} (s)	$\mathbb{E}$[QSS-L]	SISA	Speedup
Jigsaw	1.4M	0.5%	44	0.2 s	109 s	483×
CelebA Male	137K	0.5%	13	0.1 s	9.5 s	132×
MiniBooNE	130K	0.5%	30	0.2 s	17.5 s	114×
DBpedia	473K	0.5%	29	0.1 s	15.8 s	107×
Covertype†	581K	0.5%	850	4.3 s	98.7 s	23×
Airlines	500K	0.5%	50	0.3 s	11.0 s	54×
AG News	96K	0.5%	14	75 ms	3.8 s	51×
ImageNet†	998K	0.5%	227	1.1 s	52.9 s	46×
MNIST	53K	0.5%	13	68 ms	3.1 s	46×
CelebA Young	137K	0.5%	17	88 ms	3.6 s	41×
Skin Seg.	245K	0.5%	126	0.6 s	12.0 s	19×
FMNIST	53K	0.5%	64	0.3 s	2.5 s	8×
HIGGS	11M	0.5%	3,522	17.6 s	114 s	6×
KDD Cup	494K	0.5%	2,381	11.9 s	154 s	6×
SST-2	51K	0.5%	63	0.3 s	1.2 s	4×

C.4 Codebook Collapse

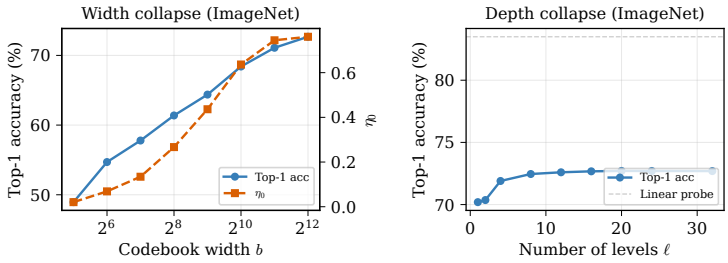


Figure 9: Codebook collapse on ImageNet (unsupervised RQ). Left: width collapse (accuracy and η_0 grow with b). Right: depth collapse (accuracy plateaus as $\eta_\ell \to 0$). See Appendix A.3 for the variance decay analysis that explains this convergence behavior.

C.5 Accuracy Retention Under Deletion

Table 7: Accuracy retention under random deletion. Unlearn and retrain-from-scratch produce bitwise identical predictions at every deletion fraction (Proposition 1).

	0%	1%	5%	10%	20%	30%	40%	50%
CelebA	99.35	99.35	99.34	99.36	99.33	99.33	99.32	99.33
ImageNet	72.67	72.67	72.66	72.65	72.66	72.65	72.48	72.44
Covertype	75.45	75.39	75.40	75.38	75.36	75.33	75.21	75.14
AG News	90.08	90.11	90.05	90.14	89.95	89.89	89.89	89.86

CelebA, ImageNet, Covertype, and AG News are stable through 50% deletion (<0.3pp loss). All four datasets are *bitwise identical* between unlearn and retrain-from-scratch at every fraction.

C.6 Membership Inference Attack

We evaluate membership privacy via LiRA [6]: 64 shadow QSS models (each trained on a random 50% subset of CelebA), auditing 1000 target points.

Table 8: LiRA membership inference on QSS (CelebA Male). The attack barely outperforms random guessing at all ε, including $\varepsilon{=}\infty$ (no DP noise).

ε	ROC AUC	TPR@FPR=0.1%	TPR@FPR=1%
1.0	0.515	0.2%	2.2%
2.0	0.518	0.8%	1.2%
4.0	0.531	0.2%	1.4%
∞ (no DP)	0.525	0.0%	0.4%

Even without DP, QSS leaks no membership information: AUC ≈ 0.52. Predictions depend on cell-level aggregates ($n_{m,j} \gg 1$), not individual points.

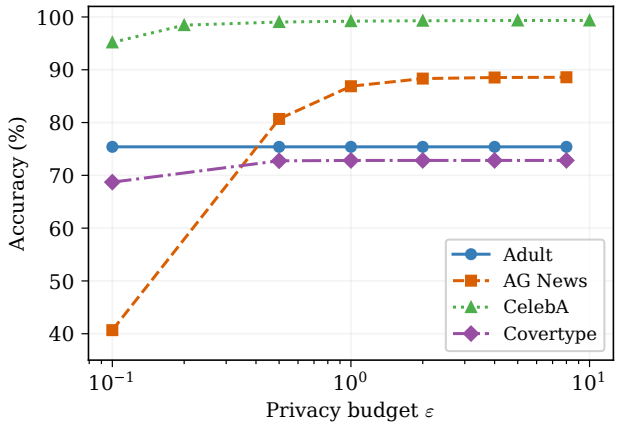


Figure 10: ε-label-DP accuracy. Adult (binary) is insensitive; Covertype (7-class) and AG News (4-class) degrade gracefully, retaining most accuracy at $\varepsilon \geq 1$.

C.7 Variance Decay and Theory Validation

Proposition 5 (Variance Decay) states that $\mathrm{Var}(r_m) = \mathrm{Var}(y^r) \prod_{j=1}^{m}(1-\eta_j)$. If $\eta_m \geq \gamma > 0$, MSE decays as $(1-\gamma)^\ell$. Proof in Appendix A.3. Excess risk scales as b/n, not b^ℓ/n (Appendix A.5).

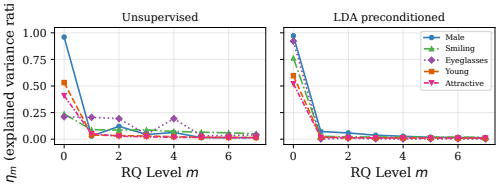


Figure 11: Per-level η_m across datasets.

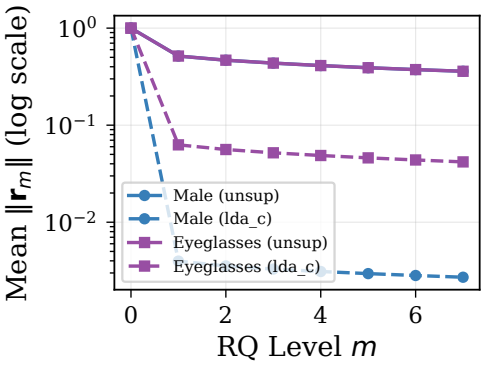


Figure 12: Residual norm decay across levels.

C.8 Ablations

We isolate the contribution of codebook width b, depth ℓ, and LDA preconditioning. All ablations use 5 seeds (0–4) on the DNN pipeline (MNIST, CIFAR-100, AG News, Covertype) at $|\mathcal{D}^\circ| \in \{500, 5{,}000\}$.

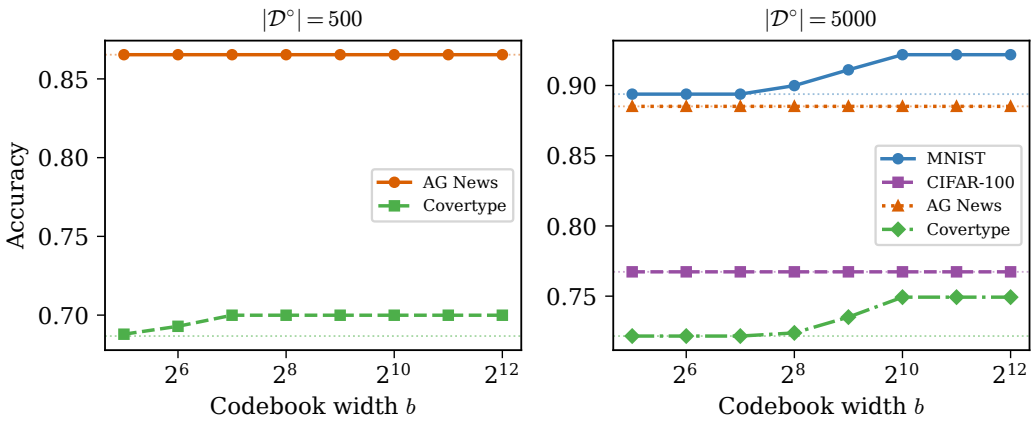


Figure 13: Accuracy vs. codebook width b (ℓ=4 fixed). Dotted lines show g-only baselines. Width matters most on MNIST and Covertype at $|\mathcal{D}^\circ|$=5,000, where larger b provides finer Voronoi cells that better separate classes. AG News and CIFAR-100 are flat ($\alpha \approx 0$).

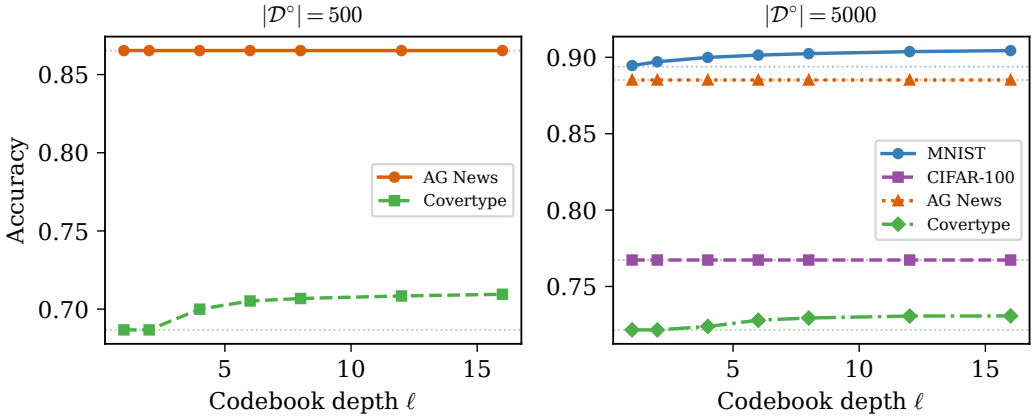


Figure 14: Accuracy vs. codebook depth ℓ (b=256 fixed). Covertype gains $\sim$2 pp from ℓ=1 to ℓ=8; other datasets saturate by ℓ=2–4, consistent with the depth collapse observed in Figure 9.

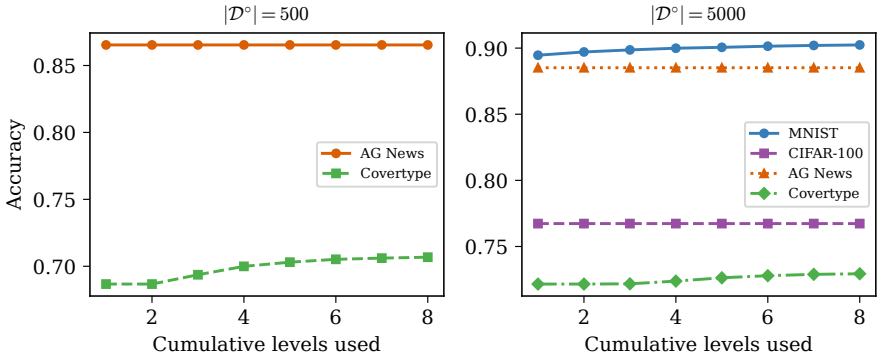


Figure 15: Cumulative accuracy as RQ levels are added. Level 1 captures most of the QSS lift; deeper levels provide diminishing returns, confirming the variance decay prediction (Proposition 5).

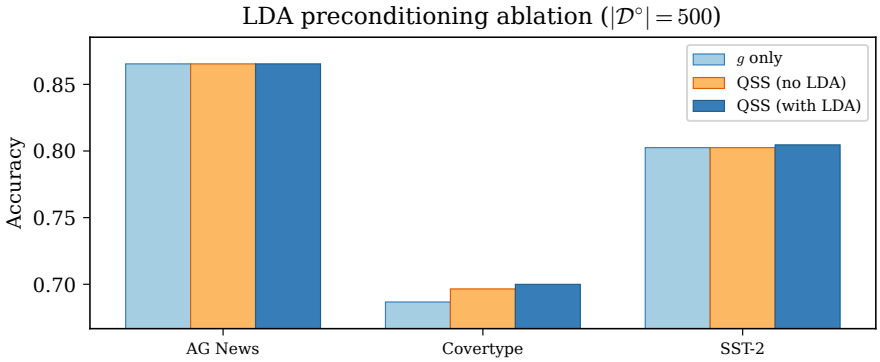


Figure 16: LDA preconditioning ablation ($|\mathcal{D}^\circ|$=500). LDA provides a modest lift on Covertype ($+1.5$ pp) where $k-1$=6 discriminant directions help the unsupervised codebook align with class boundaries. Effect is negligible on AG News and SST-2 where the pretrained SBERT embeddings already separate classes well.

C.9 Hyperparameter Sensitivity and Rebuild Efficiency

The codebook parameters (b, ℓ) jointly control accuracy and unlearning speed: $t_{\text{RQ}} = O(\ell^2 b^2 k)$ (Equation 8). Cross-validation selects (b, ℓ, α) to maximize held-out accuracy on $\mathcal{D}^\circ$; here we examine the full parameter surface to characterize robustness.

Accuracy is robust to (b, ℓ) when g is strong. Figure 17 sweeps b and ℓ independently on two representative datasets. On MiniBooNE (tabular, g-only $=$ .912), QSS accuracy is flat across the entire (b, ℓ) range: (b=32, ℓ=1) and (b=512, ℓ=8) yield the same .913 accuracy, because the blending weight $\alpha \approx 0$ lets the strong base model dominate. On FMNIST (vision, g-only $=$ .749), QSS accuracy increases with both b and ℓ: moving from ℓ=1 to ℓ=8 gains 1.5 pp, because the cell-mean correction contributes meaningfully when g is far from the ceiling. The practical consequence is that datasets where g already performs well (typically those with high-quality embeddings) can use minimal codebook parameters without sacrificing accuracy, reducing t_{RQ} by orders of magnitude (e.g., 0.02 ms at $(32, 1)$ vs. 9 ms at $(512, 8)$).

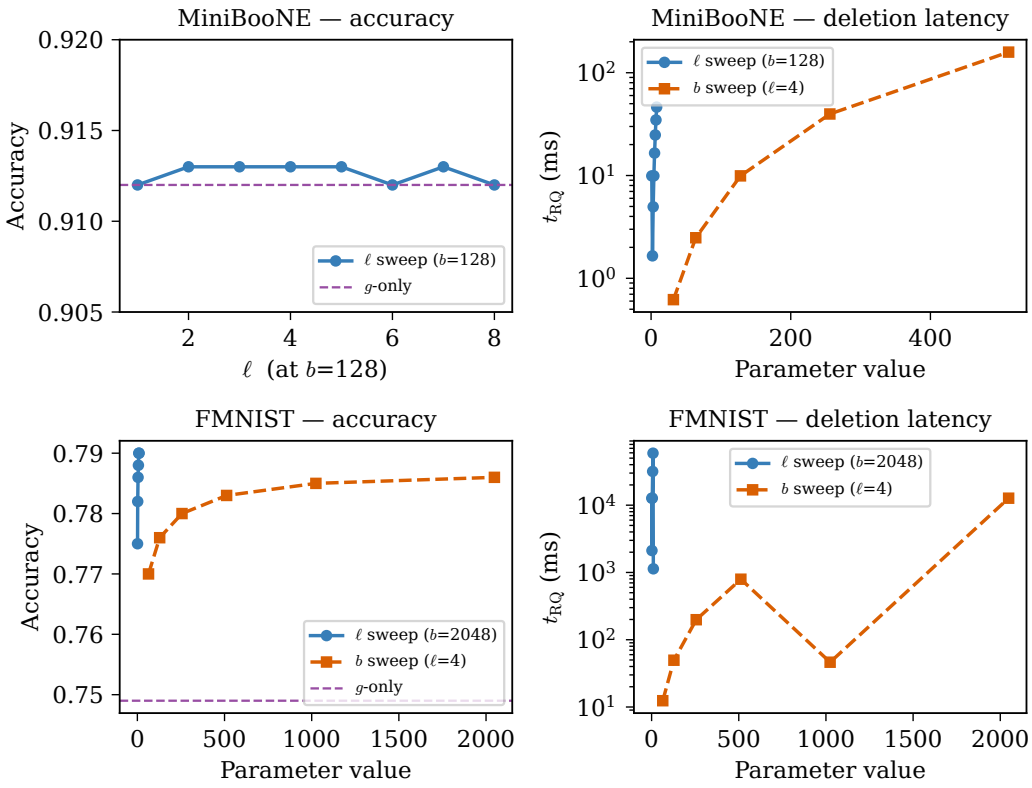


Figure 17: QSS accuracy (left) and t_{RQ} (right) as codebook parameters vary. **Top:** MiniBooNE (ℓ sweep at b=128; b sweep at ℓ=4). Accuracy is flat; t_{RQ} spans three orders of magnitude. **Bottom:** FMNIST (same sweeps). Accuracy improves with larger codebooks; we trade deletion speed for accuracy via CV.

Accumulator convergence. A full rebuild re-encodes all n points and re-accumulates their labels into the joint-count tables. Since these tables are sufficient statistics (cell-level sums and counts), they converge as the per-cell sample sizes grow. We measure accuracy as a function of the fraction of $\mathcal{D}^\bullet$ encoded into accumulators (Figure 18).

On tabular datasets with well-clustered LGBM embeddings, accumulators converge rapidly: Skin Segmentation reaches full accuracy at 1% of n (2.5K points), KDD Cup at 5%, HIGGS at 10%, and Airlines/MiniBooNE at 25% (Table 9). This reflects high cell purity: LGBM leaf embeddings place same-class points in tight clusters, so even a fraction of the data yields stable per-cell label distributions.

On DNN datasets (CLIP, MiniLM), embeddings are more diffuse and accumulators require the full dataset to converge. FMNIST loses 1 pp at 50% fill and 9.5 pp at 1%; SST-2 shows a similar pattern. QSS genuinely leverages all available data in this regime.

Table 9: Accumulator convergence on tabular datasets. *Saturation fraction*: smallest fraction of $\mathcal{D}^\bullet$ at which accuracy is within 0.15 pp of full. Subsampled rebuilds reduce the encode+accumulate cost proportionally.

Dataset	n	Saturation	Gap	Rebuild savings
Skin Seg.	245K	1% (2.5K)	−0.02 pp	100×
KDD Cup	494K	5% (25K)	−0.15 pp	20×
HIGGS	11M	10% (1.1M)	−0.02 pp	10×
Covertype	581K	10% (58K)	−0.40 pp	10×
Airlines	500K	25% (125K)	−0.05 pp	4×
MiniBooNE	130K	25% (33K)	−0.15 pp	4×

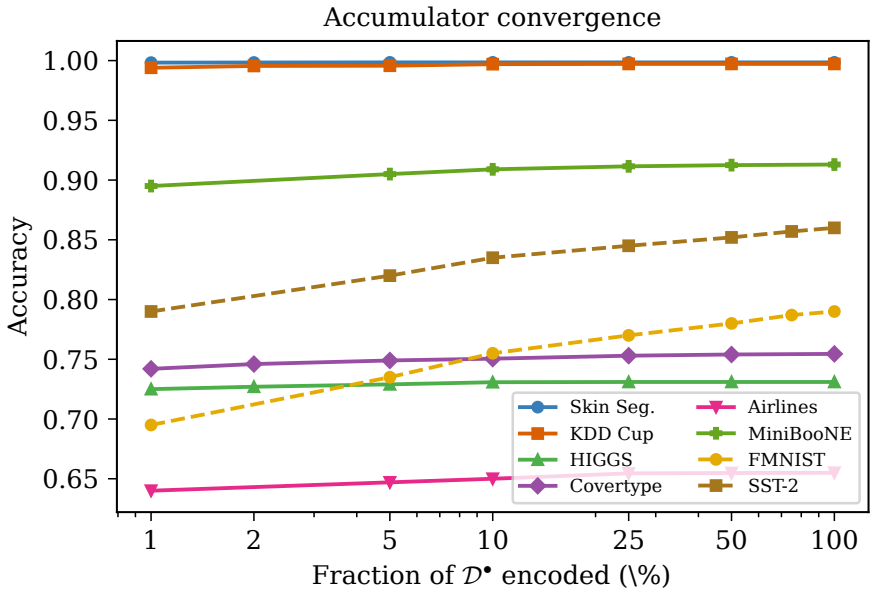


Figure 18: Accuracy vs. fraction of $\mathcal{D}^\bullet$ encoded into accumulators. Tabular datasets (solid) converge rapidly; DNN datasets (dashed) require most or all of the data. The convergence rate reflects embedding geometry: well-clustered embeddings yield high cell purity and fast convergence.

The convergence behavior has a direct practical implication for rebuild cost. The full rebuild pipeline has four stages: retrain g, retrain codebook (both on $\mathcal{D}^\circ$ only, independent of $|\mathcal{D}^\bullet|$), encode $\mathcal{D}^\bullet$, and accumulate. Only the last two stages scale with the number of points encoded. For datasets where accumulators saturate at fraction f, the encode+accumulate cost drops by $1/f$ while the g-training and codebook-training costs remain unchanged. On HIGGS (n=11M), encoding 1.1M points instead of 11M reduces the encode+accumulate phase from $\sim$9 s to $\sim$0.9 s, cutting total rebuild time and increasing the expected speedup over SISA accordingly.

C.10 Experimental Details

Base model g. Linear probe (logistic regression for classification, ridge regression for regression) trained on $\mathcal{D}^\circ$ via Optuna (200 trials, TPE sampler) over learning rate ($[10^{-5}, 10^{-1}]$), weight decay ($[10^{-6}, 10^{-2}]$), epochs ($\{10, 50, 100, 200\}$), and batch size ($\{32, 64, 128, 256\}$), minimizing validation loss on a 20% stratified holdout of $\mathcal{D}^\circ$. When $|\mathcal{D}^\circ| \geq 5{,}000$, we also evaluate a one-hidden-layer MLP ($D \to 128 \to k$, ReLU, Adam lr=1e-3, wd=1e-4, bs=256, early stopping patience=3 on validation loss) and report whichever yields higher validation accuracy.

Residual Quantization. Faiss [15] (version 1.7.4) `ResidualQuantizer` with `Train_default` (k-means++ initialization) and `max_beam_size`=1 (greedy nearest-centroid assignment), L_2 distance. Codebook hyperparameters (b, ℓ) selected by Optuna (30 trials per mode, TPE sampler) over $b \in \{64, 128, 256, 512, 1024, 2048\}$ and $\ell \in \{2, 4, 6, 8, 10, 12, 16\}$, with blending weight $\alpha \in [0, 2]$ (grid of 21 values) cross-validated on the same stratified holdout as the base model, optimizing validation accuracy (classification) or Pearson r (regression). All embeddings are L_2-normalized before codebook training.

LDA preconditioning. $W = W_{\text{LDA}} \in \mathbb{R}^{(k-1)\times d}$: all $k-1$ discriminant directions from standard LDA (scikit-learn `LinearDiscriminantAnalysis`, default solver) fitted on $\mathcal{D}^\circ$ labels and embeddings. This is analogous to Optimized Product Quantization [10], but uses label supervision from $\mathcal{D}^\circ$ rather than reconstruction error. Feature map: $\varphi(x) = L_2\text{-normalize}([f(x);\ W f(x)])$ (concatenation, not projection). When $\mathcal{D}^\circ = \varnothing$, $W = I$ (no preconditioning).

Encoders. Vision: OpenAI CLIP ViT-L/14 (`openai/clip-vit-large-patch14`, 768-d). Text: Sentence-Transformers `sentence-transformers/all-MiniLM-L6-v2` (384-d). Standard preprocessing per encoder (CLIP: 224×224 center crop; SBERT: default tokenizer, max 512 tokens).

Tabular encoder (LightGBM leaf contributions). For tabular datasets (Skin Segmentation, Adult, Covertype, California Housing), we train a LightGBM model [16] on $\mathcal{D}^\circ$ and extract per-tree leaf indices as the embedding. Configuration: 200 trees, max depth 6, learning rate 0.1, min data in leaf 20, bagging fraction 0.8, feature fraction 0.8, `num_leaves`=31. Each input is mapped to a 200-d binary vector (one-hot leaf index per tree, concatenated), yielding D=200. The LightGBM model is trained only on $\mathcal{D}^\circ$ and frozen, satisfying the same independence requirement as the CLIP/SBERT encoders. Leaf contributions are extracted via `model.predict(X, pred_leaf=True)`.

SISA. S=5 shards. Each shard trains the same model architecture as g on $\mathcal{D}^\circ \cup \text{shard}_k$. Predictions are majority-voted (classification) or averaged (regression).

Data splits. Each dataset is split into $\mathcal{D}^\circ$ (n° points selected uniformly at random, stratified by class for classification), $\mathcal{D}^\bullet$ (remainder minus test), and a held-out test set. Datasets with standard splits (ImageNet, CIFAR-100, MNIST, AG News, SST-2, DBpedia) use their official test sets; all others use a 25% random stratified test set. Results are averaged over 5 random splits (seeds 0–4); the seed controls $\mathcal{D}^\circ$ selection, RQ k-means initialization, and Optuna sampling.

Timing experiments. All timing comparisons are measured on the same GPU (NVIDIA A100), same process, median of 10 runs. SISA deletion retrains `nn.Linear`+SGD on $\mathcal{D}^\circ \cup$ affected shard with the same Optuna-selected hyperparameters. Table 10 reports the raw RQA data structure update time (the $\mathcal{D}^\bullet$ component of the expected cost in Table 6).

Table 10: RQA data structure update latency (median of 10 runs, A100 GPU): encode the deleted point, decrement accumulators, and re-materialize prediction tables. This is the t_{RQ} component of the expected QSS cost (Table 6).

Dataset	n	RQ (ms)	SISA (ms)	Speedup
CalHousing	21K	3.9	994	257×
UTKFace	24K	5.7	68,622	12,096×
DTD	5.6K	6.1	26,105	4,278×
Adult	45K	8.0	1,188	148×
Skin Seg.	245K	10.9	1,025	94×
SST-2	68K	12.0	1,303	109×
CIFAR-100	60K	12.0	31,563	2,620×
MNIST	70K	13.0	21,614	1,661×
AG News	128K	23.2	4,924	213×
Covertype	581K	88.4	10,495	119×
DBpedia	630K	186.7	20,642	111×
CelebA Young	1.3M	389.3	527,271	1,354×
ImageNet	1.3M	429.8	376,799	877×

C.11 Detailed Pipeline Flowchart

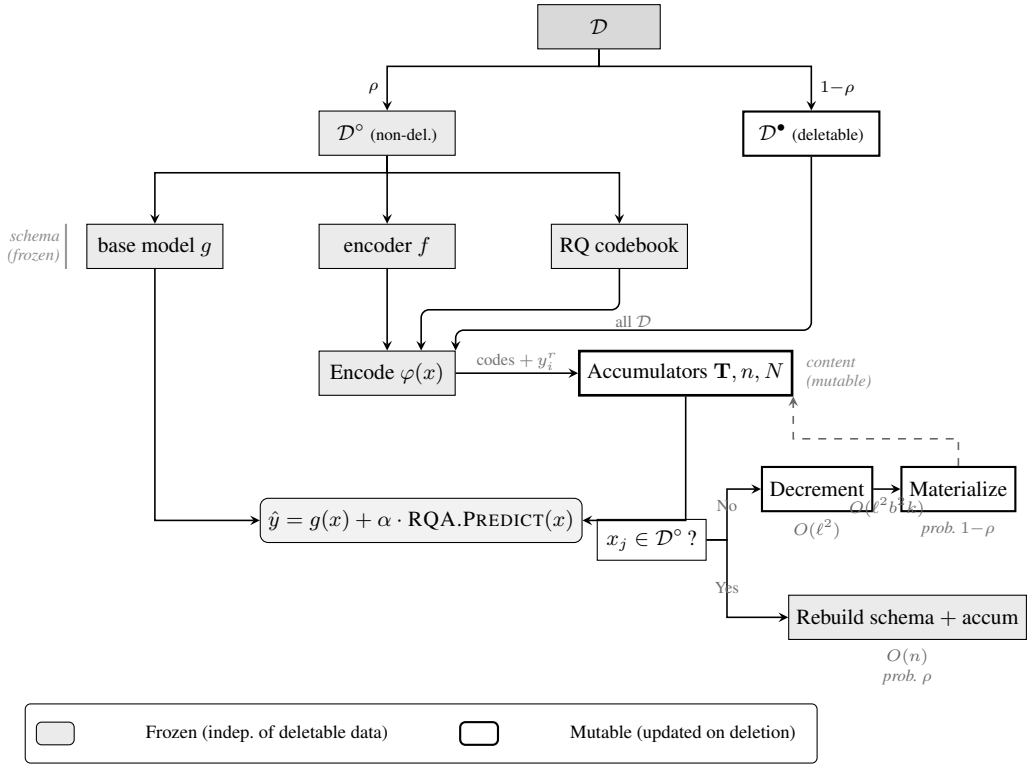


Figure 19: End-to-end QSS pipeline. A random fraction ρ of $\mathcal{D}$ forms the non-deletable subset $\mathcal{D}^\circ$, which defines the *schema*: encoder f, base model g, and the RQ codebook—all frozen before any deletion request. All of $\mathcal{D}$ is encoded and accumulated into mutable sufficient statistics (*content*). To unlearn a point from $\mathcal{D}^\bullet$ (probability $1-\rho$): decrement accumulators in $O(\ell^2)$ and re-materialize in $O(\ell^2 b^2 k)$—the schema is untouched. If the point is in $\mathcal{D}^\circ$ (probability ρ): a full rebuild is required at cost $O(n)$.

D Guarantee-Aligned and Analytic Comparisons

D.1 Fixed QSS-L and QSS-E results

Table 11 uses one fixed construction per guarantee rather than selecting between QSS-E and QSS-L using test accuracy. QSS-E trains its codebook on the retained schema; QSS-L uses the fixed primary label-deletion construction for each feature pipeline. QSS-L is more accurate on all 15 tasks (median difference 5.27 pp), quantifying the cost of requiring the codebook to be independent of every deletable input.

Table 11: Fixed-mode accuracy (%, mean ± standard deviation over five seeds) at $\rho = 0.5\%$. The required neighboring relation, not test accuracy, determines the mode.

Dataset	QSS-E	QSS-L	Dataset	QSS-E	QSS-L
MNIST	81.83±1.42	94.43±0.34	FMNIST	75.03±1.56	82.49±1.40
CelebA Young	89.24±0.10	89.34±0.52	CelebA Male	99.21±0.05	99.25±0.05
SST-2	79.79±0.65	85.36±1.97	AG News	84.62±0.53	89.89±0.42
DBpedia	91.24±0.97	95.80±0.42	Jigsaw	92.03±0.05	92.35±0.06
ImageNet	46.24±0.31	66.19±1.92	CIFAR-100	39.03±2.55	68.42±4.86
Skin Seg.	99.46±0.14	99.71±0.08	Airlines	57.80±0.17	64.02±0.22
MiniBooNE	86.88±0.21	91.24±0.37	HIGGS	63.52±0.05	72.98±0.08
KDD Cup	99.30±0.03	99.88±0.01			

For QSS-L, we additionally deleted batches of 1, 10, 100, and 1,000 labels on AG News. In every case, the source statistics, materialized cell values, and prediction scores were bitwise identical to rebuilding from $\mathcal{D}^{-y}$; prediction disagreement was zero.

D.2 Exact analytic heads

Ridge regression on a fixed representation is an important exact baseline. It stores $A = X^\top X + \lambda I$ and $B = X^\top Y$; deleting (x, y) applies $A \leftarrow A - xx^\top$ and $B \leftarrow B - xy^\top$, followed by a Sherman–Morrison downdate. We reconstructed the five submitted AG News and SST-2 splits, selected λ using only an internal split of each training set, and then fit the final head using all training labels.

Table 12: Exact ridge and fixed QSS-L on identical frozen-SBERT features and submitted splits. The QSS-L advantage on SST-2 is not statistically significant.

Dataset	Exact ridge	QSS-L	Difference	Paired p
AG News	89.468	89.894	+0.426 pp	0.048
SST-2	83.709	85.356	+1.647 pp	0.121

The Sherman–Morrison result agrees with a solve from rebuilt statistics to at most 1.11×10^{-12} coefficient error on AG News and 1.10×10^{-13} on SST-2. Neither family dominates computationally:

Table 13: State and ordinary-deletion costs, both independent of dataset size n. Here d is analytic feature dimension and K the number of outputs.

Method	State	Ordinary deletion
Ridge/ACU	$O(d^2 + dK)$	$O(d^2 + dK)$
QSS	$O(\ell^2 b^2 + \ell b K)$	$O(\ell^2 b^2 K)$ conservative

D.3 Orthogonal schema and correction choices

QSS combines two design decisions that need not be made together. The first is *what target to model*: one may fit the target directly on a fixed representation, or first fit a retained-schema predictor g and model the residual $y - g(x)$. This choice determines whether the correction depends on a sampled schema. Whenever g changes after a schema deletion, every residual defined relative to that copy of g must be rebuilt, independently of how the residual is stored.

The second decision is *how to represent the mutable correction*. A linear or ridge correction stores global analytic sufficient statistics such as $X^\top X$ and $X^\top Y$, as in ACU and the centralized construction of Quan et al. [14, 24]. QSS instead stores local sums and counts indexed by a frozen quantizer. This choice determines the hypothesis class and the state/update costs, rather than whether a schema is present.

Consequently, a standalone analytic head has no sampled-schema event; a QSS predictor uses a schema-relative quantized residual; and a composed method can use a schema predictor with an analytic residual head. In the composed case, each replicated schema owns its base predictor and residual statistics, so a schema hit invalidates only that copy. Analytic heads are preferable when a modest-dimensional linear correction is expressive, while quantized statistics provide local nonlinear capacity without introducing an explicit high-dimensional feature expansion.

D.4 Larger SISA shard counts

Increasing S can make an affected shard faster to retrain, but it does not remove the coupling between deletion work and data per constituent model. Table 14 holds each model architecture fixed while extending SISA through $S = 200$.

D.5 Schema capacity and retained-set size

The schema head should be selected jointly with ρ. At the intended $\rho = 0.5\%$ regime, additional depth is often variance-limited: linear, one-layer, and three-layer heads obtain $81.06/75.72/78.39\%$

Table 14: SISA accuracy and isolated affected-shard retraining time.

Dataset	$S = 5$	$S = 50$	$S = 200$
Adult	83.30% / 6.1 ms	83.32% / 2.2 ms	81.91% / 1.2 ms
Covertype	71.99% / 3.379 s	71.46% / 0.160 s	70.60% / 0.0256 s
SST-2	80.73% / 0.1947 s	79.82% / 0.1050 s	78.56% / 0.0035 s

on Adult, $71.17/69.53/71.82\%$ on Covertype, and $72.40/73.47/74.46\%$ on SST-2. Three-layer training is $8.2\times$ the linear-head time on Covertype and $3.7\times$ on SST-2.

More retained data can make depth useful, at the cost of a larger schema. On Covertype, the three-layer advantage over linear grows from 0.65 pp at $\rho = 0.5\%$ to 5.75 pp at 2% and 10.74 pp at 5%; its training time at 5% is 14.65 s versus 0.86 s for linear. At 5%, corresponding three-seed QSS-L predictors obtain $72.90/77.20/83.83\%$ with linear, one-layer, and three-layer schemas. The pattern is not universal: SST-2 favors three layers at 0.5%, one layer at 2%, and one/two layers are effectively tied at 5%. This is the practical scope boundary: richer schemas can improve accuracy, but increasing ρ raises both rebuild probability and rebuild cost.

E Unsupervised Quality Diagnostics

Table 15: Without preconditioning, η_0 predicts quality.

Attribute	η_0	PR-AUC
Male	0.960	0.998
Young	0.532	0.959
Attractive	0.409	0.832
Smiling	0.237	0.887
Eyeglasses	0.210	0.851

E.1 Cell Purity Across Modalities

Figure 20 shows UMAP projections of test-set embeddings from four datasets, shaded by level-0 cell purity (the fraction of each cell's points belonging to the majority class). MNIST (CLIP, k=10) achieves 96.6% average purity at level 0: the unsupervised RQ codebook already separates classes well, which explains QSS's strong performance. At the other extreme, Covertype (k=7, b=64) reaches only 63.1%. CIFAR-100 (k=100) and AG News (k=4) fall between.

Multi-level RQ drives effective purity above 99.8% on all four datasets (level-0 impurity is corrected by deeper levels), but the per-level improvement depends on the base purity: when level 0 is already high, fewer levels suffice.

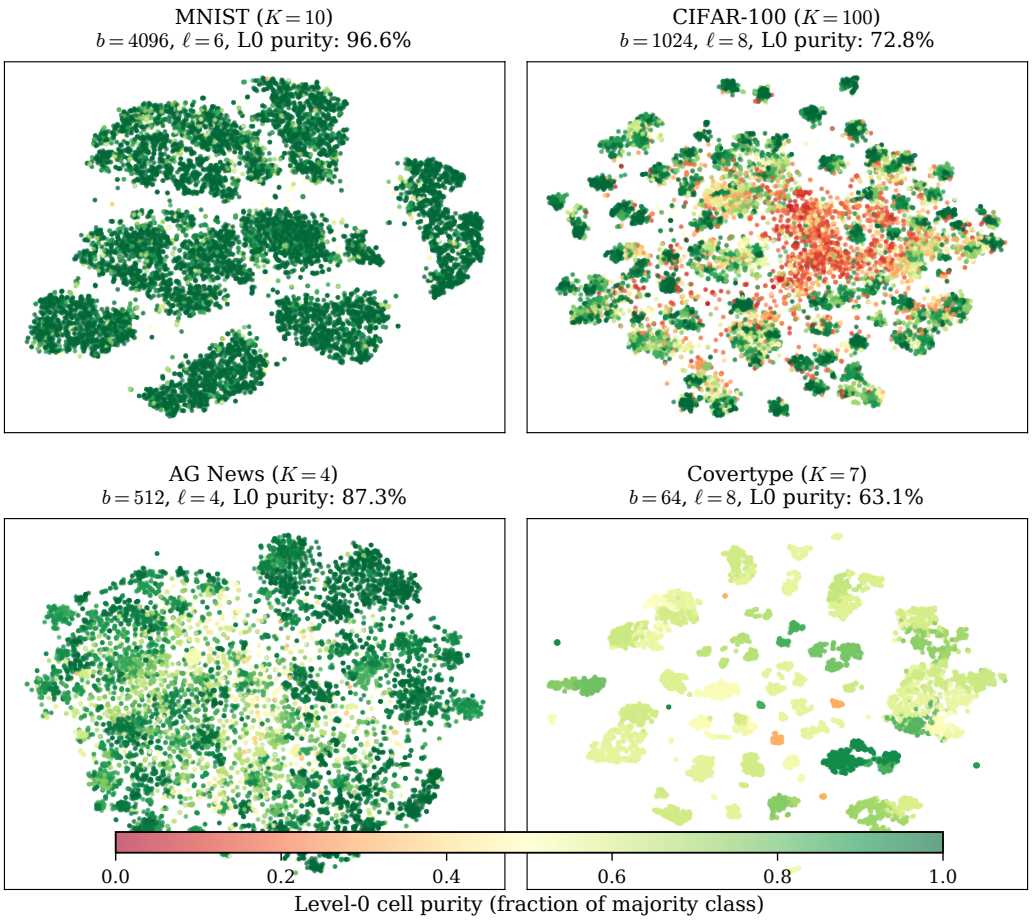


Figure 20: UMAP projections of test-set embeddings, shaded by level-0 cell purity (fraction of each cell belonging to the majority class). Dark regions indicate pure cells (>90%); light regions indicate mixed cells. MNIST (96.6% average purity) achieves the smallest QSS-L–SISA gap (+3.8 pp); Covertype (63.1%) has the largest among low-k tasks (-10.7 pp).

NeurIPS Paper Checklist

1. **Claims**

   Question: Do the main claims made in the abstract and introduction accurately reflect the paper's contributions and scope?

   Answer: [Yes]

   Justification: The abstract and introduction distinguish exact label deletion (QSS-L) from exact instance deletion (QSS-E), state the randomized rebuild event, and scope the empirical claims to the corresponding construction. Theoretical results appear in §3 and guarantee-aligned experiments in §4 and the appendices.

2. **Limitations**

   Question: Does the paper discuss the limitations of the work performed by the authors?

   Answer: [Yes]

   Justification: §4.5 discusses high-k accuracy gaps to SISA, rebuild costs, and schema capacity. Remark 6 discusses depth collapse, §3.2 treats adversarial deletion batches, and Appendix D.2 compares exact analytic heads.

3. **Theory assumptions and proofs**

   Question: For each theoretical result, does the paper provide the full set of assumptions and a complete (and correct) proof?

   Answer: [Yes]

   Justification: §3 states the assumptions for exact unlearning, label DP with Gaussian noise, rebuild scheduling, and variance decay. Proofs and supporting analysis appear in Appendix A. The sharper sensitivity bound discussed in Appendix A.4 is presented as empirical evidence rather than a proved guarantee.

4. **Experimental result reproducibility**

   Question: Does the paper fully disclose all the information needed to reproduce the main experimental results of the paper to the extent that it affects the main claims and/or conclusions of the paper (regardless of whether the code and data are provided or not)?

   Answer: [Yes]

   Justification: Algorithm 1 specifies the data structure. §4.1 describes all datasets, encoders, data splits, baselines, and hyperparameter selection. All datasets are publicly available.

5. **Open access to data and code**

   Question: Does the paper provide open access to the data and code, with sufficient instructions to faithfully reproduce the main experimental results, as described in supplemental material?

   Answer: [Yes]

   Justification: Code is available at `https://github.com/atavory/rqu`. All datasets used are publicly available (CelebA, ImageNet, CIFAR-100, Covertype, etc.).

6. **Experimental setting/details**

   Question: Does the paper specify all the training and test details (e.g., data splits, hyperparameters, how they were chosen, type of optimizer) necessary to understand the results?

   Answer: [Yes]

   Justification: §4.1 specifies data splits ($|\mathcal{D}^\circ|$, random selection), base model architecture (linear probe or MLP with stated dimensions), cross-validation procedure for (b, ℓ, α), and SISA configuration ($S{=}5$).

7. **Experiment statistical significance**

   Question: Does the paper report error bars suitably and correctly defined or other appropriate information about the statistical significance of the experiments?

   Answer: [Yes]

Justification: Main results are 5-seed means over random train/test/$\mathcal{D}^\circ$ splits. The number of seeds is stated in the table captions.

8. **Experiments compute resources**

   Question: For each experiment, does the paper provide sufficient information on the computer resources (type of compute workers, memory, time of execution) needed to reproduce the experiments?

   Answer: [Yes]

   Justification: Appendix C.10 identifies the NVIDIA A100 GPU used for timing comparisons. Table 6 reports rebuild and expected deletion times.

9. **Code of ethics**

   Question: Does the research conducted in the paper conform, in every respect, with the NeurIPS Code of Ethics `https://neurips.cc/public/EthicsGuidelines`?

   Answer: [Yes]

   Justification: The research addresses data privacy (right to deletion) and uses only publicly available datasets. No human subjects or personally identifiable data were collected.

10. **Broader impacts**

    Question: Does the paper discuss both potential positive societal impacts and negative societal impacts of the work performed?

    Answer: [Yes]

    Justification: §1 motivates deletion for privacy compliance and removal of poisoned or biased training data. §3.2 discusses operational risks from adversarial deletion batches, and §4.5 discusses accuracy and rebuild-cost limitations.

11. **Safeguards**

    Question: Does the paper describe safeguards that have been put in place for responsible release of data or models that have a high risk for misuse (e.g., pre-trained language models, image generators, or scraped datasets)?

    Answer: [N/A]

    Justification: The paper does not release pre-trained models, generative models, or scraped datasets. QSS is an unlearning method, not a generative system.

12. **Licenses for existing assets**

    Question: Are the creators or original owners of assets (e.g., code, data, models), used in the paper, properly credited and are the license and terms of use explicitly mentioned and properly respected?

    Answer: [Yes]

    Justification: All datasets and models are cited (CLIP, SBERT, Faiss, CelebA, ImageNet, etc.). All are publicly available under their respective licenses.

13. **New assets**

    Question: Are new assets introduced in the paper well documented and is the documentation provided alongside the assets?

    Answer: [Yes]

    Justification: The QSS implementation is available at `https://github.com/atavory/rqu` with documentation and scripts to reproduce the main experiments.

14. **Crowdsourcing and research with human subjects**

    Question: For crowdsourcing experiments and research with human subjects, does the paper include the full text of instructions given to participants and screenshots, if applicable, as well as details about compensation (if any)?

    Answer: [N/A]

    Justification: No crowdsourcing or human subjects research was conducted.

15. **Institutional review board (IRB) approvals or equivalent for research with human subjects**

    Question: Does the paper describe potential risks incurred by study participants, whether such risks were disclosed to the subjects, and whether Institutional Review Board (IRB) approvals (or an equivalent approval/review based on the requirements of your country or institution) were obtained?

    Answer: [N/A]

    Justification: No human subjects research was conducted.

16. **Declaration of LLM usage**

    Question: Does the paper describe the usage of LLMs if it is an important, original, or non-standard component of the core methods in this research? Note that if the LLM is used only for writing, editing, or formatting purposes and does *not* impact the core methodology, scientific rigor, or originality of the research, declaration is not required.

    Answer: [N/A]

    Justification: LLMs were used only for writing assistance and editorial polishing. They are not a component of the QSS method or experimental pipeline.