Privacy-Utility Optimization for Data Minimization

Zhendong Guo*, Shuhao Liu†, Wenfei Fan†‡, Jiahui Jin*
*Southeast University, China †Shenzhen Institute of Computing Sciences, China ‡University of Edinburgh, UK
zdguo@seu.edu.cn, shuhao@sics.ac.cn, wenfei@inf.ed.ac.uk, jjin@seu.edu.cn

***Abstract*—This paper studies compliance-aware relational data minimization for machine learning training. We formulate the problem as deriving a minimized dataset that reduces privacy risk while preserving model utility within a specified tolerance, and show that the problem is NP-complete. To address the challenge, we propose TRIM, a framework that jointly optimizes vertical and horizontal minimization. The key idea is a *retention view*: starting from a fully suppressed dataset, TRIM progressively restores information by attribute refinement and retention-class inclusion. Guided by a ratio-marginal objective, TRIM explicitly balances utility recovery against privacy leakage. To efficiently evaluate candidate operations, TRIM employs a two-stage utility estimator, avoiding retraining the target model. TRIM guarantees to terminate: it returns a dataset that passes a final utility check; it also logs the utility gain and privacy cost of every committed operation, producing auditable records for regulatory reviews. Using real-world and synthetic datasets, we experimentally show that TRIM reduces utility degradation by up to 92.9% over the best data-minimization baselines at matched privacy level, scales to 950K records, and supports DPIA review.**

I. Introduction

Machine learning (ML) models are often trained on datasets containing rich personal information [1], such as the patient records in Table I. Under the General Data Protection Regulation (GDPR), training data should be limited to what is necessary for the task, and this minimization should be auditable [2], [3] for, *e.g.,* a Data Protection Impact Assessment (DPIA) [4]. When a controller retains personal data that the task does not need, downstream training pipelines can access it, and a breach may expose it [5], [6]. Removing it reduces this exposure and prevents leaks through the model [7]–[9].

Prior approaches to training-data minimization typically perform either *vertical minimization*, which removes or generalizes attributes [3], [10], or *horizontal minimization*, which removes records through sampling, coresets, or record selection [11]–[16]. Neither suffices alone, as Example 1 shows.

Example 1: Table I shows medical records adapted from the `Diabetes` dataset [17] for training a readmission classifier. An adversary who knows a patient's `age` and `diagnosis` can pinpoint their record, as no two records share the same pair. Every patient is thus uniquely identifiable.

Vertical minimization coarsens attributes, *e.g.,* `age` to decades and `diagnosis` to ICD-9 families, which leaves t_1 and t_2 indistinguishable, as well as t_3 and t_5. However, t_4, the only patient with a hypertensive disease, is still unique; hiding it needs the far coarser *circulatory disease* level, which blurs every other record with the same value and hurts the classifier utility. Horizontal minimization fares no better: deleting records leaves whoever is kept just as identifiable. At a given utility tolerance, vertical methods may leave sensitive attributes insufficiently generalized and retain low-utility records that increase re-identification risk, and horizontal methods may preserve highly identifying attribute combinations.

Table I. Records from `Diabetes` [17] (†: QI, quasi-identifier).

tuple	age†	diagnosis†	A1C	readmitted
t_1	79	410.41 (Ischemic / Heart attack)	>8	Yes
t_2	72	414.01 (Ischemic / Atherosclerosis)	None	No
t_3	86	428.0 (Other heart / Heart failure)	Norm	Yes
t_4	82	402.00 (Hypertensive / Heart disease)	None	No
t_5	88	427.31 (Other heart / Arrhythmia)	>7	Yes

Vertical minimization via coarsening (`age`→decade, `diagnosis`→ICD-9 family): $t_1, t_2 \rightarrow$ (70s, Ischemic), $t_3, t_5 \rightarrow$ (80s, Other heart) become alike; yet $t_4 \rightarrow$ (80s, Hypertensive) remains uniquely identifiable.
Horizontal minimization via deletion (*e.g.,* keep t_1, t_2, t_3): each kept record is unique.
Both (coarsen as above and drop t_4): no retained record is unique

Combining both succeeds: coarsening age to decades and diagnosis to families, then dropping t_4, leaves every retained patient indistinguishable from at least one other while preserving diagnosis-family information for classification. □

Effective training-data minimization must thus jointly determine (a) which records to retain and (b) at what level of detail their attributes should be represented. This raises a question: what constitutes a "good" minimized training dataset?

The GDPR answers this via the DPIA [4], which yields three requirements: (1) *utility preservation*: the retained data must be necessary for the purpose, *i.e.,* maintain the utility required by the target ML task; (2) *privacy minimization*: the remaining risk must be proportionate to that purpose, *i.e.,* re-identification risk must be minimized; and (3) *decision accountability*: retention decisions must be documented, *i.e.,* each retained attribute/record needs an auditable justification.

Meeting these requirements is challenging. (1) Utility and privacy conflict: generalizing attributes or removing records lowers risk but may hurt utility, and the risk of a record depends on which other records are retained, *e.g.,* t_4 is unique at the family level only because no other hypertensive patient in their 80s is retained. (2) Utility is expensive to evaluate: the space of record-retention and attribute-generalization decisions is combinatorial (over $2^{50,000}$ record subsets alone for the `Diabetes` training data), and retraining the model for each candidate is prohibitive. (3) Optimization alone is not adequate: a DPIA reviewer needs to know why each attribute or record is retained, not only the resulting dataset, *e.g.,* why t_4 is dropped but `diagnosis` is kept at the family level.

Contributions & Organization. To address these challenges, we develop TRIM, a compliance-aware TRaIning data Minimization framework that jointly optimizes privacy and

utility, and produces full decision traces for DPIA audits.

(1) Compliance-aware data minimization (Section III). We formulate training-data minimization as an optimization problem that jointly decides which records to retain and how to generalize their attributes. It minimizes privacy risk, measured by the worst-case re-identification probability of any individual, while preserving the utility of a target model within a tolerance ϵ. We show that it is NP-complete.

(2) Retention-view optimization (Section IV). TRIM integrates vertical and horizontal minimization in a *retention view*: it starts from a fully suppressed dataset and progressively restores information via attribute refinement and retention-class inclusion. The process follows a *ratio-marginal optimization strategy* that prioritizes the operation with the largest estimated utility gain of the trained model per unit privacy cost.

(3) Efficient and auditable minimization (Sections V–VI). To make this strategy practical, TRIM enumerates candidate operations from a bounded retention-based search space that is both semantically meaningful and auditable. A two-stage utility estimator then ranks the candidates and evaluates only the top-ranked ones with a proxy model, so that TRIM avoids repeated and expensive retraining of the target model. Together, these components ensure that TRIM terminates after polynomially many operations with a dataset that meets the utility tolerance, and logs an auditable justification for every retention decision.

(4) Experimental study (Section VII). Using real-world and synthetic datasets, we show that TRIM (a) improves the privacy-utility frontier, reducing utility degradation by up to 92.9% over the best data-minimization baselines at matched privacy, with consistent gains in tail risk and across downstream models; (b) is up to $115\times$ faster than worst-case risk-reducing baselines, scaling to 950K records; (c) produces auditable DPIA traces, with stable default hyperparameters; and (d) derives its gains from joint horizontal-vertical minimization, ratio-marginal selection, and two-stage utility estimation.

We discuss related work in Section II and summarize the novelty in Section VIII. We defer detailed proofs to [18].

II. Background and Related Work

This section outlines regulatory concepts of data minimization and reviews existing techniques potentially applicable.

Key concepts. We start with relevant concepts.

Personal data. Under modern privacy regulations such as GDPR [4], personal data refers to any information relating to an identified or identifiable natural person. This includes both *direct identifiers* (*e.g.,* names or IDs) and *quasi-identifiers (QIs)* (*e.g.,* locations) whose combinations may enable re-identification. In ML training datasets, direct identifiers are typically removed due to their irrelevance to the task, therefore we focus on QIs as the main source of privacy risk.

DPIAs. A Data Protection Impact Assessment (DPIA) is an *ex ante* assessment process for high-risk processing of personal data [4]. It requires organizations to justify the *necessity* and *proportionality* of their use of personal data and to document the resulting privacy risks and mitigation measures. In this paper, training-data minimization serves as a practical measure both to evaluate privacy risks and to support DPIA obligations; the goal is to reduce and document residual risk in the minimized dataset rather than to certify anonymity.

Table II. Training-data minimization methods (△: partial).

Method	Horizontal	Vertical	Quantified Risk	Utility Constraint	Auditable
Horizontal minimization	✓	✗	✗	✓	✗
Vertical minimization	✗	✓	△	△	△
k-anonymization	△	✓	✓	△	△
Feature selection	✗	✓	✗	✗	✓
Differential privacy	✗	✗	✓	✗	✗
Synthetic data	△	✗	△	✗	✗
TRIM	✓	✓	✓	✓	✓

Example 2: Continuing with Example 1, a DPIA for the readmission classifier should justify why the granularity of every retained attribute and tuple of Table I is the least intrusive one that remains adequate, quantify the re-identification risk of the retained dataset (*e.g.,* of matching the patient of t_4 to that record), and record each minimization decision with its utility impact and privacy-risk change. □

Datasets. Consider a relational dataset $\mathcal{D}$ over schema $\mathcal{R} = \{R_k\}_{k=1}^{K}$. For ML training, we assume that $\mathcal{R}$ contains a single relation R, and direct identifiers are removed from R. The QIs, specified for each dataset, are the attributes of R that an adversary can obtain from external sources, *e.g.,* demographics; a controller designates any attribute it deems obtainable as a QI. We retain the other attributes that do not affect re-identification as given, since their necessity is a feature-selection question [19]. Let $A = \{A_1, \ldots, A_p\} \subseteq \mathsf{attr}(R)$ be the QIs. For each tuple $t_i \in \mathcal{D}$, $A(t_i)$ is the QI record associated with person i in population $\mathcal{I}$.

Data minimization. Article 5(1)(c) of the GDPR [4] requires personal data to be limited to what is *necessary* for a *specified purpose*. For ML training, this means deriving a minimized training dataset that preserves the required model utility while reducing the amount and granularity of retained personal data. We model minimization as a sequence of *minimization operations* on $\mathcal{D}$, in two categories:

- *Horizontal minimization*: reducing the retained records by removing tuples not required for the target ML task.
- *Vertical minimization*: reducing the information retained per record by removing attributes or replacing fine-grained values with coarser generalizations.

Minimization aims at a dataset that balances utility and privacy risk with auditable justifications for retained data.

Related work. We categorize prior approaches as follows, with a comparison of their key characteristics in Table II.

Data minimization for ML pipelines. Previous approaches typically focus on either *horizontal* or *vertical* minimization. Horizontal approaches [3], [11], [12], [20], [21] often adopt sampling or coreset selection, but may leave highly identifying attributes intact and provide limited support for auditability.

Vertical approaches [3], [10], [22], [23] typically optimize heuristic privacy-utility objectives. Across both categories, privacy risk and model utility are rarely enforced jointly through explicit objectives and constraints.

TRIM (1) minimizes an explicit privacy-risk objective subject to a utility constraint, (2) combines horizontal and vertical minimization, (3) efficiently estimates the utility impact of candidate operations without repeated retraining, and (4) provides auditable justifications for data-retention decisions.

Privacy measures. Prior work adopts the following metrics for dataset privacy risks. (1) *Data volume*, *e.g.,* the number of retained samples or entries [3], [11], [20]. It is misaligned with re-identification risk [3], as deleting records may leave every kept record unique. (2) *Empirical attacks*, *e.g.,* evaluated with attack suites [6], [10], [22]. Their success is average-case, depends on the attacks chosen, and is costly to measure for every candidate dataset. (3) *Formal guarantees*, including predicate-singling-out (PSO) [24] and differential privacy (DP) [25]. Both certify release mechanisms for anonymity claims, rather than quantify the risk of a given dataset, which a DPIA requires because the minimized dataset remains personal data. (4) *Attribute disclosure*, *e.g.,* ℓ-diversity [26] and t-closeness [27]. They limit what the QIs reveal about the label, which is the association the model must learn [28].

TRIM measures privacy by matching risk, a graded form of k-anonymity. Under our threat model, it is the success probability of the optimal attack on the most exposed individual, computed from class sizes for every candidate. This is auditable and better aligned with the data minimization principle: it assigns a risk to each dataset, data subject, and minimization operation, and leaves the QI-label association intact.

Training data reduction. Several related techniques can reduce training data, although designed for different objectives.

(1) k-anonymization. k-anonymity [29] and its utility-aware variants [2], [30]–[38] generalize QIs and suppress records until each record shares its QI values with at least $k-1$ others. TRIM is close to full-domain k-anonymization with suppression [29] and top-down specialization [39], [40], with key differences: (1) it bounds the loss of the trained model rather than an information loss; (2) it removes records that k protects but the task does not need; and (3) it jointly optimizes attribute refinement and record retention rather than refining attributes over a fixed set of records: refining an attribute changes which records are worth keeping, and evicting records changes the privacy cost of a refinement.

(2) Data de-identification. Data de-identification and sanitization techniques [41]–[45] detect and remove personally identifiable information (PII), including direct identifiers and sensitive attributes. Industrial tools (*e.g.,* [46], [47]) can effectively reduce privacy exposure, but are largely task-agnostic. In contrast, TRIM explicitly balances privacy risk against model utility and retains only the data justified by the intended use.

(3) Synthetic data. Dataset distillation [48]–[50] and DP data synthesis [51], [52] replace the collected records with a small or private synthetic dataset. The residual risk of distilled data is often unclear, and neither bounds the utility loss on a given task or shows why each collected attribute and record is necessary, which a DPIA of training on the collected data asks. In contrast, TRIM retains real records at reduced granularity and logs why each retention decision was made.

(4) Feature selection. Feature selection [19], [53], [54] retains or discards features based primarily on predictive utility. As a result, it neither models privacy risk nor supports intermediate levels of disclosure between full retention and removal. In contrast, TRIM enables hierarchical feature generalization.

(5) Differential privacy (DP). DP [25], [55] can protect ML training against membership-inference attacks [56], *e.g.,* via DP-SGD [57] and its variants [58]–[60]. Complementary to data minimization, DP limits the influence of any individual record, and thus mitigates membership-inference risks outside TRIM's threat model. The data-minimization techniques can therefore serve as preprocessing steps for DP, whose guarantee then holds *w.r.t.* the minimized dataset, not the original $\mathcal{D}$.

III. Compliance-Aware ML Training

This section introduces the privacy and utility metrics that capture the objectives of minimization, and formalizes the training-data minimization problem for ML pipelines.

A. Privacy and Utility Foundations

Threat model. We consider an adversary $\mathcal{A}$ that tries to re-identify the record of a target individual in the released dataset [61]. Assume that $\mathcal{A}$ knows whether the target is in the original dataset, and knows some of the target's QI values from auxiliary sources. We assume one record per individual, and give $\mathcal{A}$ the same auxiliary information for every candidate dataset, so that candidates are compared on equal terms.

Formally, let S be a reduced candidate dataset obtained from $\mathcal{D}$ by removing records and generalizing QIs. Let $\mathcal{I}$ be the individuals in $\mathcal{D}$, and $\mathcal{I}(S) \subseteq \mathcal{I}$ be those whose records remain in S. For a target individual i, $\mathcal{A}$ knows whether $i \in \mathcal{I}$, as in the prosecutor model [62], and a set B_i of QI value combinations that contains i's true one, *e.g.,* "aged 70–79".

This adversary $\mathcal{A}$ follows EU anonymization guidance [63] in three respects. (1) *Attack goal*: finding the target's record is a singling-out and linkage attack, two of the three risks that the guidance assesses. (2) *Adversary knowledge*: the guidance analyzes an attack model that knows that a specific individual is in the data and knows some of their attribute values, as $\mathcal{A}$ does. (3) *Risk measure*: if k records share a target's generalized values, the target can still be linked to its record with probability $1/k$, which is exactly the risk we quantify.

The threat model covers the training dataset after release. A trusted controller minimizes the original $\mathcal{D}$ before releasing the result to downstream training pipelines, where a breach may expose it; $\mathcal{D}$ and the validation split stay with the controller. The trained model also cannot leak removed information [64].

Privacy risk. Under exact knowledge of $\mathcal{A}$, the set B_i holds only i's true combination. A record of S is compatible with i if its generalized QI values cover some combination in B_i and

equal those of i's record on each QI whose value $\mathcal{A}$ knows. These records form the *candidate set* of i, of size $k_i(S)$.

The *matching probability* $r_i(S)$ is the probability that the best guess of $\mathcal{A}$ for i's record is correct, *i.e.,* the Bayes vulnerability [65] of $\mathcal{A}$'s posterior belief about which record is i's. As is standard for class-size risk [62], we assume that $\mathcal{A}$ has no evidence, from non-QI values, record order, or how S was produced, that favors one candidate over another. Then $\mathcal{A}$ can only guess uniformly, so $r_i(S) = 1/k_i(S)$ if $i \in \mathcal{I}(S)$, and $r_i(S) = 0$ otherwise. We define the privacy risk of S as the natural log of its worst-case matching probability:

$$\mathsf{leak}(S) := \log \max_{i \in \mathcal{I}} r_i(S). \tag{1}$$

Intuitively, $\mathsf{leak}(S)$ is set by the most exposed individual, and a lower value means that even this individual is hard to match. In Table I, every patient is unique: $\mathsf{leak} = \log 1 = 0$. Coarsening to decades and families pairs up t_1, t_2 and t_3, t_5, but t_4 alone keeps leak at 0; without t_4, it is $-\log 2$.

Proposition 1: *If $\mathcal{A}$ knows exact QI values, then* $\mathsf{leak}(S) \leq -\log k$ *if and only if S is k-anonymous.* □

Proposition 1 holds because the candidate set of a retained target is the equivalence class of its record (full proof in [18]).

Rationale. Since the training dataset generally remains personal data, a DPIA asks how identifiable each data subject remains and whether a less identifying dataset would still serve the purpose [4]. Then leak answers both for every candidate S.

- *It is operational*: $e^{\mathsf{leak}(S)}$ is the maximum prosecutor risk, for which thresholds can be set to meet regulatory guidance, *e.g.,* 0.09 on publishing clinical data [66]. It bounds the matching probability of every considered adversary, since less QI knowledge only enlarges candidate sets and unknown membership only lowers the posterior (see [18]).
- *It is individual*: outliers (*e.g.,* t_4) cannot be averaged away.
- *It responds to both minimization axes*: generalization enlarges candidate sets, while removal zeroes the owner's risk but can shrink others' sets (dropping t_5 would expose t_3).
- *It is graded and auditable*: by Proposition 1, $\mathsf{leak}(S) = -\log k^*$, where k^* is the largest k for which S is k-anonymous, so every minimization operation receives an auditable marginal risk on a continuous scale, and the definition extends to partial knowledge.

Scope. Our threat model excludes three kinds of attacks. (1) Attacks that break the uniform-guess assumption, *e.g.,* by exploiting the minimality of generalization [67], [68], correlations learned from S [69], or the values of other records. Fixing what $\mathcal{A}$ knows is unavoidable: no measure computed from S bounds an adversary with arbitrary auxiliary information [70]. (2) Such attacks on the audit log of TRIM, which need access to it. (3) Attribute inference: inferring labels from QIs is the purpose of training, and any dataset that preserves its utility preserves the QI-label association [28].

Utility. We quantify *the utility of a candidate dataset S* with respect to a fixed model $\mathcal{M}$ using the negative Expected Generalization Error (EGE) of the model $\mathcal{M}_S$ trained on S:

$$U(\mathcal{M}_S) := -\mathbb{E}_{(x,y)\sim\mathcal{P}}\big[\ell(\mathcal{M}_S(x), y)\big], \tag{2}$$

where $\ell(\cdot,\cdot)$ is the loss function and $\mathcal{P}$ is the distribution over feature-label pairs. A higher utility U means that $\mathcal{M}_S$ better preserves task utility on unseen data in the validation split. For a fair comparison, all candidates share the same training protocol, random seeds, $\mathcal{P}$, and ℓ, and $\mathcal{M}_S$ includes the input transformation that maps test inputs to the granularity of S.

To quantify the utility impact of minimization, we compare $\mathcal{M}_S$ with the model $\mathcal{M}_\mathcal{D}$ trained on the original dataset:

$$\Delta U(S) := U(\mathcal{M}_\mathcal{D}) - U(\mathcal{M}_S). \tag{3}$$

We say that a minimized dataset S is ϵ *utility-preserving* when $\Delta U(S) \leq \epsilon$ for a user-specified tolerance ϵ.

B. Problem Formulation and Challenges

Problem statement. Given a relational dataset $\mathcal{D}$ with schema $\mathcal{R}$ and a target model $\mathcal{M}$, our goal is to derive a *minimized dataset* S^* by applying a sequence of minimization operations $o_1, \ldots, o_n$ to $\mathcal{D}$. An operation removes a record, or generalizes one QI by one level in all records; thus, every attribute keeps one encoding for training and test inputs, without requiring *e.g.,* local or multidimensional recoding [71], [72].

We state the Data Minimization Problem (DM) as follows:

- *Input*: Dataset $\mathcal{D}$, model $\mathcal{M}$, and utility tolerance $\epsilon > 0$.
- *Output*: A minimized dataset $S^* = o_n \circ \cdots \circ o_1(\mathcal{D})$.
- *Objective*: Minimize the residual privacy risk $\mathsf{leak}(S^*)$ while bounding the utility degradation $\Delta U(S^*)$:

$$\min_{S^* = o_n \circ \cdots \circ o_1(\mathcal{D})} \mathsf{leak}(S^*) \quad \text{s.t.} \quad \Delta U(S^*) \leq \epsilon.$$

Consider the decision version of DM: given $\mathcal{D}$, $\mathcal{M}$, and thresholds τ and ϵ, decide whether there exists a minimized dataset S^* such that $\mathsf{leak}(S^*) \leq \tau$ and $\Delta U(S^*) \leq \epsilon$. We assume that, for the model class and protocol considered, both $\mathsf{leak}(S^*)$ and $\Delta U(S^*)$ are computable in PTIME.

Theorem 2: *The decision version of* DM *is* NP*-complete.* □

Proof sketch: A candidate solution can be verified in PTIME by assumption; hence the problem is in NP. For the lower bound, we reduce from NP-complete Independent Set [73] with a learner that only counts records (see [18]). This extends the NP-hardness of optimal k-anonymization [74], [75]. ■

Challenges. Solving DM raises three challenges.

(1) Privacy-utility optimization. Privacy risk and utility are tightly coupled: removing or generalizing data may reduce disclosure risk, but can also lose information needed by the model. Moreover, their effects may be non-monotonic: an operation that appears beneficial in isolation may become redundant or harmful after other operations are applied. Hence, finding a Pareto-optimal minimized dataset requires searching a large, interdependent privacy-utility space.

(2) Efficient utility estimation. Retraining the target model for every candidate is expensive. Prior influence-based methods are cheaper, but their estimates diverge from the actual utility degradation as operations accumulate (see [18]).

(3) Auditability. Data minimization is not merely an optimization task: a DPIA reviewer needs each decision's privacy cost, utility contribution, and role in the final privacy-utility tradeoff.

IV. TRIM: Data Minimization Framework

Since DM is intractable, TRIM is a greedy heuristic that combines a retention view with a ratio-marginal rule.

Retention view. Existing minimization methods follow a *removal view* [3], [10], [11], [22]: starting from the original dataset $\mathcal{D}$, they iteratively remove records or generalize QIs, treating minimization as deletion. Instead, TRIM treats it as a retention problem by adopting a *retention view*. It begins with a dataset $\mathcal{D}_0$ of minimal data exposure, in which only a small label-covering seed of records is retained and all QIs are suppressed. It then progressively restores information whose utility contribution justifies its privacy cost. This extends the top-down specialization of [39], which operates on the QIs of each record independently, to (1) the choice of records, (2) a uniform treatment of all QIs, and (3) with an estimate of the trained model's utility gain in place of information gain.

The retention view also lets early iterations operate on small datasets, further reducing the cost of utility estimation. Our evaluation (see [18]) suggests a 14–557× speedup of retention-view search over a greedy removal-view search.

Ratio-marginal optimization. Under the retention view, an operation restores information, which typically recovers utility but also adds privacy risk. To satisfy the data-minimization principle, additional personal data should be retained only when justified by its contribution to the target task. TRIM thus prioritizes operations according to their *ratio-marginal score*, the utility gain per unit of privacy cost.

Let $\mathcal{V}_t$ denote the set of candidate operations applicable to the current dataset $\mathcal{D}_{t-1}$ in round t. For each candidate $o \in \mathcal{V}_t$, let $G(o)$ denote its marginal utility recovery and $\Delta\mathsf{leak}(o) := \mathsf{leak}(\mathcal{D}_{t-1} \oplus o) - \mathsf{leak}(\mathcal{D}_{t-1})$ denote its marginal change in privacy risk. Because leak is set by the most exposed individual, a candidate o may leave it unchanged or lower it. TRIM therefore charges each operation the *privacy cost* $c(o)$ and defines the *ratio-marginal score* of o as

$$\rho(o) := \frac{G(o)}{c(o)} = \frac{U(\mathcal{D}_{t-1} \oplus o) - U(\mathcal{D}_{t-1})}{\max\{\Delta\mathsf{leak}(o),\, \kappa\}}, \tag{4}$$

where $\mathcal{D}_{t-1} \oplus o$ denotes applying o to $\mathcal{D}_{t-1}$, and $\kappa > 0$ is a floor that ensures a positive denominator. Candidates that leave leak unchanged compete on their utility gain; with $\kappa = \log(1+x)$, an increase of at most a factor $1 + x$ in the worst-case matching probability counts as negligible. Such candidates add no risk for the most exposed individual, so only necessity, a positive gain, decides among them. TRIM then selects the operation with the largest ratio-marginal score:

$$o_t^* := \arg \max_{o \in \mathcal{V}_t:\, G(o) > 0} \rho(o), \tag{5}$$

and commits exactly this operation in round t.

Framework overview. Figure 1 illustrates the TRIM workflow. Starting from the suppressed dataset $\mathcal{D}_0$, TRIM works

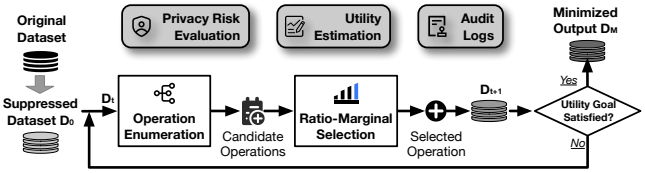


Figure 1. Overview of the TRIM workflow.

in rounds to produce a sequence of retained datasets $\mathcal{D}_0, \mathcal{D}_1, \ldots, \mathcal{D}_t$, where $\mathcal{D}_t$ denotes the output of round t.

Suppressed dataset. Following the retention view, TRIM first constructs a suppressed dataset $\mathcal{D}_0$: every QI is generalized to its coarsest level, and only a small label-covering seed is retained. The nonempty seed is the minimal initialization needed to train the current model and estimate candidate utility from the first round. On Table I, a seed $\{t_3, t_4\}$ (one per label) forms one class with both QIs suppressed, so $\mathsf{leak}(\mathcal{D}_0) = -\log 2$; Example 3 later evicts t_4.

After $\mathcal{D}_0$ is obtained, TRIM iterates the following steps.

(1) Operation enumeration. Given the current dataset $\mathcal{D}_{t-1}$, TRIM enumerates candidate retention operations from a bounded search space consisting of retention-class inclusions (horizontal) and attribute refinements (vertical). These define the candidate set used to construct $\mathcal{D}_t$ (Section V).

(2) Ratio-marginal evaluation. For each candidate o, TRIM calculates its privacy cost $c(o)$ directly. Computing $G(o)$ exactly would require retraining the target model for every candidate. To avoid this cost, TRIM employs a two-stage utility estimator: a lightweight ranking stage that prioritizes promising candidates, followed by a proxy-evaluation stage that retrains a lightweight proxy model to estimate the utility gain of the top-ranked operations (Section VI).

(3) Selection and application. TRIM next applies the ratio-marginal greedy rule to select the most promising operation o_t^* from the set $\mathcal{V}_t$, yielding the updated dataset $\mathcal{D}_t = \mathcal{D}_{t-1} \oplus o_t^*$. If no top-ranked candidate has a positive estimated gain, TRIM *widens* proxy evaluation to the top $\bar{b}$ per family; if none has one before the check passes, it returns $\mathcal{D}$ and logs the failure.

Final check. When the proxy model on $\mathcal{D}_t$ has $\Delta U \le \epsilon$, TRIM retrains the target model on the *release* of $\mathcal{D}_t$, adding each record outside $\mathcal{D}_t$ at the finest level of $\mathcal{D}_0, \ldots, \mathcal{D}_t$ where it does not raise leak, and checks $\Delta U \le \epsilon$ on the validation split before terminating. If the check fails, the search continues. Each round thus retrains the target model at most once.

Addressing DM challenges. TRIM directly addresses the three challenges identified in Section III.

(1) Privacy-utility optimization. TRIM optimizes DM via the ratio-marginal greedy rule, retaining information only when its utility gain justifies the induced privacy exposure. This reflects the DPIA principles of necessity and proportionality. TRIM terminates after polynomially many commits and returns a dataset that passes the final check (Section V).

(2) Efficient utility estimation. TRIM avoids exhaustive search and repeated retraining through structured candidate enumeration and an efficient two-stage utility estimator (Section VI).

(3) Auditability. TRIM logs each committed operation with its estimated utility gain, privacy cost, and the candidates it was compared with. The gain supports necessity, and the comparisons support proportionality. A reviewer can check the risk from the minimized dataset alone, but needs $\mathcal{D}$ to check the gains; the controller keeps $\mathcal{D}$ and the log under access control.

V. Retention-Based Search Space

This section develops the search space over which the retention view and ratio-marginal optimization operate.

Overview. At round t, TRIM enumerates a set $\mathcal{V}_t$ of candidate retention operations applicable to the current dataset $\mathcal{D}_{t-1}$. To support compliance-aware minimization, a key challenge is to make the search space both tractable and auditable.

(1) Tractability. Enumeration must avoid the exponential space of record subsets and arbitrary QI groupings. To address this challenge, TRIM exposes a bounded set $\mathcal{V}_t$ of structured local operations, each restoring an interpretable unit of information.

(2) Semantic coherence. Each operation must have a clear semantic meaning that can be justified in the audit log. Horizontal operations admit records that naturally belong together, while vertical ones follow predefined generalization hierarchies such as disease taxonomies, rather than ad hoc partitions.

Since suppression affects both record coverage and attribute granularity, TRIM supports two operation families: retention-class inclusion and attribute refinement. Next we introduce retention classes, both families, and the enumeration procedure.

Retention classes. TRIM groups records by their generalized QIs. For each record $r \in \mathcal{D}$, let $A_{t-1}(r)$ denote its *QI signature*, *i.e.,* the generalized QI values under the representation used by the current dataset $\mathcal{D}_{t-1}$. The *retention class* of r is defined as $C_{\mathsf{ret}}(r) = \{\, r' \in \mathcal{D} \mid A_{t-1}(r') = A_{t-1}(r) \,\}$, *i.e.,* the maximal set of records in $\mathcal{D}$ that share this signature. Hence, at round t, retention classes partition $\mathcal{D}$ into equivalence classes by the equality of current QI signatures.

Each class contains records indistinguishable under the current representation. It is analogous to a k-anonymity equivalence class [29], but evolves as the QI representation changes. Since class sizes affect re-identification risk (Proposition 1), retention classes form the basic units of both horizontal and vertical retention. The records of a class that are in $\mathcal{D}_{t-1}$ form a *retained class*; its size sets its members' matching risk.

Retention-class inclusion, the horizontal operation in TRIM. At a fixed QI granularity, the current dataset $\mathcal{D}_{t-1}$ may omit regions of the data distribution that are important for the target task; inclusion therefore restores such coverage.

TRIM enumerates one inclusion candidate for each retention class with not-yet-admitted records. Let k_{t-1} denote the size of the smallest retained class in $\mathcal{D}_{t-1}$. The candidate of a class admits all its not-yet-admitted records if there are at most k_{t-1} of them, and a random sample of k_{t-1} of them otherwise. Records of one class are indistinguishable under the current representation, so any sample is as private as another, and a random one does not bias the data within the class. A sample of k_{t-1} records never forms a class smaller than the smallest retained one, so it does not raise leak. Later inclusions can admit more of the class at no risk, so each admission is justified by its own gain, and any shift it induces lowers that gain, as the proxy is evaluated on the unshifted validation split.

Since retention classes are defined by exact equality of current QI signatures, one linear scan enumerates all candidates, one per signature, instead of the $2^{|\mathcal{D}|}$ subsets induced by arbitrary record-level admission. For a DPIA, the log names the class of every inclusion and the number of records it admits.

Attribute refinement, the vertical operation in TRIM. Each operation advances one QI in $\mathcal{D}_{t-1}$ to a finer level in a precomputed generalization hierarchy and restores a finer-grained representation. Refinement may increase re-identification risk by splitting existing classes without adding records. To address this problem, TRIM couples each refinement with retention-class swaps in a *refine-and-swap* operation.

Offline hierarchy construction. TRIM constructs a generalization hierarchy for each QI in the schema $\mathcal{R}$ of $\mathcal{D}$. Given an attribute's name, type, observed domain, and optional metadata (*e.g.,* categorical dictionaries), an LLM generates semantically meaningful generalization levels from the finest values to full suppression, denoted by '$*$' (see [18] for LLM usage details). Only attribute names and value domains, never records, reach the LLM, and the controller reviews each hierarchy. The levels can thus be organized as a generalization tree, with '$*$' as the root and the finest observed values as leaves.

This one-time preprocessing requires one LLM call per QI. The finite fan-out of the hierarchies bounds the candidate space. Unlike performance-driven hierarchies [10], TRIM uses only semantically meaningful levels, improving auditability.

Refine-and-swap. TRIM enumerates refinement targets directly from this generalization tree. In $\mathcal{D}_{t-1}$, each QI q is represented at some level λ of its hierarchy. If λ is not the finest, a *refinement target* $q : \lambda \mapsto \lambda'$ advances it to any finer level λ'.

A bare refinement can split a retention class into small fragments, raising leak. For example, refining `diagnosis` to the family level in Table I may isolate the only patient in their 80s with a hypertensive disease (t_4). TRIM thus uses *refine-and-swap*, an atomic operation that couples refinement with class-level repair. The algorithm in Figure 2 gives the details.

Algorithm. To construct a refine-and-swap candidate, it first advances the refinement target q from level λ to the finer level λ', fragmenting the affected retention classes (line 1). It then compares the smallest retained class C^- with C^+, the largest part of a class that is not yet admitted (line 2). While C^+ is larger, it evicts the smallest retained classes, at most $|C^+|$ records in total, and admits C^+ in their place (line 4). Each exchange admits at least as many records as it evicts and keeps or enlarges the smallest retained class of the refined dataset (lines 5–7). The resulting composite operation (line 8) is then estimated and ranked like any other candidate.

The swap repairs a refinement rather than selecting records: it uses class sizes alone, evicting the most exposed individuals first, and its exchanges enter the candidate's estimated gain,

Input: Retained dataset $\mathcal{D}_{t-1}$, original $\mathcal{D}$, a refinement target $q : \lambda \mapsto \lambda'$.
Output: A refine-and-swap o for $\mathcal{V}_t^{\text{attr}}$.

1 $\mathcal{D}' \leftarrow \mathcal{D}_{t-1}$ with QI q advanced from λ to λ';
2 $C^- \leftarrow$ a smallest retained class of $\mathcal{D}'$;
3 $C^+ \leftarrow$ the largest not-yet-admitted class in $\mathcal{D} \setminus \mathcal{D}'$ at the new level;
4 **while** $|C^+| > |C^-|$ **do**
5 $\quad X \leftarrow$ the smallest retained classes of $\mathcal{D}'$, taken in increasing size, with total size at most $|C^+|$;
6 $\quad \mathcal{D}' \leftarrow (\mathcal{D}' \setminus X) \cup C^+$; /* exchange */
7 $\quad$ update C^- and C^+ as in line 2;
8 **return** $o \leftarrow (\mathcal{D}_{t-1} \mapsto \mathcal{D}')$; /* the composite candidate */

Figure 2. TRIM: refine-and-swap at round t.

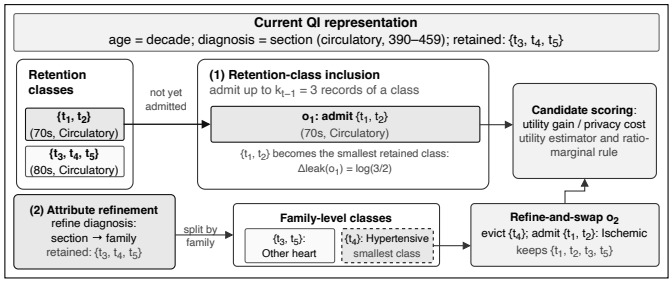


Figure 3. Enumeration example.

so they are committed only if the candidate wins the ratio-marginal rule. The seed of $\mathcal{D}_0$ is a retained class like any other, so the swap can evict seed records. Early in the search, when few records are retained, the swap can replace every retained class by a single group; the refined QI is then constant among the retained records, so the candidate's estimated gain is low and the selection rule passes it over. Early rounds thus proceed mainly by inclusion, and refinements become effective as admission grows. Once every record is retained, no part is left to admit, and the candidate is the bare refinement, which the ratio rule charges its full increase in leak.

Refine-and-evict. A refinement that isolates a few records can lower k sharply, as the swap admits only parts larger than the smallest retained class. TRIM thus also offers a *refine-and-evict* variant of each refinement, which further evicts the retained classes smaller than $\lceil k_{t-1}/4 \rceil$, so k falls by at most $4\times$. The release keeps evicted records at a coarser, earlier level where possible, and the proxy evaluates the variant with them. TRIM prefers candidates within $4\times$ if any has $\bar{G}(o) > 0$.

Example 3: Figure 3 illustrates both operations on `Diabetes` of Example 1. Suppose `age` is retained at the decade level and `diagnosis` at the section level (codes 390–459 become *Circulatory*), so the retained records t_3, t_4, t_5 form the class (80s, Circulatory) and $\mathsf{leak}(\mathcal{D}_{t-1}) = -\log 3$.

(1) Retention-class inclusion. The unretained class $\{t_1, t_2\}$ (70s, Circulatory) yields candidate o_1, which admits both records since $2 \leq k_{t-1} = 3$; as the new smallest retained class, it gives $\Delta\mathsf{leak}(o_1) = \log(3/2)$.

(2) Attribute refinement. Refining `diagnosis` to the family level splits the retained class into $\{t_3, t_5\}$ (80s, Other heart diseases) and the singleton $\{t_4\}$ (80s, Hypertensive diseases). The swap evicts $C^- = \{t_4\}$, admits $C^+ = \{t_1, t_2\}$ (70s, Ischemic heart diseases), and stops since the largest unadmitted part, $\{t_4\}$, is smaller than every retained class. The resulting candidate o_2 retains t_1, t_2, t_3, t_5 with $\Delta\mathsf{leak}(o_2) = \log(3/2)$. Without the swap, the refinement would isolate t_4 and cost $\log 3$, so the ratio rule would demand $2.7\times$ the gain of o_2.

For a small κ, TRIM breaks the tie by estimated utility gain. If it commits o_2, every retained record shares its age decade and diagnosis family with another: t_4 is retained while its class has three records and evicted once refinement isolates it, which neither vertical nor horizontal minimization alone achieves. The commit's log entry records the refinement, the eviction of t_4, the admission of t_1, t_2, and the gains and costs of o_2 and o_1, as the DPIA of Example 2 asks. □

Properties. The audit log relies on three properties of the entire search. Let H be the sum of the heights of the QI hierarchies and $N = |\mathcal{D}|$. An estimate $\bar{G}(o)$ of the true utility recovery $G(o)$ is ξ-*accurate* if $|\bar{G}(o) - G(o)| \leq \xi$.

Proposition 3: TRIM *satisfies the following: (a)* Termination*: it commits at most* $H + (H+1)N$ *operations, and each round retrains the proxy at most* $4b+1$ *times (*$4b+4\bar{b}+1$ *if widened) and the target once; (b)* Feasibility*: its output passes the final check, or it returns* $\mathcal{D}$*; and (c)* Per-decision proportionality*: in a round whose estimates are* ξ*-accurate, the committed operation* o^* *satisfies* $\rho(o^*) \geq \rho(o) - \xi/c(o^*) - \xi/c(o)$ *for every candidate* o *within* $4\times$ *evaluated in that round.* □

We defer a full proof of this proposition to [18]. Properties (a) and (b) hold for any estimate. Two candidates charged κ add no worst-case risk, and for them (c) reads $G(o^*) \geq G(o) - 2\xi$. Since TRIM reuses the validation split across rounds, (c) documents the selection rule, and feasibility rests on (b).

VI. Efficient Utility Estimation

The ratio-marginal rule needs the utility recovery $G(o)$ of each candidate $o \in \mathcal{V}_t$. Computing $G(o)$ exactly would require retraining the target model $\mathcal{M}$ once per candidate, which is prohibitively expensive. In light of this, this section develops a two-stage utility estimator that mitigates this cost.

Overview. The greedy rule does not need exact utility values. It needs *estimates* that (a) score retention-class inclusions and attribute refinements under one measure, and (b) are cheap enough to screen candidates yet accurate for the few finalists.

Existing estimators satisfy neither. Influence-based methods [76]–[78] and approximate unlearning [79] approximate the effect of retraining through parameter updates, Hessian inverses, or training checkpoints. These are costly to repeat for every candidate, and their estimates drift as operations accumulate. Per-record valuation such as Data Shapley [80] values records rather than operations that rewrite existing records, and misses interactions among jointly admitted records.

TRIM meets both requirements in two stages:

- *Ranking*: Learning Gradient Alignment (LGA) scores every candidate from training gradients alone for both operation families, and TRIM keeps the top-b of each family; and
- *Proxy evaluation*: for each candidate, TRIM retrains a lightweight *proxy model* in place of $\mathcal{M}$ to estimate $G(o)$.

(1) Ranking with LGA. LGA scores a candidate by how much it changes the training gradient in the direction that lowers the validation loss. It uses a lightweight *ranking model*, *e.g.,* logistic regression, whose parameters θ are fitted on $\mathcal{D}_{t-1}$

and then fixed. For a candidate o, with $\mathcal{D}_{t-1} \oplus o$ the provisional dataset obtained by applying o, we define

$$\mathsf{LGA}_\theta(\mathcal{D}_{t-1}, o) = \Big\langle \nabla_\theta L_{\mathsf{v}}(\theta),\ \nabla_\theta L_{\mathsf{train}}(\theta, \mathcal{D}_{t-1} \oplus o) - \nabla_\theta L_{\mathsf{train}}(\theta, \mathcal{D}_{t-1}) \Big\rangle,$$

where $L_{\mathsf{train}}(\theta, \cdot)$ and $L_{\mathsf{v}}(\theta)$ are the mean ranking model losses over the training and the validation split, respectively. That is, LGA aligns the validation-loss gradient with the marginal training-loss gradient induced by o, isolating the learning signal that o adds. For o_1 of Example 3, the gap is proportional to the loss gradient of t_1, t_2: LGA is high if fitting them fixes the current model's errors on validation patients in their 70s.

Since θ minimizes the training loss on $\mathcal{D}_{t-1}$, retraining on $\mathcal{D}_{t-1} \oplus o$ lowers the validation loss to first order, by the inner product above with the gradient gap multiplied by H^{-1}, the inverse Hessian of that loss [76]. LGA drops H^{-1}, which avoids inverting it and preserves the order of candidates outside a margin set by H's condition number of H (see [18]). Thus a large positive score indicates large $G(o)$.

LGA applies the one-step validation-gradient gain of GLISTER [81] to the gradient *gap* induced by an operation, whereas GRAD-MATCH [82] and gradient matching [48], [49] match the gradients of a subset or of synthetic data. LGA needs only gradients at the current parameters, which TRIM refits every round, and it serves only to rank candidates for the proxy.

Top-b selection. TRIM ranks inclusions, refinements and their variants separately by $\mathsf{LGA}_\theta(\mathcal{D}_{t-1}, o)/c(o)$, mirroring the ratio-marginal score of Equation (4), and keeps the top-b of each, plus the variants of the top-b refinements, which LGA can underrate. Both families thus reach proxy evaluation, with at most $4b$ candidates per round. Per unit cost, this order agrees with the retrained one outside the same margin divided by $c(o)$ (see [18]). Our experiments show that LGA orders inclusions more accurately than prior gradient-based methods.

(2) Proxy evaluation. A *proxy* $\bar{\mathcal{M}}$ is a model much cheaper to retrain than $\mathcal{M}$, *e.g.,* a narrower network of $\mathcal{M}$'s family or logistic regression, as in selection via proxy [83]. For each retained candidate o, TRIM retrains $\bar{\mathcal{M}}$ on $\mathcal{D}_{t-1}$ and on $\mathcal{D}_{t-1} \oplus o$, and estimates $G(o)$ by the utility recovery of $\bar{\mathcal{M}}$:

$$\bar{G}(o) = \bar{U}(\mathcal{D}_{t-1} \oplus o) - \bar{U}(\mathcal{D}_{t-1})$$

on the validation split. Since $\bar{G}(o)$ is a difference of two retrainings of $\bar{\mathcal{M}}$, any gap between $\bar{\mathcal{M}}$ and $\mathcal{M}$ that o leaves unchanged cancels. The remaining error comes from validation noise and from how differently the two models respond to the records that o changes; for uniformly stable models [84], it is $O(|o|/n)$, which can be large for a refine-and-swap (see [18]).

TRIM uses $\bar{G}(o)$ only to select operations; feasibility rests on the final check, which retrains $\mathcal{M}$. The proxy thus affects how much risk TRIM spends, not whether its output meets the tolerance, and yields one minimized dataset per task; a proxy from the target's family lowers the error when the target relies on interactions that a linear proxy cannot represent.

Greedy selection. The two stages produce a proxy estimate $\bar{G}(o)$ for each retained candidate. TRIM applies the ratio-marginal rule to these estimates: among the retained candidates with $\bar{G}(o) > 0$ (within $4\times$ if any), it commits the one that maximizes $\bar{G}(o)/c(o)$, an estimate of $\rho(o)$ in Equation (4).

Complexity. Ranking computes only gradients at the current parameters, and proxy evaluation retrains $\bar{\mathcal{M}}$ for at most $4b \ll |\mathcal{V}_t|$ candidates per round ($4\bar{b}$ if widened). Privacy costs and swaps need only class sizes. The target model is retrained only in the final check, at most once per round (Section IV).

Table III. Summary of used datasets.

Dataset	#Rows	#Features	#QIs	Type	Prediction Task
Income	183,941	10	10	real	Personal income (>$50k)
Diabetes	71,518	47	18	real	Hospital readmission
PubCov	152,676	19	19	real	Public health coverage
BM	45,211	14	7	real	Term-deposit subscription
BNG	1,000,000	20	20	synthetic	Credit risk (good or bad)

VII. Experimental Study

Using real-world and synthetic datasets, we experimentally evaluate TRIM for its privacy-utility tradeoff, scalability, utility estimator, key design choices, and audit record supports.

Experimental settings. We start with the settings.

Datasets. We evaluate TRIM on five datasets summarized in Table III. Income and PubCov are U.S. Census-based prediction tasks from [85]. Diabetes [17] contains hospitals records for readmission prediction. Following [17], we kept the first encounter of each patient (71,518 of 101,766 encounters), so that each record belongs to a distinct individual. BM [86] captures Portuguese bank telemarketing campaigns. BNG (credit-g) [87] is a Bayesian-network-generated German credit-risk benchmark. The full list of QIs in each dataset is given in [18].

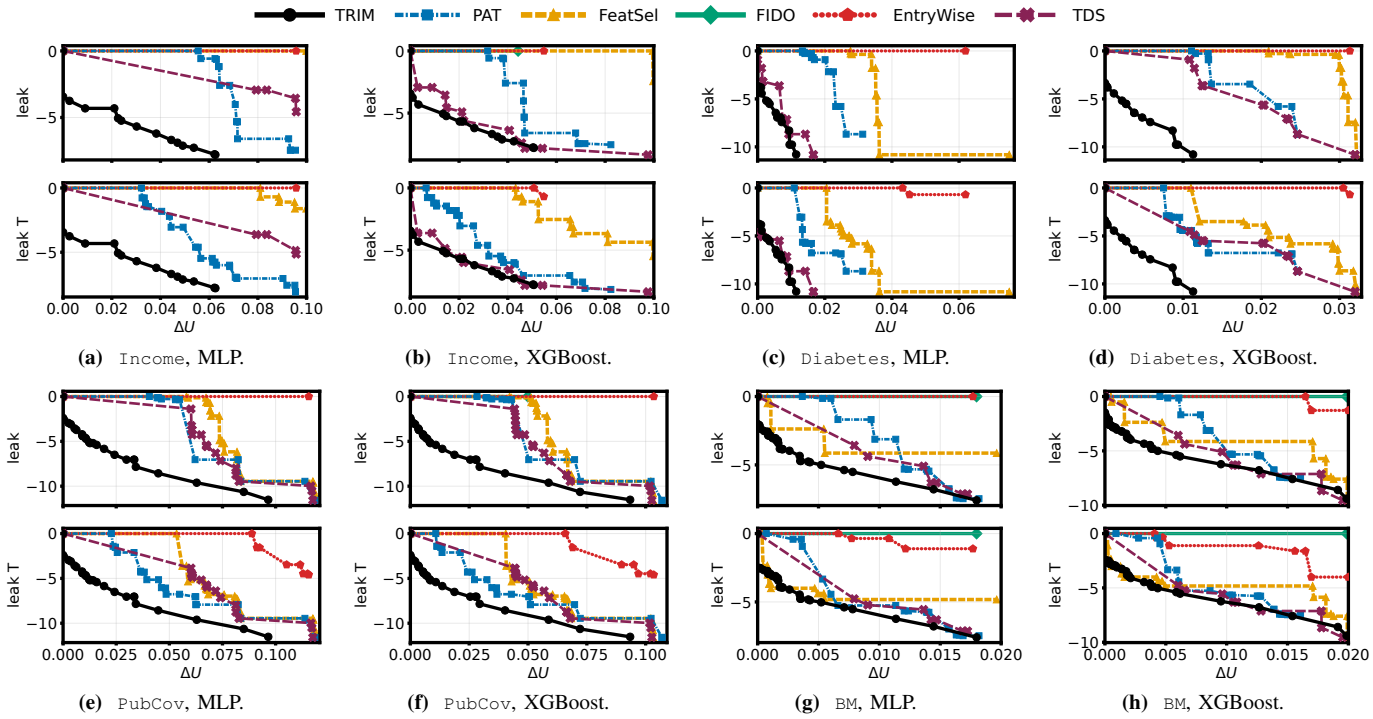
For each dataset, we creat stratified 70/15/15 training, validation and test splits, and evaluate all methods on the same splits. The splits are disjoint at the level of individuals.

Downstream models. We use two target models $\mathcal{M}$: an MLP with two 32-unit hidden layers, and XGBoost [88] with 100 trees of depth 3. A model trained on the output of TRIM receives test records generalized to the levels of that output.

Baselines. We compare TRIM with four representative removal-based data-minimization baselines and one k-anonymization baseline: (1) *Horizontal-only minimization*: FIDO [11], which suppresses samples under a validation-loss bound. (2) *Vertical-only minimization*: FeatSel [22], which searches all QI subsets (on PubCov, Diabetes, and BNG, those dropping or keeping at most 3, 2, and 2 QIs), and PAT [10], which generalizes QIs using a learned tree minimizer. (3) *Hybrid minimization*: EntryWise [3], which suppresses random cells. (4) *k-anonymization*: KAnon [29] via TDS [40], refining QIs top-down with $\geq k$ records per class.

We run the public implementation of each baseline; PAT and KAnon learn their own generalization, and the others generalize no values. The data-minimization baselines do not target leak, so we report it for their outputs as an audit of the risk they leave. Since KAnon fixes the risk instead of the utility, we compare it with TRIM at the same leak.

We implemented and tested TRIM in ∼12k LOC of Python, including several variants for ablation study in Exp-6.



(a) `Income`, MLP. **(b)** `Income`, XGBoost. **(c)** `Diabetes`, MLP. **(d)** `Diabetes`, XGBoost.

(e) `PubCov`, MLP. **(f)** `PubCov`, XGBoost. **(g)** `BM`, MLP. **(h)** `BM`, XGBoost.

Figure 4. Tradeoffs between privacy risk and utility degradation.

Metrics. For each output dataset $\mathcal{D}_M$, we report privacy risk $\mathsf{leak}(\mathcal{D}_M)$, tail risk $\mathsf{leak}_T(\mathcal{D}_M) = \mathrm{P99}_{i\in\mathcal{I}}\big[\log r_i(\mathcal{D}_M)\big]$, and utility degradation $\Delta U(\mathcal{D}_M)$. Here leak_T is the 99th percentile of individual risk, complementing the worst-case risk leak. The loss ℓ is the log loss. TRIM's estimates and final check use the validation split; ΔU is the increase in test log loss over each method's best output, the full split included for baselines.

Configurations. By default, we set the cost floor to $\kappa = \log 1.05$ and use XGBoost as the downstream model. The initial $\mathcal{D}_0$ is a fixed-seed 1.2% sample of the training rows with all QIs generalized to their coarsest levels. Both stages of the utility estimator use logistic regression, independent of the downstream model. The proxy stage evaluates the top $b = 7$ LGA-ranked candidates per family ($\bar{b} = 140$ if widened).

All experiments ran on NVIDIA RTX 3090 servers with two Xeon Gold 5320 CPUs, or on A100 and L4 GPUs, using Python 3.10–3.11 and PyTorch 2.9. All methods use identical environments, data splits, and random seeds within each run. We repeated each experiment five times and report the mean. Representative results are shown; the rest are consistent.

Experimental results. We report six experiments on TRIM's privacy-utility tradeoff (Exp-1), empirical risk mitigation (Exp-2), scalability (Exp-3), audit logs (Exp-4), hyperparameter sensitivity (Exp-5), and ablation study (Exp-6).

Exp-1: Privacy vs. utility. Figure 4 reports the privacy-utility tradeoff of each method on the real datasets. For TRIM, each point is a dataset that the search reaches, with its leak and its test ΔU; the target model is retrained on each such dataset for evaluation only. Each baseline sweeps its own parameter: the loss bound of FIDO, the QI subsets of FeatSel, the parameter α of PAT, and the kept-cell fraction and bias of EntryWise. We compute leak of every output by its equivalence classes: a removed record has $r_i = 0$, and a suppressed cell matches only $*$. To compare methods at the same leak, we take each baseline at its own operating points, and TRIM at its smallest ΔU among the datasets with no larger leak.

Privacy-utility frontier. At the same utility degradation, TRIM attains a worst-case risk leak lower than or comparable to that of every data-minimization baseline in all settings.

(1) Under stringent utility tolerances (*e.g.,* $\Delta U \le 0.02$), TRIM already reduces worst-case leakage, while PAT, the strongest of these baselines on most datasets, often leaves it unreduced (*i.e.,* $\mathsf{leak} = 0$). At the same leak level, PAT incurs 1.09–$7.80\times$ and 1.10–$14.0\times$ as much utility degradation as TRIM for MLP and XGBoost, respectively, excluding the points where TRIM reduces leak at no cost ($\Delta U < 0.001$) and the coarsest ones, where both generalize most QIs to the coarsest levels.

(2) On `BM` (Figures 4g–4h), FeatSel and PAT have Pareto frontiers close to TRIM. Most predictive variables are non-QIs (*e.g.,* contact channel and campaign history), while useful QIs are largely binary (*e.g.,* housing/personal loans). So vertical baselines retain most utility with coarse QI exposure, leaving limited room for TRIM's horizontal suppression.

(3) FIDO and EntryWise cannot protect high-risk records, since they optimize indirect targets (*e.g.,* the number of retained records/attributes) rather than the worst-case risk; as they keep exact QI values, their leak stays at or near 0.

(4) As shown in Figures 4a–4h, TRIM achieves similar privacy-utility frontiers on MLP and XGBoost: leak varies by at most 20.4% across the two downstream models. Both estimator stages use logistic regression, so the two targets are evaluated on one sequence of datasets; the similar frontiers show that this sequence serves both targets.

Tail privacy. Here leak_T follows the same trend as the leak-utility frontier. The gap between TRIM and PAT/FeatSel is

Table IV. Utility degradation with DPSGD training (Income, MLP). Entries for TRIM and KAnon are increases over the original $\mathcal{D}$ in the same column.

DPSGD $e^{\varepsilon_{\mathrm{DP}}}$	**No Noise**	5	2	1.5
Original Dataset	0	0.043	0.041	0.043
TRIM (leak $= -5.50$)	$+0.082$	$+0.043$	$+0.048$	$+0.050$
KAnon ($k = 235$, Mondrian [71])	$+0.094$	$+0.066$	$+0.085$	$+0.102$
KAnon ($k = 235$, TDS, $\leq 5\%$)	$+0.098$	$+0.075$	$+0.084$	$+0.092$

smaller, because leak_T discounts the most exposed 1% of records, where vertical-only generalization is weakest. Even under this less sensitive metric, TRIM achieves lower tail risk at small to moderate tolerances with XGBoost, while FeatSel has lower tail risk at the smallest tolerances on BM.

Flexibility. TRIM traces a smooth privacy-utility curve across the achieved utility degradation (Figures 4a–4h), letting an operator choose a DPIA-approved operating point anywhere along this range, flexibly adapting to different levels of regulatory requirements. FeatSel offers a few discrete operating points, one per selected QI subset, and PAT's parameter α specifies neither a privacy nor a utility budget.

Interplay with DP. We evaluate TRIM as a preprocessing step for DPSGD [57], which limits membership-inference by noisy training. This test focuses on utility. Table IV compares TRIM with training on the original data and with KAnon [29], implemented with either Mondrian [71] or top-down specialization (TDS) [39] with suppression. We configure KAnon and TRIM to the same worst-case privacy risk on all 184K records of Income. All models use SGD for 20 epochs with Poisson sampling, $\varepsilon_{\mathrm{DP}} = \log x$ for $x \in \{5, 2, 1.5\}$, and $\delta = 10^{-5}$. Because TRIM's selection is data-dependent, the DP guarantee applies to the minimized dataset rather than the original $\mathcal{D}$.

Across the settings, TRIM incurs 0.012–0.042 less utility degradation than both variants of KAnon. When DP is enabled, TRIM adds only 0.043–0.050 of utility degradation over the corresponding model trained on the original $\mathcal{D}$, compared with 0.066–0.102 for KAnon. Moreover, as $e^{\varepsilon_{\mathrm{DP}}}$ decreases from 5 to 1.5, TRIM's advantage over KAnon widens to 0.042, showing that data minimization complements DP training.

Exp-2: Empirical risk mitigation. We run empirical privacy attacks on all 184K records of Income at $\epsilon = 0.05$, with a 6% seed and an MLP as target and proxy.

QI reconstruction and inference. Figure 5a compares attack errors across methods. An adversary who knows the original values of a fifth of the records trains a model that recovers the original QI values of a minimized record (QI reconstruction), or that predicts race, a special category under the GDPR, from the minimized values of eight other QIs (QI inference). Both attacks learn only from released records; we report the fraction of wrong predictions on held-out test records generalized as in each output, averaged over the QIs for reconstruction, capped at the error of guessing the mode (the mean for numerical QIs). They measure how well minimized data hide exact QI values, rather than re-identification. TRIM has the highest errors under both attacks: its reconstruction error is $\geq 34.7\%$ above that of every baseline, and its inference error (0.36) is $\geq 40.0\%$ above. For matching, an adversary within our threat model of Section III-A succeeds with probability $1/k_i$, which leak reports exactly; attacks outside it are beyond the scope.

Operational relevance of leak. We assess whether leak tracks the success of realizable attacks. We conduct both attacks against TRIM outputs spanning a range of leak values. Figure 5b reports the advantage Δ of each attack, *i.e.,* the guess error minus the attack error. As leak decreases from -0.7 to -9.3, Δ shrinks from 0.23 to 0.02 for reconstruction and from 0.06 to 0.02 for inference over all QIs. Thus, on TRIM's outputs, leak tracks the success of both.

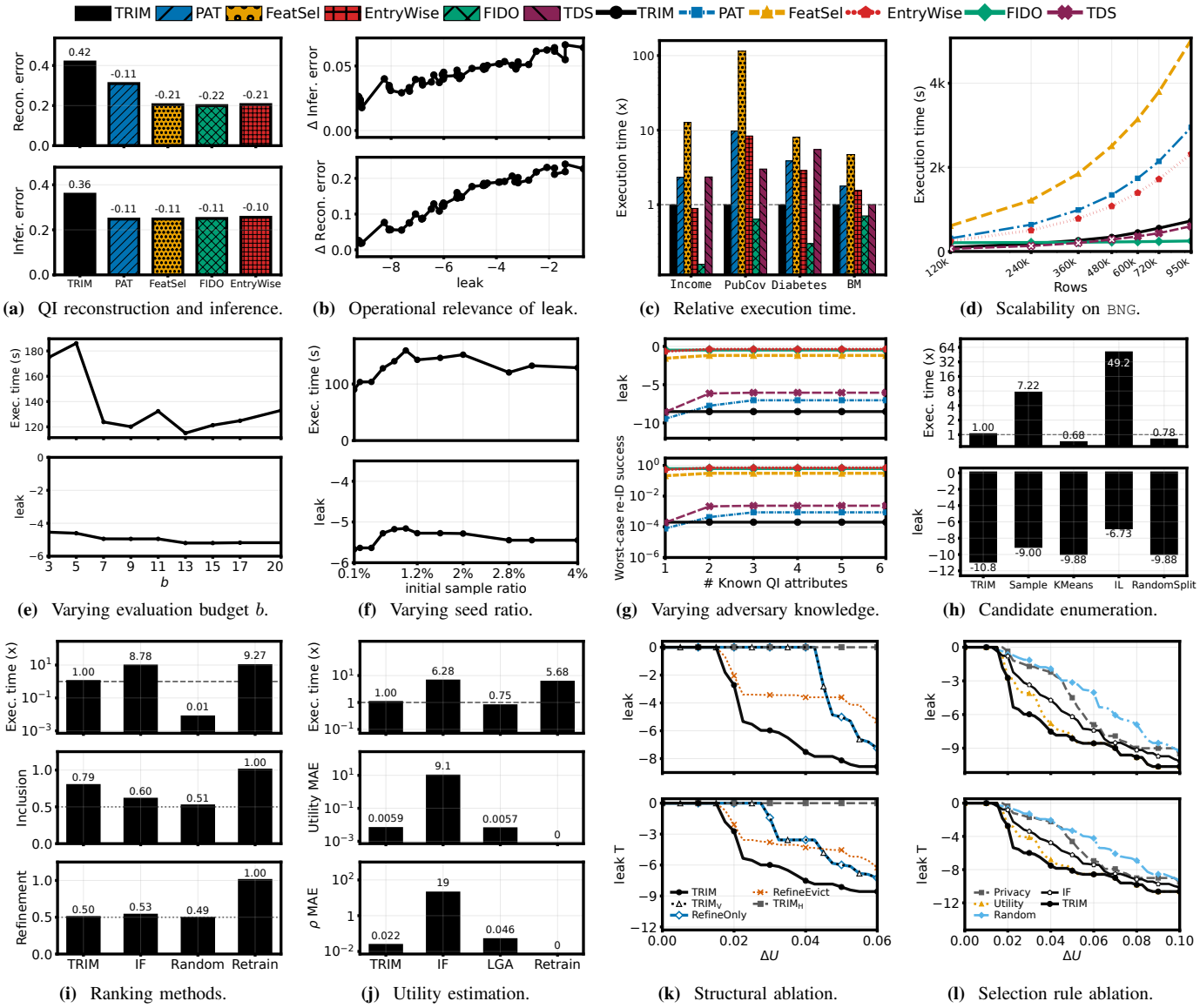
Exp-3: Efficiency. We compare the execution times.

Execution time. Figure 5c reports the execution time on the real datasets. For TRIM, we measure the time to first satisfy $\epsilon = 0.09$ on Income, 0.1 on PubCov, 0.005 on Diabetes, and 0.01 on BM, with the MLP target, all records, and a 1.2% seed. TRIM is 1.78–$9.77\times$ faster than PAT on all four datasets. FeatSel retrains the model per candidate subset, so it takes 4.72–$115\times$ as long as TRIM. EntryWise fits the model once per random mask and takes 0.893–$8.35\times$ as long. KAnon fits the model once per k and takes 1.00–$5.49\times$ as long, but at the same ϵ on Income, its leak is at least -3.14, versus at most -3.58 for TRIM. FIDO takes 0.159–$0.708\times$ as long, but incurs high worst-case risk.

Scalability. To isolate data-size effects, we use synthetic BNG instances with fixed distribution and varying scale factors. In Figure 5d, TRIM is 2.59–$7.33\times$ faster than EntryWise, PAT, and FeatSel, and its runtime grows by $7.6\times$ from 120K to 950K rows ($7.9\times$). This is because each round enumerates one inclusion per retention class with not-yet-admitted records, not arbitrary record subsets, and many rows share the same QI signature. KAnon takes 0.65–$0.83\times$ as long as TRIM, but misses $\epsilon = 0.05$ on 8 of 35 runs (hollow markers), and its leak is 1.35–3.64 higher than that of TRIM.

```
[run] data=Income purpose=income>50K target=XGBoost
     epsilon=5.0e-2 b=7
[log1] action=COMMIT op=refine-and-swap target=AGEP
      step=ANY->lifecycle-band
      evict={seed} admit={RC-ABB}
      gain_est=1.3e-2 leakage_delta=-3.5e+0 cost=kappa
      rho=1.3e-2/kappa reason=max-rho
[log2] action=COMMIT op=retention-class-inclusion
     target=RC-4A9 lga_rank=2
      scope={same-QI-signature; all-class-members}
      gain_est=2.9e-3 leakage_delta=+1.3e-1 cost=1.3e-1
      rho=2.2e-2 reason=max-rho
      alt={op=refine-and-swap; target=MAR; lga_rank=5;
        gain_est=1.6e-2; leakage_delta=+8.0e-1; rho=2.0e-2}
      alt={op=class-inclusion; target=RC-F5A; lga_rank=1;
        gain_est=-3.2e-3; leakage_delta=+2.7e-2; rho=-6.5e-2}
[stop] state=D[T] check=target-on-validation
     DeltaU(D[T])<=epsilon result=PASS
```

Exp-4: Auditability. We evaluate auditability on Income with $\epsilon = 0.05$. TRIM logs the declared task, risk and utility policies, estimator configuration, dataset states, and each committed operation's scope, estimated utility gain, exact leakage change, privacy cost, and the candidates it was compared with. The excerpt above shows two commits: one that lowers leak and is charged the floor κ, and one charged its leakage increase.



(a) QI reconstruction and inference. **(b)** Operational relevance of leak. **(c)** Relative execution time. **(d)** Scalability on BNG.

(e) Varying evaluation budget b. **(f)** Varying seed ratio. **(g)** Varying adversary knowledge. **(h)** Candidate enumeration.

(i) Ranking methods. **(j)** Utility estimation. **(k)** Structural ablation. **(l)** Selection rule ablation.

Figure 5. Efficiency, effectiveness, hyperparameter sensitivity, and privacy diagnostics of TRIM.

The header fixes the run's purpose, utility tolerance, and risk policy. In `log1`, a refine-and-swap refines `AGEP` to lifecycle bands, evicts the seed, and admits `RC-ABB`; it lowers leak by 3.53, so it is charged κ, and its utility gain decides its score. In log2, `RC-4A9` ranks second among inclusions by LGA but is selected because its $\rho = 0.0216$ exceeds the scores of the other evaluated candidates. Its positive utility estimate supports *necessity*, and its maximum ρ supports *proportionality*. The release publishes 128,747 training records, 49.8% at earlier, coarser levels. Taken together, these support a DPIA review.

Exp-5: Sensitivity to hyperparameters. Beyond ϵ, we vary TRIM's hyperparameters around their defaults with the MLP target on all 184K records of `Income`, except b and adversary knowledge (on `PubCov`; XGBoost target for b).

Varying evaluation budget b. Figure 5e varies b from 3 to 20 at $\epsilon = 0.02$ with a 3% seed; TRIM passes its final check on all five seeds for every b. For $b \leq 5$, it reaches leak $= -4.54$ to -4.61 in 175–186 s. From $b = 7$, it reaches leak $= -4.95$ to -5.20 in 115–133 s, as the selector can choose from more evaluated candidates. We thus use $b = 7$ by default, the smallest b reaching this level.

Varying cost floor κ. We vary $\kappa = \log(1 + x)$ for $x \in \{0.01, 0.05, 0.1, 0.25, 0.5\}$ at $\epsilon = 0.09$. Across these, leak ranges only from -5.28 to -4.30. Thus, TRIM is insensitive to κ.

Varying seed ratio. Figure 5f varies the seed $\mathcal{D}_0$ from 0.1% to 4% of the training rows at $\epsilon = 0.09$; each point is the first dataset of the search that satisfies ϵ. TRIM is insensitive to the seed ratio, since its first round evicts the whole seed in all 60 runs. Across ratios, leak ranges only from -5.67 to -5.16, and the time from 90 to 159 s. The other datasets behave alike (see [18]). We thus keep the 1.2% default.

Varying adversary knowledge. We vary the number of known QIs from 1 to 6 on `PubCov`, over all such combinations, and report leak and the worst-case re-identification success rate at $\epsilon = 0.07$ with a 6% seed and of each baseline at its most private output within ϵ (Figure 5g). TRIM's leak stays at -8.52, the lowest from two QIs on; PAT's and KAnon's, lower with one, rise to -7.04 and -6.05, and FeatSel, EntryWise,

and FIDO leave (nearly) unique records.

Exp-6: Ablation studies. We evaluate TRIM's key design decisions on `PubCov`, except in Figures 5i–5j, which use `Income` with an MLP as the target model.

Quality of retention classes. Figure 5h compares retention classes with four horizontal admission units: Sample admits individual records, KMeans forms clusters, IL groups records by information loss, and RandomSplit splits groups randomly. Retention classes get the lowest residual risk (leak $= -10.83$) and are 7.22–$49.2\times$ faster than Sample and IL. KMeans and RandomSplit take only $0.68\times$ and $0.78\times$ as long, respectively, while both stop at leak $= -9.88$; Sample stops at -9.00, and IL at -6.73. Retention classes reduce residual risk by 0.95–4.10 over the alternatives.

LGA-based ranking. Figure 5i compares LGA with random ranking (Random) and influence-based scoring (IF) by their pairwise accuracy against retraining (Retrain), *i.e.,* the fraction of candidate pairs within a round and operation family that a method orders as Retrain does (0.5 is chance), averaged over the rounds with at least three candidates. Retrain refits the proxy $\bar{\mathcal{M}}$ for inclusions (eight rounds of one run) and the target $\mathcal{M}$ for refinements (five runs). For inclusions, LGA reaches 0.79, above IF (0.60) and Random (0.51); for refinements, all three methods are near chance (0.49–0.53). IF and retraining $\mathcal{M}$ take $8.78\times$ and $9.27\times$ as long as LGA, respectively.

End-to-end estimation fidelity. Figure 5j tests marginal-utility estimates for TRIM's top-b horizontal and vertical candidates per round, comparing its two-stage estimator with influence functions (IF) [89] and LGA against full retraining (Retrain) as ground truth; LGA is scaled by a factor fitted on three other seeds. The two-stage estimator has the lowest error in the ratio-marginal score $\rho(o)$, which TRIM uses to select operations, and runs $5.68\times$ faster than Retrain, which retrains the target model $\mathcal{M}$. LGA is $1.33\times$ faster than the two-stage estimator and has a 3.3% lower utility error, but its error in $\rho(o)$ is $2.09\times$ higher. IF is $6.28\times$ slower than the two-stage estimator, and its errors are 851–$1{,}542\times$ higher. For the selected operation, the two-stage estimator reduces the error in $\rho(o)$ by 23.8% and over 99.9% relative to LGA and IF, respectively. The two-stage estimator thus balances accuracy and efficiency.

Structural ablations. Figure 5k compares TRIM with two single-axis variants on `PubCov`, from five runs with a 0.25% seed: TRIM_V retains all records and only refines attributes, and TRIM_H keeps exact QI values and only admits records, grouped by k-means clustering. They are the single-axis extremes: TRIM_V keeps every record, even isolated ones, and TRIM_H releases exact QI values, so its leak is 0 by construction. TRIM attains lower leak and leak_T than both, so the two operations are complementary: vertical refinement restores detail within retained classes, while horizontal inclusion selects which QI groups enter training.

Figure 5k also isolates the effect of refine-and-swap with two variants that keep both operation families of TRIM but replace refine-and-swap: RefineOnly applies the bare refinement, and RefineEvict evicts the classes that a refinement leaves with fewer than 30 records, without admitting others. RefineOnly creates persistent privacy-risk spikes and thus achieves no risk reduction (leak $= 0$) under most utility tolerances. At the same leak, RefineEvict incurs up to $2.67\times$ as much utility degradation as TRIM. By evicting risky fragments and admitting larger classes in their place, refine-and-swap avoids privacy spikes and consistently yields the best privacy-utility tradeoff.

Selection-rule ablation. Figure 5l, from five runs with a 1% seed, replaces the ratio-marginal rule with utility-only (Utility), privacy-only (Privacy), random (Random) and influence-function (IF) selection. The ratio-marginal rule has the best privacy-utility tradeoff by normalizing utility gain by leakage cost. At $\epsilon = 0.02$, it lowers leak by 1.22, 4.10, and 3.18 relative to Utility, Privacy and IF, respectively. Utility over-spends privacy, Privacy favors low-risk but low-utility operations, and IF poorly captures composed retention effects. Thus, the ratio-marginal score is the most effective of these selection rules.

Summary. We find that: (1) At the same leak, TRIM reduces utility degradation by up to 92.9% over the strongest data-minimization baselines for both MLP and XGBoost. (2) It yields the highest QI reconstruction and inference errors among all methods. (3) It is up to $115\times$ faster than the baselines that reduce the worst-case privacy risks, and its runtime grows about linearly, by $7.6\times$ for $7.9\times$ more data. (4) Its logs support DPIA-style review of necessity and proportionality; it is insensitive to κ and the seed ratio, and on `PubCov` the advantage holds from two known QIs on. (5) Each key design choice outperforms its alternatives in the ablation studies.

VIII. Conclusion

The novelty of the work lies in (1) a formulation that minimizes residual re-identification risk under an explicit utility constraint; (2) a retention view that builds a minimized dataset from a fully suppressed one, coupling attribute refinement with record exchanges; (3) a ratio-marginal strategy with a two-stage utility estimator that avoids retraining the target model per candidate; and (4) an audit log of the utility gain and privacy cost of each committed operation. Our experimental study has verified that TRIM is promising in practice.

A DPIA using TRIM should consider three limitations: (1) leak is set by the smallest retained class, so it ignores how many records larger classes hold; (2) Rare QI combinations form small classes, the costliest to admit and first evicted (like t_4), so the minimized dataset may underrepresent rare subgroups, whose disparate impact the DPIA should assess [20]; and (3) leak does not bound what an adversary who knows other class members' values can infer, so a DPIA that retains sensitive non-QIs may add diversity checks [26].

Future research for TRIM includes (a) extending it from supervised ML training to broader data-driven workflows, such as analytics, data sharing, and model evaluation; and (b) generalizing it beyond relational data to multimodal settings, including text, images, time series, and genomic data.

ACKNOWLEDGMENT

We used AI tools in three parts of this work. (1) Generalization hierarchies (Section V): ChatGPT with web search generated the generalization hierarchy for each QI from its attribute name and value domain, rather than from the records; we reviewed each hierarchy and corrected it where necessary (see [18]). (2) Experiments (Section VII): Codex and the GPT-5.6 model assisted with running the experiments and debugging the scripts used to execute them; we reviewed the code and checked the results. (3) *Writing*: Codex + the GPT-5.6 model helped polish the wording and grammar of the text.

REFERENCES

[1] N. Subramani, S. Luccioni, J. Dodge, and M. Mitchell, "Detecting Personal Information in Training Corpora: An Analysis," in *Workshop on Trustworthy Natural Language Processing (TrustNLP)*, Jul. 2023, pp. 208–220. [Online]. Available: https://aclanthology.org/2023.trustnlp-1.18

[2] A. Goldsteen, G. Ezov, R. Shmelkin, M. Moffie, and A. Farkash, "Data minimization for GDPR compliance in machine learning models," *AI and Ethics*, vol. 2, no. 3, pp. 477–491, 2022. [Online]. Available: https://link.springer.com/10.1007/s43681-021-00095-8

[3] P. Ganesh, C. Tran, R. Shokri, and F. Fioretto, "The Data Minimization Principle in Machine Learning," in *ACM Conference on Fairness, Accountability, and Transparency (FAccT)*, Jun. 2025, pp. 3075–3093. [Online]. Available: https://doi.org/10.1145/3715275.3732195

[4] European Parliament and Council of the European Union, "Regulation (EU) 2016/679 of the European Parliament and of the Council of 27 April 2016 on the protection of natural persons with regard to the processing of personal data and on the free movement of such data, and repealing Directive 95/46/EC (General Data Protection Regulation)," 2016. [Online]. Available: https://eur-lex.europa.eu/eli/reg/2016/679/oj

[5] S. Garfinkel, J. M. Abowd, and C. Martindale, "Understanding database reconstruction attacks on public data," *Communications of the ACM (CACM)*, vol. 62, no. 3, pp. 46–53, 2019.

[6] M. Giomi, F. Boenisch, C. Wehmeyer, and B. Tasnádi, "A Unified Framework for Quantifying Privacy Risk in Synthetic Data," *Proceedings on Privacy Enhancing Technologies (PETS)*, vol. 2023, no. 2, pp. 312–328, Apr. 2023. [Online]. Available: https://petsymposium.org/popets/2023/popets-2023-0055.php

[7] Q. Li, J. Hong, C. Xie, J. Tan, R. Xin, J. Hou, X. Yin, Z. Wang, D. Hendrycks, Z. Wang, B. Li, B. He, and D. Song, "LLM-PBE: Assessing Data Privacy in Large Language Models," *PVLDB*, vol. 17, no. 11, pp. 3201–3214, Jul. 2024. [Online]. Available: https://dl.acm.org/doi/10.14778/3681954.3681994

[8] S. Kim, S. Yun, H. Lee, M. Gubri, S. Yoon, and S. J. Oh, "ProPILE: Probing Privacy Leakage in Large Language Models," in *Advances in Neural Information Processing Systems (NeurIPS)*, vol. 36, Dec. 2023, pp. 20 750–20 762. [Online]. Available: https://proceedings.neurips.cc/paper_files/paper/2023/hash/420678bb4c8251ab30e765bc27c3b047-Abstract-Conference.html

[9] N. Carlini, C. Liu, Ú. Erlingsson, J. Kos, and D. Song, "The secret sharer: Evaluating and testing unintended memorization in neural networks," in *USENIX Security Symposium (Security)*, Aug. 2019, pp. 267–284.

[10] R. Staab, N. Jovanović, M. Balunović, and M. Vechev, "From Principle to Practice: Vertical Data Minimization for Machine Learning," in *IEEE Symposium on Security and Privacy (SP)*, May 2024, pp. 4733–4752. [Online]. Available: https://ieeexplore.ieee.org/document/10646763/

[11] D. Shanmugam, F. Diaz, S. Shabanian, M. Finck, and A. Biega, "Learning to Limit Data Collection via Scaling Laws: A Computational Interpretation for the Legal Principle of Data Minimization," in *ACM Conference on Fairness, Accountability, and Transparency (FAccT)*, Jun. 2022, pp. 839–849. [Online]. Available: https://dl.acm.org/doi/10.1145/3531146.3533148

[12] X. Niu, R. Rahman, X. Wu, Z. Fu, D. Xu, and R. Qiu, "Leveraging Uncertainty Quantification for Reducing Data for Recommender Systems," in *IEEE International Conference on Big Data (BigData)*, Dec. 2023, pp. 352–359. [Online]. Available: https://ieeexplore.ieee.org/abstract/document/10386790

[13] B. Mirzasoleiman, J. Bilmes, and J. Leskovec, "Coresets for data-efficient training of machine learning models," in *International Conference on Machine Learning (ICML)*, vol. 119, Jul. 2020, pp. 6950–6960.

[14] M. Paul, S. Ganguli, and G. K. Dziugaite, "Deep Learning on a Data Diet: Finding Important Examples Early in Training," in *Advances in Neural Information Processing Systems (NeurIPS)*, vol. 34, Nov. 2021, pp. 20 596–20 607. [Online]. Available: https://openreview.net/forum?id=Uj7pF-D-YvT

[15] Y. Yang, H. Kang, and B. Mirzasoleiman, "Towards sustainable learning: Coresets for data-efficient deep learning," in *International Conference on Machine Learning (ICML)*, vol. 202, Jul. 2023, pp. 39 314–39 330.

[16] M. Saar-Tsechansky, P. Melville, and F. Provost, "Active Feature-Value Acquisition," *Management Science*, vol. 55, no. 4, pp. 664–684, Apr. 2009, *https://pubsonline.informs.org/doi/10.1287/mnsc.1080.0952.*

[17] B. Strack, J. P. DeShazo, C. Gennings, J. L. Olmo, S. Ventura, K. J. Cios, and J. N. Clore, "Impact of HbA1c measurement on hospital readmission rates: Analysis of 70,000 clinical database patient records," *BioMed Research International*, vol. 2014, 2014.

[18] "Full version with appendices," 2026, https://shuhaoliu.github.io/assets/papers/trim-full.pdf.

[19] C. F. Aliferis, A. R. Statnikov, I. Tsamardinos, S. Mani, and X. D. Koutsoukos, "Local causal and markov blanket induction for causal discovery and feature selection for classification part I: Algorithms and empirical evaluation," *J. Mach. Learn. Res.*, vol. 11, pp. 171–234, 2010.

[20] A. J. Biega, P. Potash, H. Daumé, F. Diaz, and M. Finck, "Operationalizing the Legal Principle of Data Minimization for Personalization," in *ACM Conference on Research and Development in Information Retrieval (SIGIR)*, Jul. 2020, pp. 399–408. [Online]. Available: https://dl.acm.org/doi/10.1145/3397271.3401034

[21] B. Sorscher, R. Geirhos, S. Shekhar, S. Ganguli, and A. S. Morcos, "Beyond neural scaling laws: Beating power law scaling via data pruning," in *Advances in Neural Information Processing Systems (NeurIPS)*, vol. 35, Oct. 2022, pp. 19 523–19 536. [Online]. Available: https://openreview.net/forum?id=UmvSlP-PyV

[22] T. Shaowang, S. Liu, J. Marques, N. Feamster, and S. Krishnan, "Algorithmic data minimization for machine learning over internet-of-things data streams," *PVLDB*, vol. 18, no. 13, pp. 5652–5661, 2025.

[23] S. M. Lundberg and S.-I. Lee, "A unified approach to interpreting model predictions," in *Advances in Neural Information Processing Systems (NIPS)*, vol. 30, Dec. 2017, pp. 4768–4777.

[24] A. Cohen and K. Nissim, "Towards formalizing the GDPR's notion of singling out," *Proceedings of the National Academy of Sciences (PNAS)*, vol. 117, no. 15, pp. 8344–8352, Apr. 2020. [Online]. Available: https://pnas.org/doi/full/10.1073/pnas.1914598117

[25] C. Dwork and A. Roth, "The Algorithmic Foundations of Differential Privacy," *Foundations and Trends in Theoretical Computer Science*, vol. 9, no. 3–4, pp. 211–407, Aug. 2014.

[26] A. Machanavajjhala, D. Kifer, J. Gehrke, and M. Venkitasubramaniam, "l-diversity: Privacy beyond k-anonymity," *ACM Transactions on Knowledge Discovery from Data (TKDD)*, vol. 1, no. 1, pp. 3–es, 2007.

[27] N. Li, T. Li, and S. Venkatasubramanian, "t-Closeness: Privacy beyond k-Anonymity and ℓ-Diversity," in *IEEE International Conference on Data Engineering (ICDE)*, 2007, pp. 106–115.

[28] J. Brickell and V. Shmatikov, "The cost of privacy: Destruction of data-mining utility in anonymized data publishing," in *ACM SIGKDD International Conference on Knowledge Discovery and Data Mining (KDD)*, 2008, pp. 70–78.

[29] P. Samarati, "Protecting respondents identities in microdata release," *IEEE Transactions on Knowledge and Data Engineering (TKDE)*, vol. 13, no. 6, pp. 1010–1027, 2001. [Online]. Available: http://ieeexplore.ieee.org/document/971193/

[30] V. S. Iyengar, "Transforming data to satisfy privacy constraints," in *ACM SIGKDD International Conference on Knowledge Discovery and Data Mining (KDD)*, 2002, pp. 279–288.

[31] K. Wang, P. S. Yu, and S. Chakraborty, "Bottom-up generalization: A data mining solution to privacy protection," in *IEEE International Conference on Data Mining (ICDM)*, 2004, pp. 249–256.

[32] K. LeFevre, D. J. DeWitt, and R. Ramakrishnan, "Workload-aware anonymization," in *ACM SIGKDD International Conference on Knowledge Discovery and Data Mining (KDD)*, 2006, pp. 277–286.

[33] ——, "Workload-aware anonymization techniques for large-scale datasets," *ACM Transactions on Database Systems (TODS)*, vol. 33, no. 3, pp. 17:1–17:47, 2008.

[34] F. Prasser, J. Eicher, H. Spengler, R. Bild, and K. A. Kuhn, "Flexible data anonymization using ARX—Current status and challenges ahead," *Software: Practice and Experience*, vol. 50, no. 7, pp. 1277–1304, 2020.

[35] D. Kifer and J. Gehrke, "Injecting utility into anonymized datasets," in *ACM International Conference on Management of Data (SIGMOD)*, 2006, pp. 217–228.

[36] B. Rastegarpanah, K. Gummadi, and M. Crovella, "Auditing Black-Box Prediction Models for Data Minimization Compliance," in *International Conference on Neural Information Processing Systems (NeurIPS)*, vol. 34, 2021, pp. 20 621–20 632. [Online]. Available: https://proceedings.neurips.cc/paper_files/paper/2021/hash/ac6b3cce8c74b2e23688c3e45532e2a7-Abstract.html

[37] A. Goldsteen, G. Ezov, R. Shmelkin, M. Moffie, and A. Farkash, "Anonymizing machine learning models," in *Data Privacy Management, Cryptocurrencies and Blockchain Technology (DPM)*, ser. LNCS, vol. 13140, 2021.

[38] D. Slijepčević, M. Henzl, L. D. Klausner, T. Dam, P. Kieseberg, and M. Zeppelzauer, "k-anonymity in practice: How generalisation and suppression affect machine learning classifiers," *Computers & Security*, vol. 111, p. 102488, 2021.

[39] B. C. M. Fung, K. Wang, and P. S. Yu, "Top-down specialization for information and privacy preservation," in *IEEE International Conference on Data Engineering (ICDE)*, 2005, pp. 205–216.

[40] ——, "Anonymizing classification data for privacy preservation," *IEEE Transactions on Knowledge and Data Engineering (TKDE)*, vol. 19, no. 5, pp. 711–725, 2007.

[41] S. Liu, W. Fan, and Y. Xu, "Shielding PII to prevent re-identification and preserve utility," *Proc. ACM Manag. Data*, vol. 4, no. 3, May 2026.

[42] P. Lison, I. Pilán, D. Sanchez, M. Batet, and L. Øvrelid, "Anonymisation Models for Text Data: State of the art, Challenges and Future Directions," in *Annual Meeting of the Association for Computational Linguistics (ACL)*, Aug. 2021, pp. 4188–4203. [Online]. Available: https://aclanthology.org/2021.acl-long.323/

[43] A. Riabi, M. Mahamdi, V. Mouilleron, and D. Seddah, "Cloaked Classifiers: Pseudonymization Strategies on Sensitive Classification Tasks," in *Workshop on Privacy in Natural Language Processing (PrivateNLP)*, Aug. 2024, pp. 123–136. [Online]. Available: https://aclanthology.org/2024.privatenlp-1.13/

[44] V. T. Chakaravarthy, H. Gupta, P. Roy, and M. K. Mohania, "Efficient techniques for document sanitization," in *ACM Conference on Information and Knowledge Management (CIKM)*, Oct. 2008, pp. 843–852. [Online]. Available: https://dl.acm.org/doi/10.1145/1458082.1458194

[45] M. I. Szawerna, S. Dobnik, T. Lindström Tiedemann, R. Muñoz Sánchez, X.-S. Vu, and E. Volodina, "Pseudonymization Categories across Domain Boundaries," in *Joint International Conference on Computational Linguistics, Language Resources and Evaluation (LREC-COLING)*, May 2024, pp. 13 303–13 314. [Online]. Available: https://aclanthology.org/2024.lrec-main.1164/

[46] Microsoft, "Presidio," 2025. [Online]. Available: https://microsoft.github.io/presidio/

[47] spaCy Developers, "spaCy·Industrial-strength Natural Language Processing in Python," 2025. [Online]. Available: https://spacy.io/

[48] B. Zhao, K. R. Mopuri, and H. Bilen, "Dataset condensation with gradient matching," in *International Conference on Learning Representations (ICLR)*. OpenReview.net, 2021. [Online]. Available: https://openreview.net/forum?id=mSAKhLYLSsl

[49] T. Wang, J.-Y. Zhu, A. Torralba, and A. A. Efros, "Dataset distillation," *CoRR*, vol. 1811.10959, 2018. [Online]. Available: https://arxiv.org/abs/1811.10959

[50] B. Zhao and H. Bilen, "Dataset condensation with distribution matching," in *IEEE/CVF Winter Conference on Applications of Computer Vision (WACV)*. IEEE, 2023, pp. 6503–6512.

[51] J. Zhang, G. Cormode, C. M. Procopiuc, D. Srivastava, and X. Xiao, "PrivBayes: Private data release via Bayesian networks," *ACM Transactions on Database Systems (TODS)*, vol. 42, no. 4, 2017.

[52] R. McKenna, B. Mullins, D. Sheldon, and G. Miklau, "AIM: An adaptive and iterative mechanism for differentially private synthetic data," *PVLDB*, vol. 15, no. 11, pp. 2599–2612, 2022.

[53] K. Yu, L. Liu, J. Li, W. Ding, and T. D. Le, "Multi-source causal feature selection," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 42, no. 9, pp. 2240–2256, 2020.

[54] P. A. Estévez, M. Tesmer, C. A. Perez, and J. M. Zurada, "Normalized mutual information feature selection," *IEEE Trans. Neural Networks*, vol. 20, no. 2, pp. 189–201, 2009.

[55] C. Dwork, F. McSherry, K. Nissim, and A. Smith, "Calibrating Noise to Sensitivity in Private Data Analysis," in *Theory of Cryptography (TCC)*, 2006, pp. 265–284.

[56] R. Shokri, M. Stronati, C. Song, and V. Shmatikov, "Membership inference attacks against machine learning models," in *IEEE Symposium on Security and Privacy (SP)*, 2017, pp. 3–18.

[57] M. Abadi, A. Chu, I. Goodfellow, H. B. McMahan, I. Mironov, K. Talwar, and L. Zhang, "Deep learning with differential privacy," in *ACM Conference on Computer and Communications Security (CCS)*, 2016, pp. 308–318.

[58] G. Andrew, O. Thakkar, B. McMahan, and S. Ramaswamy, "Differentially private learning with adaptive clipping," in *Advances in Neural Information Processing Systems (NeurIPS)*, vol. 34, 2021, pp. 17 455–17 466.

[59] B. Balle, J. Bell, A. Gascón, and K. Nissim, "The privacy blanket of the shuffle model," in *International Cryptology Conference (Crypto)*, 2019, pp. 638–667.

[60] K. A. Bonawitz, H. Eichner, W. Grieskamp, D. Huba, A. Ingerman, V. Ivanov, C. Kiddon, J. Konečný, S. Mazzocchi, B. McMahan, T. V. Overveldt, D. Petrou, D. Ramage, and J. Roselander, "Towards federated learning at scale: System design," in *Proceedings of Machine Learning and Systems (MLSys)*, vol. 1, 2019, pp. 374–388.

[61] A. Narayanan and V. Shmatikov, "Robust de-anonymization of large sparse datasets," in *IEEE Symposium on Security and Privacy (SP)*, 2008, pp. 111–125.

[62] K. El Emam and F. K. Dankar, "Protecting privacy using k-Anonymity," *Journal of the American Medical Informatics Association (JAMIA)*, vol. 15, no. 5, pp. 627–637, 2008.

[63] ARTICLE 29 DATA PROTECTION WORKING PARTY, "Opinion 05/2014 on "Anonymisation Techniques"," 2014. [Online]. Available: https://ec.europa.eu/justice/article-29/documentation/opinion-recommendation/files/2014/wp216_en.pdf

[64] N. Carlini, F. Tramèr, E. Wallace, M. Jagielski, A. Herbert-Voss, K. Lee, A. Roberts, T. Brown, D. Song, Ú. Erlingsson, A. Oprea, and C. Raffel, "Extracting Training Data from Large Language Models," in *USENIX Security Symposium (Security)*, 2021, pp. 2633–2650. [Online]. Available: https://www.usenix.org/conference/usenixsecurity21/presentation/carlini-extracting

[65] G. Smith, "On the foundations of quantitative information flow," in *International Conference on Foundations of Software Science and Computational Structures (FoSSaCS)*, 2009, pp. 288–302.

[66] European Medicines Agency, "External guidance on the implementation of the European Medicines Agency policy on the publication of clinical data for medicinal products for human use," 2016, eMA/90915/2016.

[67] R. C.-W. Wong, A. W.-C. Fu, K. Wang, and J. Pei, "Minimality attack in privacy preserving data publishing," in *International Conference on Very Large Data Bases (VLDB)*, 2007, pp. 543–554.

[68] A. Cohen, "Attacks on deidentification's defenses," in *USENIX Security Symposium (Security)*, 2022.

[69] D. Kifer, "Attacks on privacy and deFinetti's theorem," in *ACM International Conference on Management of Data (SIGMOD)*, 2009, pp. 127–138.

[70] C. Dwork and M. Naor, "On the difficulties of disclosure prevention in statistical databases or the case for differential privacy," *Journal of Privacy and Confidentiality*, vol. 2, no. 1, 2010.

[71] K. LeFevre, D. J. DeWitt, and R. Ramakrishnan, "Mondrian multidimensional k-anonymity," in *IEEE International Conference on Data Engineering (ICDE)*, 2006, p. 25.

[72] A. Inan, M. Kantarcioglu, and E. Bertino, "Using anonymized data for classification," in *IEEE International Conference on Data Engineering (ICDE)*, 2009, pp. 429–440.

[73] M. Garey and D. Johnson, *Computers and Intractability: A Guide to the Theory of NP-Completeness*. W. H. Freeman and Company, 1979.

[74] A. Meyerson and R. Williams, "On the complexity of optimal k-anonymity," in *ACM SIGMOD-SIGACT-SIGART Symposium on Principles of Database Systems (PODS)*, 2004, pp. 223–228.

[75] G. Aggarwal, T. Feder, K. Kenthapadi, R. Motwani, R. Panigrahy, D. Thomas, and A. Zhu, "Anonymizing tables," in *International Conference on Database Theory (ICDT)*, 2005, pp. 246–258.

[76] P. W. Koh and P. Liang, "Understanding black-box predictions via influence functions," in *International Conference on Machine Learning (ICML)*, vol. 70. PMLR, 2017, pp. 1885–1894. [Online]. Available: https://proceedings.mlr.press/v70/koh17a.html

[77] A. Warnecke, L. Pirch, C. Wressnegger, and K. Rieck, "Machine Unlearning of Features and Labels," in *Network and Distributed System Security Symposium (NDSS)*, 2023. [Online]. Available: https://www.ndss-symposium.org/wp-content/uploads/2023/02/ndss2023_s87_paper.pdf

[78] G. Pruthi, F. Liu, S. Kale, and M. Sundararajan, "Estimating training data influence by tracing gradient descent," in *Advances in Neural Information Processing Systems (NeurIPS)*, vol. 33, 2020, pp. 19 920–19 930.

[79] S. Basu, X. You, and S. Feizi, "On second-order group influence functions for black-box predictions," in *International Conference on Machine Learning (ICML)*. PMLR, 2020, pp. 715–724.

[80] A. Ghorbani and J. Zou, "Data shapley: Equitable valuation of data for machine learning," in *International Conference on Machine Learning (ICML)*. PMLR, 2019, pp. 2242–2251.

[81] K. Killamsetty, D. Sivasubramanian, G. Ramakrishnan, and R. Iyer, "GLISTER: Generalization based data subset selection for efficient and robust learning," in *AAAI Conference on Artificial Intelligence (AAAI)*, vol. 35, no. 9, 2021, pp. 8110–8118.

[82] K. Killamsetty, D. S, G. Ramakrishnan, A. De, and R. Iyer, "GRAD-MATCH: Gradient matching based data subset selection for efficient deep model training," in *International Conference on Machine Learning (ICML)*, ser. Proceedings of Machine Learning Research, vol. 139, 2021, pp. 5464–5474. [Online]. Available: https://proceedings.mlr.press/v139/killamsetty21a.html

[83] C. Coleman, C. Yeh, S. Mussmann, B. Mirzasoleiman, P. Bailis, P. Liang, J. Leskovec, and M. Zaharia, "Selection via proxy: Efficient data selection for deep learning," *arXiv preprint arXiv:1906.11829*, 2019.

[84] O. Bousquet and A. Elisseeff, "Stability and generalization," *Journal of Machine Learning Research*, vol. 2, no. Mar, pp. 499–526, 2002. [Online]. Available: https://www.jmlr.org/papers/v2/bousquet02a.html

[85] F. Ding, M. Hardt, J. Miller, and L. Schmidt, "Retiring adult: New datasets for fair machine learning," in *Advances in Neural Information Processing Systems (NeurIPS)*, vol. 34, 2021, pp. 6478–6490.

[86] S. Moro, P. Cortez, and P. Rita, "A data-driven approach to predict the success of bank telemarketing," *Decision Support Systems*, vol. 62, pp. 22–31, 2014. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S016792361400061X

[87] O. Platform, "Bng (credit-g) dataset," 2026, *https://www.openml.org/search?type=data&id=260&sort=runs&status=active*.

[88] T. Chen and C. Guestrin, "Xgboost: A scalable tree boosting system," in *ACM International Conference on Knowledge Discovery and Data Mining (KDD)*, 2016, pp. 785–794.

[89] S. Yang, Z. Xie, H. Peng, M. Xu, M. Sun, and P. Li, "Dataset pruning: Reducing training data by examining generalization influence," in *International Conference on Learning Representations (ICLR)*, 2023. [Online]. Available: https://openreview.net/forum?id=4wZiAXD29TQ

[90] A. Jacot, F. Gabriel, and C. Hongler, "Neural tangent kernel: Convergence and generalization in neural networks," in *Advances in Neural Information Processing Systems (NeurIPS)*, vol. 31. Curran Associates, Inc., 2018. [Online]. Available: https://proceedings.neurips.cc/paper/2018/hash/5a4be1fa34e62bb8a6ec6b91d2462f5a-Abstract.html

[91] S. Arora, S. Du, W. Hu, Z. Li, and R. Wang, "Fine-grained analysis of optimization and generalization for overparameterized two-layer neural networks," in *International Conference on Machine Learning (ICML)*, vol. 97. PMLR, 2019, pp. 322–332. [Online]. Available: https://proceedings.mlr.press/v97/arora19a.html

[92] S. Moro, P. Rita, and P. Cortez, "Bank Marketing," UCI Machine Learning Repository, 2014. [Online]. Available: https://archive.ics.uci.edu/dataset/222/bank+marketing

[93] United States Congress, "Health insurance portability and accountability act of 1996," 1996. [Online]. Available: https://www.congress.gov/bill/104th-congress/house-bill/3103

APPENDIX A
PROOF OF PROPOSITION 1

Proposition 1. If $\mathcal{A}$ knows exact QI values, then $\mathsf{leak}(S) \leq -\log k$ if and only if S is k-anonymous. □

Proof. Consider a candidate dataset S and an integer $k \geq 1$. Under exact QI knowledge, by the definition of compatibility, the candidate set of each retained individual $i \in \mathcal{I}(S)$ is exactly the equivalence class of records that share its generalized QI values, of size $k_i(S) \geq 1$. The matching model thus gives $r_i(S) = 1/k_i(S)$ for retained individuals and $r_i(S) = 0$ for individuals absent from S. By Equation (1),

$$\begin{aligned}\mathsf{leak}(S) &= \log \max_{i \in \mathcal{I}} r_i(S) = \log \max_{i \in \mathcal{I}(S)} \frac{1}{k_i(S)} \\ &= -\log \min_{i \in \mathcal{I}(S)} k_i(S).\end{aligned}$$

Every equivalence class in S contains a retained individual. Thus, S is k-anonymous exactly when $\min_{i \in \mathcal{I}(S)} k_i(S) \geq k$. Since $-\log x$ is strictly decreasing for $x > 0$, this condition is equivalent to $\mathsf{leak}(S) \leq -\log k$. □

Conservativeness. Computed with exact QIs and known membership, $\mathsf{leak}(S)$ bounds the matching probability of every adversary in the threat model of Section III-A, under its uniform-guess assumption. Fix S and a target i. If $i \notin \mathcal{I}(S)$, every adversary fails, since $r_i(S) = 0$; so let $i \in \mathcal{I}(S)$.

We consider two weaker adversaries. (1) One that knows less about i, *i.e.,* a set $B_i' \supseteq B_i$ of QI combinations. Each record compatible with i under B_i is compatible under B_i', and the record of i stays compatible because B_i contains its true combination. The candidate set can thus only grow, and the uniform guess succeeds with probability at most $r_i(S)$. (2) The other that is unsure whether $i \in \mathcal{I}(S)$, and assigns it probability $\pi_i \leq 1$. Its guess succeeds only if $i \in \mathcal{I}(S)$ and it picks the record of i, with probability at most $\pi_i / k_i(S) \leq r_i(S)$. Taking the maximum over i shows that no such adversary matches any target with probability $> e^{\mathsf{leak}(S)}$.

APPENDIX B
PROOF OF THEOREM 2

Theorem 2. The decision version of DM is NP-complete. □

Given a dataset $\mathcal{D}$, a model $\mathcal{M}$, and thresholds τ and ϵ, the decision version of DM asks whether some minimized dataset S^* satisfies $\mathsf{leak}(S^*) \leq \tau$ and $\Delta U(S^*) \leq \epsilon$.

Proof. We show membership in NP and give a polynomial-time reduction from Independent Set.

Upper bound. We give an NP verifier for the decision version of DM. It guesses a minimized dataset $S^* = o_n \circ \cdots \circ o_1(\mathcal{D})$, represented by the sequence of minimization operations applied to $\mathcal{D}$. Every operation only coarsens or removes information in a finite collection of cells, so no operation needs to be applied twice, and the number of distinct operations is polynomial in $|\mathcal{D}|$ and the schema size. Hence S^* admits a description of polynomial size. Given S^*, the verifier computes the residual privacy risk $\mathsf{leak}(S^*)$ and the utility degradation $\Delta U(S^*)$, and accepts if and only if $\mathsf{leak}(S^*) \leq \tau$ and $\Delta U(S^*) \leq \epsilon$. Both quantities are computable in PTIME, so the verifier runs in PTIME and DM is in NP.

Table V. The eight records created for an edge $e = \{u, v\}$. The last column gives every A_j with $j \notin \{u, v\}$.

C	A_u	A_v	Other A_j
$(e, 0)$	0	0	0
$(e, 0)$	0	1	0
$(e, 0)$	1	0	0
$(e, 0)$	1	1	0
$(e, 1)$	0	0	1
$(e, 1)$	0	1	1
$(e, 1)$	1	0	1
$(e, 1)$	1	1	1

Lower bound. We reduce from NP-complete *Independent Set* [73]: given a graph $\mathcal{G} = (V, E)$ and an integer K, decide whether at least K vertices can be selected without selecting both endpoints of any edge. Write $n = |V|$ and $m = |E|$. We consider $m \geq 1$ and $1 \leq K \leq n$. These restrictions preserve hardness: trivial values of K can be handled separately, and adding two new vertices joined by one edge, while increasing K by one, preserves the answer and ensures $m \geq 1$.

Given an instance $(\mathcal{G}, K)$ of Independent Set, we construct an instance $(\mathcal{D}, \mathcal{M}, \tau, \epsilon)$ of DM and show that $\mathcal{G}$ has an independent set of size at least K if and only if the constructed instance has a minimized dataset S^* with $\mathsf{leak}(S^*) \leq \tau$ and $\Delta U(S^*) \leq \epsilon$.

The reduction proceeds in two steps. Step 1 encodes vertex selection by attribute generalization, with all records retained. Step 2 shows that also allowing record removal leaves the answer unchanged. We fix a polynomial-time training and evaluation protocol for the constructed instances. Since $\mathcal{M}$ is part of the input of DM, a reduction that uses this protocol suffices to prove hardness of the general problem.

Dataset construction. Create a binary QI A_j for each vertex $j \in V$, and a categorical QI C whose value identifies a group of four individuals. For every edge $e = \{u, v\}$, create two groups with distinct values $C = (e, 0)$ and $C = (e, 1)$, where e denotes the edge's index in $\{1, \ldots, m\}$. Table V lists their records. In each group, (A_u, A_v) takes all four binary combinations, and every other vertex attribute is constant: zero in the first group and one in the second. Every record gets the label vector $y = (1, \ldots, 1)$ with $n + 1$ components.

The resulting table $\mathcal{D}$ has $N = 8m$ records and $b = 2m$ group values, and each record represents a different individual. Every vertex column contains exactly $N/2$ ones, including the columns of isolated vertices: an incident edge contributes two ones in each of its groups, and a nonincident edge contributes four ones in its second group.

All QIs have two-level generalization trees, with their exact values as leaves and $*$ as the root. Each QI column may be generalized to $*$ as a whole, and any row may be removed. A missing column is treated as unavailable. In particular, C is a QI subject to the same operations. The adversary knows each individual's exact QI values. The constant labels add no distinctions between individuals, and the prediction rules below

specify what happens when a required label is unavailable.

A fixed learning and evaluation protocol. The model has binary outputs $h_0, h_1, \ldots, h_n$. Output h_0 uses the group column, and output h_j uses the vertex column A_j. The model schema specifies the original n vertex columns and b group values, even when a candidate removes columns or records. For this model family, the training rules and evaluation formulas below are fixed parts of the protocol specified by $\mathcal{M}$, and they read their parameters from this original schema. Predictions are evaluated on the common original input domain, with input $x = (x_C, x_1, \ldots, x_n)$. For each group value c, let $d_c(S)$ count the records in S whose C value is exactly c and whose label y_0 is available and equals one; generalized values do not count as exact group values. The group predictor is

$$h_0^S(x) = \mathbf{1}\{d_{x_C}(S) \geq 4\}.$$

Thus, a group is predicted positive only when it has at least four exactly assigned positive training records. This rule uses counts in S and needs no copy of $\mathcal{D}$.

For each $j \in \{1, \ldots, n\}$, set $h_j^S(x) = 0$ if A_j is missing, if any retained value in that column is generalized, or if any required training label y_j is unavailable. Otherwise, select whichever of $x \mapsto 0$ and $x \mapsto x_j$ has fewer training errors, breaking ties in favor of zero. These rules also define training on the empty table, where all outputs equal zero. All candidates share the same test distribution: x_C is uniform over the original b group values; the x_j's are independent fair binary variables, independent of x_C; and $y = (1, \ldots, 1)$. The loss is

$$\ell(\widehat{y}, y) = b(n+1)\mathbf{1}\{\widehat{y}_0 \neq y_0\} + 2\sum_{j=1}^{n} \mathbf{1}\{\widehat{y}_j \neq y_j\}. \quad (6)$$

This rule depends only on the original schema sizes n and b. Its distribution and loss are identical for every candidate S, and it inspects neither the adjacency relation of $\mathcal{G}$ nor S. Thus, the construction uses the negative EGE of Equation (2).

Utility values and thresholds. On $\mathcal{D}$, every group has four positive records, so $h_0^{\mathcal{D}} = 1$ on the whole test domain. Every vertex column is half ones and all its training labels are one, so its trained predictor is $h_j^{\mathcal{D}}(x) = x_j$, with test error $1/2$ (the constant-zero predictor has test error 1). Thus $\mathrm{EGE}(\mathcal{D}) = n$.

For any candidate S, let $T(S)$ index the vertex outputs whose selected predictor is $x \mapsto x_j$, and let $z(S)$ count the groups with fewer than four exact positive training records:

$$z(S) = |\{c : d_c(S) < 4\}|.$$

Each vertex output outside $T(S)$ increases EGE by $2(1 - 1/2) = 1$, and each group counted in $z(S)$ increases it by $b(n+1)/b = n+1$. Taking the expectation of Equation (6) therefore gives the exact utility degradation

$$\Delta U(S) = \mathrm{EGE}(S) - \mathrm{EGE}(\mathcal{D}) = (n+1)z(S) + n - |T(S)|. \quad (7)$$

Both $(n+1)z(S)$ and $n - |T(S)|$ are nonnegative, so no improvement in a vertex predictor can offset the loss of a failed group predictor.

We set $\epsilon = n - K + \frac{1}{2} > 0$ and $\tau = -\frac{1}{2}$. For a nonempty release, the privacy constraint then requires at least two candidate records for every retained target: one candidate gives risk $\log 1 = 0 > \tau$, whereas two or more give risk at most $-\log 2 < -1/2$. The rational threshold τ thus implements this requirement under the matching-risk definition.

Step 1: attribute operations only. First, retain every record and allow arbitrary attribute operations. By Equation (7), every utility-feasible release has $z(S) = 0$, since $z(S) \geq 1$ gives $\Delta U(S) \geq n + 1 > \epsilon$. Generalizing column C drops the exact positive count of an original group below four, so a feasible release retains all exact C values. Its utility constraint becomes

$$n - |T(S)| \leq n - K + \tfrac{1}{2},$$

which, by integrality, is equivalent to $|T(S)| \geq K$. Every column indexed by $T(S)$ is complete and exact.

For the forward direction, let $T \subseteq V$ be an independent set with $|T| \geq K$. Retain C, the labels, and every vertex column indexed by T exactly, and replace every value of the other vertex columns by $*$. For each $j \in T$, all rows and the column A_j are unchanged, so $x \mapsto x_j$ still has $N/2$ training errors versus N for zero. Then $T(S) = T$, $z(S) = 0$, and $\Delta U(S) = n - |T| \leq n - K < \epsilon$. Consider any target in a group of edge $e = \{u, v\}$. Its exact C value excludes all other groups from its candidate set. Within its group, all nonendpoint columns are constant, and the endpoint columns take the four combinations of Table V. Since T is independent, at most one endpoint column is retained, so the target has four candidates if neither is retained and two if exactly one is. This covers every individual in $\mathcal{D}$, so the release satisfies the privacy constraint.

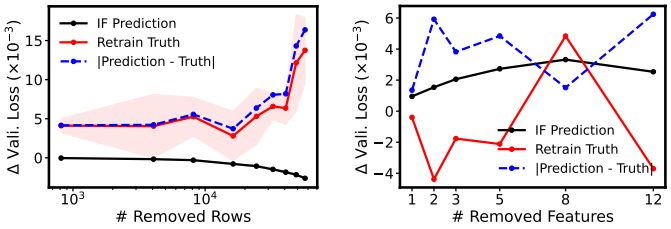
For the reverse direction, let S be any feasible release obtained by attribute operations. By the above, C remains exact and $|T(S)| \geq K$. Suppose $T(S)$ contains both endpoints of an edge $e = \{u, v\}$, so both endpoint columns are exact in every row. In either group of e, their four combinations distinguish each record from the other three, while C excludes every record outside the group. Each target in these groups thus has a single candidate, so $\mathsf{leak}(S) = 0 > \tau$, a contradiction. Hence $T(S)$ is an independent set of size at least K, which proves the equivalence for attribute operations alone.

Step 2: allowing record removal. Now also allow arbitrary row removals, without changing the instance or its evaluation protocol. Deleting any record leaves its original group c with $d_c(S) \leq 3$, and no other record can enter that exact group value, since the operations only remove rows or replace values by $*$. Thus, every candidate that removes a row has $z(S) \geq 1$ and, by Equation (7), $\Delta U(S) \geq n + 1 > \epsilon$. Such a candidate is utility-infeasible regardless of any simultaneous attribute operations; in particular, so is the empty release.

Therefore, every feasible release of the full problem retains all rows and is covered by Step 1.

Correctness. We now verify the correctness of this reduction.

($\Rightarrow$) Suppose $\mathcal{G}$ has an independent set $T \subseteq V$ with $|T| \geq K$. Let S^* be the release built from T in the forward direction of Step 1: it retains every record, C, the labels, and the vertex columns indexed by T, and replaces every value of the other vertex columns by $*$. Its utility degradation is $\Delta U(S^*) = n - |T| \leq n - K < \epsilon$, so the utility constraint is met. Every target has two or four candidates, so $\mathsf{leak}(S^*) \leq -\log 2 < \tau$,



(a) Record removal. **(b)** Feature removal.

Figure 6. IF estimates against actual utility degradation (`Diabetes`)

and the privacy constraint is met. Since S^* uses only attribute operations, it remains legal when row removal is allowed. Thus, S^* is a solution to the constructed DM instance.

($\Leftarrow$) Suppose that the constructed DM instance has a solution S^*, *i.e.,* $\mathsf{leak}(S^*) \leq \tau$ and $\Delta U(S^*) \leq \epsilon$. By Step 2, S^* removes no record, since any removal gives $\Delta U(S^*) \geq n+1 > \epsilon$. Hence S^* is obtained by attribute operations alone, and Step 1 applies. The utility constraint keeps C exact and implies $|T(S^*)| \geq K$. The privacy constraint implies that $T(S^*)$ contains at most one endpoint of every edge, since retaining both endpoint columns of an edge exactly gives $\mathsf{leak}(S^*) = 0 > \tau$. Let $T = T(S^*) \subseteq V$. Then no edge has both endpoints in T, and $|T| \geq K$. Therefore, T is an independent set of $\mathcal{G}$ of size at least K.

Polynomial size and evaluation. The table has $8m$ rows and $O(n)$ fields per row, with group values encoded in $O(\log m)$ bits. The two-level hierarchies, schema parameters, loss weights, and thresholds all have polynomial-size descriptions. The training rules need only scans and counts over the candidate table. Although the common test domain contains exponentially many feature combinations, Equation (7) gives the exact EGE difference without enumerating it. The candidate counts for privacy are computed directly from the released records and the exact target QIs. The construction and evaluation are therefore polynomial-time, which yields a polynomial-time reduction from Independent Set to DM. Together with the upper bound, this proves the theorem. □

Inapproximability. The reduction also separates the optimal values. In a yes-instance, the release in the forward direction of Step 1 gives every target two or four candidates, so its matching probability e^{leak} is at most $1/2$. In a no-instance, every feasible release retains both endpoint columns of some edge exactly, so the records of its groups are unique and $e^{\mathsf{leak}} = 1$. An algorithm that approximates the minimum of e^{leak} within a factor below 2 would distinguish the two cases; hence such an approximation is NP-hard.

Appendix C
Limitations of Influence Functions

This appendix supports challenge (2) of Section III-B.

Example 4: Figure 6 revisits Example 1. It compares influence-function (IF) estimates with the actual utility degradation after retraining on `Diabetes`, under cumulative record removals (Figure 6a) and feature removals (Figure 6b). In both cases, the estimated and actual degradations diverge as operations accumulate, showing the limitations of local influence. □

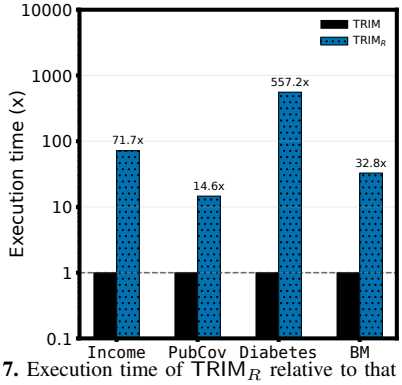


Figure 7. Execution time of TRIM_R relative to that of TRIM.

Appendix D
Retention View vs. Removal View

This appendix evaluates the efficiency advantage of the retention view over the removal view. The two views admit the same feasible minimized datasets, but they search in opposite directions and can thus incur very different computational costs. The retention view starts from $\mathcal{D}_0$, which has minimal data exposure (Section IV), so its early iterations evaluate candidates on small retained datasets. The following experiment isolates the resulting efficiency gain.

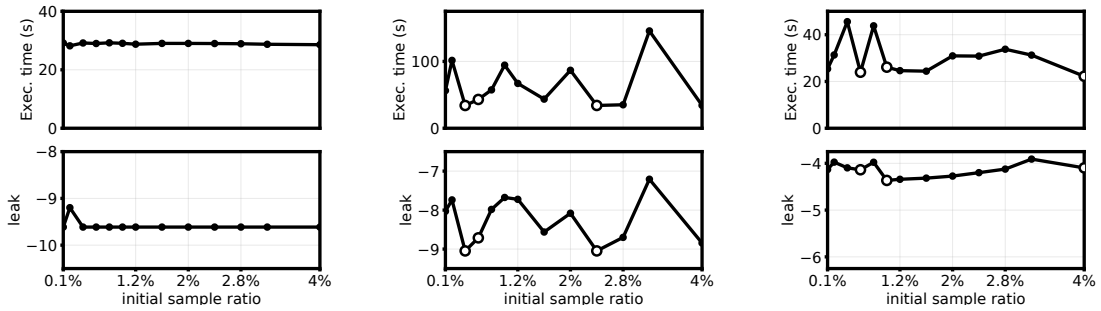
Experiment setup. We compare TRIM with its removal-view counterpart, denoted by TRIM_R. Starting from the full training set at the finest generalization levels, TRIM_R reduces the retained information in each iteration by coarsening one attribute by one level, removing one retention class, or exchanging a retained class for a removed one. As in the ratio-marginal rule of TRIM, it ranks operations by privacy progress per estimated utility loss, proxy-evaluates the top b (the next b if none meets the tolerance), and selects the largest ratio under the proxy. Progress is the increase in k^*, the smallest class size, or, if k^* is unchanged (*e.g.,* one of several such classes is removed), the relative drop in their number. Thus, TRIM_R is a greedy search over the same operations and the same minimization space as TRIM, but in the opposite direction.

To isolate the effect of the search direction, both methods use an MLP as the downstream model and share one stratified 70/15/15 data split (seed 42), the utility estimator, and $b = 7$. We set $\epsilon = 0.09$ for `Income`, $\epsilon = 0.05$ for `PubCov`, $\epsilon = 0.02$ for `Diabetes`, and $\epsilon = 0.005$ for `BM`.

Results. Figure 7 reports the execution times, normalized to that of TRIM. On `Income`, `PubCov`, `Diabetes`, and `BM`, TRIM_R takes $71.7\times$, $14.6\times$, $557.2\times$, and $32.8\times$ the execution time of TRIM, respectively. These ratios are lower bounds: TRIM_R stops at its cap of 50 iterations with leak still 0. Thus, the retention view is more than $10\times$ faster on all four datasets.

This difference has two causes.

(1) TRIM starts from the suppressed dataset $\mathcal{D}_0$, so its early iterations enumerate and evaluate candidates on small datasets. In contrast, TRIM_R starts from the full training set, where the



(a) PubCov. **(b)** Diabetes. **(c)** BM.

Figure 8. Varying seed ratio on the other datasets.

number of retention classes is close to the number of records, and must thus evaluate a large candidate set from the outset.

(2) In every iteration, TRIM_R removes a one-record class, which has the lowest estimated utility loss, and each removal shrinks the candidate set only marginally. Moreover, removing a class rarely changes the smallest class size, so many removals are needed before leak decreases, and TRIM_R repeats similarly expensive evaluations over many iterations. TRIM avoids this cost by admitting only the information needed to recover the target utility.

APPENDIX E
SEED RATIO ON OTHER DATASETS

This appendix repeats the seed-ratio experiment of Exp-5 (Figure 5f) on PubCov, Diabetes, and BM.

Experiment setup. We use the setting of Exp-3: the MLP target, all records, and $\epsilon = 0.1$ on PubCov, 0.005 on Diabetes, and 0.01 on BM. For each of 13 seed ratios from 0.1% to 4% and five seeds, we take the first dataset of the search that satisfies ϵ, and report its leak and the time to reach it, averaged over the seeds that satisfy ϵ; a hollow marker denotes a ratio at which a seed misses ϵ.

Results. As Figure 8 shows, TRIM is insensitive to the seed ratio on these datasets, as on Income. TRIM satisfies ϵ in 188 of the 195 runs. Seeds below 1% and above 1.2% differ in mean leak by at most 0.16 and in time by at most $1.19\times$. On PubCov, every ratio except 0.2% reaches leak $= -9.61$ in about 29 s. On Diabetes and BM, the points vary more, since a different seed can change the first operation, after which the search follows another path; this variation shows no trend in the ratio. As on Income, the first round evicts the whole seed in all 65 runs on PubCov and in 63 of 65 on BM. On Diabetes, it does so in 27 runs; in the other 38, the first round admits a retention class and keeps the seed.

APPENDIX F
SWAP INVARIANTS AND INCLUSION COST

This appendix states and proves the properties of the two operation families used in Section V.

Proposition 4: *Let o be an operation enumerated by* TRIM *at round t. (a) If o is a refine-and-swap, then every exchange in its swap keeps or increases both the number of retained records and the size of the smallest retained class. The swap ends after finitely many exchanges, and only when no not-yet-admitted part of a class is larger than the smallest retained class. (b) If o is a retention-class inclusion whose class has no retained member, then $\Delta\mathsf{leak}(o) \geq 0$, with equality when o admits at least as many records as the smallest retained class holds. If the class has retained members, then $\Delta\mathsf{leak}(o) \leq 0$.* □

Proof. We work at round t with the current dataset $\mathcal{D}_{t-1}$, in the setting of Proposition 1: $\mathcal{A}$ knows exact QI values, and each QI has the same granularity in all retained records. After a preliminary, the proof proceeds in two steps, one per claim.

Preliminaries. For a retained dataset S, call the retained records that share a QI signature a *retained class* of S. In this setting, the candidate set of a retained individual i is its retained class, so $r_i(S) = 1/k_i(S)$, where $k_i(S)$ is the size of that class, while $r_i(S) = 0$ for every removed individual. By Equation (1), $\mathsf{leak}(S) = -\log \min_C |C|$, where C ranges over the retained classes of S.

(a) Exchanges of refine-and-swap. Consider one exchange of the algorithm in Figure 2, applied to a dataset $\mathcal{D}'$ whose smallest retained class is C^-, with $|C^+| > |C^-|$ (line 4). The exchange evicts a set X of whole retained classes, taken in increasing size, whose total size is at most $|C^+|$, and it admits the $|C^+|$ records of C^+. The number of retained records therefore changes by $|C^+| - \sum_{C \in X} |C| \geq 0$. Since $|C^-| < |C^+|$, the class C^- is the first class placed in X. Every retained class that is not evicted keeps all its records, because X consists of whole classes, and it has at least $|C^-|$ records, because C^- is a smallest retained class. The class that receives C^+ has at least $|C^+| > |C^-|$ retained records afterwards. Hence the smallest retained class after the exchange has at least $|C^-|$ records. The loop at line 4 continues while $|C^+| > |C^-|$, where C^+ is the largest not-yet-admitted part of a class, so the swap stops only when no such part is larger than the smallest retained class.

It remains to show that the swap ends. An exchange with $X = \{C^-\}$ admits $|C^+| - |C^-| > 0$ more records than it evicts. An exchange with $|X| \geq 2$ removes at least one retained class without reducing the number of retained records, because C^+ joins at most one class. Thus, every exchange increases, in lexicographic order, the pair formed by the number of retained records and the negated number of retained

classes, and this pair takes finitely many values.

(b) Privacy cost of retention-class inclusion. Let o admit a set C of not-yet-admitted members of a retention class (all of them, or a sample of k_{t-1} of them). First, suppose that no member of the class is retained in $\mathcal{D}_{t-1}$. Because retention classes are defined by equality of QI signatures, no record of $\mathcal{D}_{t-1}$ shares the signature of C. Admitting C thus leaves every retained class of $\mathcal{D}_{t-1}$, and hence every r_i with $i \in \mathcal{I}(\mathcal{D}_{t-1})$, unchanged, and adds C as a new retained class. Hence,

$$\mathsf{leak}(\mathcal{D}_{t-1} \oplus o) = \max\{\mathsf{leak}(\mathcal{D}_{t-1}),\ -\log|C|\} \geq \mathsf{leak}(\mathcal{D}_{t-1}),$$

with equality exactly when $|C|$ is at least the size of the smallest retained class of $\mathcal{D}_{t-1}$.

Next, suppose that the class already has retained members, which form a retained class C_0 of $\mathcal{D}_{t-1}$. Admitting C replaces C_0 by the larger class $C_0 \cup C$ and leaves every other retained class unchanged. Hence $\Delta\mathsf{leak}(o) \leq 0$, with strict inequality when C_0 is the unique smallest retained class of $\mathcal{D}_{t-1}$. □

Appendix G
Proof of Proposition 3

Proposition 3. TRIM satisfies the following properties. (a) *Termination*: it commits at most $H+(H+1)N$ operations, and each round retrains the proxy at most $4b+1$ times ($4b+4\bar{b}+1$ if widened) and the target once; (b) *Feasibility*: its output passes the final check, or it is $\mathcal{D}$ and the log reports that no minimized dataset passed; and (c) *Per-decision proportionality*: in a round whose estimates are ξ-accurate, the committed operation o^* satisfies $\rho(o^*) \geq \rho(o) - \xi/c(o^*) - \xi/c(o)$ for every candidate o within $4\times$ evaluated in that round. □

Here $\bar{G}(o)$ is ξ-*accurate* if $|\bar{G}(o) - G(o)| \leq \xi$, and the candidates evaluated in a round are those scored by the proxy, *i.e.,* the top-b of each operation family and of the refine-and-evict variants, plus the variants of the top-b refine-and-swaps (with $\bar{b}$ in place of b if widened). In (b), passing the final check means that the target model retrained on the output satisfies $\Delta U \leq \epsilon$ on the validation split. In (c), each committed operation has a positive estimated utility gain and the largest estimated score $\bar{G}(o)/c(o)$ among these candidates, first among those within $4\times$, *i.e.,* whose smallest retained class has at least $k_{t-1}/4$ records.

Proof. The proof proceeds in three steps, one per property.

(a) Termination. Each committed operation is a refine-and-swap, its refine-and-evict variant, or a retention-class inclusion, and each round commits one of them (Equation (5)). A refine-and-swap or its variant advances one QI by one level of its hierarchy, and no operation coarsens a QI. A QI q can thus advance at most h_q times, where h_q is the height of its hierarchy, so at most $H = \sum_q h_q$ refine-and-swap operations are committed. These commits split the execution into at most $H+1$ intervals in which only inclusions are committed. Each inclusion admits at least one not-yet-admitted record of $\mathcal{D}$ and evicts none, so within one interval the retained records grow strictly inside $\mathcal{D}$, and at most $N = |\mathcal{D}|$ inclusions are committed. The total number of commits, and hence of rounds that do not terminate, is thus at most $H + (H+1)N$. In each round, the proxy is retrained once on $\mathcal{D}_{t-1}$ and once for each of the at most $4b$ evaluated candidates ($4b + 4\bar{b}$ if widened), and the target at most once, in the final check.

(b) Feasibility. TRIM stops in one of two ways. First, it stops with $\mathcal{D}_T$ when the final check passes, *i.e.,* the target model retrained on $\mathcal{D}_T$ satisfies $\Delta U(\mathcal{D}_T) \leq \epsilon$ on the validation split. If the check fails, the search continues, so no dataset that fails the check is returned. Second, it stops when no candidate has a positive estimated utility gain before the check passes. It then returns $\mathcal{D}$, for which $\Delta U(\mathcal{D}) = 0 \leq \epsilon$ by Equation (3), and logs that no minimized dataset passed.

(c) Per-decision proportionality. Let K be the set of candidates scored in a round (the top-b of each operation family and of the variants, plus the variants of the top-b refine-and-swaps, with $\bar{b}$ if widened; Section VI), and $K_4 \subseteq K$ those within $4\times$. For $o \in K$, write $\hat{\rho}(o) = \frac{\bar{G}(o)}{c(o)}$, and $\rho(o) = \frac{G(o)}{c(o)}$. By Equation (5), with $G(o)$ replaced by $\bar{G}(o)$ (Section VI), the committed operation o^* has $\bar{G}(o^*) > 0$ and maximizes $\hat{\rho}$ on the candidates in K_4 with a positive estimated gain, or on those in K if K_4 has none. Every candidate $o \in K$ with $\bar{G}(o) \leq 0$ has $\hat{\rho}(o) \leq 0 < \hat{\rho}(o^*)$, so $\hat{\rho}(o^*) \geq \hat{\rho}(o)$ for all $o \in K_4$.

Suppose that the estimates of the round are ξ-accurate, *i.e.,* $|\bar{G}(o) - G(o)| \leq \xi$ for every $o \in K$. The privacy cost $c(o) = \max\{\Delta\mathsf{leak}(o), \kappa\}$ is computed exactly and is positive, so for every $o \in K$, $|\hat{\rho}(o) - \rho(o)| = \frac{|\bar{G}(o) - G(o)|}{c(o)} \leq \frac{\xi}{c(o)}$. Hence, for every $o \in K_4$, $\rho(o^*) \geq \hat{\rho}(o^*) - \frac{\xi}{c(o^*)} \geq \hat{\rho}(o) - \frac{\xi}{c(o^*)}$ and $\geq \rho(o) - \frac{\xi}{c(o)} - \frac{\xi}{c(o^*)}$. Since $c(o) \geq \kappa$, this implies the weaker bound $\rho(o^*) \geq \max_{o \in K_4} \rho(o) - 2\xi/\kappa$. When $c(o^*) = c(o) = \kappa$, multiplying by κ gives $G(o^*) \geq G(o) - 2\xi$. □

Appendix H
LGA as a Hessian-Free Influence Score

We now prove the property of LGA used in Section VI.

Theorem 5: *Let g and g_o be the gradients of L_{train} at θ on $\mathcal{D}_{t-1}$ and on $\mathcal{D}_{t-1} \oplus o$. Assume that L_{v} is L-smooth and that $\|g\|, \|g_o\| \leq G$ for every candidate o. For a step size $\eta > 0$, let the* one-step gain $\Delta_\eta(o) := L_{\mathsf{v}}(\theta - \eta g) - L_{\mathsf{v}}(\theta - \eta g_o)$ *be the reduction in validation loss when the step is taken on $\mathcal{D}_{t-1} \oplus o$ instead of $\mathcal{D}_{t-1}$. Then $|\Delta_\eta(o) - \eta\,\mathsf{LGA}_\theta(\mathcal{D}_{t-1}, o)| \leq LG^2\eta^2$. Hence any two candidates whose values of $\Delta_\eta(o)/c(o)$ differ by more than $2LG^2\eta^2/\kappa$ are ordered the same way by $\mathsf{LGA}_\theta(\mathcal{D}_{t-1}, o)/c(o)$.* □

Proof. The proof proceeds in three steps: (1) we fix the notation and state the two assumptions; (2) we show, by a second-order expansion, that the one-step gain $\Delta_\eta(o)$ equals $\eta\,\mathsf{LGA}_\theta(\mathcal{D}_{t-1}, o)$ up to a remainder of at most $LG^2\eta^2$; and (3) we divide by the privacy cost and compare two operations.

(1) Setup and assumptions. Fix the parameters θ of the ranking model and the current dataset $\mathcal{D}_{t-1}$. Write

$$v = \nabla_\theta L_{\mathsf{v}}(\theta),\ g = \nabla_\theta L_{\mathsf{train}}(\theta, \mathcal{D}_{t-1}),\ g_o = \nabla_\theta L_{\mathsf{train}}(\theta, \mathcal{D}_{t-1} \oplus o)$$

for the validation gradient and for the training gradients before and after applying a candidate o, respectively. The LGA score $\mathsf{LGA}_\theta(\mathcal{D}_{t-1}, o) = \langle v,\ g_o - g \rangle$ then aligns the validation

gradient with the marginal training gradient $g_o - g$ that o induces. We use the two assumptions in the following form.

(A1) *Smoothness*: L_{v} is L-smooth, so $|L_{\mathsf{v}}(\theta+\delta) - L_{\mathsf{v}}(\theta) - \langle v, \delta\rangle| \le \frac{L}{2}\|\delta\|^2$ for every δ.

(A2) *Bounded gradients*: $\|g\|, \|g_o\| \le G$ for each candidate o.

(2) LGA is a first-order surrogate of the one-step gain. The one-step gain of o is the reduction in validation loss from the step that includes o, relative to the baseline step on $\mathcal{D}_{t-1}$:

$$\Delta_\eta(o) = L_{\mathsf{v}}(\theta - \eta g) - L_{\mathsf{v}}(\theta - \eta g_o).$$

The two updates displace θ by $-\eta g$ and $-\eta g_o$. Applying (A1) to each displacement expands the two losses as

$$L_{\mathsf{v}}(\theta - \eta g) = L_{\mathsf{v}}(\theta) - \eta \langle v, g\rangle + r,$$
$$L_{\mathsf{v}}(\theta - \eta g_o) = L_{\mathsf{v}}(\theta) - \eta \langle v, g_o\rangle + r_o,$$

with second-order remainders $|r| \le \frac{L}{2}\eta^2\|g\|^2$ and $|r_o| \le \frac{L}{2}\eta^2\|g_o\|^2$. Subtracting the second line from the first, the base losses $L_{\mathsf{v}}(\theta)$ cancel, the linear terms combine into $\eta\langle v, g_o - g\rangle = \eta\,\mathsf{LGA}_\theta(\mathcal{D}_{t-1}, o)$, and the remainders form $\zeta(o) := r - r_o$: $\Delta_\eta(o) = \eta\,\mathsf{LGA}_\theta(\mathcal{D}_{t-1}, o) + \zeta(o)$. By (A2), the remainder is bounded uniformly over candidates: $|\zeta(o)| \le |r| + |r_o| \le \frac{L}{2}\eta^2\big(\|g\|^2 + \|g_o\|^2\big) \le LG^2\eta^2$. Thus, the one-step gain equals the LGA score scaled by the shared factor η, up to a remainder of at most $LG^2\eta^2$.

(3) Ranking preservation. Fix two operations o_1 and o_2, and write $c_i = c(o_i)$. The privacy cost is computed exactly and satisfies $c_i \ge \kappa > 0$ (Equation (4)). By Step (2),

$$\frac{\Delta_\eta(o_i)}{c_i} = \eta\,\frac{\mathsf{LGA}_\theta(\mathcal{D}_{t-1}, o_i)}{c_i} + \frac{\zeta(o_i)}{c_i},$$

where $|\zeta(o_i)/c_i| \le LG^2\eta^2/\kappa$. Subtracting the two relations, the gap between the one-step gains per unit cost equals η times the gap between the LGA scores per unit cost, plus the perturbation $\zeta(o_1)/c_1 - \zeta(o_2)/c_2$, whose magnitude is at most $2LG^2\eta^2/\kappa$. Thus, if $\left|\frac{\Delta_\eta(o_1)}{c_1} - \frac{\Delta_\eta(o_2)}{c_2}\right| > \frac{2LG^2\eta^2}{\kappa}$, this perturbation cannot reverse the sign of the gap. Since $\eta > 0$, $\Delta_\eta(o_1)/c_1 - \Delta_\eta(o_2)/c_2$ and $\mathsf{LGA}_\theta(\mathcal{D}_{t-1}, o_1)/c_1 - \mathsf{LGA}_\theta(\mathcal{D}_{t-1}, o_2)/c_2$ have the same sign, thus the theorem. □

Role of step size. The step size η is shared by all candidates, so it never alters the LGA order itself; it controls how faithfully that order reflects the one-step gains. The signal $\eta\,\mathsf{LGA}_\theta(\mathcal{D}_{t-1}, o)/c(o)$ is of order η, and the remainder $\zeta(o)/c(o)$ is of order η^2. Hence LGA correctly orders every pair whose scores per unit cost differ by more than $2LG^2\eta/\kappa$, and this narrows as η decreases. Exp-6 (Section VII) tests how well the LGA order agrees with the order by retrained utility.

Relation to retraining. Theorem 5 compares LGA with a single gradient step. As TRIM refits the ranking model every round, LGA can be compared with the effect of retraining it.

Theorem 6: *Let θ minimize $F(\theta) = L_{\mathsf{train}}(\theta, \mathcal{D}_{t-1}) + \Omega(\theta)$, where Ω is a data-independent regularizer, and let $F_o(\theta) = L_{\mathsf{train}}(\theta, \mathcal{D}_{t-1} \oplus o) + \Omega(\theta)$. Assume that the Hessian H of F_o at θ has eigenvalues in $[\mu, M]$ with $\mu > 0$, and let $\Delta^{(1)}(o) := \langle v, H^{-1}(g_o - g)\rangle$ be the first-order reduction in validation loss when the ranking model is retrained on $\mathcal{D}_{t-1} \oplus o$ [76]. Then, with $\alpha = (1/\mu + 1/M)/2$, $\big|\Delta^{(1)}(o) - \alpha\,\mathsf{LGA}_\theta(\mathcal{D}_{t-1}, o)\big| \le \frac{1}{2}\Big(\frac{1}{\mu} - \frac{1}{M}\Big)\|v\|\,\|g_o - g\|$.* □

Proof. Since θ minimizes F, $\nabla F(\theta) = g + \nabla\Omega(\theta) = 0$, so $\nabla F_o(\theta) = g_o + \nabla\Omega(\theta) = g_o - g$. One Newton step on F_o from θ moves the parameters to $\theta - H^{-1}(g_o - g)$, and a first-order expansion of L_{v} at θ gives the reduction $\Delta^{(1)}(o) = \langle v, H^{-1}(g_o - g)\rangle$. The eigenvalues of H^{-1} lie in $[1/M, 1/\mu]$, so those of $H^{-1} - \alpha I$ lie in $[-(1/\mu - 1/M)/2,\ (1/\mu - 1/M)/2]$. By the Cauchy-Schwarz inequality,

$$\begin{aligned}\big|\Delta^{(1)}(o) - \alpha\langle v, g_o - g\rangle\big| &= \big|\langle v, (H^{-1} - \alpha I)(g_o - g)\rangle\big| \\ &\le \frac{1}{2}\Big(\frac{1}{\mu} - \frac{1}{M}\Big)\|v\|\,\|g_o - g\|,\end{aligned}$$

and $\langle v, g_o - g\rangle = \mathsf{LGA}_\theta(\mathcal{D}_{t-1}, o)$. □

Ranking. Write $e(o)$ for the right-hand side of Theorem 6. Since $\alpha > 0$, if $\frac{\Delta^{(1)}(o_1)}{c(o_1)} - \frac{\Delta^{(1)}(o_2)}{c(o_2)} > \frac{e(o_1)}{c(o_1)} + \frac{e(o_2)}{c(o_2)}$, then o_1 also precedes o_2 by $\mathsf{LGA}_\theta(\mathcal{D}_{t-1}, o)/c(o)$. Relative to the scale $\alpha\|v\|\|g_o - g\|$ of the score, the error $e(o)$ is a fraction $(M - \mu)/(M + \mu)$, which depends only on the condition number of H. LGA is thus a Hessian-free influence score: it avoids inverting H, at the cost of a margin that is small when H is well conditioned and loose otherwise. Unlike influence estimates reused across successive operations, it is recomputed at the refit parameters in every round.

Appendix I
Accuracy of Proxy Evaluation

We bound the error of the proxy estimate $\bar{G}(o)$ (Section VI).

Theorem 7: *Let o be a candidate operation independent of the validation split, let $|o|$ be the number of single-record changes that turn $\mathcal{D}_{t-1}$ into $\mathcal{D}_{t-1} \oplus o$, and let $n = |\mathcal{D}_{t-1} \oplus o|$. Assume that the loss takes values in $[0, \ell_{\max}]$, and that $\mathcal{M}$ and $\bar{\mathcal{M}}$ satisfy the* local-response condition *with rate β: each of these changes shifts the population risks of the retrained $\mathcal{M}$ and $\bar{\mathcal{M}}$ by amounts that differ by at most β/n. Then, with probability at least $1 - \delta$ over a validation split of m records,*

$$\big|\bar{G}(o) - G(o)\big| \;\le\; \beta\,\frac{|o|}{n} + 2\ell_{\max}\sqrt{\frac{\log(4/\delta)}{2m}}. \qquad (8)$$

□

Proof. The proof proceeds in four steps: (1) we decompose the proxy-target recovery error into validation noise and the operation-induced change in proxy bias; (2) we bound the validation noise by concentration; (3) we bound the bias change by tracing o through a neighboring-record path; and (4) we combine the two bounds into Equation (8). We then extend the bound to several candidates.

(1) Setup and error decomposition. Fix a candidate operation o independently of the validation split, write $\mathcal{D}_t = \mathcal{D}_{t-1} \oplus o$, and let $n = |\mathcal{D}_t|$. Assume that $0 \le \ell \le \ell_{\max}$ and that the validation split consists of m records drawn i.i.d. from $\mathcal{P}$, independently of the training data.

For a dataset $\mathcal{D}'$, let $\theta_{\mathcal{D}'}$ and $\bar{\theta}_{\mathcal{D}'}$ be the parameters obtained by training the target and the proxy on it, and let $R(\theta) := \mathbb{E}_{(x,y)}[\ell(x,y;\theta)]$ denote the population risk. For $j \in \{t-1,t\}$, let V_j be the validation sample for the proxy at state $\mathcal{D}_j$, with empirical validation risk $\widehat{R}_j(\bar{\theta}) := \frac{1}{m}\sum_{(x,y)\in V_j} \ell(x,y;\bar{\theta})$. The samples V_{t-1} and V_t may coincide, as they do in TRIM, where both are the validation split, but each is independent of training. Since utility is negative risk,

$$G(o) = R(\theta_{\mathcal{D}_{t-1}}) - R(\theta_{\mathcal{D}_t}),$$
$$\bar{G}(o) = \widehat{R}_{t-1}(\bar{\theta}_{\mathcal{D}_{t-1}}) - \widehat{R}_t(\bar{\theta}_{\mathcal{D}_t}).$$

Let $\Phi(\mathcal{D}') := R(\theta_{\mathcal{D}'}) - R(\bar{\theta}_{\mathcal{D}'})$ be the gap between the population risks of the target and the proxy trained on $\mathcal{D}'$. Then

$$\begin{aligned}\bar{G}(o) - G(o) = &\left[\widehat{R}_{t-1}(\bar{\theta}_{\mathcal{D}_{t-1}}) - R(\bar{\theta}_{\mathcal{D}_{t-1}})\right] \\ &+ \left[R(\bar{\theta}_{\mathcal{D}_t}) - \widehat{R}_t(\bar{\theta}_{\mathcal{D}_t})\right] \\ &+ \left[\Phi(\mathcal{D}_t) - \Phi(\mathcal{D}_{t-1})\right]. \end{aligned} \tag{9}$$

Equation (9) follows by adding and subtracting the population risks $R(\bar{\theta}_{\mathcal{D}_{t-1}})$ and $R(\bar{\theta}_{\mathcal{D}_t})$ of the proxy, and by grouping the risks of the target and the proxy trained on the same dataset into Φ. The first two terms are the validation errors before and after o, and the last is the change in proxy bias induced by o. Hence the target and the proxy may differ arbitrarily in parameters or absolute risk: any persistent gap cancels when their recoveries are compared.

(2) Validation concentration. The proxy recovery uses empirical validation risks, so the first two terms of Equation (9) are ordinary sampling noise. Conditioned on the trained proxy parameters, the losses in each validation sample are independent and lie in $[0,\ell_{\max}]$. By Hoeffding's inequality and a union bound over the two evaluations, with probability at least $1-\delta$,

$$\left|\widehat{R}_j(\bar{\theta}_{\mathcal{D}_j}) - R(\bar{\theta}_{\mathcal{D}_j})\right| \le \ell_{\max}\sqrt{\frac{\log(4/\delta)}{2m}},\ j \in \{t-1,t\}. \tag{10}$$

(3) Local response along an operation path. It remains to bound the change in the target-proxy risk gap induced by o. Write o as a path of $s = |o|$ neighboring single-record changes, $\mathcal{D}^{(0)} = \mathcal{D}_{t-1}, \ldots, \mathcal{D}^{(s)} = \mathcal{D}_t$, each of which adds, removes, or rewrites one record. This path is only an accounting device: the telescoping sum below is exact despite interactions among record changes. Neighboring-dataset paths are standard in stability analysis [84] and influence-function analysis [76].

For neighboring datasets $\mathcal{D}' \sim \mathcal{D}''$, define

$$\Delta_{\mathcal{M}}(\mathcal{D}'',\mathcal{D}') := R(\theta_{\mathcal{D}''}) - R(\theta_{\mathcal{D}'}),$$
$$\Delta_{\bar{\mathcal{M}}}(\mathcal{D}'',\mathcal{D}') := R(\bar{\theta}_{\mathcal{D}''}) - R(\bar{\theta}_{\mathcal{D}'}).$$

The local-response condition with rate β states that the target and the proxy respond similarly to each single-record change on the path: for $1 \le i \le s$,

$$\left|\Delta_{\mathcal{M}}(\mathcal{D}^{(i)},\mathcal{D}^{(i-1)}) - \Delta_{\bar{\mathcal{M}}}(\mathcal{D}^{(i)},\mathcal{D}^{(i-1)})\right| \le \frac{\beta}{n}. \tag{11}$$

The factor $1/n$ reflects that one record carries a weight of order $1/n$ in a dataset of size of order n. Equation (11) requires only local, not global, agreement: the two models may differ arbitrarily in parameters and absolute risk, as long as their risks move alike under the same data change.

At each step, the change in Φ is the difference between the target and proxy responses. Summing these changes along the path telescopes from $\mathcal{D}_{t-1}$ to $\mathcal{D}_t$, and Equation (11) gives

$$\begin{aligned}&|\Phi(\mathcal{D}_t) - \Phi(\mathcal{D}_{t-1})| \\ &\quad\le \sum_{i=1}^{s}\left|\Delta_{\mathcal{M}}(\mathcal{D}^{(i)},\mathcal{D}^{(i-1)}) - \Delta_{\bar{\mathcal{M}}}(\mathcal{D}^{(i)},\mathcal{D}^{(i-1)})\right| \\ &\quad\le \frac{s\beta}{n} = \beta\,\frac{|o|}{n}. \end{aligned} \tag{12}$$

This bound separates two effects: $|o|/n$ is the fraction of affected records, while β captures the response mismatch between the proxy and the target. The bound is small for an operation that touches few records, and of order β when $s = \Theta(n)$: a refinement rewrites every retained record whose value is refined, and an exchange or a class-level inclusion may move a constant fraction of the records.

When $\bar{\mathcal{M}}$ is a network of width w in the model family of $\mathcal{M}$ and d is the input dimension, one may posit $\beta = c_0 C_L\, d/w$, where C_L absorbs fixed architecture, norm, and optimization factors, and c_0 is independent of n, $|o|$, d and w. Wide-network analyses [90], [91] provide context for such capacity-dependent approximation, but the rate is an assumption, not a consequence of those results. Section VII measures the estimation error, and the selections it leads to, empirically.

A sufficient condition. The local-response condition holds when both learners are uniformly stable [84]. Call a learner β_A-*stable* if for neighboring datasets $\mathcal{D}' \sim \mathcal{D}''$ with about n records, the losses of the two trained models differ by at most β_A/n at every example. Since population risks are averages of losses, $|\Delta_{\mathcal{M}}(\mathcal{D}'',\mathcal{D}')| \le \beta_{\mathcal{M}}/n$ and $|\Delta_{\bar{\mathcal{M}}}(\mathcal{D}'',\mathcal{D}')| \le \beta_{\bar{\mathcal{M}}}/n$, so Equation (11) holds with $\beta = \beta_{\mathcal{M}} + \beta_{\bar{\mathcal{M}}}$. Regularized convex empirical risk minimization with a Lipschitz loss is uniformly stable with such a rate [84], which covers a logistic-regression proxy; for the target model, stability is an assumption. The resulting error $\beta|o|/n$ is small for an inclusion that admits few records and of order β for a refine-and-swap.

(4) Combining the bounds. By Equation (9), no other error source remains. Applying the triangle inequality to Equation (9) and bounding its terms by Equations (10) and (12) yields Equation (8) with probability at least $1-\delta$. Its first term is the response mismatch accumulated along the operation path, and its second is finite-sample validation noise, which depends on the validation split alone.

Several candidates. Consider Q candidate operations, each fixed independently of the validation split, as when each round evaluates its candidates on a fresh validation sample. Their proxy estimates involve at most $2Q$ empirical risks. Applying Hoeffding's inequality to each of these risks, conditionally on the trained proxy parameters, together with a union bound shows that, with probability at least $1-\delta$, Equation (8) holds for all Q candidates simultaneously when $\log(4/\delta)$ is replaced by $\log(4Q/\delta)$. TRIM evaluates at most $4b+4\bar{b}$ candidates per round, and by the proof of Proposition 3(a) it runs at most

Table VI. QIs of the datasets.

Dataset	QIs
Income	AGEP, COW, SCHL, MAR, OCCP, POBP, RELP, WKHP, SEX, RAC1P
PubCov	AGEP, SCHL, MAR, SEX, DIS, ESP, CIT, MIG, MIL, ANC, NATIVITY, DEAR, DEYE, DREM, PINCP, ESR, ST, FER, RAC1P
Diabetes	age, race, gender, admission_type_id, admission_source_id, discharge_disposition_id, time_in_hospital, number_outpatient, number_emergency, number_inpatient, num_lab_procedures, num_procedures, num_medications, payer_code, medical_specialty, diag_1, diag_2, diag_3
BM	age, job, marital, education, housing, loan, contact_half_month
BNG	checking_status, duration, credit_history, purpose, credit_amount, savings_status, employment, installment_commitment, personal_status, other_parties, residence_since, property_magnitude, age, other_payment_plans, housing, existing_credits, job, num_dependents, own_telephone, foreign_worker

$(H+1)(N+1)$ rounds: at most $H+(H+1)N$ rounds that commit, and one that stops. Hence $Q \leq (4b+4\bar{b})(H+1)(N+1)$, and the logarithm is at most $\log(16(b+\bar{b})(H+1)(N+1)/\delta)$. The right-hand side, maximized over these candidates, gives the error ξ of Proposition 3(c). In TRIM, however, the LGA ranking and the earlier commits use the validation split, so the evaluated candidates depend on it; TRIM therefore relies on the final check for feasibility (Proposition 3). □

APPENDIX J
QUASI-IDENTIFIERS OF THE DATASETS

Table VI lists the QIs of the datasets of Section VII, *i.e.,* the attributes that an adversary can obtain from external sources (Section II). Every feature of Income, PubCov, and BNG is a QI. The non-QIs of Diabetes are the patient's weight, the number of diagnoses, the two test results, the 23 medication indicators, and the two medication flags. The non-QIs of BM are the default status, the account balance, and the five attributes of the telemarketing contacts. We drop the call duration, which is known only after the call and strongly affects the label [92]. The month and the day of the last contact are two parts of one date, so we join them into one QI, contact_half_month, instead of releasing the exact day as a non-QI; date elements other than the year are identifiers under the HIPAA Safe Harbor rule [93]. Its tree follows the calendar (quarter, month, half-month). Half-months (days 1–15 and 16 to the end) are the finest level, since exact dates are sparse: 33 of the 318 observed month-day pairs have fewer than 5 records.

APPENDIX K
CONSTRUCTION OF THE GENERALIZATION HIERARCHIES

For each dataset, we exported the observed domain of every QI and gave it to an LLM (GPT-5.6) with Web search, using four instructions. (1) For every coded attribute, find the official data dictionary of the dataset and group the values by their documented meaning, not by numeric proximity; a code whose meaning is not confirmable stays a separate leaf. (2) Build one tree per QI in which every observed value appears in exactly one leaf, sibling nodes cover disjoint domains, the intervals of a numeric QI do not overlap, and the root covers the full observed domain. (3) Add intermediate levels, so that TRIM can select among granularities. (4) Report, for each attribute, whether coverage, parent links and sibling disjointness hold. Each tree records that these three checks pass. The full prompt is given with our code. The LLM calls cost <$0.1 in total.