LAUNDERBENCH: IS YOUR DATA LAUNDERED AND THEN USED IN LLM TRAINING?

Anonymous authors
Paper under double-blind review

ABSTRACT

Data laundering refers to an emerging threat that rewrites documents before training a language model. This threat will hide evidence of copyrighted data use from training-data provenance detection while preserving knowledge of such source data. However, the community has yet to establish an evaluation protocol that distinguishes whether the detector is simply underpowered, the laundering succeeded, or the model failed to learn. To address this confounding, we introduce **`LaunderBench`**, a benchmark grounded in three diagnostic criteria, namely *baseline detector auditability*, *audit evasion efficacy*, *source utility viability*. Specifically, **`LaunderBench`** trains three compute-matched models from a shared checkpoint: laundered (on rewrites), unlaundered (on originals), and filler-only (on background text alone), and audits all three on the original documents. This setup allows us to separate audit evasion from detector weakness, and distinguish whether the model actually learned from the sources or merely drifted during background training. Evaluating across model scales (Pythia-1.4B to 6.9B), architectures (Pythia and GPT-Neo), 4 data corpora, and 9 rewriting methods reveals three findings: ① rewrite-trained models consistently learn source knowledge, but the utility gap lost relative to unlaundered training widens as training proceeds; ② the acquired source knowledge does not translate into higher detectability; ③ under strict false-positive constraints, the relative evasion rankings of rewriting methods can invert. These findings show that reliable provenance audits must evaluate detector baseline power alongside strict false-alarm budgets, which **`LaunderBench`** operationalizes to enable rigorous benchmarking of laundering-resilient data provenance audits.

1 INTRODUCTION

Large language models (LLMs) are trained on diverse web-scale corpora, yet model providers often disclose only limited information about the data used to train them (Wan et al., 2025; OpenAI, 2023; Longpre et al., 2023). To audit whether their copyrighted or proprietary data was used in training, data owners can apply a provenance detector to a model's predictions on such data (i.e., *originals*) and look for evidence such as lower prediction loss than on held-out non-member data (Henderson et al., 2023; Carlini et al., 2022). This reliance on originals verbatim gives model providers who want to conceal unauthorized data use an incentive to train on *rewrites* of originals (i.e., non-verbatim sources) while keeping the source knowledge they want the model to learn. Recent studies have shown that this practice, referred to as *data laundering*, can weaken originals-based audit evidence (i.e., originals verbatim become harder to distinguish from non-member data) while preserving some source-derived capabilities in the models trained on the rewrites. As LLMs make such rewriting cheap at scale, data laundering forms a practical threat to obfuscate data provenance (Li et al., 2026).

However, a data owner whose audit finds only weak evidence against a possibly laundered model cannot *fairly* evaluate whether laundering occurred or how well it worked. Laundering succeeds only if rewriting weakens the audit evidence and the model still learns from the originals. Weak evidence *alone* does not establish the first, because the same detector may be already weak on a model trained on verbatim source. Nor does good prediction of the originals *alone* establish the second, since an LLM never trained on them may predict them well too. Figure 1 demonstrates both problems.

Starting from a shared Pythia-1.4B pre-trained checkpoint (Biderman et al., 2023), we compare models post-trained on two rewriting methods, lexical paraphrase and structure-guided paraphrase (Ertan

Reviewers: please read the Reviewer Guidelines (iclr.cc/Conferences/2027/ReviewerGuidelines) and the AI Policy for Reviewers (iclr.cc/Conferences/2027/AIPolicyForReviewers). If you used AI to expand, edit, or polish your review, please provide the input text to the LLM. Better still, consider skipping the LLM and submitting your original text: we, and the authors, are much more interested in your unedited thoughts than in what an LLM has to say. AI-assisted or not, you are putting your name and reputation behind your review: LLM-generated falsehoods, hallucinations or misrepresentations are subject to disciplinary action, which may include desk-rejecting all papers you have authored.

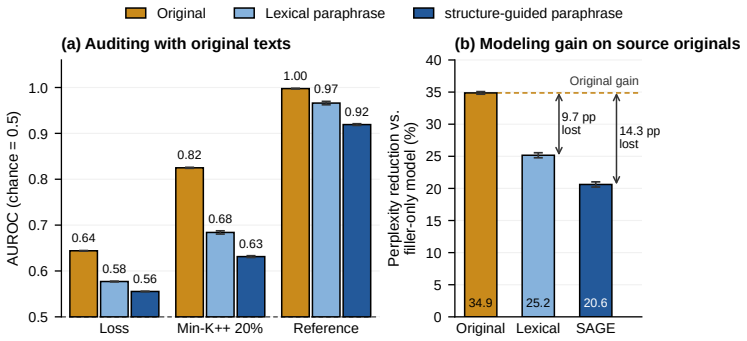

Figure 1: We post-train a shared Pythia-1.4B checkpoint on Wikipedia data, with source documents supplied either as originals or as rewrites generated by two rewriting methods, and a matched filler-only comparator exclusively on disjoint background text, respectively. **(a)** AUROC for distinguishing members from non-members under unlaundered training and two rewrite methods (i.e., source data-use detectability). **(b)** The reduction in perplexity on the source originals relative to the filler-only model (i.e., source data modeling utility).

et al., 2026), against matched comparators trained under an identical schedule: an *unlaundered* model trained on verbatim sources from the Wikipedia subset of MIMIR (Duan et al., 2024), and a *filler-only* model trained exclusively on disjoint background text. As Figure 1(a) shows, while two detectors, Loss (Yeom et al., 2018) and Min-K++ (Zhang et al., 2025a), both report weak audit evidence after rewriting, their "unlaundered" baselines lead to different conclusions. For the Loss detector, the unlaundered baseline itself achieves only marginal separation between members and non-members, so much of its weakness was present even before any rewriting. In contrast, Min-K++ identifies verbatim members effectively; its post-rewriting collapse reflects a genuine laundering-induced drop in detectability. When an auditor has additional access to the filler-only checkpoint, the Reference detector (Watson et al., 2022) separates unlaundered members almost perfectly and remains effective after both rewriting methods. On the other hand, Figure 1(b) assesses utility by evaluating models' perplexity on the source originals relative to the filler-only comparator. While rewrite-trained models improve upon the filler-only baseline, they incur disparate utility shortfalls compared to unlaundered training. Therefore, a laundering outcome should not be assessed by a single metric, but comprises three coupled quantities: the change in detection relative to an unlaundered baseline, the source modeling gain retained over an unexposed baseline, and the gain lost relative to unlaundered training.

Gap. Recent studies have begun evaluating detectability alongside source utility. For example, Ertan et al. (2026) evaluate detectability degradation against question-answering retention under structure-guided paraphrasing; Zhang et al. (2025b) examine the trade-off under selective obfuscation during fine-tuning. Yet these evaluations lack a unified, strictly paired experimental protocol. Particularly, changing the training condition or the auditor's resource tier in existing setups often silently alters the underlying document distributions, evaluation queries, or comparator models, hindering an apples-to-apples comparison of what laundering conceals versus what the model actually learns (broader related works are discussed in Appendix G).

Solution. To resolve this confounding, we introduce **`LaunderBench`**, a controlled benchmark that evaluates post-laundering detection and modeling utility on *identical* documents and checkpoints. Specifically, **`LaunderBench`** anchors every training derivative directly to its source document, trains laundered, unlaundered, and filler-only models from a shared initialization under the *same* optimization schedule, and evaluates all models on *fixed* sets of verbatim source queries. By systematically reporting detectability across explicit auditor access tiers alongside the decomposed modeling gain (retained vs. lost), **`LaunderBench`** isolates true laundering efficacy from shifts in data, training compute, or auditor capabilities, providing a testbed for *any* data provenance detector.

Construction. We instantiate **`LaunderBench`** as a structured evaluation suite across four MIMIR corpora (Duan et al., 2024), including arXiv, Wikipedia, Pile-CC, and PubMed, and 10 training conditions (unlaundered training alongside 9 rewriting procedures) across 2 random seeds (§3). To thoroughly investigate target LLM and detector behaviors under laundering, the benchmark varies three axes: ① *training exposure and quotas*, tracking 80 source-trained trajectories on Pythia-1.4B across 240 checkpoint snapshots (313, 626, and 1,252 optimization updates) alongside 6 seed- and update-matched filler-only checkpoints, with source-token quota extending from 5% to 50% in the training materials (i.e., portion of source data mixed within the background texts); ② *model capacity*,

scaling across Pythia-1.4B, 2.8B, and 6.9B; and ③ *architecture diversity*, incorporating Pythia and GPT-Neo (Black et al., 2021) backbones.

Findings. ① Modeling gain and detectability decouple along training trajectories. Rewrite-trained models continuously accumulate source modeling capabilities, yet original-text detectability does not increase in step, widening the gap between retained utility and auditability (§4.1). ② Unlaundered audit baselines scale heavily with training exposure. Target-only detectors (e.g., Loss) applied to originals shift from near-chance at low quotas to near-perfect at high quotas, meaning an identical post-rewriting score can indicate an underpowered audit in one regime, but severe laundering evasion in another (§4.2). ③ Laundering efficacy hinges on the operating threshold. Under low-false-positive constraints, the relative rankings of rewriting methods can invert, and high aggregate AUROCs often conceal large proportions of missed source documents (§4.3).

In brief, we establish **`LaunderBench`** as a principled testbed alongside a multi-axis characterization of LLM behaviors under data laundering. To facilitate future research on laundering-resilient data provenance audits, we release our code at `https://shorturl.at/Faa3J`.

2 How Should We Evaluate Data Laundering?

As established in §1, evaluating data laundering requires separating genuine evasion from detector weakness, and retained source utility from background training drift. We formulate this evaluation around two controlled comparisons across models trained under the same update schedule, namely detection relative to unlaundered training, and utility decomposed against an unexposed filler-only baseline. Below, §2.1 specifies the training controls under data laundering, §2.2 formalizes originals-based provenance auditing, and §2.3 introduces the evaluation metrics. Figure 2 outlines the workflow.

2.1 Data Laundering and Training Controls

From source document to training data. Let $x \in \mathcal{X}$ denote a candidate text document. Under data laundering, a model provider applies a transformation $T : \mathcal{X} \to \mathcal{X}$ to x, seeking to learn its underlying content while evading detection by an auditor querying the verbatim original (Li et al., 2026; Ertan et al., 2026). Unlaundered training is equivalent to the identity transformation, $T_{\text{id}}(x) = x$. For a fixed configuration, corpus, and training seed, let $\mathcal{S}_T^{\text{train}} \subseteq \mathcal{X}$ denote the original documents whose derivatives $T(x)$ enter continued training (i.e., *source injection*).

Paired training conditions. An interpretable evaluation requires isolating two distinct counterfactual effects: the penalty (on detection) induced by rewriting relative to unlaundered training, and the utility retained beyond unexposed background training. To resolve both without optimization confounds, **`LaunderBench`** pairs each target evaluation with two compute-matched controls branched from a shared pre-trained checkpoint θ under an identical training protocol, including optimizer state, update budget, batch size, sequence length, and seed: the *laundered* model θ_T^+, trained on derivatives $T(x)$ for $x \in \mathcal{S}_T^{\text{train}}$ mixed with disjoint background text (*i.e., filler data*); the *unlaundered* model $\theta_{T_{\text{id}}}^+$, trained on verbatim originals $x \in \mathcal{S}_{T_{\text{id}}}^{\text{train}}$ mixed with filler data; and the *filler-only* model θ^-, trained exclusively on a different set of filler data $\mathcal{F}$. Throughout all three training conditions, the total number of gradient updates is fixed (details are in Appendix B).

2.2 Data Provenance Audit

Audit queries and labels. The auditor queries a target model θ with the original verbatim x to test whether x contributed to training, even when training occurred on $T(x)$. The positive evaluation cohort $\mathcal{S} \subseteq \mathcal{S}_T^{\text{train}}$ comprises original documents whose training derivatives were injected, while the negative cohort $\mathcal{H} \subseteq \mathcal{X}$ comprises held-out documents (i.e., $\mathcal{H} \cap \mathcal{S}_T^{\text{train}} = \varnothing$). The ground-truth exposure label $y : \mathcal{S} \cup \mathcal{H} \to \{0, 1\}$ assigns $y(x) = 1$ for $x \in \mathcal{S}$ and $y(x) = 0$ for $x \in \mathcal{H}$.

Matched evaluation queries. To prevent evaluation distribution shifts from confounding cross-method comparisons, evaluations across transformations should restrict source queries to their shared intersection, $\mathcal{S} \subseteq \bigcap_T \mathcal{S}_T^{\text{train}}$, evaluated against an invariant held-out cohort $\mathcal{H}$.

Auditor information conditions. An auditor's capability depends on accessible target outputs, reference corpora, or auxiliary models, which we formalize as the *information condition* $\mathcal{I}$. A

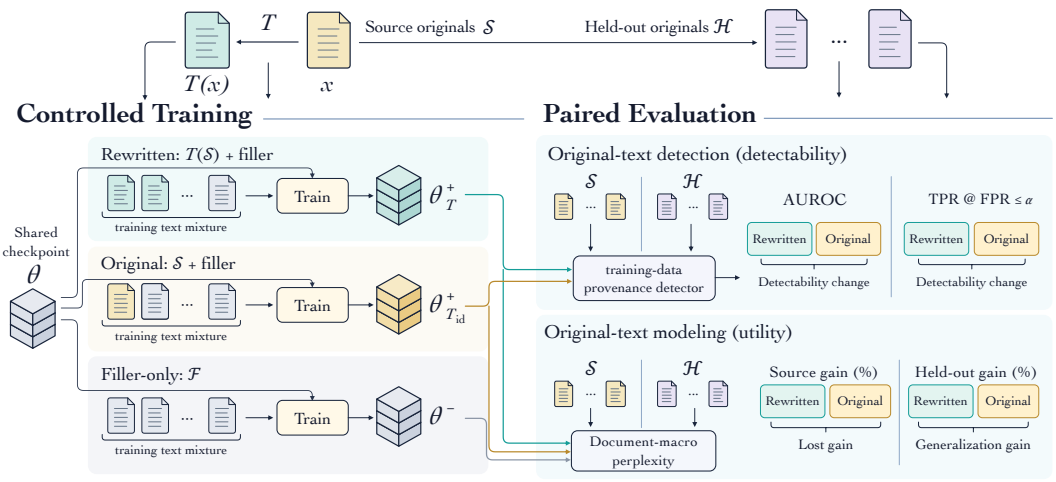

Figure 2: **Overview of `LaunderBench`.** Models branch from a shared pre-trained checkpoint and continue training under an identical update schedule on source rewrites mixed with filler data (θ^+_T), verbatim originals mixed with filler data ($\theta^+_{T_{\text{id}}}$), or filler data alone (θ^-). All audits query *verbatim* originals. The evaluation of **`LaunderBench`** considers both **(A) Detectability comparison:** for a fixed detector under an explicit auditor information condition, AUROC measures separation between source originals ($\mathcal{S}$) and held-out originals ($\mathcal{H}$) for θ^+_T relative to $\theta^+_{T_{\text{id}}}$, and **(B) Utility comparison:** relative perplexity reduction against the seed- and update-matched filler-only baseline θ^- decomposes source modeling utility into retained gain ($\Delta\text{PPL}_{\mathcal{S}}(T)$), originals gain ($\Delta\text{PPL}_{\mathcal{S}}(T_{\text{id}})$), and lost gain; held-out gains ($\Delta\text{PPL}_{\mathcal{H}}(T)$) evaluate out-of-cohort generalization.

detector d computes a score $s_d(x;\theta^+_T,\mathcal{I})$, where larger values indicate stronger evidence of source injection. Appendix C.1 summarizes the existing data provenance detectors.

2.3 Evaluation Metrics and Diagnostic Protocol

Source detectability metrics. An audit must evaluate both global discriminability and performance under strict false-alarm constraints. For a detector d under information condition $\mathcal{I}$, we evaluate threshold-free separation between source originals $\mathcal{S}$ and held-out originals $\mathcal{H}$ via

$$D(T,d,\mathcal{I}) \triangleq \text{AUROC}\big(\{\big(y(x), s_d(x;\theta^+_T,\mathcal{I})\big) : x \in \mathcal{S}\cup\mathcal{H}\}\big) \in [0,1].$$

Since practical provenance claims require bounded false positives, we also evaluate discrimination at an explicit evidentiary operating threshold. For a maximum false-positive budget $\alpha \in (0,1)$, we determine the empirical decision threshold on held-out data, $\tau_\alpha = \inf\{\tau \in \mathbb{R} : \frac{1}{|\mathcal{H}|}\sum_{x\in\mathcal{H}} \mathbf{1}[s_d(x;\theta^+_T,\mathcal{I}) \geq \tau] \leq \alpha\}$, and report the corresponding true-positive rate:

$$\text{TPR@}\alpha\text{FPR}(T,d,\mathcal{I}) \triangleq \frac{1}{|\mathcal{S}|}\sum_{x\in\mathcal{S}} \mathbf{1}\big[s_d(x;\theta^+_T,\mathcal{I}) \geq \tau_\alpha\big].$$

Source utility metrics. We quantify source modeling gain by evaluating how well a model predicts verbatim source text. Let $\text{PPL}(x;\theta)$ denote document perplexity over valid target tokens, and let $\text{PPL}_{\mathcal{S}}(\theta) = \frac{1}{|\mathcal{S}|}\sum_{x\in\mathcal{S}}\text{PPL}(x;\theta)$ be mean perplexity over the cohort. To isolate the modeling utility gained specifically from source injection beyond an unexposed model, we measure relative perplexity reduction against the seed- and update-matched filler-only baseline θ^-: $\Delta\text{PPL}_{\mathcal{S}}(T) = (\text{PPL}_{\mathcal{S}}(\theta^-) - \text{PPL}_{\mathcal{S}}(\theta^+_T))/\text{PPL}_{\mathcal{S}}(\theta^-)$. We define $\Delta\text{PPL}_{\mathcal{S}}(T)$ as the *retained gain* under rewriting, and $\Delta\text{PPL}_{\mathcal{S}}(T_{\text{id}})$ as the *originals gain*. Their difference defines the *lost gain* forfeited by laundering:

$$\Delta\text{PPL}_{\mathcal{S}}(T_{\text{id}}) - \Delta\text{PPL}_{\mathcal{S}}(T) \triangleq \big(\text{PPL}_{\mathcal{S}}(\theta^+_T) - \text{PPL}_{\mathcal{S}}((\theta^+_{T_{\text{id}}})\big)/\text{PPL}_{\mathcal{S}}(\theta^-).$$

Applying the same formulation to $\mathcal{H}$ yields the held-out gain $\Delta\text{PPL}_{\mathcal{H}}(T)$, which assesses whether acquired modeling benefits extend beyond the injected data to unseen sources from the same distribution. Normalizing by θ^- is critical across both cohorts because filler-only training induces domain

Condition	Text Sample (Wikipedia: Caerhendy transmitting station)
Original	The Caerhendy television relay station was **sited at Caerhendy in the Afan Valley** … **It was originally built** in the 1980s as a fill-in relay for UHF analogue television.
Lexical	The Caerhendy television relay station was **located in Caerhendy within the Afan Valley** … **Initially constructed** in the 1980s, it functioned as a fill-in relay for UHF analogue television.
Syntactic	The Caerhendy television relay station was **located in Caerhendy, within the Afan Valley** … **Initially constructed** in the 1980s, **the station served** as a fill-in relay for UHF analogue television
SAGE	**Located in the Afan Valley** , the Caerhendy television relay station **was constructed** in the 1980s to function as an auxiliary relay for UHF analogue television.
SAGE-R	Located in the `<<FACT_1>>` , the `<<FACT_2>>` television relay station was constructed in the `<<FACT_3>>` to function as an auxiliary relay for UHF analogue television.

Figure 3: Example training derivative versions of source x from a Wikipedia article.

drift. Because θ^- perplexity rises across checkpoints, relative gains isolate retained utility even when absolute target losses drift.

How to interpret laundering? An empirical measurement constitutes valid evidence of laundering if and only if it decouples audit degradation from both detector insufficiency and model collapse. We formalize this interpretation through three *necessary* criteria:

① *Baseline auditability:* The detector must exhibit non-trivial discriminative power on the unlaundered control ($D(T_{\mathrm{id}}, d, \mathcal{I}) \gg 0.5$). If unlaundered separation collapses to chance, post-rewriting detectability is statistically indistinguishable from detector failure.

② *Evasion efficacy:* Laundering requires a counterfactual detectability penalty relative to the unlaundered control, $D(T, d, \mathcal{I}) < D(T_{\mathrm{id}}, d, \mathcal{I})$. Low absolute number alone cannot establish evasion.

③ *Utility viability:* Provenance obfuscation constitutes laundering only if the model extracts positive source utility beyond filler-only exposure ($\Delta\mathrm{PPL}_{\mathcal{S}}(T) > 0$). A collapse in detectability accompanied by non-positive gain ($\Delta\mathrm{PPL}_{\mathcal{S}}(T) \leq 0$) does not reflect provenance-audit evasion.

3 LaunderBench: Benchmark Construction

We next instantiate the evaluation framework established in §2 as **`LaunderBench`**, which standardizes source corpora, rewriting methods, training controls, and auditor information condition tiers into a reproducible evaluation suite.

3.1 Corpora, Laundering Suite, and Matched Queries

Source and held-out corpora. We curate **`LaunderBench`** across four domains from the MIMIR benchmark (Duan et al., 2024): arXiv, Wikipedia, Pile-CC, and PubMed. From each corpus, we draw 500 candidate source originals ($\mathcal{S}$) and 60 held-out originals ($\mathcal{H}$) from the non-member splits of the Pile test set (Duan et al., 2024; Biderman et al., 2023), ensuring *neither cohort was exposed* during pretraining of the target LLM. After quality filtering across transformations, the shared positive evaluation cohorts contain 471 arXiv, 460 Wikipedia, 464 Pile-CC, and 474 PubMed originals, paired with the 60 held-out documents per corpus. To enforce the matched-query control of §2.2, all evaluation metrics are evaluated on these invariant document intersections. Details are in Appendix A.1.

Laundering procedures. We implement 9 rewriting procedures covering 3 rewriting mechanisms[1], evaluated alongside unlaundered training (T_{id}). ① *Surface paraphrasing* as static heuristic rewrite. Lexical rewrites (i.e., local vocabulary substitutions) and syntactic rewrites (i.e., clause restructuring) generated via fixed prompts. ② *Surrogate-optimized rewriting* to simulate adaptive laundering by an adversary who pre-screens text for evasion. We generate 5 candidate lexical and syntactic rewrites per document and select the one that minimizes an ensemble proxy detection score (Loss and Min-K %) on a Pythia-160M surrogate. ③ *Structure-guided laundering*. We leverage SAGE (Ertan et al., 2026),

[1]All rewritings were executed with both Qwen3-32B and Llama-3.3-70B-Instruct, unless otherwise specified.

which applies narrative summarization, and SAGE-R, which further flattens discourse and masks key entity markers during training. See Figure 3 for example rewrites and Appendix A.1 for more details.

3.2 Training Controls and Multi-Axis Coverage

Controlled training protocol. In the primary study, the evaluated models are post-trained from a shared pre-trained Pythia-1.4B checkpoint (Biderman et al., 2023). In laundered (θ^{+}_{T}) and unlaundered ($\theta^{+}_{T_{\text{id}}}$) models, sources are mixed into disjoint background text sampled from 1,000 documents of the Pile-uncopyrighted split; the filler-only baseline (θ^{-}) trains exclusively on this background pool under the identical optimizer schedule and seed (Appendix B). Each comparison evaluates the laundering procedure end-to-end under strictly matched update budgets, capturing both text transformation and token-level loss masking (details are in Appendices B.2 and B.3).

Primary evaluation axes. To capture dynamics across exposure regimes, the evaluation of Pythia-1.4B study spans two complementary axes: ① *Training-duration trajectories:* With source-token quota fixed at 5%, models are checkpointed at 313, 626, and 1,252 optimizer updates across 10 training conditions, 4 corpora, and 2 random seeds. This yields 80 longitudinal trajectories (240 snapshots) paired with 6 matched filler-only baselines to track joint utility and detectability dynamics. ② *Source-quota scaling:* At a fixed budget of 1,252 updates, the source-token proportion scales across 5%, 30%, and 50%. As higher quotas displace filler data and alter text composition (Appendix B.4), this axis isolates detectability scaling across source concentrations.

Model capacity and architectural diversity. To examine whether laundering and auditing behaviors generalize across model scales and model families, **`LaunderBench`** extends evaluation along two additional axes: ① *Capacity scaling:* Scaling model parameters across Pythia-1.4B, Pythia-2.8B, and Pythia-6.9B under identical source injection schedules to evaluate whether larger models exhibit stronger verbatim retention or greater laundering resistance. ② *Architectural diversity:* Evaluating across distinct transformer implementations by incorporating gptneo backbones (Black et al., 2021) alongside the Pythia family under matched compute budgets.

3.3 Auditing Protocols and Benchmark Extensibility

Detector suite across information tiers. We evaluate 6 canonical provenance detectors mapped directly to the access tiers of §2.2: ① *Target-only access ($\mathcal{I}_{\text{target}}$):* Loss (Yeom et al., 2018), Min-K% (Shi et al., 2024), Min-K++ (Zhang et al., 2025a), and Zlib (Carlini et al., 2021); ② *Prefix-augmented access ($\mathcal{I}_{\text{prefix}}$):* ReCaLL (Xie et al., 2024), which conditions target predictions on non-member prefix text; ③ *Reference-model access ($\mathcal{I}_{\text{ref}}$):* The Reference detector (Watson et al., 2022), which evaluates target log-likelihood ratios against the seed- and update-matched filler-only baseline θ^{-}. Spanning these detector families, we evaluate 10 concrete variants across hyperparameter configurations (e.g., truncation ratios for Min-K and Min-K++; details are in Appendix C.1).

Operating points and uncertainty. We evaluate both threshold-free AUROC and true-positive rates at bounded false-positive rates ($\text{TPR@}\alpha\text{FPR}$) for different operating thresholds $\alpha \in \{1\%, 5\%, 10\%\}$ (Appendix C.2). All reported metrics include 95% confidence intervals computed via 2,000 paired document-bootstrap resamples conditional on the checkpoints (Appendix C.3), while independent training seeds are analyzed separately to capture optimization variability across training runs.

Extensibility. **`LaunderBench`** is structured as a modular platform to support ongoing community evaluation (Appendix C.4): ① *Evaluating new detectors:* Researchers can benchmark new detectors directly on our public checkpoint suite across all trajectory snapshots and quota regimes *without* retraining checkpoints. ② *Evaluating new laundering defenses:* Practitioners can plug in new text transformation pipelines $T(x)$, train under our standardized background mixture schedule, and benchmark retained utility and detection evasion against our established baselines.

4 Results

We evaluate data laundering through the three diagnostic criteria (§2.3), namely baseline auditability, evasion efficacy, and utility viability. Unless noted otherwise, all comparisons evaluate shared source originals under fixed update budgets, source quotas, corpora, and random seeds.

Table 1: **Audit detectability and modeling utility trade-offs at trajectory endpoints** (5% quota, Qwen3-32B). Rewriting reduces Reference detectability while preserving positive source modeling gain, but lost gain widens as training extends. Values are equal-weight means across eight corpus–seed cells. Reference AUROC and strict TPR@1%FPR evaluate separation from 60 held-out originals. Retained gain ($\Delta\text{PPL}_{\mathcal{S}}$) is macro-perplexity reduction relative to the seed- and update-matched filler baseline (θ^{-}); lost gain measures the shortfall relative to unlaundered training ($\theta^{+}_{T_{\text{id}}}$). Table 11 (in Appendix D) reports all ten rewriting conditions.

	313 updates				1,252 updates			
Rewriting Condition	AUROC	TPR (%)	Gain (%)	Lost (pp)	AUROC	TPR (%)	Gain (%)	Lost (pp)
Original	0.983	86.1	5.89	—	0.996	80.9	30.04	—
Lexical	0.878	37.3	3.11	2.78	0.950	52.4	19.48	10.56
Syntactic	0.907	44.5	3.61	2.28	0.971	55.2	21.55	8.49
SAGE	0.812	31.3	1.89	4.00	0.882	36.4	15.53	14.51
SAGE-R	0.662	8.9	0.76	5.13	0.764	19.2	8.53	21.51

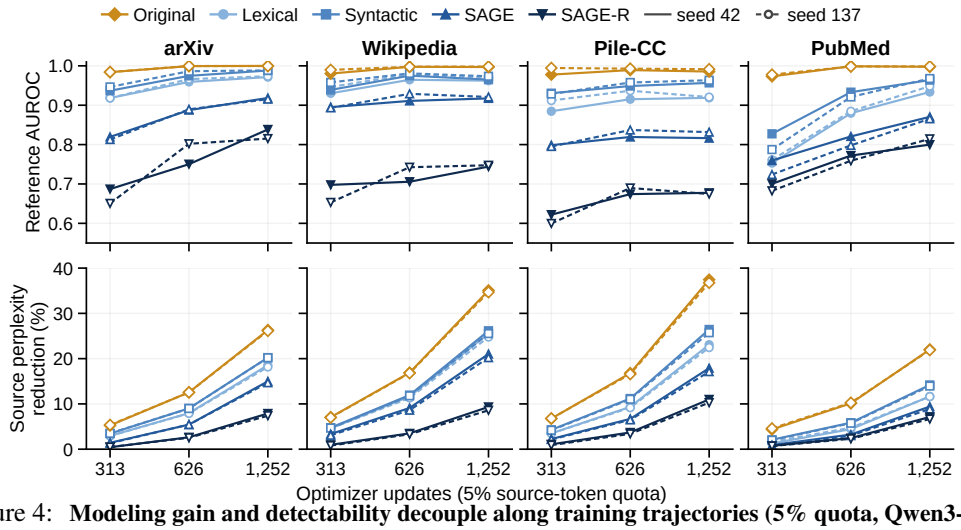

Figure 4: **Modeling gain and detectability decouple along training trajectories (5% quota, Qwen3-32B).** While source modeling gain accumulates monotonically across training, Reference AUROC plateaus or declines (e.g., between 626 and 1,252 updates), showing that acquired utility decouples from surface auditability. **Top:** Reference AUROC separating source originals from 60 held-out originals. **Bottom:** Macro-perplexity reduction on source originals relative to the matched filler-only baseline θ^{-}. Full results are in Appendix D.

4.1 How Does Rewriting Change Detection and Modeling Along Training?

Rewriting retains source utility while widening lost gain. Across all trajectories, models trained on rewrites consistently improve source prediction over the unexposed filler-only baseline (θ^{-}) while remaining less detectable than unlaundered training ($\theta^{+}_{T_{\text{id}}}$) under Reference (Table 1). All evaluated source gains yield positive bootstrap intervals, confirming *utility viability*. As training proceeds, retained utility accumulates across all rewriting conditions; however, unlaundered models acquire source capabilities substantially faster. Consequently, the lost gain widens across training in all 72 rewritten trajectories (Figure 4), demonstrating that while laundering captures increasing domain utility beyond filler text, it incurs an expanding capability penalty relative to verbatim training.

Background drift explains why relative gains grow as absolute prediction degrades. Continued training on background filler text causes the unexposed baseline's perplexity on source originals to rise monotonically across checkpoints, reflecting progressive domain drift away from source distributions. Because this baseline drifts upward, a laundered model's relative gain ($\Delta\text{PPL}_{\mathcal{S}}$) can increase even as its absolute cross-entropy on originals worsens. This divergence occurs in 62 out of 72 rewritten trajectories (Appendix D.3), demonstrating that tracking relative gains against an unexposed baseline is necessary to distinguish genuine source retention from background degradation.

Audit detectability decouples from modeling gain along training. While source modeling gain increases monotonically across all trajectories, Reference AUROC fails to scale in step, plateauing or

Table 2: **AUROC of six detection methods along both training axes.** Entries are equal-weight means over corpus–seed cells: 8 for unlaundered training (Orig.) and 72 for the nine rewriting conditions (Rew.). Scores use each condition's retained cohort (Pile-CC: 465 sources; 464 for SAGE-R), and ReCaLL carries the prefix-allocation limitation of Appendix C.1. The 1,252-update, 5% models belong to both axes. Appendix E reports all ten variants (Table 12).

		Updates at 5% quota				Quota at 1,252 updates			
		313 updates		1,252 updates		30% quota		50% quota	
Access	Detector	Orig.	Rew.	Orig.	Rew.	Orig.	Rew.	Orig.	Rew.
Target	Loss	0.507	0.489	0.627	0.539	0.947	0.693	0.997	0.745
	Zlib	0.502	0.486	0.616	0.532	0.940	0.678	0.997	0.729
	Min-K	0.525	0.503	0.668	0.560	0.982	0.727	0.999	0.775
	Min-K++	0.588	0.541	0.807	0.632	0.995	0.771	0.998	0.763
+ prefix	ReCaLL	0.579	0.545	0.757	0.616	0.993	0.792	1.000	0.825
+ reference	Reference	0.983	0.820	0.996	0.901	1.000	0.965	1.000	0.952

declining between 626 and 1,252 updates in nearly a third of runs (Figure 4). For instance, SAGE on Pile-CC exhibits continuous utility growth while its AUROC stalls completely. This decoupling stems from their distinct objectives: Reference AUROC measures ranking separation across source and held-out cohorts, whereas modeling gain averages absolute likelihood reductions exclusively on sources. Consequently, accumulating source domain utility does not necessarily enhance non-member separation, confirming that detection-only audits cannot track retained utility.

4.2 When Is an Audit Informative Before Rewriting?

Target-only audits are uninformative under low training exposure. Under low training budget (313 updates at 5% quota), all target-only detectors hover near random on unlaundered originals, while Reference achieves near-perfect separation (Table 2 and Figure 5). In this regime, their low detectability does not reflect successful laundering.

Auditor access tiers generally govern detection resilience. ① *Reference auditing* ($\mathcal{I}_{\text{ref}}$) maintains near-perfect unlaundered separation and remains robust after rewriting across all quotas, showing only a slight non-monotonic decline at 50% quota as filler is displaced (Appendix B.4). ② *Prefix conditioning* (ReCaLL, $\mathcal{I}_{\text{prefix}}$) achieves intermediate sensitivity, delivering substantial post-rewriting separation at high quotas without an auxiliary reference model. ③ *Target-only detectors* ($\mathcal{I}_{\text{target}}$) rely on exposure to separate members. Zlib mirrors raw Loss without distinct advantage, while Min-K++ achieves the strongest separation under moderate exposure before saturating at higher quotas.

Heterogeneity across corpora and rewriting procedures. ① *Corpus difficulty:* Across all detectors, Pile-CC consistently yields the lowest post-rewriting detectability. ② *Rewriting resilience:* Syntactic restructuring is the most detectable procedure across all settings, whereas SAGE-R is exceptionally evasive, keeping target-only detectors and ReCaLL near chance even after extended training and constraining Reference detectability well below other procedures. ③ *Optimization stability:* Variation between training seeds remains negligible across detectors, confirming that reported rankings are robust to random initialization.

4.3 Do AUROC Comparisons Hold at Strict Operating Points?

High aggregate AUROC conceals extensive missed sources under strict FPR budgets. For auditors requiring bounded false-alarm rates (TPR@αFPR), AUROC provides an overly optimistic assessment of forensic efficacy. Under a strict zero-false-alarm budget ($\alpha = 0.01$), high AUROC frequently coexists with severe detection failure: e.g., Syntactic detects barely half of its sources despite near-perfect aggregate AUROC (Table 1). Even under unlaundered training, strict detection rates fall as training extends, despite rising AUROC. Because the zero-FP threshold is governed by the single highest-scoring non-member, strict detection is sensitive: admitting merely three false alarms ($\alpha = 0.05$) frequently restores detection from poor levels to near-completion (e.g., on Pile-CC).

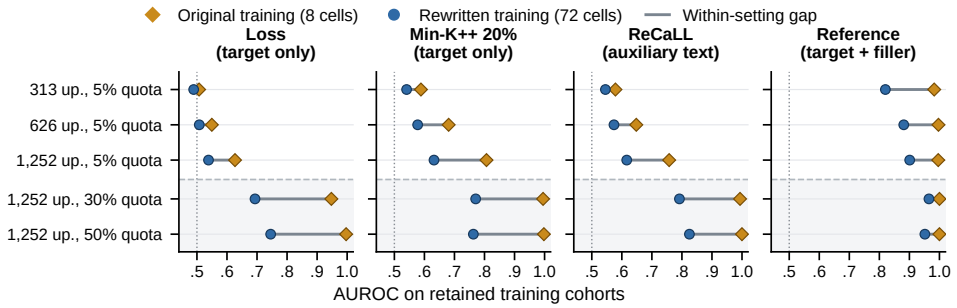

Figure 5: **Unlaundered audit baselines scale with training exposure.** Mean AUROC of four representative detector variants on retained cohorts across five training settings: Loss and Min-K++ 20% (target-only), ReCaLL (prefix-augmented), and Reference (reference-model). Amber diamonds denote unlaundered Original training; blue circles denote rewritten training (72 cells across nine conditions). Segments link the unlaundered baseline to rewritten detection; dotted vertical lines indicate random chance (AUROC = 0.5). See Appendix E for complete results.

Operating thresholds invert relative procedure rankings. Evaluating detection at discrete operating points frequently overturns AUROC-based conclusions. Across pairwise comparisons between rewritten conditions, 38 of 288 pairs invert their relative order when evaluated by zero-false-positive TPR rather than AUROC. For instance, on Pile-CC, SAGE Llama achieves higher aggregate AUROC than SAGE Qwen, yet SAGE Qwen detects substantially more sources above every negative control. Comparative laundering resistance is therefore fundamentally threshold-dependent (Appendix E).

5 Discussion and Outlook

Decoupling utility retention from surface detectability. Our findings show that source modeling utility and verbatim auditability decouple across training trajectories (§4.1). Because standard provenance detectors rely predominantly on verbatim token-level likelihood difference between member and non-member data, laundering allows models to acquire source domain utility while diminishing the surface statistical traces that detectors exploit. Controlled counterfactual benchmarking makes this divergence measurable, showing that verbatim detectability does not reliably track retained source influence under text transformations.

Implications for provenance auditing. We also highlight two practical considerations for auditing. ① *Resource dependence:* While reference-based detectors remain comparatively resilient to rewriting, target-only methods are significantly less capable under low training exposure (§4.2). The provenance auditors lacking access to unexposed background checkpoints therefore face constrained detection capabilities. ② *Operating-point sensitivity:* Aggregate AUROCs can obscure detection limits in low-false-positive regimes (TPR@1%FPR), where TPRs decline substantially, and method rankings can invert (§4.3). Provenance assessments under bounded false-alarm budgets therefore require evaluation at concrete operating thresholds.

Scope and limitations. We outline three clear boundaries of this study: ① *Query modality:* Audits here evaluate verbatim original sequences. They do not encompass active derivative-reconstruction audits (Li et al., 2026) or invasive watermarking schemes. ② *Utility characterization:* Currently, source utility is quantified via language modeling perplexity on original texts. While this isolates sequence-level acquisition without task-construction confounds, it does not directly measure downstream factual question-answering or conceptual reasoning retention. ③ *Empirical operating resolution:* Operating points are constrained by the held-out documents per corpus, which represents empirical sample bounds rather than population-calibrated error guarantees.

Future work. These boundaries outline immediate extensions for the community. A natural next step is mapping laundering scaling laws across larger parameter regimes ($>$ 7B) and instruction-tuned architectures. Equally important is developing effective source utility measures beyond perplexity, e.g., through LLM-as-a-judge (Ertan et al., 2026). Ultimately, establishing confound-free benchmarks like **`LaunderBench`** marks a step toward legally viable copyright auditing, enabling future research to attribute data influence beyond surface-level lexical memorization, in the era of generative AI.

AI USE STATEMENT

In accordance with the ICLR Author Guidelines, we disclose that ChatGPT and Claude Code were used as an auxiliary tool during the preparation of this work. Specifically, AI assistance was used to help polish academic language, craft implementation and utility code, and refine script syntax for collecting data and plotting experimental figures. All core scientific conceptualizations, designs, experimental logic, and final manuscript revisions were independently executed, manually verified, and approved by the authors, who maintain full responsibility for the integrity of this work.

REFERENCES

Colin Bannard and Chris Callison-Burch. Paraphrasing with bilingual parallel corpora. In *ACL*, pp. 597–604, 2005.

Regina Barzilay and Kathleen R. McKeown. Extracting paraphrases from a parallel corpus. In *ACL*, pp. 50–57, 2001.

Rahul Bhagat and Eduard Hovy. Squibs: What is a paraphrase? *CL*, 39(3):463–472, 2013. doi: 10.1162/COLI_a_00166.

Stella Biderman, Hailey Schoelkopf, Quentin Anthony, Herbie Bradley, Kyle O'Brien, Eric Hallahan, Mohammad Aflah Khan, Shivanshu Purohit, USVSN Sai Prashanth, Edward Raff, Aviya Skowron, Lintang Sutawika, and Oskar van der Wal. Pythia: A Suite for Analyzing Large Language Models Across Training and Scaling. In *ICML*, 2023.

Sid Black, Gao Leo, Phil Wang, Connor Leahy, and Stella Biderman. GPT-Neo: Large Scale Autoregressive Language Modeling with Mesh-Tensorflow, 2021. URL https://doi.org/10.5281/zenodo.5297715.

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

Nicholas Carlini, Steve Chien, Milad Nasr, Shuang Song, Andreas Terzis, and Florian Tramer. Membership inference attacks from first principles. *arXiv preprint arXiv:2112.03570v2*, 2022.

Hoagy Cunningham, Aidan Ewart, Logan Riggs, Robert Huben, and Lee Sharkey. Sparse autoencoders find highly interpretable features in language models. *arXiv preprint arXiv:2309.08600v1*, 2023.

Debeshee Das, Jie Zhang, and Florian Tramèr. Blind baselines beat membership inference attacks for foundation models. In *IEEE SPW*, pp. 118–125, 2025.

Michael Duan, Anshuman Suri, Niloofar Mireshghallah, Sewon Min, Weijia Shi, Luke Zettlemoyer, Yulia Tsvetkov, Yejin Choi, David Evans, and Hannaneh Hajishirzi. Do membership inference attacks work on large language models? *arXiv preprint arXiv:2402.07841*, 2024.

André Vicente Duarte, Xuandong Zhao, Arlindo L. Oliveira, and Lei Li. DE-COP: Detecting copyrighted content in language models training data. In *ICML*, volume 235, pp. 11940–11956, 2024.

B. Efron. Bootstrap methods: Another look at the jackknife. *Ann. Statist.*, 7(1):1–26, 1979. doi: 10.1214/aos/1176344552.

Murat Bilgehan Ertan, Emirhan Böge, Min Chen, Kaleel Mahmood, and Marten van Dijk. On the evidentiary limits of membership inference for copyright auditing. *arXiv preprint arXiv:2601.12937*, 2026.

Peter Henderson, Xuechen Li, Dan Jurafsky, Tatsunori Hashimoto, Mark A. Lemley, and Percy Liang. Foundation models and fair use. *JMLR*, 24(400):1–79, 2023.

Kalpesh Krishna, Yixiao Song, Marzena Karpinska, John Wieting, and Mohit Iyyer. Paraphrasing evades detectors of AI-generated text, but retrieval is an effective defense. In *NeurIPS*, volume 36, pp. 27469–27500, 2023.

Muxing Li, Zesheng Ye, Sharon Li, and Feng Liu. Combating data laundering in LLM training. *arXiv preprint arXiv:2604.01904*, 2026.

Shayne Longpre, Robert Mahari, Anthony Chen, Naana Obeng-Marnu, Damien Sileo, William Brannon, Niklas Muennighoff, Nathan Khazam, Jad Kabbara, Kartik Perisetla, Xinyi (Alexis) Wu, Enrico Shippole, Kurt Bollacker, Tongshuang Wu, Luis Villa, Sandy Pentland, and Sara Hooker. The data provenance initiative: A large scale audit of dataset licensing & attribution in AI. *arXiv preprint arXiv:2310.16787v3*, 2023.

Pratyush Maini, Hengrui Jia, Nicolas Papernot, and Adam Dziedzic. LLM dataset inference: Did you train on my dataset? *arXiv preprint arXiv:2406.06443*, 2024.

Matthieu Meeus, Igor Shilov, Manuel Faysse, and Yves-Alexandre de Montjoye. Copyright traps for large language models. In *ICML*, volume 235, pp. 35296–35309, 2024.

Matthieu Meeus, Igor Shilov, Shubham Jain, Manuel Faysse, Marek Rei, and Yves-Alexandre de Montjoye. SoK: Membership inference attacks on LLMs are rushing nowhere (and how to fix it). In *SaTML*, pp. 385–401, 2025.

Fatemehsadat Mireshghallah, Kartik Goyal, Archit Uniyal, Taylor Berg-Kirkpatrick, and Reza Shokri. Quantifying privacy risks of masked language models using membership inference attacks. In *EMNLP*, pp. 8332–8347, 2022.

OpenAI. GPT-4 technical report. *arXiv preprint arXiv:2303.08774*, 2023.

Weijia Shi, Anirudh Ajith, Mengzhou Xia, Yangsibo Huang, Daogao Liu, Terra Blevins, Danqi Chen, and Luke Zettlemoyer. Detecting pretraining data from large language models. *arXiv preprint arXiv:2310.16789v3*, 2024.

Reza Shokri, Marco Stronati, Congzheng Song, and Vitaly Shmatikov. Membership Inference Attacks against Machine Learning Models. In *IEEE S&P*, 2017.

Alexander Wan, Kevin Klyman, Sayash Kapoor, Nestor Maslej, Shayne Longpre, Betty Xiong, Percy Liang, and Rishi Bommasani. The 2025 foundation model transparency index. *arXiv preprint arXiv:2512.10169*, 2025.

Cheng Wang, Yiwei Wang, Bryan Hooi, Yujun Cai, Nanyun Peng, and Kai-Wei Chang. Con-ReCall: Detecting pre-training data in LLMs via contrastive decoding. In *COLING*, pp. 1013–1026, 2025.

Lauren Watson, Chuan Guo, Graham Cormode, and Alex Sablayrolles. On the importance of difficulty calibration in membership inference attacks. *arXiv preprint arXiv:2111.08440v2*, 2022.

Sam Witteveen and Martin Andrews. Paraphrasing with large language models. In *WNGT*, pp. 215–220, 2019.

Roy Xie, Junlin Wang, Ruomin Huang, Minxing Zhang, Rong Ge, Jian Pei, Neil Zhenqiang Gong, and Bhuwan Dhingra. ReCaLL: Membership inference via relative conditional log-likelihoods. In *EMNLP*, pp. 8671–8689, 2024.

Samuel Yeom, Irene Giacomelli, Matt Fredrikson, and Somesh Jha. Privacy risk in machine learning: Analyzing the connection to overfitting. *arXiv preprint arXiv:1709.01604v5*, 2018.

Jingyang Zhang, Jingwei Sun, Eric Yeats, Yang Ouyang, Martin Kuo, Jianyi Zhang, Hao Frank Yang, and Hai Li. Min-K%++: Improved Baseline for Detecting Pre-Training Data from Large Language Models. In *ICLR*, 2025a.

Kaiyuan Zhang, Siyuan Cheng, Hanxi Guo, Yuetian Chen, Zian Su, Shengwei An, Yuntao Du, Charles Fleming, Ashish Kundu, Xiangyu Zhang, and Ninghui Li. SOFT: Selective data obfuscation for protecting LLM fine-tuning against membership inference attacks. In *USENIX*, 2025b.

Appendix

The appendices first describe how the benchmark is built and then extend its evidence. Appendices A–C specify how the training texts are produced, how they enter continued training, and how the resulting models are audited and their modeling measured. Appendices D and E extend the primary results of § 4, and Appendix F reports the supplementary scale and architecture studies. The displays summarize all primary cells; the per-cell estimates, bootstrap intervals and ROC vertices behind them accompany the paper as data records.

A Rewriting procedures and aligned examples

This appendix describes the rewriting procedures, shows one source document under each of them, and defines the source cohorts on which all conditions are compared.

A.1 Procedures, generators, and selection

Table 3 lists every training condition in the paper. The primary study compares Original training with nine rewriting procedures. They differ in three respects that the tables and figures keep separate: the rewriting operation, the generator that produces the text, and the rule that selects a prompt or a candidate output. Two further procedures were trained only in the GPT-Neo-2.7B study of Appendix F.

Table 3: Training conditions and the studies that use them. Qwen denotes Qwen3-32B and Llama denotes Llama-3.3-70B-Instruct; both are rewrite generators, not audited models. Every rewriting procedure is applied to the same source originals, and every model is audited on the originals.

Condition	Rewriting operation	Prompt or candidate selection	Studies
Original	None; the source text itself	Identity	Primary
Lexical Qwen	Lexical paraphrase	Fixed prompt	Primary, Pythia scale, GPT-Neo
Lexical Llama	Lexical paraphrase	Fixed prompt	Primary
Selected lexical Qwen	Lexical paraphrase	Surrogate-selected prompt	Primary, GPT-Neo
Syntactic Qwen	Syntactic restructuring	Fixed prompt	Primary
Selected syntactic Qwen	Syntactic restructuring	Surrogate-selected prompt	Primary
SAGE Qwen	Paraphrase guided by sparse-autoencoder features	Structure and overlap scores	Primary
SAGE Llama	Paraphrase guided by sparse-autoencoder features	Structure and overlap scores	Primary
SAGE-R Qwen	SAGE output, then narrative flattening and fact markers	Two-stage procedure	Primary
SAGE-R Llama	SAGE output, then narrative flattening and fact markers	Two-stage procedure	Primary
Discourse restructuring Qwen	Reorganization of the document's discourse	Fixed prompt	GPT-Neo only
Cross-lingual round-trip Qwen	Translation into Chinese and back into English	Fixed prompt	GPT-Neo only

Lexical and syntactic rewriting. Lexical and syntactic variation are longstanding objects of paraphrase research, although approximate equivalence need not preserve every detail (Barzilay & McKeown, 2001; Bhagat & Hovy, 2013). Language-model-based paraphrasing has also been demonstrated for both sentences and longer passages (Witteveen & Andrews, 2019). In **`LaunderBench`**,

lexical paraphrasing asks the generator to change local phrasing, whereas syntactic restructuring asks it to reorganize sentences; both prompts request that the facts be preserved. The five primary conditions of this kind use fixed, previously generated derivatives. Three use one fixed prompt. The two surrogate-selected conditions instead choose among five prompt variants using the first 20 source documents of each corpus. Each candidate prompt rewrites these documents, Pythia-160M scores the rewrites with Loss and Min-K ($k = 10\%$), both standardized against a fixed reference population of the raw prompt-selection texts, and the prompt with the lowest mean combined score rewrites the remaining documents. The surrogate is a much smaller model than the audited Pythia-1.4B target. The 20 prompt-selection documents never enter training, which leaves at most 480 derivatives per corpus for these conditions before generation failures and the shared-cohort restrictions of Appendix A.3.

Structure-guided rewriting. The SAGE derivatives were supplied by the method's authors (Ertan et al., 2026), with 500 attempted source documents per condition and corpus, decoded at temperature 0.7 and top-p 0.95. SAGE generates up to three candidates per document and scores each twice: a structure-preservation score compares last-token-pooled features of a sparse autoencoder on Gemma-2B (release `gemma-2b-res-jb`, layer `blocks.12.hook_resid_post`) between original and candidate, and a lexical-overlap score combines word and character overlap[2]. A candidate is accepted when its structure score is at least 0.6 and its overlap score at most 0.35. When no candidate meets both thresholds, the candidate with the largest structure score minus overlap score is kept, so not every retained derivative satisfies both thresholds; rejected outputs are excluded from training.

SAGE-R starts from the corresponding SAGE output, flattens its narrative, and replaces source facts with deterministic markers such as `<<FACT_1>>`. We use the supplied two-stage outputs rather than regenerating them from the originals. The markers stay in the training input, but their tokens are removed as prediction targets (Appendix B.2), so SAGE-R changes both the text and how the text supervises the model.

Procedures used only for GPT-Neo-2.7B. Discourse restructuring reorganizes a document's discourse structure, and the cross-lingual round trip translates the document into Chinese and back into English. Bilingual pivoting has also been used to extract paraphrases from parallel corpora (Bannard & Callison-Burch, 2005); our round trip instead applies document-level translation with an LLM. Both use Qwen3-32B with a fixed prompt. Together with Lexical Qwen and Selected lexical Qwen, they are the four procedures of the GPT-Neo learning-rate study, which trains on arXiv and Wikipedia.

Whether a rewrite is usable and whether its source is labeled as a member are separate properties. The source originals come from the MIMIR non-member splits, and a usable derivative makes its source a positive example of this benchmark because the derivative enters continued training.

A.2 One source under every procedure

Table 4 shows the training texts produced for one source by all eleven rewriting procedures, beside the original. The source is the Wikipedia article *Caerhendy transmitting station*, the example of Figure 3 and a member of the shared Wikipedia cohort. All twelve texts are taken verbatim from the stored training inputs: each file contains one accepted row for this document, every file records the same original text, and the two procedures used only for GPT-Neo pass their quality check for it. The original is printed in full. Each rewrite is shown through two windows chosen by a fixed rule, the beginning of the text through the sentence, line, or list item that contains its 40th word and the end of the text from the sentence, line, or list item that contains its 20th word from the end. Cutting at a line or list item as well as at a sentence keeps list-formatted output intact. The windows show how each procedure opens the document and what it keeps at the end, where the reference and link sections are.

The display makes the procedures' different operations concrete. The five lexical and syntactic rewrites keep the document's order, its named places, and its reference section while changing phrasing: "sited at" becomes "located in" or "located at", a "fill-in relay" becomes a "supplementary relay" in Lexical Llama and Selected lexical Qwen, and the syntactic rewrites merge clauses, as in "From the 1980s to March 2010, Caerhendy transmitted …". Lexical Llama also retitles the article, and Selected syntactic Qwen keeps long stretches of the original wording, such as "originally built in

[2]Sparse autoencoders provide a learned sparse decomposition of model activations into features (Cunningham et al., 2023); the specific structure score and candidate-selection rule here follow SAGE (Ertan et al., 2026)

the 1980s as a fill-in relay". The two SAGE rewrites change sentence structure as well as phrasing. SAGE Qwen opens with the valley rather than the station, and both drop the station's location relative to the M4 motorway and Port Talbot. Outside the displayed windows, SAGE Llama also omits that the served houses lie about 500 m across the valley.

Each SAGE-R rewrite keeps the sentences of its SAGE input but replaces names, places, and dates with markers and removes the title, the channel list, and the links. The two generators mark different spans in this document: SAGE-R Qwen also replaces the 500 m and 2 km distances and the 0.5 W output with markers, whereas SAGE-R Llama keeps the 2 km distance and the 0.5 W output and omits the final sentence about S4C. Marker numbering is local to each output, so `<<FACT_1>>` denotes the valley in one and the station in the other.

The two procedures used only for GPT-Neo change the form of the document further. Discourse restructuring turns the article into a nested Markdown list with bold headings and replaces each external link with a placeholder, `(link)`. The round trip through Chinese preserves the structure but changes names: Caerhendy becomes "Cae Rhendy" and the Afan Valley the "Afon valley", and outside the windows Ynys Park, Mynydd Emroch, Alltwen, Cwmafan and Wenvoe become "Einion Park", "Minnedrumroch", "Alderton", "Cumyvan" and "Wenwood". Its closing window also turns "Channel 4" into "the fourth channel" and "UKFree" into "Freeview UK". These are properties of one document's outputs and are reproduced as generated; the benchmark does not score how faithful each rewrite is.

Table 4: Training versions of one source document, the Wikipedia article *Caerhendy transmitting station*, under every rewriting procedure in this paper. Each rewritten excerpt reproduces the training text from its beginning through the sentence, line or list item containing word 40, then [. . .], then the text from the sentence, line or list item containing the 20th word from the end. Wording, markup, markers and errors are reproduced as generated; blank lines and trailing spaces are not shown. Text outside the windows, including differences discussed in Appendix A.2, is omitted.

Condition	**Training text**
Original training text (shown in full)	
Original	Caerhendy transmitting station
source document	The Caerhendy television relay station was sited at Caerhendy in the Afan Valley, a few kilometres north of Junction 40 on the M4 motorway at Port Talbot. It was originally built in the 1980s as a fill-in relay for UHF analogue television. It was sited at the clubhouse of the soccer ground at Ynys Park and consisted of a mere 3 m aluminium mast carrying a vertically stacked yagi array. The transmitter served about 50 houses in Caerhendy, about 500 m across the valley to the east, which for reasons of geography couldn't get a signal from Mynydd Emroch transmitter just to the south. Alltwen transmitter re-radiated a signal received off-air from Cwmafan about 2 km farther to the northeast, itself a relay of Wenvoe. It was possibly the lowest ERP officially provided TV relay in the UK, radiating just 0.5 W on each of its four UHF channels. When it came, the digital switchover process rendered Caerhendy redundant. It is no longer in service. Channels listed by frequency Analogue television 1980s - March 2010 Caerhendy (being in Wales) transmitted the S4C variant of Channel 4. References External links The Transmission Gallery: Caerhendy UKFree: Caerhendy TV transmitter

Continued on next page

Table 4 (continued)

Condition	Training text
Lexical and syntactic rewriting (primary study)	
Lexical Qwen Qwen3-32B fixed prompt	Caerhendy transmitting station The Caerhendy television relay station was located in Caerhendy within the Afan Valley, a short distance to the north of Junction 40 on the M4 motorway at Port Talbot. Initially constructed in the 1980s, it functioned as a fill-in relay for UHF analogue television. [. . .] Caerhendy (being in Wales) transmitted the S4C version of Channel 4. References External links The Transmission Gallery: Caerhendy UKFree: Caerhendy TV transmitter
Lexical Llama Llama-3.3-70B-Instruct fixed prompt	Caerhendy Transmission Facility The Caerhendy television relay station was located in Caerhendy, within the Afan Valley, a short distance north of Junction 40 on the M4 motorway, near Port Talbot. Initially constructed in the 1980s, its purpose was to serve as a supplementary relay for UHF analogue television broadcasts. [. . .] As it was located in Wales, Caerhendy transmitted the S4C variant of Channel 4. References External links The Transmission Gallery: Caerhendy UKFree: Caerhendy TV transmitter
Selected lexical Qwen Qwen3-32B surrogate-selected prompt	Caerhendy transmitting station The Caerhendy television relay station was located in Caerhendy, within the Afan Valley, a short distance north of Junction 40 on the M4 motorway at Port Talbot. It was initially constructed in the 1980s as a supplementary relay for UHF analogue television. [. . .] Caerhendy (being in Wales) transmitted the S4C version of Channel 4. References External links The Transmission Gallery: Caerhendy UKFree: Caerhendy TV transmitter
Syntactic Qwen Qwen3-32B fixed prompt	Caerhendy transmitting station The Caerhendy television relay station was located in Caerhendy, within the Afan Valley, a few kilometers north of Junction 40 on the M4 motorway at Port Talbot. Initially constructed in the 1980s, the station served as a fill-in relay for UHF analogue television. [. . .] From the 1980s to March 2010, Caerhendy transmitted the S4C version of Channel 4, as it was located in Wales. References External links The Transmission Gallery: Caerhendy UKFree: Caerhendy TV transmitter
Selected syntactic Qwen Qwen3-32B surrogate-selected prompt	Caerhendy transmitting station The Caerhendy television relay station was located at Caerhendy in the Afan Valley, a few kilometres north of Junction 40 on the M4 motorway at Port Talbot, and was originally built in the 1980s as a fill-in relay for UHF analogue television. [. . .] From the 1980s to March 2010, Caerhendy, being in Wales, transmitted the S4C variant of Channel 4. References External links The Transmission Gallery: Caerhendy UKFree: Caerhendy TV transmitter

Continued on next page

Table 4 (continued)

Condition	Training text
Structure-guided rewriting and fact markers (primary study)	
SAGE Qwen Qwen3-32B SAE-structure and overlap scores	Caerhendy transmitting station Located in the Afan Valley, the Caerhendy television relay station was constructed in the 1980s to function as an auxiliary relay for UHF analogue television. It was strategically placed at the clubhouse of the soccer ground in Ynys Park, featuring a modest 3-meter aluminum mast equipped with a vertically stacked Yagi antenna array. [. . .] As a facility located in Wales, the Caerhendy transmitter was responsible for broadcasting the S4C version of Channel 4. References External links The Transmission Gallery: Caerhendy UKFree: Caerhendy TV transmitter
SAGE Llama Llama-3.3-70B-Instruct SAE-structure and overlap scores	Caerhendy transmitting station The Caerhendy television relay station, situated in the Afan Valley near Caerhendy, was constructed in the 1980s to serve as a supplementary relay for UHF analogue television broadcasts. Specifically located at the Ynys Park soccer ground's clubhouse, the installation comprised a modest 3-meter aluminum mast equipped with a vertically stacked yagi array. [. . .] Caerhendy (being in Wales) transmitted the S4C variant of Channel 4. References External links The Transmission Gallery: Caerhendy UKFree: Caerhendy TV transmitter
SAGE-R Qwen Qwen3-32B two-stage procedure	Located in the <<FACT_1>>, the <<FACT_2>> television relay station was constructed in the <<FACT_3>> to function as an auxiliary relay for UHF analogue television. It was strategically placed at the clubhouse of the soccer ground in <<FACT_4>>, featuring a modest 3-meter aluminum mast equipped with a vertically stacked Yagi antenna array. [. . .] With the advent of the digital switchover, the <<FACT_2>> relay became obsolete and has since ceased operations. As a facility located in Wales, the <<FACT_2>> transmitter was responsible for broadcasting the S4C version of Channel 4.
SAGE-R Llama Llama-3.3-70B-Instruct two-stage procedure	The <<FACT_1>> television relay station, situated in the <<FACT_2>> near <<FACT_1>>, was constructed in the <<FACT_3>> to serve as a supplementary relay for UHF analogue television broadcasts. Specifically located at the <<FACT_4>> soccer ground's clubhouse, the installation comprised a modest 3-meter aluminum mast equipped with a vertically stacked yagi array. [. . .] The advent of digital switchover ultimately led to the redundancy of the <<FACT_1>> transmitter, which is now out of operation.
Procedures used only in the GPT-Neo-2.7B study	
Discourse restructuring Qwen Qwen3-32B fixed prompt	- **Caerhendy transmitting station** - Located at Caerhendy in the Afan Valley, a few kilometres north of Junction 40 on the M4 motorway at Port Talbot. - Built in the 1980s as a fill-in relay for UHF analogue television. - Sited at the clubhouse of the soccer ground at Ynys Park. [. . .] - Caerhendy transmitted the S4C variant of Channel 4 (being in Wales). - **References** - **External links** - [The Transmission Gallery: Caerhendy](link) - [UKFree: Caerhendy TV transmitter](link)

Continued on next page

Table 5: Source and held-out documents per corpus. Core-cohort models train on the core sources and SAGE-R models on the SAGE-R sources; every matched comparison uses the shared sources. Counts are identical for both training seeds.

Corpus	Core sources	SAGE-R sources	Shared sources	Held-out originals
arXiv	471	471	471	60
Wikipedia	460	460	460	60
Pile-CC	465	464	464	60
PubMed	474	474	474	60

Table 4 (continued)

Condition	**Training text**
Cross-lingual round-trip Qwen Qwen3-32B fixed prompt	The Cae Rhendy Relay Station The Cae Rhendy television relay station is located in Cae Rhendy, Afon valley, a few kilometers north of junction 40 of the M4 motorway at Port Talbot. It was originally built in the 1980s as a fill-in relay for UHF analog television. [. . .] Cae Rhendy (in Wales) broadcast the S4C version of the fourth channel. References External Links Transmission Gallery: Cae Rhendy Freeview UK: Cae Rhendy Television Transmitter

A.3 Source cohorts and shared queries

Each corpus provides 500 candidate source originals and 60 held-out originals, all from MIMIR non-member splits drawn from the Pile test set (Duan et al., 2024; Biderman et al., 2023). The held-out originals are never injected. The candidate sources shrink before training for two reasons: the surrogate-selected conditions exclude their 20 prompt-selection documents, and some derivatives are missing or rejected. Comparing conditions on documents that each condition kept separately would let every procedure choose its own queries, so the benchmark defines two nested cohorts. The core cohort contains the documents with usable text under Original, all five lexical and syntactic conditions, and both SAGE conditions; the SAGE-R cohort restricts it to documents with usable outputs under both SAGE-R conditions. Table 5 gives their sizes.

The two cohorts differ only for Pile-CC, where the eight core conditions were trained on 465 sources and the SAGE-R conditions on 464. The matched analysis restricts the stored core-condition scores to the 464 shared originals, which aligns the queries without changing any fitted model or its training history. Every joint result and every fixed-variant detection result uses these shared queries. The six additional detector variants and the saved intervals for changes in AUROC were computed on each model's retained training cohort, which equals the shared cohort except for these eight Pile-CC conditions; the displays in Appendix E identify which cohort they use.

Crossing the ten conditions with four corpora, two seeds, and three update counts gives the 240 source-trained checkpoints of the training-length trajectories, which belong to 80 trajectories and are compared with six filler-only checkpoints. The source-quota comparison trains the same 80 condition–corpus–seed combinations at 30% and 50% quotas for 1,252 updates, adding 160 models; its 5% models and filler-only checkpoints are those of the trajectories.

B Training protocol and realized exposure

All primary models, source-trained and filler-only, share one continued-training recipe. They differ only in the documents that fill the fixed schedule and, for SAGE-R, in which tokens are prediction targets. This appendix gives the recipe, the treatment of masked markers, and the construction of the mixture, and then reports how many source and filler tokens the models actually read. Model-specific settings of the supplementary studies appear in Appendix F.

Table 6: Continued-training settings shared by all source-trained and filler-only runs of the primary study.

Setting	Value
Target model and tokenizer	EleutherAI/pythia-1.4b, revision `step143000`; all parameters trained
Optimizer	AdamW, $\beta_1 = 0.9$, $\beta_2 = 0.95$, $\epsilon = 10^{-8}$, weight decay 0.01, no gradient clipping
Learning rate	Peak 10^{-5}; linear warmup over 100 updates, then constant
Updates	313, 626, and 1,252 (training-length trajectories); 1,252 (source-quota comparison)
Batch	32 document sequences per update, no gradient accumulation; maximum length 2,048 tokens
Source-token quota	5% (trajectories); 5%, 30%, and 50% (quota comparison)
Filler pool	1,000 fixed documents from the Pile-uncopyrighted training split
Seeds	42 and 137
Precision	BF16 model with CUDA autocast; gradient checkpointing
Software and hardware	Python 3.11.14, PyTorch 2.10.0+cu128, Transformers 4.57.6, Tokenizers 0.22.2, CUDA 12.8; H100 GPUs for the trajectories (evaluation hardware not recorded)

B.1 Initialization and optimization

Every source-trained run and its seed-matched filler-only run start from the same Pythia-1.4B checkpoint and tokenizer revision, `step143000`, and update the full model with the settings in Table 6. The filler-only run replaces the entire source–filler mixture with filler and keeps the initialization, optimizer, seed, and update schedule. One filler-only run per seed is evaluated after 313, 626, and 1,252 updates and serves every corpus and condition at that update count.

The learning-rate multiplier rises linearly from $1/100$ to 1 over the first 100 updates and then stays at 1; each batch produces one optimizer update followed by one scheduler step. The training seed initializes the Python and PyTorch random generators and determines the data-order shuffle. Deterministic algorithms are requested in warning-only mode, so a fixed seed does not guarantee bitwise-identical runs across software or hardware.

B.2 Tokenization and SAGE-R masking

Each retained derivative and each filler document is tokenized separately with the Pythia tokenizer into one sequence of at most 2,048 tokens, padded to that length. Documents are never packed together, and text beyond the truncation limit is not trained on. The end-of-sequence token serves as padding; padding positions are excluded from the attention mask and receive the ignored label -100.

SAGE-R additionally masks its fact markers. The supplied marker grammar locates each complete marker span in the derivative, the fast tokenizer's character offsets identify every non-padding token that overlaps such a span, and those tokens receive the label -100. The marker tokens remain in the input, where they can serve as context for later text; only their prediction targets are removed. All other source and filler tokens keep their next-token targets. The mixture planner below counts non-padding tokens, so it does not see the masking: a SAGE-R model and a SAGE model can read the same number of source tokens while being supervised on different numbers of them.

B.3 Mixture construction and realized exposure

A 313-update run has $313 \times 32 = 10{,}016$ sequence slots. The planner fills them by cycling through the retained source derivatives of the condition and through the filler pool (document IDs 1000–1999). It enumerates the possible numbers of source draws and keeps the allocation whose share of non-padding source tokens is closest to the nominal quota, counting the actual tokenized length of every draw rather than assigning a percentage of slots. Ties go to an allocation at or above the quota and then to the one closest to a single pass through the source cohort. The remaining slots hold filler, the combined list is shuffled with the training seed, and a filler-only run uses filler in every slot. The

Table 7: Realized source and filler input in one 313-update block of the training-length trajectories, summed over the four corpora (identical for both seeds; the 626- and 1,252-update checkpoints repeat the block two and four times). Input tokens are non-padding tokens; supervised tokens are prediction targets after the causal shift, which exclude masked SAGE-R markers. Selected: surrogate-selected prompt.

Rewriting	Generator	Source sequences	Source input (M tokens)	Supervised (M tokens)	Supervised share (%)	Filler input (M tokens)
Original	—	4,532	1.317	1.312	99.7	25.02
Lexical	Qwen	4,364	1.323	1.319	99.7	25.14
	Llama	4,338	1.324	1.319	99.7	25.16
	Qwen, selected	4,382	1.323	1.318	99.7	25.13
Syntactic	Qwen	4,363	1.323	1.319	99.7	25.14
	Qwen, selected	4,373	1.323	1.319	99.7	25.14
SAGE	Qwen	4,746	1.310	1.305	99.6	24.89
	Llama	4,312	1.324	1.320	99.7	25.17
SAGE-R	Qwen	5,374	1.287	1.035	80.4	24.46
	Llama	4,879	1.304	1.059	81.2	24.79

same filler pool serves every condition, which holds the background text fixed across comparisons; it does not by itself rule out textual overlap between the filler and the audited corpora.

Longer runs repeat this block rather than drawing a new schedule: the 313-update multiset of 10,016 sequence occurrences and its seeded order are used twice for 626 updates and four times for 1,252. Because the planner fits whole documents to a token share, source documents are not drawn equally often. Under Original training on arXiv, for example, 89 sources are drawn three times and 382 twice per block, hence 12 and 8 times over 1,252 updates.

Table 7 shows the result for one block. Every condition reaches the 5% source share, but it does so with a different number of source sequences because derivatives differ in length: SAGE-R Qwen needs 5,374 sequences where Original needs 4,532. The supervised share separates SAGE-R from everything else. Every other condition is supervised on about 99.7% of its source input tokens, the remainder being the first token of each sequence lost to the causal shift, whereas SAGE-R Qwen and SAGE-R Llama are supervised on 80.4% and 81.2% because their marker tokens are masked. Equal updates and equal source-token shares therefore do not imply an equal number of optimized source targets, which is why each comparison in the paper evaluates a rewriting procedure together with the way its text is used in training.

B.4 Source quota and total input

The source-quota comparison keeps 1,252 updates, and therefore 40,064 sequence slots per model, while raising the source share. Source sequences are much shorter than filler sequences, about 290 against 705 non-padding tokens on average, so each additional source sequence displaces a longer filler sequence. Table 8 shows the consequence summed over the 80 source-trained models of each setting: source input grows from 105.3 million tokens at 5% to 474.2 and 659.8 million at 30% and 50%, filler input falls from 2,000.4 to 1,106.4 and 659.8 million, and total input falls by 24.9% and 37.3%. A higher quota therefore changes three things at once: more source exposure, less filler, and fewer tokens overall. This is why the paper uses the quota comparison for detection only and does not interpret it as the effect of source exposure alone.

C Auditing and modeling measurements

Every model is evaluated on the same original documents in two ways: detectors score each original for evidence that its source was trained on, and document perplexities measure how well the model predicts the originals. This appendix defines the detector scores and the resources they require, the operating-point estimator, and the modeling gains with their intervals. It ends with the inputs needed to extend the benchmark.

Table 8: Input tokens summed over the 80 source-trained models of each training setting, in millions. These are accounting totals for a setting, not the exposure of one model. The last column is the change in total input (source plus filler) relative to the 5% quota at the same 1,252 updates. Each filler-only model reads filler only (7.0M after 313 updates; 14.1M after 626 updates; 28.2M after 1,252 updates; identical for both seeds).

Setting	Source input	Filler input	Source supervised	Total input change (%)
313 updates, 5%	26.3	500.1	25.3	
626 updates, 5%	52.6	1,000.2	50.5	
1,252 updates, 5%	105.3	2,000.4	101.0	0.0
1,252 updates, 30%	474.2	1,106.4	455.8	−24.9
1,252 updates, 50%	659.8	659.8	634.7	−37.3

Table 9: Detector scores and the resources they require. Loss is mean next-token cross-entropy; LL is summed token log-likelihood.

Method	Score	Target-model output	Additional resource
Loss	Negative document loss	Scalar loss	None
Zlib	Compressed byte length divided by document loss	Scalar loss	None
Min-K (×3)	Mean of the lowest $k\%$ token log-probabilities	Token log-probabilities	None
Min-K++ (×3)	Mean of the lowest $k\%$ vocabulary-standardized token log-probabilities	Next-token distribution	None
ReCaLL	Conditional LL divided by unconditional LL	Token log-likelihoods	Held-out texts as a prefix
Reference	Reference-model loss minus target loss	Scalar loss	Seed-matched filler-only checkpoint

C.1 Detector scores and resources

We evaluate six detection methods comprising ten variants: Loss, Zlib, Reference, ReCaLL, and three variants each of Min-K and Min-K++ with $k \in \{10, 20, 30\}$. Table 9 lists what each needs. All scores are computed on original documents and oriented so that larger values indicate stronger evidence of source use.

Likelihood notation. Let $J_\theta(x)$ be the prediction positions scored for document x and $\ell_t(x;\theta) = \log p_\theta(x_t \mid x_{<t})$ the observed-token log-probability at position t. The document loss and summed log-likelihood are

$$L_\theta(x) = -\frac{1}{|J_\theta(x)|} \sum_{t \in J_\theta(x)} \ell_t(x;\theta), \qquad \mathrm{LL}_\theta(x) = \sum_{t \in J_\theta(x)} \ell_t(x;\theta).$$

The scored positions are those retained by tokenization, the context limit, and the causal next-token shift; ReCaLL's conditional score can retain a different portion of the query, as described below.

Loss and Reference. Loss assigns $s_{\mathrm{Loss}}(x;\theta) = -L_\theta(x)$ (Yeom et al., 2018). For a source-trained model θ_T^+ and its seed-matched filler-only model θ^- with the same number of updates (313, 626, or 1,252), Reference uses

$$s_{\mathrm{Reference}}(x) = L_{\theta^-}(x) - L_{\theta_T^+}(x),$$

which calibrates the target's loss on a document against a model trained with the same initialization and schedule but without source injection (Watson et al., 2022). The score uses only the two models' losses on the queried document.

Min-K and Min-K++. Let $n = |J_\theta(x)|$ and sort the observed-token log-probabilities in ascending order as $\ell_{(1)}, \ldots, \ell_{(n)}$. For percentage k, Min-K averages the $m_k = \max(1, \lfloor kn/100 \rfloor)$ least likely

observed tokens (Shi et al., 2024),

$$s_{\text{Min-K}}(x;\theta) = \frac{1}{m_k}\sum_{j=1}^{m_k} \ell_{(j)}.$$

Min-K++ first standardizes each observed-token log-probability against the model's predicted distribution p_t at that position (Zhang et al., 2025a). With

$$\mu_t = \sum_v p_t(v)\log p_t(v), \qquad \sigma_t^2 = \sum_v p_t(v)\big(\log p_t(v)\big)^2 - \mu_t^2, \qquad z_t = \frac{\ell_t - \mu_t}{\sqrt{\max(\sigma_t^2, 10^{-12})}},$$

it averages the lowest m_k values of z_t, with the same m_k as Min-K, and returns 0 when no prediction tokens are available.

Zlib. With $C(x)$ the byte length of the UTF-8 text after zlib compression at level 9, the score is $C(x)/L_\theta(x)$, or $10^6 C(x)$ when $L_\theta(x) < 10^{-6}$. It combines the document's compressibility with the model's mean loss and needs only the scalar loss.

ReCaLL. For an auxiliary prefix P, ReCaLL uses $s_{\text{ReCaLL}}(x;\theta,P) = \text{LL}_\theta(x \mid P)/\text{LL}_\theta(x)$, with both sums over their scored query tokens, and returns 1 when the absolute unconditional log-likelihood is below 10^{-8}. The prefix joins three texts, sampled with a fixed seed of 42 from the first twenty held-out originals, with blank lines; all sixty held-out originals, including these three, remain negative queries. Although the implementation stores a 512-token prefix parameter, conditional scoring tokenizes the prefix and the concatenated prefix–query string under the common 2,048-token context limit without enforcing that cap. Only tokens attributed to the query enter the conditional log-likelihood, so a long prefix can shorten the scored query, and the numerator and denominator can cover different query tokens. The ReCaLL results in Appendix E.1 use this implementation.

C.2 Empirical operating points

AUROC summarizes discrimination over all thresholds, whereas an auditor who must avoid false accusations works at one threshold. For a nominal false-positive bound α, we report the largest attainable true-positive rate,

$$\max_{\tau:\,\widehat{\text{FPR}}(\tau)\le\alpha} \widehat{\text{TPR}}(\tau),$$

maximized over the complete set of empirical ROC vertices. A query counts as positive when its score is at least the threshold, tied scores enter together, and collinear vertices are retained (`drop_intermediate=False`); true- and false-positive counts are recovered directly from the scores. Lines drawn between vertices in a figure are visual guides, not interpolated operating points.

With 60 held-out negatives, empirical false-positive rates move in steps of $1/60$, so the bounds of 1%, 5%, and 10% admit at most zero, three, and six false positives. At the 1% bound a source is detected only if it scores strictly above every held-out original, which makes the result sensitive to the single highest-scoring negative. Thresholds are chosen on the same negatives, so these rates are in-sample operating points rather than calibrated population error rates. Each condition, corpus, and seed has its own ROC; seeds are never pooled and no ROC is computed from averaged scores, and summary rates average separately evaluated cells with equal weight.

These full-vertex operating points are computed on the shared queries for Reference and the three fixed target-only variants. The six additional variants have AUROC on their retained cohorts and a zero-false-positive rate saved by an earlier operating-point calculator, which can prune a collinear vertex and therefore need not match the full-vertex estimator. For the four fixed variants, the earlier calculator's saved AUROC and zero-false-positive rate coincide exactly with the full-vertex values in every cell where the retained and shared cohorts agree. We report the full-vertex values and keep the two calculators apart.

C.3 Modeling gains and their uncertainty

Document perplexity and aggregation. Each original is scored within a 2,048-token window. After the causal shift, padding targets are excluded, the mean cross-entropy over the remaining targets is exponentiated to give the document's perplexity, and document perplexities are averaged with equal

Table 10: Relative difference between the two filler-only models (seeds 42 and 137) in source-document macro-perplexity, $B_{\mathcal{S}}$, in percent with 95% document-bootstrap intervals, for each corpus and update count.

Corpus	313 updates	626 updates	1,252 updates
arXiv	0.07 [0.01, 0.13]	0.23 [0.15, 0.30]	0.45 [0.36, 0.55]
Wikipedia	0.17 [0.10, 0.24]	0.39 [0.31, 0.47]	0.98 [0.85, 1.11]
Pile-CC	0.12 [0.04, 0.20]	0.29 [0.20, 0.38]	0.90 [0.66, 1.10]
PubMed	0.10 [0.04, 0.16]	0.13 [0.05, 0.21]	0.52 [0.42, 0.61]

weight. The relative reduction of §2.3 is then applied to the two macro means. Because perplexity is averaged after exponentiation, the mean document loss on the log scale does not determine the gain on its own; loss-scale means are kept with the per-cell records and are not used for any gain. All perplexities are derived in 64-bit precision from the same forward passes that produce the detector scores, so detection and modeling rest on one scoring path, and the filler-only losses reconstructed from these scores agree exactly with those stored by the Reference computation. A separate utility-scoring implementation, used earlier in development on a 40-cell subset, gave slightly different values (7.060% instead of 6.979% for Original training on Wikipedia at seed 42 in that subset) and some different orderings and held-out signs, which is why the two implementations are never mixed.

Remaining and lost gain. The remaining gain is the source-perplexity reduction relative to the seed-matched filler-only model at the same update count, and the lost gain subtracts it from the Original gain for the same corpus, seed, and update count, in percentage points. The two gains share the filler-only denominator. No paired interval was estimated for the lost gain, and subtracting the two marginal intervals would not provide one, so lost gain is reported as a point estimate.

Paired document bootstrap (Efron, 1979). For each source cohort, a replicate resamples documents with replacement at the cohort's size, applies the same indices to the source-trained and filler-only models, and recomputes both macro perplexities and their relative reduction. Held-out gains are resampled in the same way within the held-out cohort. For the source–held-out contrast, source and held-out indices are drawn independently, each at its own cohort size, and the difference between the two gains is computed within each replicate, so the contrast interval comes from its own distribution rather than from the marginal endpoints. We use 2,000 replicates, random seed 20260616, and percentile endpoints at 2.5% and 97.5%. The intervals are conditional on the fitted checkpoints: they describe which documents happened to be sampled, not variation over retraining, and the two training seeds are reported separately rather than resampled. The intervals are pointwise; their overlap does not establish equivalence between conditions.

Reading the source–held-out contrast. Source and held-out gains each use the filler-only perplexity of their own documents as denominator. A positive contrast means only that the source gain is larger: it can accompany improvement on both groups, improvement on the sources with deterioration on the held-out documents, or deterioration on both. The component gains distinguish these cases, and Appendix D.2 shows them together.

Two filler-only models as a scale for small gains. The two seeds' filler-only models differ slightly in how well they predict the originals, which gives a scale against which to judge a small source gain. For a source cohort $\mathcal{S}$ and update count, let

$$B_{\mathcal{S}} = \frac{\left|\mathrm{PPL}_{\mathcal{S}}(\theta^{-}_{42}) - \mathrm{PPL}_{\mathcal{S}}(\theta^{-}_{137})\right|}{\mathrm{PPL}_{\mathcal{S}}(\theta^{-}_{42})},$$

resampled with the same document indices for both controls. Table 10 reports it. A source gain exceeds this difference when the lower endpoint of its interval lies strictly above the upper endpoint of $B_{\mathcal{S}}$ for the same corpus and update count. All 240 source gains of the training-length trajectories do so; the closest case is SAGE-R Llama on arXiv at seed 137 after 313 updates, with a lower endpoint of 0.30% against an upper endpoint of 0.13%. Two fitted controls do not describe the distribution of retraining effects, so this comparison gives a scale for the smallest gains rather than a significance test.

C.4 Adding a rewriting procedure or a detector

A new rewriting procedure enters through its derivatives. Each derivative is linked to the stable identifier of its original, and the procedure declares its generation rule, its prompt or candidate selection, and any change to prediction targets, as SAGE-R's marker masking does. The derivatives are trained with the recipe of Table 6, the shared filler pool, and the token-share planner of Appendix B.3, for the same seeds and update counts. Because the filler-only checkpoints do not depend on the rewriting condition, the existing six serve the new models, and the existing Original models supply the comparator. Evaluation uses the originals that the new procedure retains together with every condition it is compared with. If the procedure drops documents, the shared cohort shrinks, and the existing models' stored per-document scores can be restricted to it, as done for Pile-CC; their bootstrap intervals must then be recomputed.

A new detector needs no training. It scores the existing source-trained checkpoints on the shared source and held-out originals, declares its resources as an information condition, and names any reference model or auxiliary text it uses. Its AUROCs and operating points are then compared with the Original baseline under the same resources, and the modeling measurements, which do not depend on the detector, are reused unchanged.

D Joint outcomes along training

In §4.1, we summarize the training-length trajectories through five canonical conditions and the first and last checkpoints (Table 1). This appendix shows the full comparison: all ten evaluated conditions across generators and selection procedures (Table 11), every rewritten cell against its Original counterpart (Figure 6), the held-out gains with their intervals, and the target and filler-only perplexities from which the gains are formed. All displays use the shared queries at the 5% source-token quota and keep the two seeds separate.

Table 11: **Detection and original-text modeling for all ten evaluated conditions at the first and last checkpoints** (5% source-token quota). Each entry is the equal-weight mean over the eight corpus–seed cells, keeping generators and prompt-selection procedures separate; the averages describe these cells rather than a population effect. AUROC and strict TPR use Reference evaluated against the 60 held-out originals; strict TPR is the percentage of source originals scored above every held-out original (the attainable point under an empirical 1% FPR bound). Gain is the reduction in source-document macro-perplexity relative to the equal-update, seed-matched filler-only model; lost gain subtracts it from the Original gain for the same corpus and seed. Selected: surrogate-selected prompt. Figure 6 compares every rewritten cell with its Original counterpart.

		313 updates				1,252 updates			
Rewriting	Generator	AUROC	TPR (%)	Gain (%)	Lost (pp)	AUROC	TPR (%)	Gain (%)	Lost (pp)
Original	—	0.983	86.1	5.89	—	0.996	80.9	30.04	—
Lexical	Qwen	0.878	37.3	3.11	2.78	0.950	52.4	19.48	10.56
	Llama	0.878	40.9	2.75	3.14	0.945	46.0	17.34	12.70
	Qwen, selected	0.859	35.8	2.85	3.05	0.938	46.4	18.48	11.56
Syntactic	Qwen	0.907	44.5	3.61	2.28	0.971	55.2	21.55	8.49
	Qwen, selected	0.904	48.6	3.59	2.30	0.971	55.5	21.45	8.59
SAGE	Qwen	0.812	31.3	1.89	4.00	0.882	36.4	15.53	14.51
	Llama	0.819	22.1	1.90	3.99	0.901	35.2	16.09	13.95
SAGE-R	Qwen	0.662	8.9	0.76	5.13	0.764	19.2	8.53	21.51
	Llama	0.663	11.3	0.72	5.17	0.788	19.3	7.98	22.06

D.1 Every rewriting condition against Original

Figure 6 places each of the 216 rewritten cells (nine conditions, four corpora, two seeds, three checkpoints) against the Original model with the same corpus, seed, and update count. The left column gives the difference in Reference AUROC and the right column the lost gain, so both

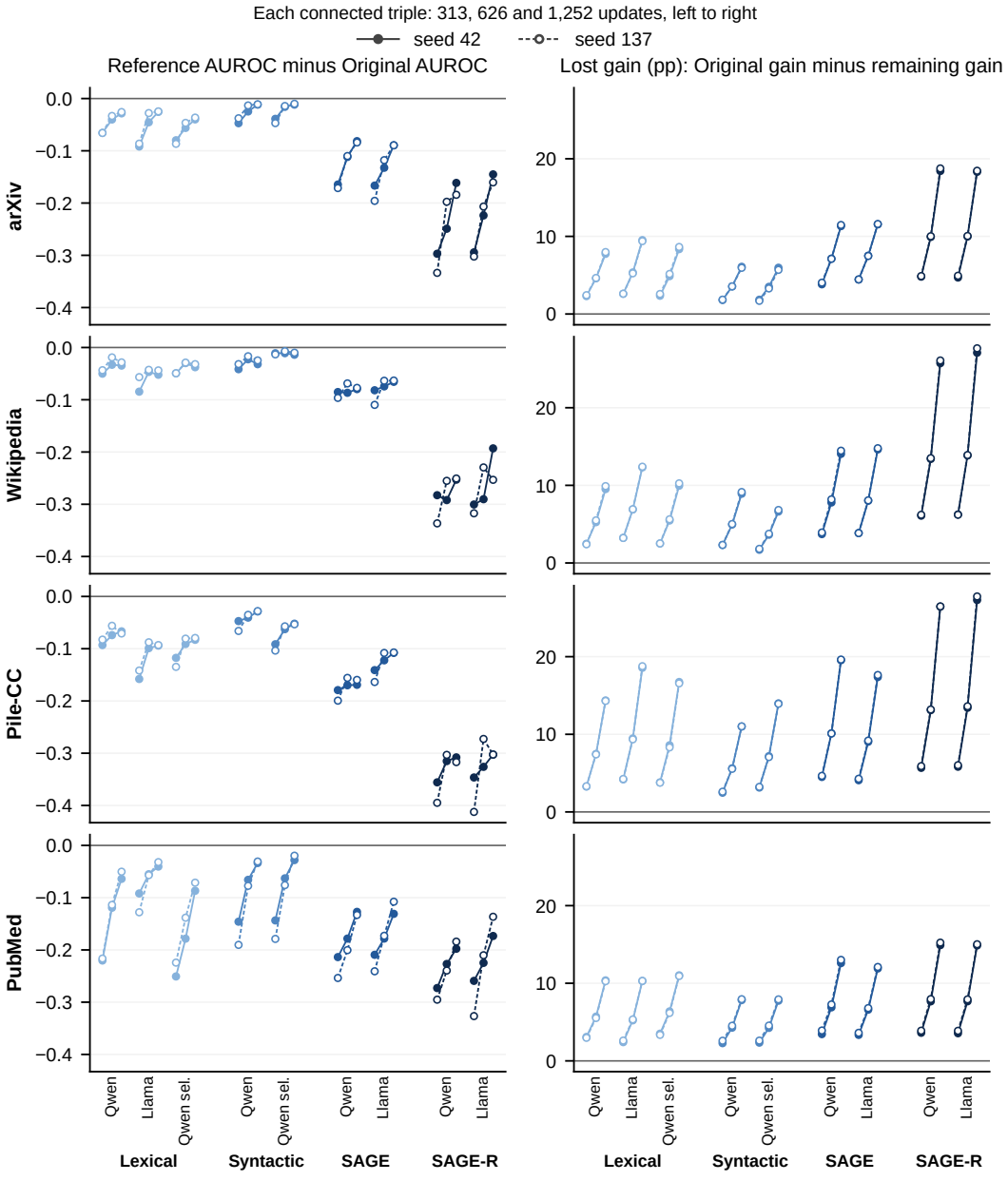

Figure 6: **Every rewritten cell against its Original comparator.** Left: Reference AUROC of the rewritten model minus that of the Original model with the same corpus, seed, and update count. Right: lost gain, the Original source gain minus the rewritten source gain, in percentage points. Within each condition, the three connected points are the checkpoints after 313, 626, and 1,252 updates (5% source quota); solid lines with filled markers show seed 42 and dashed lines with open markers seed 137. Labels under the axis give the generator; sel. marks a surrogate-selected prompt. Points are estimates for fitted checkpoints; the lost gain has no interval (Appendix C.3).

comparisons of Figure 2 appear side by side for every condition. Every AUROC difference is negative, ranging from -0.412 to -0.007, and every lost gain is positive, ranging from 1.70 to 27.76 percentage points.

The two comparisons move in opposite directions. Along training the rewritten models close part of their detection gap while falling further behind in modeling. Between 313 and 1,252 updates the AUROC difference narrows in 71 of the 72 rewritten trajectories, whereas the lost gain widens at both

transitions in all 72. For SAGE-R Qwen, the mean difference over corpora and seeds shrinks from -0.321 to -0.232 while the mean lost gain grows from 5.13 to 21.51 points. Because the Original AUROC is already near 1 after 313 updates (0.983 on average), the narrowing comes mostly from the rising AUROC of the rewritten models. The one exception is Selected syntactic Qwen on Wikipedia at seed 42, whose difference is already small and moves from -0.011 to -0.014.

Rewriting families differ consistently, while generator and selection effects vary. SAGE-R has the lowest Reference AUROC of all nine conditions in each of the 24 corpus–seed–checkpoint groups and the largest lost gain in 23; in the remaining group, PubMed at seed 137 after 313 updates, SAGE Qwen loses 3.91 points against 3.89 and 3.87 for the two SAGE-R conditions. SAGE is less detectable than every lexical and syntactic condition in 20 of the 24 groups and loses more gain than all of them in 18; five of the six exceptions are on Pile-CC, where Lexical Llama loses more gain than SAGE Llama. Within the lexical and syntactic family, Syntactic Qwen is more detectable and loses less gain than Lexical Qwen in all 24 groups. The generator and the selection rule shift these outcomes less consistently. Surrogate selection makes Lexical Qwen lose more gain in all 24 groups and lowers its AUROC in 22, but it has no consistent direction for Syntactic Qwen (more lost gain in 10 of 24 groups). Llama loses more gain than Qwen in 19 of 24 groups for lexical paraphrasing, by up to 4.47 points, but in only 10 for SAGE and 16 for SAGE-R. Table 1 therefore keeps generators and selection procedures as separate rows rather than averaging them into families.

D.2 Held-out originals

A modeling gain on the source originals could reflect learning about the sources themselves or a broader improvement on the corpus. The held-out originals, never injected, separate the two: their gain applies the same definition with the filler-only perplexity on the held-out documents as denominator. Figure 7 shows both gains with their per-seed document-bootstrap intervals at the first and last checkpoints.

Early in training, held-out prediction can fall behind filler-only training. After 313 updates, 19 of the 80 held-out gains are negative: all eight SAGE and SAGE-R cells on arXiv and eleven rewritten cells on PubMed. For twelve of them the whole interval lies below zero, seven on arXiv (both SAGE conditions in both seeds, SAGE-R Qwen at seed 42, SAGE-R Llama in both seeds) and five on PubMed (Selected lexical Qwen at seed 42 and both SAGE conditions in both seeds). In these cells, training on the derivatives left the held-out originals slightly harder to predict than training on filler alone, by up to 1.56% of the filler-only perplexity, while the same models predicted their source originals better than the filler-only model did. After 626 updates a single held-out gain remains negative (SAGE Llama on arXiv at seed 137, -0.05%, interval $[-0.59\%, 0.51\%]$), and after 1,252 updates every held-out interval lies above zero, with point estimates from 3.43% to 12.45%. Original training never produces a negative held-out gain.

The sources benefit more than the held-out documents throughout. In all 240 cells the interval for the source-minus-held-out contrast, computed within each bootstrap replicate, lies above zero. The smallest lower endpoint is 0.15 points, for SAGE-R Llama on Pile-CC at seed 137 after 313 updates. The source gains themselves have all 240 lower endpoints above zero, the smallest at 0.30%, and their intervals are narrow compared with those of the held-out gains, which rest on 60 documents.

D.3 Target and filler-only components

Each gain is one minus the ratio of the target's macro-perplexity to the filler-only model's, so it can change because either model changes. Figure 8 shows the two components directly, for every trajectory and both groups of originals.

The filler-only comparator drifts upward. Continued training on the filler pool raises the filler-only model's perplexity on the originals at both transitions in all eight corpus–seed groups. Between 313 and 1,252 updates the increase is 15.0% to 31.6% on the sources and 14.0% to 32.2% on the held-out documents; on Pile-CC at seed 42, source perplexity rises from 17.46 to 22.98. Original training moves the other way on its sources, lowering source perplexity by 5.9% to 11.7% over the same interval.

Rewritten models mostly worsen in absolute terms while their gains grow. The rewritten models' source perplexity rises between 313 and 1,252 updates in 62 of 72 trajectories, by up to 19.8%; the ten

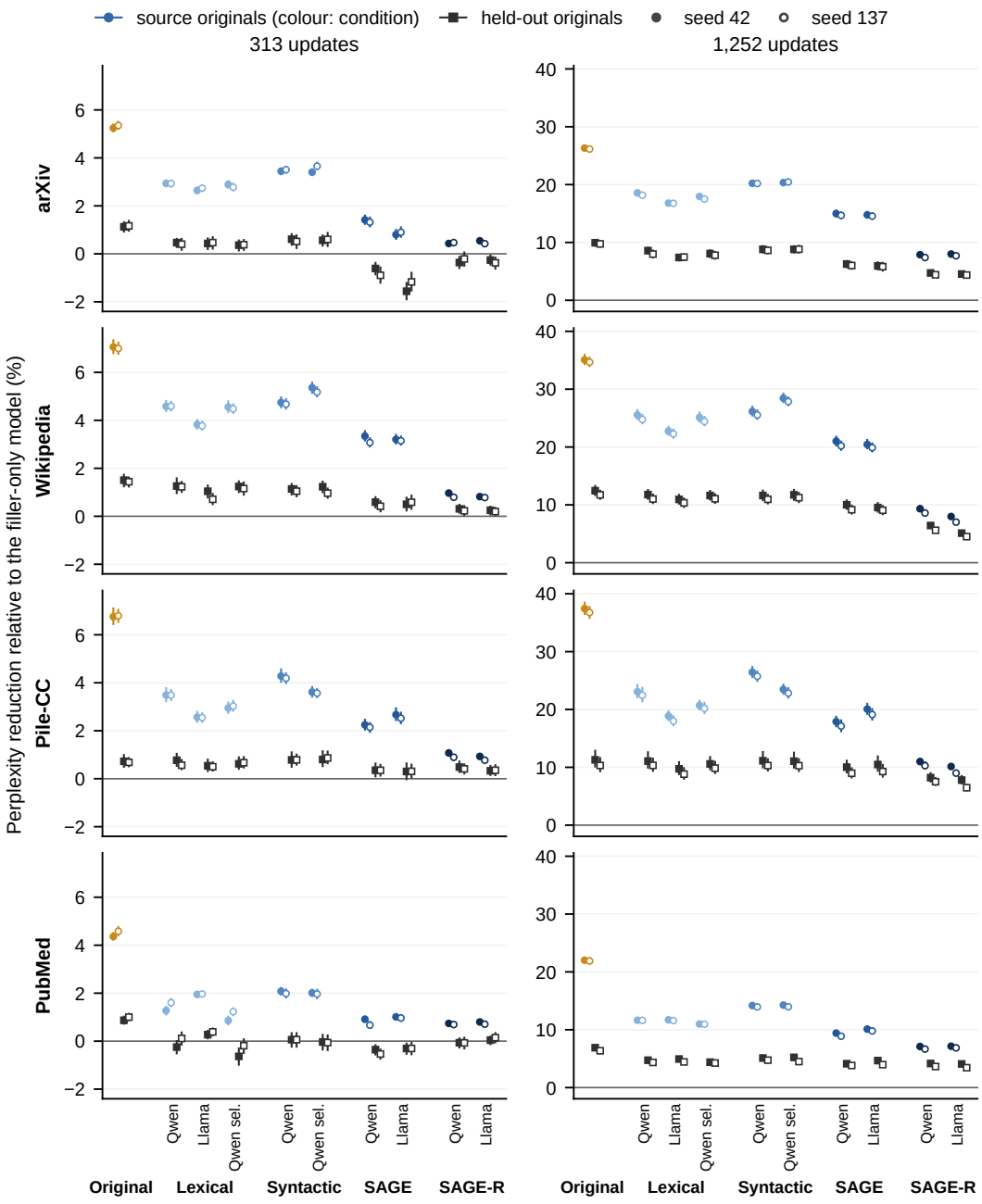

Figure 7: **Source and held-out gains with their uncertainty.** Reduction in document-macro perplexity relative to the equal-update, seed-matched filler-only model, on the shared source originals (circles, coloured by condition) and on the 60 held-out originals (black squares), after 313 and 1,252 updates at a 5% source quota. Vertical lines are 95% paired document-bootstrap intervals conditional on each fitted checkpoint; filled markers show seed 42 and open markers seed 137. Each column shares one vertical scale across corpora.

exceptions are Syntactic and Selected syntactic Qwen on arXiv and Wikipedia and Lexical Qwen on Wikipedia, whose perplexity falls by at most 3.6%. On the held-out originals every model, Original included, ends with higher perplexity than after 313 updates. In all 80 trajectories and for both groups of originals, however, the target's perplexity rises proportionally less than the filler-only model's, so every gain grows. The growing gain thus measures a widening advantage over equal-update filler-only

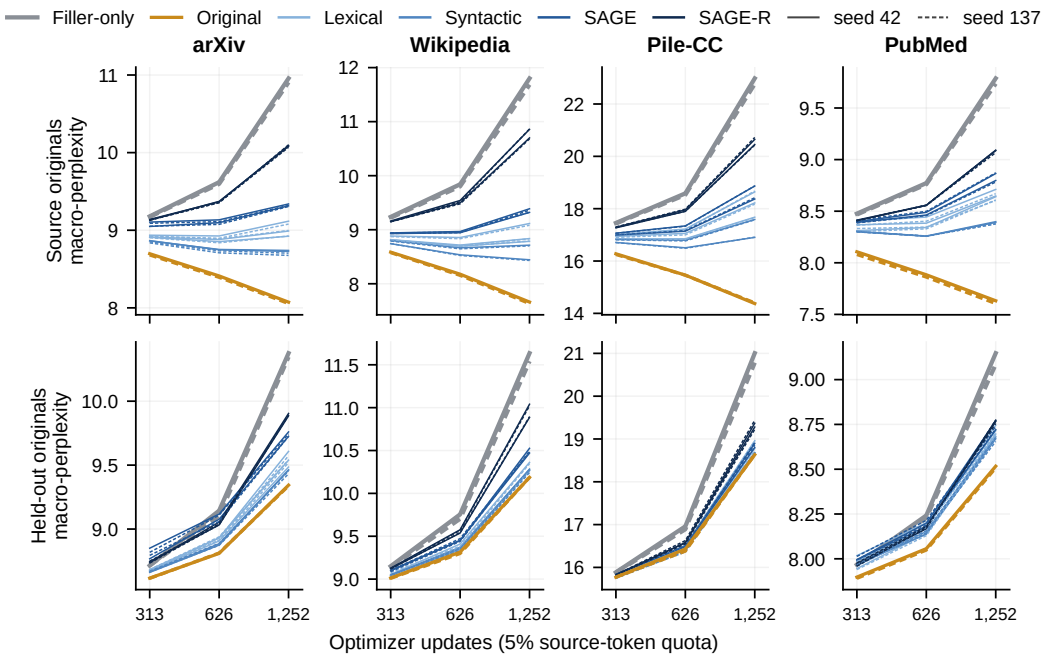

Figure 8: **The two components of every gain.** Document-macro perplexity on the shared source originals (top) and the held-out originals (bottom) along the training-length trajectories. Grey: the filler-only model, which is shared by all conditions within a corpus, seed, and update count. Amber: Original training. Blue: the nine rewriting conditions, coloured by family, with generators and selection procedures drawn as separate thin lines. Solid lines show seed 42 and dashed lines seed 137; the vertical scale differs between panels.

training, which for most rewritten models coexists with worse absolute prediction of the originals than after 313 updates.

E Detector coverage, strict operating points, and source quota

The main text fixes four detector variants and follows them along both training axes. This appendix reports all ten evaluated variants, compares AUROC with the attainable true-positive rates at strict thresholds, and gives the uncertainty attached to the changes in detection along training and across source quotas. The five training settings are the three checkpoints of the training-length trajectories and the 30% and 50% quotas at 1,252 updates; the 1,252-update, 5% models belong to both axes and are counted once.

E.1 All ten detector variants

Table 12 reports the ten variants grouped by the resources they require. Every target-only variant shows the dependence on training seen for Loss and Min-K++ 20% in Figure 5. After 313 updates at the 5% quota, the Original means of the eight target-only variants lie between 0.502 (Zlib) and 0.595 (Min-K++ 10%), so none of them has much to lose to rewriting. At the 30% quota the same means lie between 0.940 and 0.995, and at 50% between 0.997 and 1.000, while the rewritten means stay between 0.678 and 0.779. Within a family, the choice of k changes the means by at most 0.014 for Min-K and 0.047 for Min-K++. At the 5% quota Min-K++ is stronger than Min-K, Loss, and Zlib for both Original and rewritten training, and Zlib tracks Loss closely throughout.

ReCaLL, which additionally uses held-out text as a prefix, follows a similar training-dependent pattern: its Original mean rises from 0.579 to 1.000 across the settings, and at the 50% quota its rewritten mean of 0.825 is the highest of all variants other than Reference. Its prefix can shorten the scored part of each query (Appendix C.1), so these values describe this implementation. Reference,

Table 12: AUROC of all ten evaluated detector variants, grouped by the resources they require. Each entry is the equal-weight mean of separately evaluated cells: Orig. over the 8 corpus–seed cells of Original training, Rew. over the 72 corpus–seed–condition cells of the nine rewriting conditions. Scores are evaluated on the cohort retained for each model's training, which equals the shared cohort except for eight Pile-CC conditions (465 rather than 464 sources). The 1,252-update, 5% models are counted once.

	5% source quota						1,252 updates			
	313 updates		626 updates		1,252 updates		30% quota		50% quota	
Detector	Orig.	Rew.	Orig.	Rew.	Orig.	Rew.	Orig.	Rew.	Orig.	Rew.
Target model only										
Loss	0.507	0.489	0.550	0.508	0.627	0.539	0.947	0.693	0.997	0.745
Zlib	0.502	0.486	0.542	0.502	0.616	0.532	0.940	0.678	0.997	0.729
Min-K 10%	0.530	0.506	0.581	0.527	0.674	0.562	0.985	0.724	1.000	0.769
Min-K 20%	0.525	0.503	0.577	0.525	0.668	0.560	0.982	0.727	0.999	0.775
Min-K 30%	0.520	0.499	0.571	0.521	0.660	0.556	0.976	0.722	0.999	0.771
Min-K++ 10%	0.595	0.549	0.685	0.586	0.803	0.633	0.995	0.743	0.998	0.730
Min-K++ 20%	0.588	0.541	0.681	0.578	0.807	0.632	0.995	0.771	0.998	0.763
Min-K++ 30%	0.583	0.535	0.676	0.572	0.802	0.627	0.995	0.779	0.998	0.777
Auxiliary text										
ReCaLL	0.579	0.545	0.648	0.574	0.757	0.616	0.993	0.792	1.000	0.825
Additional model										
Reference	0.983	0.820	0.997	0.881	0.996	0.901	1.000	0.965	1.000	0.952

which uses the equal-update filler-only model, has the highest means in every setting: 0.983 to 1.000 for Original and 0.820 to 0.965 for rewritten training. Its advantage over the other variants remains large after rewriting, but nearly vanishes under Original training at the highest quota. For the six additional variants, a zero-false-positive rate saved by an earlier calculator is kept with the per-cell records; the operating points below use the full-vertex estimator for the four fixed variants.

E.2 Strict operating points

Table 13 places AUROC beside the attainable true-positive rates at zero, three, and six false positives among the 60 held-out originals. The strict threshold separates audits that look alike by AUROC. Under Original training at the 30% quota, Loss reaches a mean AUROC of 0.947 but detects only 38.0% of the sources above every held-out original, whereas Min-K++ 20% reaches 0.995 and 85.6%. Reference under Original training detects fewer sources at zero false positives after 1,252 updates (80.9%) than after 626 (87.3%), although its AUROC barely changes; allowing three false positives raises both to above 99%. Rewritten training keeps Reference well below these rates at every setting, for example 40.6% at zero false positives after 1,252 updates, rising to 77.2% when six false positives are allowed.

High AUROC leaves the strict rate largely undetermined. Figure 9(a) plots the zero-false-positive rate against Reference AUROC for all 400 matched cells. Among the 198 cells with AUROC of at least 0.95, the rate ranges from 5.0% to 100%; the lowest is Syntactic Qwen on Pile-CC at seed 42 after 1,252 updates, with AUROC 0.957, which detects 81.0% of its sources once three false positives are allowed. The zero-false-positive threshold is set by the single highest-scoring held-out original of each model, so one negative can decide it.

Reversals concentrate at the strictest threshold. Section 4.3 counts the pairs of rewritten conditions, within one corpus and seed, that Reference AUROC and the zero-false-positive rate order differently: 38 of 288 pairs after 1,252 updates. Comparing AUROC with the rates at three or six false positives instead gives fewer reversed pairs: 12 and 13 after 1,252 updates. After 313 updates the counts at zero, three, and six false positives are 42, 28, and 16, and after 626 updates 36, 14, and 11. At the 30% and 50% quotas they are 9, 7, and 5 and 11, 8, and 5. Panels (b) and (c) of Figure 9 show how a reversal arises. On Pile-CC, SAGE Llama has the higher AUROC in both seeds, but SAGE Qwen places more sources above every negative. With one false positive allowed, SAGE Llama already detects more at seed 42 (27.6% against 22.8%), and with two it does so at both seeds. The same

Table 13: AUROC and attainable true-positive rates of the four fixed detector variants on the shared queries. TPR columns give the percentage of source originals detected when at most 0, 3 or 6 of the 60 held-out originals score at or above the threshold (FP $\leq$ 0, 3, 6; empirical FPR bounds of 1%, 5% and 10%), read from the full empirical ROC. Entries are equal-weight means over the 8 Original cells (corpus–seed) and the 72 rewritten cells (corpus–seed–condition, nine conditions) of each setting.

	Original training				Rewritten training			
		TPR (%), FP $\leq$				TPR (%), FP $\leq$		
Detector and setting	AUROC	0	3	6	AUROC	0	3	6
Reference								
313 updates, 5%	0.983	86.1	92.9	96.0	0.820	31.2	52.7	60.9
626 updates, 5%	0.997	87.3	99.5	99.8	0.881	39.6	62.8	72.0
1,252 updates, 5%	0.996	80.9	99.6	99.8	0.901	40.6	67.3	77.2
1,252 updates, 30%	1.000	100.0	100.0	100.0	0.965	81.2	88.6	91.7
1,252 updates, 50%	1.000	100.0	100.0	100.0	0.952	77.7	85.8	88.2
Loss								
313 updates, 5%	0.507	1.5	5.9	8.5	0.489	1.3	5.3	7.8
626 updates, 5%	0.550	2.1	7.4	10.3	0.508	1.6	5.9	8.3
1,252 updates, 5%	0.627	3.4	10.1	15.1	0.539	2.0	6.6	9.7
1,252 updates, 30%	0.947	38.0	70.1	82.3	0.693	8.1	18.2	25.8
1,252 updates, 50%	0.997	88.7	99.1	99.8	0.745	13.2	27.9	36.5
Min-K 20%								
313 updates, 5%	0.525	1.6	6.3	10.5	0.503	1.4	5.4	9.2
626 updates, 5%	0.577	2.7	8.7	13.3	0.525	1.7	6.7	9.8
1,252 updates, 5%	0.668	5.8	14.3	21.6	0.560	2.8	8.2	11.4
1,252 updates, 30%	0.982	65.8	90.6	95.1	0.727	12.8	25.9	34.0
1,252 updates, 50%	0.999	98.0	99.9	99.9	0.775	19.9	37.9	46.6
Min-K++ 20%								
313 updates, 5%	0.588	1.6	10.4	16.7	0.541	1.4	7.9	13.5
626 updates, 5%	0.681	3.7	20.4	26.6	0.578	2.2	11.8	17.3
1,252 updates, 5%	0.807	9.5	39.5	49.3	0.632	3.9	17.3	23.9
1,252 updates, 30%	0.995	85.6	99.7	99.7	0.771	18.1	42.8	52.7
1,252 updates, 50%	0.998	99.1	99.8	99.8	0.763	23.9	42.3	51.2

panels show Original training at seed 42 rising from 26.5% to 88.1% of its sources when a single false positive is allowed.

E.3 Source quota and the uncertainty of detection changes

The quota comparison raises the source share at a fixed 1,252 updates, which also displaces filler and reduces total input (Appendix B.4), so it describes a combined change in training. The matched point estimates in Section 4.2 show every one of the 80 cells higher at both 30% and 50% than at 5%, with 57 of the 72 rewritten cells lower at 50% than at 30%. The saved intervals attach uncertainty to the first of these patterns but not to the second: they compare each higher quota with 5%, and no interval was saved for the difference between 30% and 50%, so the non-monotonic pattern rests on point estimates.

Figure 10 shows the saved intervals for the change in Reference AUROC from the 5% quota. They were computed on the retained cohorts, with 2,000 row-bootstrap replicates that resample documents for fixed checkpoints, and are not intervals for the matched AUROCs. All 160 point estimates are positive. For the rewritten models the interval lies above zero in 71 of 72 cells at 30% and 59 of 72 at 50%, and every rewritten interval that includes zero belongs to SAGE-R. For Original training the change is small because the AUROC is already near 1 at 5%, and its interval includes zero in three cells at 30% and four at 50%.

Table 14 summarizes all saved intervals for detection changes, including the change in each condition's AUROC gap to its comparator. At the higher quotas the gap between each lexical, syntactic, or SAGE condition and Original narrows with an interval above zero in all 56 cells at both quotas, whereas the comparison between the two SAGE-R generators includes zero in 15 of 16 cells. Along training, the

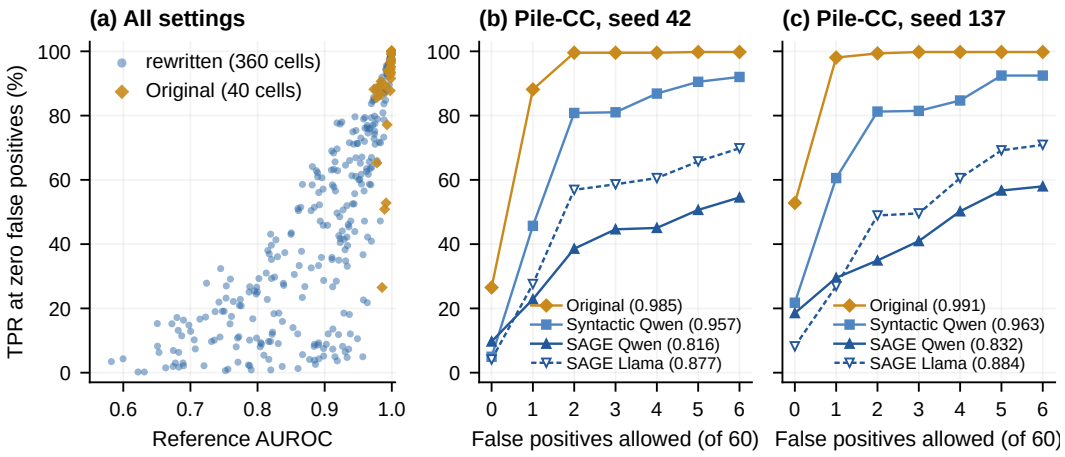

Figure 9: **AUROC and strict operating points of Reference.** (a) Percentage of sources scored above every held-out original against AUROC, for all 400 matched Reference cells: ten conditions, four corpora, two seeds, and five training settings. (b, c) The Pile-CC example discussed in Section 4.3, after 1,252 updates at a 5% quota: the largest attainable TPR when at most $0, 1, \ldots, 6$ of the 60 held-out originals may score at or above the threshold, read from the full empirical ROC; the legend gives each model's AUROC. Lines between integer false-positive counts are visual guides. This example is shown because the main text discusses it, not as a typical cell.

Table 14: Saved checkpoint-conditional 95% row-bootstrap intervals for detection changes, classified by whether they lie above zero, include zero or lie below zero. All intervals come from the retained-cohort analyses (2,000 resamples; seed 20260916 for the training-length gaps, 20260918 for the quota comparisons) and describe document resampling for fixed checkpoints. The AUROC gap is the condition's AUROC minus its comparator's at the same setting.

Comparison	Setting	Intervals	Above 0	Include 0	Below 0
Reference AUROC at the higher quota minus at 5%					
Original	30%	8	5	3	0
Original	50%	8	4	4	0
Nine rewriting conditions	30%	72	71	1	0
Nine rewriting conditions	50%	72	59	13	0
Change in the AUROC gap from the 5% quota					
Lexical, syntactic, SAGE vs. Original	30%	56	56	0	0
Lexical, syntactic, SAGE vs. Original	50%	56	56	0	0
SAGE-R Qwen vs. SAGE-R Llama	30%	8	1	7	0
SAGE-R Qwen vs. SAGE-R Llama	50%	8	0	8	0
Change in the AUROC gap from 313 updates					
Lexical, syntactic, SAGE vs. Original	626	56	40	16	0
Lexical, syntactic, SAGE vs. Original	1,252	56	38	18	0
SAGE-R Qwen vs. SAGE-R Llama	626	8	0	7	1
SAGE-R Qwen vs. SAGE-R Llama	1,252	8	0	6	2

same gap to Original narrows with an interval above zero in 40 of 56 cells by 626 updates and 38 of 56 by 1,252 updates, and no interval lies below zero. On these retained-cohort intervals, about two-thirds of the narrowing described in Appendix D.1 for the lexical, syntactic, and SAGE conditions therefore exceeds document-resampling variation. For SAGE-R Qwen against SAGE-R Llama, three of the sixteen training-length intervals lie below zero and the rest include it.

F Supplementary scale and architecture studies

Two studies from an earlier stage of benchmark development extend the evidence beyond Pythia-1.4B: a budget sweep at three Pythia scales and a learning-rate comparison on GPT-Neo-2.7B. Their design

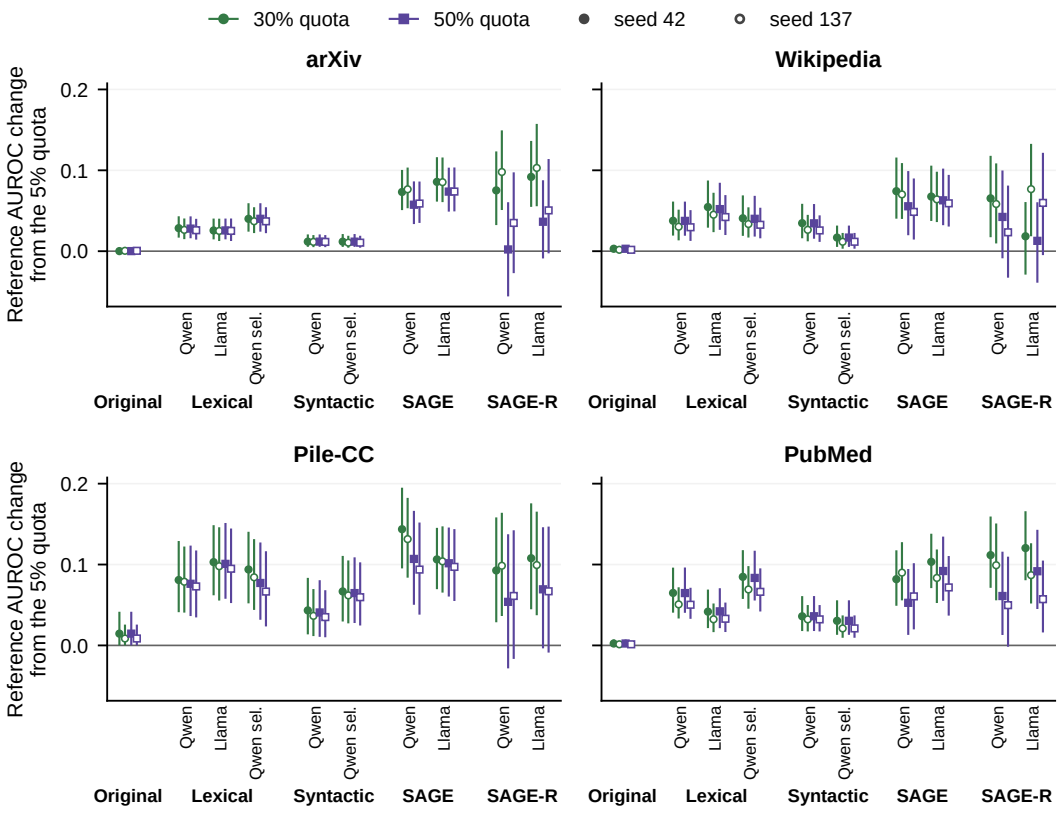

Figure 10: **Change in Reference AUROC from the 5% to the 30% and 50% source quotas** at 1,252 updates, with saved checkpoint-conditional 95% row-bootstrap intervals (2,000 replicates). Filled markers show seed 42 and open markers seed 137. The intervals are computed on each model's retained training cohort (465 rather than 464 Pile-CC sources for the eight conditions other than SAGE-R) and describe document resampling for fixed checkpoints.

differs from the primary study in two measurements. Reference uses the pretrained checkpoint of the same architecture as its reference model instead of an equal-update filler-only model, and the modeling measurement is *derivative uptake*, the relative perplexity reduction on the rewritten training texts against that pretrained checkpoint, instead of the source gain on the originals against filler-only training. Neither study trains Original or filler-only models. They therefore show how detection on the original queries and uptake of the training text respond to model scale, update budget, and learning rate, but they do not give remaining or lost gains. Both evaluate 500 source originals and 60 held-out originals per corpus and seed, and both use seeds 42 and 137, 32 sequences per update, and a nominal 5% source-token share. Figure 11 collects both studies; Tables 15 and 16 give the values.

F.1 Pythia scale and update budget

The budget study trains Lexical Qwen derivatives of arXiv into Pythia-1.4B, Pythia-2.8B, and Pythia-6.9B for 313, 626, and 1,252 updates with learning rate 10^{-5}. Detection and uptake rise together with the budget at the two larger scales: Reference AUROC increases from 0.747 to 0.863 and 0.905 for Pythia-2.8B and from 0.768 to 0.889 and 0.919 for Pythia-6.9B, while uptake rises from 18.3% to 29.8% and from 21.2% to 36.4%. At 626 and 1,252 updates the ordering follows scale for both quantities, with Pythia-1.4B at 0.762 and 0.841 and uptake of 21.8% and 25.4%. Target-only detection stays much weaker. Selecting the variant with the largest AUROC within each seed yields two-seed means no higher than 0.745 in the later sweep. These means do not follow scale, Pythia-2.8B being the lowest at every budget. Because this series takes a maximum over variants before averaging the seeds, it is an upper envelope of fixed target-only detectors rather than the result of any one of them;

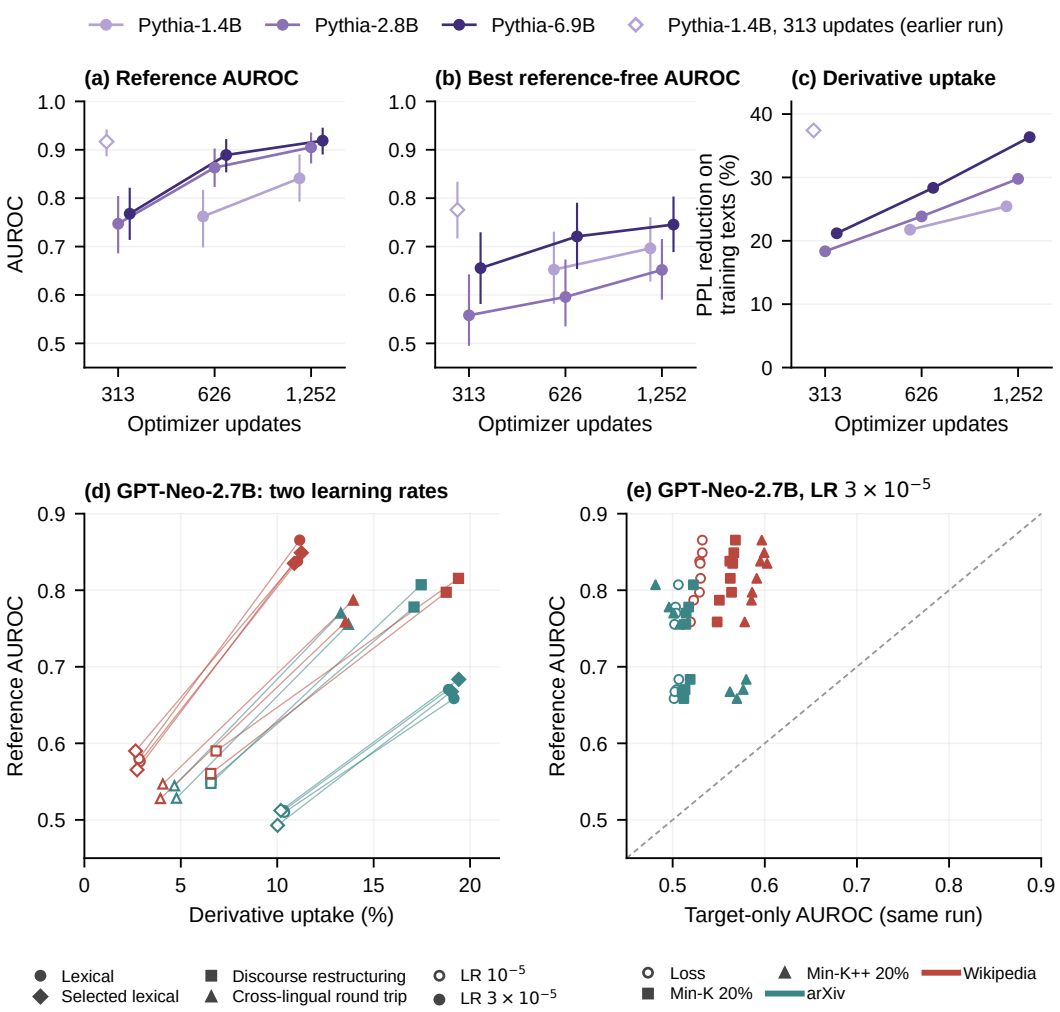

Figure 11: **Supplementary scale and architecture studies,** both with a pretrained reference model and without Original or filler-only training. (a–c) Lexical Qwen on arXiv at three Pythia scales after 313, 626, and 1,252 updates: Reference AUROC, the best target-only AUROC selected within each seed, and uptake of the training texts. Points are two-seed means; vertical lines are pointwise 95% document-bootstrap intervals. The open diamond is the earlier Pythia-1.4B run at 313 updates, which is not joined to the later sweep. (d) GPT-Neo-2.7B after 313 updates: each line joins the same procedure, corpus, and seed at learning rates 10^{-5} (open) and 3×10^{-5} (filled); colour gives the corpus. (e) The 16 higher-rate GPT-Neo runs: Reference AUROC against the AUROC of three fixed target-only variants of the same run; the dashed line marks equality.

the selected variant is Min-K++ 30% everywhere except Pythia-1.4B at 626 updates, where both seeds select Min-K++ 20%.

The Pythia-1.4B point at 313 updates comes from an earlier execution whose detector scores and checkpoint diagnostics could not be fully matched to the later sweep. Its uptake (37.4%), Reference AUROC (0.917), and best target-only AUROC (0.776) exceed those of the later 626- and 1,252-update runs, so we show it as a separate point and do not read the 1.4B series as one trajectory. The Reference AUROCs of this study are not comparable with those of the primary study, whose reference model is continued-trained on filler for the same number of updates as the target.

Table 15: Lexical Qwen on arXiv at three Pythia scales and three update budgets (learning rate 10^{-5}, 32 sequences per update, nominal 5% source exposure). Uptake is the perplexity reduction on the rewritten training texts relative to the pretrained model, averaged over seeds 42 and 137. AUROCs are two-seed means with pointwise 95% intervals from 1,000 class-stratified document-bootstrap replicates that apply the same resampled documents to both seeds. Reference uses the pretrained model of the same architecture; the reference-free column takes, within each seed, the variant with the largest AUROC. †Earlier execution, not joined to the later budget sweep.

Model	Updates	Uptake (%)	Reference AUROC	Best reference-free AUROC	Selected variant
Pythia-1.4B†	313	37.4	0.917 [0.889, 0.940]	0.776 [0.719, 0.832]	Min-K++ 30%
Pythia-1.4B	626	21.8	0.762 [0.700, 0.815]	0.652 [0.584, 0.729]	Min-K++ 20%
Pythia-1.4B	1,252	25.4	0.841 [0.795, 0.888]	0.696 [0.630, 0.758]	Min-K++ 30%
Pythia-2.8B	313	18.3	0.747 [0.688, 0.802]	0.558 [0.497, 0.640]	Min-K++ 30%
Pythia-2.8B	626	23.8	0.863 [0.825, 0.900]	0.596 [0.537, 0.671]	Min-K++ 30%
Pythia-2.8B	1,252	29.8	0.905 [0.874, 0.933]	0.652 [0.592, 0.713]	Min-K++ 30%
Pythia-6.9B	313	21.2	0.768 [0.716, 0.819]	0.655 [0.584, 0.727]	Min-K++ 30%
Pythia-6.9B	626	28.4	0.889 [0.856, 0.920]	0.721 [0.656, 0.788]	Min-K++ 30%
Pythia-6.9B	1,252	36.4	0.919 [0.892, 0.943]	0.745 [0.691, 0.801]	Min-K++ 30%

F.2 GPT-Neo-2.7B learning rates

The architecture study trains GPT-Neo-2.7B for 313 updates on four Qwen procedures, Lexical, Selected lexical, Discourse restructuring, and Cross-lingual round trip (Appendix A.1), on arXiv and Wikipedia, at learning rates 10^{-5} and 3×10^{-5}. The two rates come from separate executions with different training-code commits; the runs share their recipe settings, but identical data ordering between the two rates was not established.

Higher-rate runs show greater uptake and detection. Uptake increases in all 16 seed-matched pairs, from 2.7%–10.4% (mean 6.0%) to 10.9%–19.4% (mean 15.5%), and the two-seed Reference AUROC increases for all eight procedure–corpus pairs, from 0.503–0.578 to 0.664–0.852. At the lower rate every run stays close to chance. The runs also differ beyond the injected texts: perplexity on held-out Pile text falls by 0.9% to 1.1% relative to the pretrained model at the lower rate but rises by 1.2% to 2.0% at the higher rate.

Uptake and detection diverge across corpora. Under the higher rate, the lexical and selected-lexical procedures reach about 19% uptake on arXiv but only 11% on Wikipedia, yet their Reference AUROC is 0.66–0.68 on arXiv and 0.84–0.85 on Wikipedia (Figure 11d). The primary study found that detection does not follow the modeling gain along training; here, with different measures, larger uptake of the training text likewise does not come with stronger detection on the originals.

Target-only audits remain near chance. In the 16 higher-rate runs every saved target-only variant, and ReCaLL, stays at or below an AUROC of 0.603: Loss ranges from 0.502 to 0.532, Min-K 20% from 0.510 to 0.568, and Min-K++ 20% from 0.481 to 0.603. Reference, at 0.658 to 0.865, exceeds all nine of these variants in every run while using the additional pretrained model. Target-only scores were not saved for the lower-rate runs.

G Broader related work

G.1 Training-data auditing and the unit of inference

Membership inference tests whether a sample was used to train a model by examining differences in the model's behavior on seen and unseen examples (Shokri et al., 2017; Carlini et al., 2022). For language models, Loss and token-level scores such as Min-K and Min-K++ use the target's predictive probabilities (Yeom et al., 2018; Shi et al., 2024; Zhang et al., 2025a), and reference calibration accounts for differences in sample difficulty by comparing the target with an additional model (Watson et al., 2022; Carlini et al., 2022). For masked language models, likelihood-ratio attacks likewise use an additional reference model rather than the target score alone (Mireshghallah et al., 2022). Wang et al. (2025) further contrasts member and non-member contexts, illustrating how additional context

Table 16: GPT-Neo-2.7B runs at learning rates 10^{-5} (L) and 3×10^{-5} (H), 313 updates, 32 sequences per update and nominal 5% source exposure; all four procedures use Qwen3-32B. Uptake is the perplexity reduction on the rewritten training texts relative to the pretrained model. Reference subtracts the target loss from the pretrained model's loss. Held-out change is the relative change in perplexity on held-out Pile text against the pretrained model, so negative values mean improvement.

			Uptake (%)		Reference AUROC		Held-out change (%)	
Procedure	Corpus	Seed	L	H	L	H	L	H
Lexical	arXiv	42	10.3	18.9	0.512	0.670	−0.97	+1.83
		137	10.4	19.2	0.510	0.658	−0.96	+1.75
	Wikipedia	42	2.9	11.1	0.576	0.838	−0.97	+1.30
		137	2.8	11.2	0.580	0.865	−1.10	+1.35
Selected lexical	arXiv	42	10.0	19.0	0.493	0.667	−0.99	+1.63
		137	10.2	19.4	0.512	0.684	−0.96	+1.69
	Wikipedia	42	2.7	10.9	0.590	0.835	−1.04	+1.22
		137	2.7	11.3	0.565	0.849	−1.08	+1.38
Discourse restructuring	arXiv	42	6.6	17.1	0.552	0.778	−1.01	+1.93
		137	6.6	17.5	0.548	0.807	−0.97	+1.69
	Wikipedia	42	6.6	18.8	0.561	0.797	−0.92	+1.96
		137	6.8	19.4	0.590	0.816	−0.89	+1.71
Cross-lingual round trip	arXiv	42	4.7	13.3	0.544	0.770	−1.01	+1.66
		137	4.8	13.7	0.528	0.755	−1.01	+1.65
	Wikipedia	42	3.9	13.5	0.528	0.759	−1.07	+1.35
		137	4.1	13.9	0.547	0.787	−0.97	+1.30

resources change the inference setup. These approaches differ in the resources they require as well as in their scores. **LaunderBench** records those requirements as information conditions $\mathcal{I}$ and evaluates Reference with a seed-matched filler-only checkpoint, which is one controlled choice of reference model rather than one an external auditor can be assumed to have.

Difference between LaunderBench and MIMIR. MIMIR studies membership inference for language models and shows how sensitive its evaluation is to the construction of members and non-members (Duan et al., 2024); distribution differences between nominal members and non-members can also support strong text-only baselines, confounding the interpretation of membership scores (Meeus et al., 2025; Das et al., 2025). Therefore, the MIMIR's non-member splits supply both the source and the held-out originals used in **LaunderBench**. Building upon this, **LaunderBench** adds a controlled continued-training intervention, so that a positive label means that a source's original or derivative entered that training stage. Under rewriting, exposure of the source and verbatim inclusion of the audit query become different events, and the benchmark asks whether querying the original reveals the former.

Difference between LaunderBench and other copyright-auditing setup. Dataset inference aggregates signals across a collection to decide whether the collection was used in training (Maini et al., 2024), **LaunderBench** instead reports document-level detection on original queries and pairs it with modeling comparisons on the same documents, so its AUROCs over document scores should be distinguished from a dataset-level test and its error guarantees. Meeus et al. (2024) modify documents before training to create a detectable signal, whereas **LaunderBench** audits retained originals without inserting traps. DE-COP probes preferences between verbatim excerpts and paraphrases using multiple-choice questions (Duarte et al., 2024), a different query design from scoring each fixed original directly.

G.2 Laundering and recovery-based auditing

Prior laundering studies have evaluated auditing and learning together[3]. Ertan et al. (2026) use sparse-autoencoder guidance (SAGE) to generate meaning-preserving rewrites and measure membership-

[3]Paraphrasing can also evade detectors of AI-generated text (Krishna et al., 2023), but that task identifies generated outputs rather than whether a source contributed to model training.

inference degradation alongside source-derived question answering, with pretrained and original-text fine-tuned models as comparators. SOFT (Zhang et al., 2025b) evaluates question answering generated from fine-tuning data under selective obfuscation. **LaunderBench** complements these studies by fixing the original audit queries, pairing Original and rewritten training with an identical filler-only continued-training control, and evaluating detection and original-text modeling on matched cohorts across several corpora and auditor resource conditions.

Li et al. (2026) study the mismatch between original queries and transformed training data and introduce a method to recover training-like queries. Their recovery-based setting complements the original-query setting evaluated here: **LaunderBench** measures how specified audits on the retained originals respond to rewriting and how those changes relate to learning under the same training conditions, not the strongest audit attainable with originals recovery.