DOES DISTILLATION LEAVE A TRACE? SOURCE DETECTABILITY IN COMPLEX POST-TRAINING PIPELINES

Anonymous authors
Paper under double-blind review

ABSTRACT

Model post-training pipelines routinely distill data from multiple source models, across different domains and over multiple stages of training. Without proper reporting, it becomes difficult to determine which sources contributed to a released model, raising concerns around IP enforcement and safeguard circumvention. This raises a fundamental question: *do complex post-training decisions affect distillation source detectability*? We present the first systematic study of this question. We build a controlled testbed of 169 models distilled from ten source models across 14 settings, varied across different post-training design decisions. For each source, we test whether the model distilled from it shows greater similarity to that source than reference models trained on alternative sources. We find that while detectability magnitude varies by source and domain, it is largely preserved across post-training choices such as multi-domain distillation, target-model changes, and further on-policy distillation. One key exception: mixing multiple sources within the same domain generally degrades detectability for involved sources, though cross-domain mixing does not. Finally, under more realistic audit assumptions, where reference models do not exactly match the distilled model's pipeline, we find that signals based on the distilled model's likelihoods prove fragile, while stylistic signals remain robust. We release our models, data, and code to support future work on distillation detection and improving our understanding of post-training mechanisms.

1 INTRODUCTION

Model distillation is a cornerstone of modern post-training, enabling capabilities to be transferred from one or more source models to a target model. These distillation pipelines are becoming increasingly complex: they combine synthetic data from multiple sources and domains and may include several stages of training before release. However, transparent reporting regarding distillation sources has not kept pace (Wan et al., 2025). This lack of transparency complicates IP and terms-of-service enforcement, enables distillers to bypass safeguards placed around increasingly capable models, obscures how capabilities and biases propagate across models, and limits transparency into how ostensibly independent systems are actually constructed (Anthropic, 2026a;b)

This opacity motivates the distillation detection problem: determining whether a distilled model was trained on data generated by a given candidate source. Prior work has proposed various detection methods for this problem (Rawat et al., 2026), but experiments are usually relegated to smaller-scale testbeds using simple single-source, single-stage pipelines. Audits of real-world models allow stress-testing under more complex pipelines, but conclusions remain speculative due to a lack of ground-truth attribution (Panfilov et al., 2026). Together, this leaves open whether detectability generalizes to real-world post-training complexity. We therefore ask: *do complex post-training decisions affect distillation detectability*?

We study this question along four axes of the post-training pipeline: (1) *source mixture*, the choice of source(s) for distillation; (2) *domain mixture*, the choice of domain(s) for distillation; (3) *target model*, the model being fine-tuned; and (4) *subsequent post-training*, additional post-training applied after distillation. We construct a controlled suite of 169 distilled models across 14 settings spanning

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.

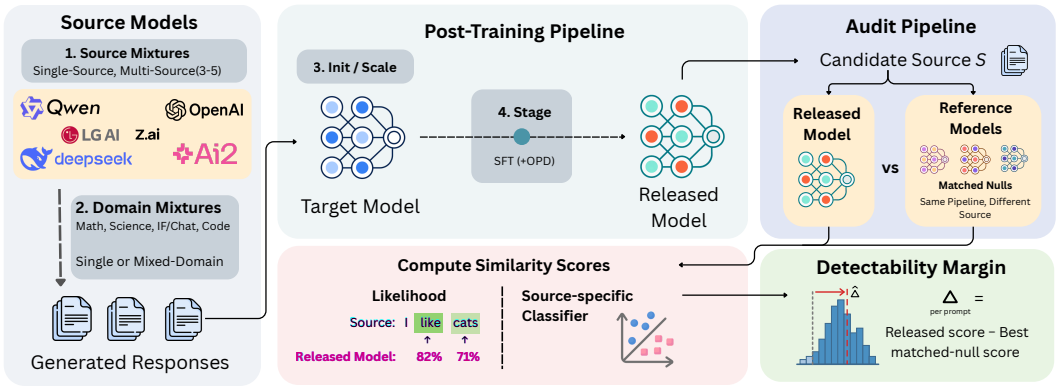


Figure 1: **Overview of our experimental and audit pipeline.** We vary four aspects of the distillation setting: the (1) source mixture, (2) domain mixture, (3) target initialization/scale, and (4) post-training stage. Source models generate responses that are used to train a target model, producing a released model after SFT or additional on-policy distillation. To audit whether a candidate source S remains detectable, we compare the released model against matched-null reference models trained with the same pipeline but different sources. We measure source similarity using likelihood-based scores and a source-specific classifier, and quantify detectability relative to the matched nulls using the testing framework in §3.

these axes, fine-tuned on reasoning traces from ten source models. To study different settings along each axis, we pair distilled models with matched *reference models* that share their training setting, differing only in their distillation source(s). Because lineage is known by construction, we can directly measure detectability – we test whether a model distilled from a given source is more similar to that source than its matched reference models are. The resulting similarity gap provides evidence of source-specific signal: its statistical significance indicates whether the source is detectable, while its size measures how strongly it is detected. We compute this gap using complementary views of model behavior, ranging from the likelihood assigned to source outputs by the distilled model (logit) to source-specific stylistic features captured by a classifier (lexical).

We find that distillation leaves a surprisingly durable footprint across a range of post-training choices: source detectability persists across multi-domain distillation, changes in target model, and further on-policy distillation. However, the strength of detectability depends largely on the source itself and the domain in which it is audited. The one setting that weakens detectability is mixing multiple sources within the same domain, which generally erodes the signal of sources involved. Mixing sources across different domains, by contrast, leaves each source roughly as detectable as if it had been distilled alone in its contributing domain. Among the choices we study, the central challenge is therefore not source mixing itself, but overlap between sources within the same domain.

Our controlled comparisons assume that the reference models match the audited model in every respect except their distillation sources. A real auditor is unlikely to know the full training pipeline well enough to construct such references. We therefore introduce realistic mismatches between the distilled model and the references; these include different target models (§4.3) and amount of source-generated training data (Appendix C.3). These tests expose a sharp contrast between detection signals. Likelihood signals degrades substantially because differences in model capability can overwhelm the source signal. Classifier signals, which rely only on lexical style in black-box outputs, remain robust. This result suggests that likelihood comparisons require carefully matched controls, whereas style-based signals may transfer more reliably to real-world audits.

Together, our findings replace a simple binary view of distillation detection with a more conditional picture. Distillation sources often remain detectable after substantial post-training, but the reliability of that detection depends on both how sources are mixed and how an auditor constructs the comparison set. Reference model design is therefore part of the detection problem, not merely an implementation detail. We release our full suite of distilled models, generated data, and code, and invite the community to contribute to and build on this resource, both to advance source attribution in increasingly complex, realistic pipelines and to deepen our understanding of how post-training shapes model behavior.

2 Related Work

Distillation in modern post-training. Distillation from strong models, often through supervised fine-tuning on source-generated outputs, is a standard way to transfer capabilities to new models (Guo et al., 2025; Guha et al., 2025a). Modern pipelines increasingly mix outputs from multiple sources and domains (Guha et al., 2025a; Lambert et al., 2025; Bercovich et al., 2025a), vary the target model (Li et al., 2025c; Panigrahi et al., 2025), and continue with later stages such as on-policy distillation or RLVR (Lu & Lab, 2025; Yang et al., 2025a; Team Olmo, 2025). Prior work primarily studies how these choices affect capability; we study how they affect the detectability of the sources used during training.

Provenance. A growing body of work in this field has studied whether models or training data can be linked to their origins. Existing approaches either recover model ancestry (Wu et al., 2026; Yax et al., 2025; Nikolic et al., 2025; Kuditipudi et al., 2025; Qiu et al., 2026) or test whether particular examples or datasets were used during training (Carlini et al., 2022; Shi et al., 2024; Maini et al., 2024). Distillation creates a different relationship entirely: the source shares neither weights nor training data with the resulting model, influencing it only through generated supervision. This can link a single model to several otherwise unrelated sources. Because the exact synthetic data used for distillation is rarely available to an auditor, these relationships are difficult to establish and remain under-disclosed (Wan et al., 2025), despite being important for reconstructing modern model lineages.

Distillation detection. When distillation is undisclosed, it must be inferred post-hoc from the distilled model itself: given a candidate source, was the model trained on its outputs? This has become increasingly relevant as large-scale distillation from frontier models enables capabilities to be transferred cheaply, potentially bypassing source safeguards (Anthropic, 2026a). Unlike preventive approaches such as query monitoring or defensive output generation (Li et al., 2025b), distillation detection asks this question after training. Existing methods compare likelihoods against reference models not trained on that source (Rawat et al., 2026; Liu et al., 2026) or use lexical classifiers to detect source-specific idiosyncrasies (Sun et al., 2025; Panfilov et al., 2026). Watermarking offers a complementary approach when the source owner can deliberately embed detectable signals, although these signals may weaken with additional training (Sander et al., 2024; Pan et al., 2025). Existing evidence, however, comes largely from simple controlled settings, real-world models without ground-truth training histories, or work which treats detection as a binary question. This leaves open how the strength of detectability changes under more complex post-training pipelines.

3 Method

3.1 Problem Formulation and Testing Framework

Let S denote a source model and d_S a distilled model post-trained on outputs generated by S. In this work, we focus primarily on supervised fine-tuning (SFT), a simple and widely used form of distillation that directly trains the target model on source-generated outputs (Wadhwa et al., 2025). Our goal is to characterize how strongly this relationship remains detectable as we vary properties of d_S's post-training pipeline.

For each distilled model d_S, let $\Omega_S = \{\omega_1, \ldots, \omega_n\}$ denote a set of *reference (null) distilled models* whose post-training did not use supervision from S. Within each comparison, d_S and every $\omega \in \Omega_S$ are constructed identically except for their distillation sources.

Let $\mathcal{X}$ denote a distribution over prompts and $X = \{x_1, \ldots, x_m\}$ for $x_i \sim \mathcal{X}$ the probes used for a test. For a similarity score σ, let $\sigma(S, q, x)$ measure the similarity between source S and model $q \in \{d_S\} \cup \Omega_S$ on probe x, with larger values indicating greater similarity. We define the per-probe distillation margin as

$$\Delta_x = \sigma(S, d_S, x) - \max_{\omega \in \Omega_S} \sigma(S, \omega, x). \tag{1}$$

A positive margin requires d_S to exhibit greater similarity to S than every matched null distilled model on that probe. Intuitively, if this advantage persists across probes, it provides evidence that d_S contains source-specific signal not explained by otherwise matched construction. We test

$$H_0 : \mathbb{E}_{x \sim \mathcal{X}}[\Delta_x] \leq 0, \qquad H_1 : \mathbb{E}_{x \sim \mathcal{X}}[\Delta_x] > 0. \tag{2}$$

Our test statistic is the mean margin $\widehat{\Delta} = \frac{1}{m}\sum_{j=1}^{m}\Delta_j$ and we estimate significance using a one-sided sign-flip randomization test (Algorithm 1).[1] We report both the p-value from the significance test and $\widehat{\Delta}$: a small p-value indicates evidence against a non-positive expected margin between d_S and Ω_S, while $\widehat{\Delta}$ quantifies the observed separation under σ. Under matched nulls, we interpret this separation as the magnitude of detectability. Furthermore, the test is conditional on the chosen null set; when d_S and Ω_S differ along other properties, changes in $\widehat{\Delta}$ may reflect those mismatches rather than stronger or weaker source-specific signal. We therefore separately study sensitivity to mismatched null construction in this work.

We assume black-box access to generated outputs from all models and, when required by the similarity score, token-level output logits; access to model weights is not required. This reflects an auditing setting in which model internals are typically unavailable.

3.2 Similarity Scores

We instantiate σ using three simple, established measures of model similarity: reference likelihood, a source-specific classifier, and embedding similarity (Wu et al., 2026), capturing logit-, lexical-, and semantic-level signals, respectively. We find embedding similarity generally ineffective and therefore focus our analysis on likelihood and classifier scores; implementation details and embedding results are provided in Appendix B.1.

Reference likelihood. A distilled model trained on outputs from S may assign elevated probability to continuations generated by S. We compute the average log-likelihood per token under model q for probe x as:

$$\sigma_{\mathrm{LL}}(S, q, x) = \frac{1}{L}\sum_{\ell=1}^{L}\log p_q\left(y_\ell^S \mid x, y_{<\ell}^S\right), \qquad y^S = (y_1^S, \ldots, y_L^S) \sim S(\cdot \mid x). \tag{3}$$

Higher values indicate that q assigns greater probability to the continuation produced by S. This score requires token-level logit access.

Source-specific classifier. A distilled model may also reproduce idiosyncracies that distinguish its true source from other candidate sources. For each S, we train a classifier h_S to distinguish generations from S from those of alternative source models. We define

$$\sigma_{\mathrm{clf}}(S, q, x) = h_S(y^q), \qquad y^q \sim q(\cdot \mid x). \tag{4}$$

A larger value indicates that q's generation exhibits more characteristics associated with S.

Note that, because different scores can have different scales, we compare $\widehat{\Delta}$ across post-training conditions within a score σ rather than directly across scores.

3.3 Probe Selection

Probe choice can substantially affect which source-specific behaviors are exposed to the similarity score. We therefore use *heuristically selected probes* by default, ranking a larger candidate pool for each source by how strongly each prompt elicits behavior that distinguishes that source from the alternatives. We then audit using the highest-ranked prompts. Appendix B.2 gives the full procedure and compares it against random sampling, where heuristic selection generally yields greater separation between d_S and $\omega \in \Omega_S$.

Unless otherwise specified, we also match the probe domain to the domain in which d_S was distilled from S, so that the audit targets the setting in which the source signal was introduced. Additionally, we study how this choice affects detectability using out-of-domain probes in Appendix C.1.

4 Experiments

We organize our analysis across four axes of synthetic post-training: (1) **Source Mixture** (§4.1) examines single-source versus multi-source distillation under a fixed domain, testing whether combining signal from multiple sources obscures or preserves each source's fingerprint; (2) **Domain**

[1] Exact sign-flip calibration at the boundary null additionally requires invariance of the probe-level margins to sign changes; symmetry about zero is sufficient (Hemerik & Goeman, 2018).

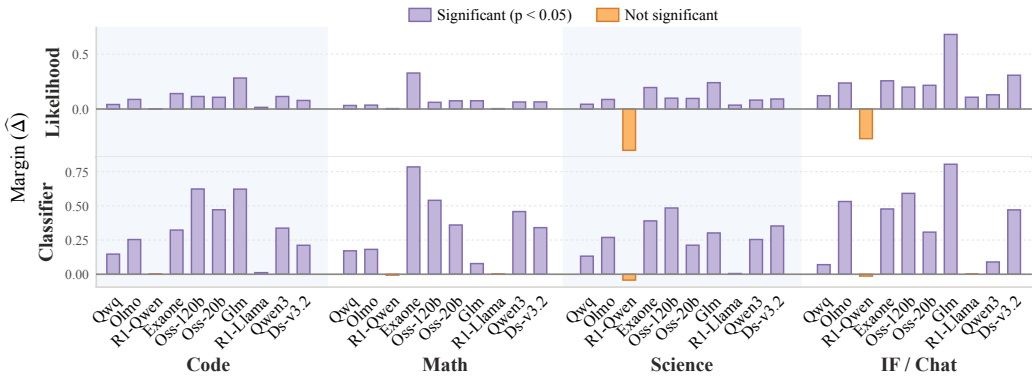


Figure 2: Single-source detectability by domain. Bars show the mean per-prompt margin $\hat{\Delta}$ under likelihood (top) and classifier (bottom) scores per source; higher is more detectable, and color marks significance ($p < 0.05$).

Mixture (§4.2) examines how domain composition affects detectability for single-source distillation within different domains, single-source distillation across mixed domains, and multi-source distillation where each source contributes a distinct domain; (3) **Choice of Target Model** (§4.3) examines the impact of base model initialization and size on detectability; and (4) **Stage of Distillation** (§4.4) provides a preliminary investigation into how later stages of post-training impact earlier stage distillation detectability, examining first stage SFT source detection after an additional stage of on-policy distillation (OPD).

Source Selection. We select ten reasoning-capable source models that are broadly comparable in capability and release timeframe: QwQ-32B (*Qwq*), OLMo-3.1-Think-32B (*Olmo*), DeepSeek-R1-Distill-Qwen2.5-32B (*R1-Qwen*), DeepSeek-R1-Distill-Llama3-70B (*R1-Llama*), GLM-4.7-Flash (*Glm*), GPT-OSS-20B (*Oss-20b*), GPT-OSS-120B (*Oss-120b*), EXAONE-4.0-32B (*Exaone*), Qwen3-235B-A22B (*Qwen3*), and DeepSeek-V3.2 (*Ds-v3.2*). Full model checkpoints and generation details are provided in Appendix A.1.

For any training setting being studied, we repeat training over each source S. As a result, when auditing source S and distilled model d_S, the other nine distilled models serve as Ω_S. Notably, our source set includes multiple models from the same organization, allowing us to study model-level attribution within organizations. We separately consider cross-organization attribution in Appendix C.2.

Training Overview. From each source model, we generate 18k continuations from the same set of prompts per domain using the same sampling configuration. These continuations consist of thinking and final answer blocks. We distill across four commonly used reasoning and instruction-following domains: *code, math, science*, and *instruction-following/chat*. Further details on distillation data generation are provided in Appendix A.2.

Unless otherwise specified, the target model of distillation is Qwen2.5-1.5B-Instruct (*Qwen-base*) and the domain of distillation is **code**. We note that our selected sources were all released after the Qwen-base was published, minimizing the chance of the target model having already seen data generated by these sources in its own training. Specific details on the distilled model subsets used for each post-training factor appear in their corresponding experimental subsection.

4.1 How the Source(s) of Distillation Affect Detectability

We investigate how the choice of source model influences distillation detectability, beginning with the simpler single-source setting before extending to multi-source mixtures that reflect common post-training datasets (Nathawani et al., 2025; Team Olmo, 2025).

Setup. *Single-source.* We first train a distilled model d_S on code data from each of the ten source models and audit it against the remaining nine distilled models as nulls (Ω_S). We use this matched single-source, single-domain setting as the *baseline* for comparisons within each audit domain. *Multi-source.* We then evaluate five randomly sampled 3-source configurations under two dataset-

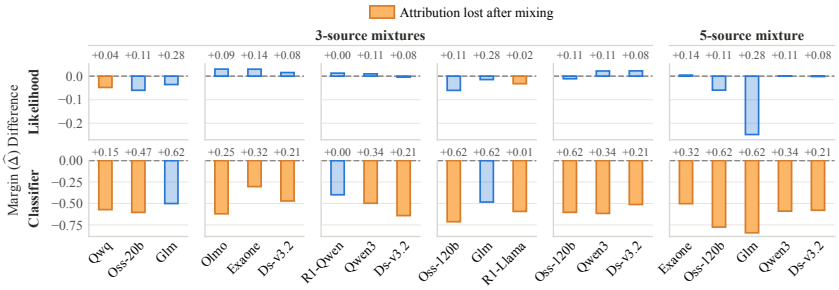


Figure 3: Effect of mixing sources (3s-54k/5s-90k) in the same domain (code). Bars show the change in $\widehat{\Delta}$ relative to the corresponding single-source *baseline* per indicated source audit. Orange indicates the mixing caused detection to become insignificant. Numbers above each bar represent the default $\widehat{\Delta}$ under the respective single-source *baseline*.

size settings: **3-source-54k** (3s-54k), which retains all 18k traces from each source, and **3-source-18k** (3s-18k), which keeps the total dataset size fixed at 18k. We additionally evaluate one 5-source configuration, **5-source-90k** (5s-90k). Full mixture construction and source configurations are provided in Appendix A.4. For each mixture, we audit every constituent source separately, using the single-source models for the remaining sources as nulls. We analyze the 3s-18k setting under the context of cardinality mismatch in Appendix C.3.

Single-source distillation is largely detectable, but its strength depends on the source. Figure 2 (code domain) shows that single-source distillation is generally detectable, with most sources significant under both likelihood and classifier signals. Detectability strength, however, differs sharply by source. Under likelihood, Glm is a clear outlier, likely because its focused coding development (Z.ai, 2025; Team et al., 2025) produces distinctive outputs that models not distilled from it struggle to predict. Under the lexical classifier, this gap narrows, with the two Oss models approaching Glm's detectability and producing distinguishable signals even relative to each other. At the other extreme, R1-Qwen and R1-Llama have the smallest margins, likely because their shared final-stage DeepSeek-R1 post-training data confounds them in both logit and output space.

Mixing sources generally weakens detectability, though less consistently under likelihood. Figure 3 shows that classifier-based detectability drops sharply relative to the single-source baseline when sources are mixed, likely because each source's lexical idiosyncrasies are diluted in the distilled model's outputs. Under likelihood, however, the effect of source mixing depends on the constituent sources. We suspect that degradation is greater when a source is more dissimilar to its co-sources (e.g., Glm in the {Qwq, Oss-20b, Glm} and {Oss-120b, Glm, R1-Llama} mixtures), possibly because the target model must learn several more distinct distributions with limited capacity and therefore models each less faithfully. In contrast, mixing similar sources can raise margins above the single-source baseline (e.g., {Olmo, Exaone, Ds-v3.2}). We caution, however, that this may not reflect stronger source-specific signal, but rather a general increase in modeling capability due to the additional data relative to the nulls. Testing larger targets could clarify whether added capacity prevents this degradation. Crucially, mixing sources does not necessarily eliminate detectability outright, particularly under likelihood.

4.2 How Distillation Domain Affects Detectability

We test whether detectability generalizes across domains and survives mixing. To do so, we consider three settings: single-source distillation within each domain, a single source distilled across all four domains, and multi-source mixtures where each source contributes a distinct domain.

Setup. *Single-source, single-domain.* We extend the baseline from §4.1 to all four domains, using the other same-domain distilled models as nulls. *Single-source, mixed-domain.* We merge each source's traces across domains as either **4d-72k** (18k per domain) or **4d-18k** (subsampled 4.5k per domain), with matched mixed-domain nulls unless stated otherwise. *Multi-source, one domain per source.* Using the 3-source configurations from §4.1, we assign each source a distinct domain and merge their traces; When auditing a given source, we use single-source models from the corresponding domains as nulls.

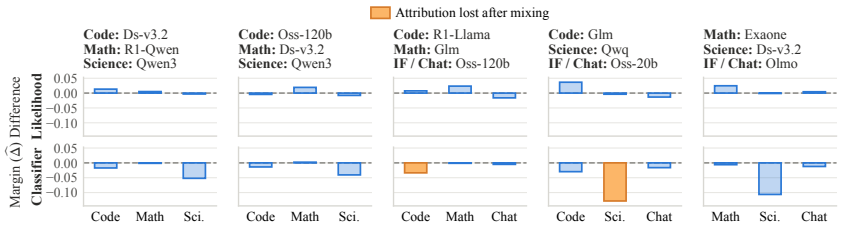


Figure 5: Effect of mixing different sources each contributing different domains. Bars show the change in $\widehat{\Delta}$ relative to the corresponding single-source, single-domain *baseline* per indicated source/domain audit.

Detectability extends across domains, but each source's strength varies by domain. Figure 2 shows that single-source detectability generalizes across domains, though not uniformly. Exaone is considerably more detectable in math than in any other domain; similar to Glm's coding specialization, this may reflect math-focused post-training. Under likelihood, IF/Chat shows notably higher detectability than other domains (average per-source margin 0.078 above the next-highest domain), possibly because its prompts allow more room for source-specific variation. Conversely, math shows the lowest detectability for all sources except Exaone, perhaps because solutions follow more similar reasoning across sources. Under the classifier, this domain-level separation largely disappears, suggesting that the lexical features distinguishing sources appear consistently across domains.

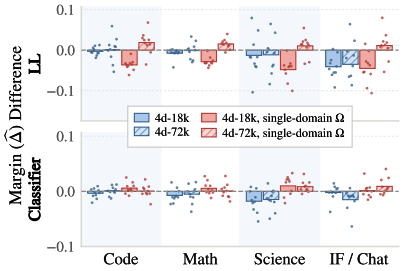


Figure 4: Effect of domain mixing. Per audited domain, bars show the average change in $\widehat{\Delta}$ from the single-domain *baseline*; dots show individual sources. Solid: *4d-18k* distilled models; hatched: *4d-72k* distilled models. Blue uses mixed-domain nulls; red uses single-domain nulls.

Mixing domains largely preserves detectability for each included domain. When both the distilled model and its nulls are mixed-domain (blue bars, Figure 4), detectability changes little relative to the single-domain *baseline*: differences are small and centered near zero across domains and signals. The main exception is IF/Chat under likelihood, where margins decrease by 0.04. Classifier decreases are small relative to single-domain margins (at most 7%). This suggests that per-domain detectability depends mainly on source signal from within that domain, with traces from other domains having little effect. The IF/Chat drop under likelihood may reflect slight leakage from other domains due to content overlap, since IF/Chat prompts can include STEM problems. Notably, 4d-18k and 4d-72k models yield similar margins despite a fourfold difference in in-domain data, suggesting that what matters is not the absolute amount of in-domain data, but whether it matches that of the nulls.

Single-domain nulls suffice when in-domain cardinality is matched. In a real-world audit, it may be impractical to construct nulls with the same domain mixture as the candidate child. Instead, given a target domain, an auditor may use single-domain nulls. The red bars in Figure 4 show detectability change when auditing mixed-domain distilled models against single-domain nulls, relative to the single-domain *baseline*. Under the classifier, detectability remains largely unchanged, with some minor margin increase. Under likelihood, however, results depend on the distilled model's *in-domain cardinality*, i.e., the number of its traces from the target domain. 4s-18k distilled models, with 4.5k in-domain traces against nulls trained on 18k, consistently show weaker detectability. 4s-72k models, whose 18k in-domain traces match that of the nulls, instead show increased margins. We attribute this to traces from other domains, which carry additional source-specific signal that slightly amplifies detectability but cannot compensate for an in-domain cardinality mismatch. The earlier results with the mixed-domain nulls largely avoid this phenomena since they contain the same amount of out-of-domain data as the distilled model. Single-domain experiments confirm that likelihood is sensitive to cardinality mismatch (Appendix C.3).

Mixing sources that each contribute a distinct domain generally preserves per-source detectability. Figure 5 shows that, across all five 3-source configurations, each source remains about

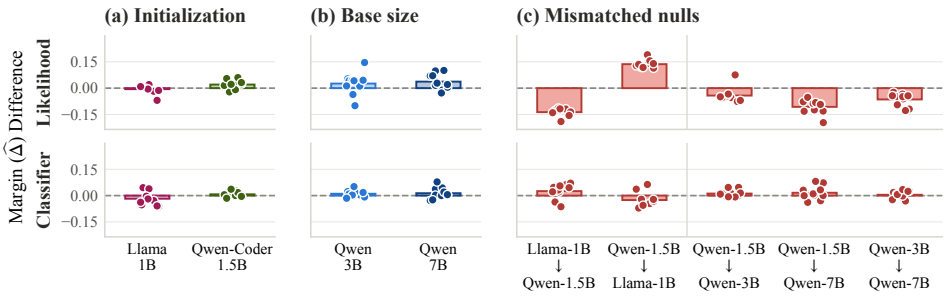


Figure 6: Effect of target initialization (a) and base size (b) on distillation detectability. Bars show the mean change in $\widehat{\Delta}$ relative to the Qwen-base (1.5B) *baseline*; dots show change for each source. Negative values indicate detectability degradation. (c) shows settings where the target of the distilled model differs from its nulls; above the ↓ is the distilled model target, below is the null target.

as detectable in its contributed domain as in the corresponding single-source setting. This contrasts with the more frequently observed degradation in the multi-source code-domain setting. We hypothesize that the separability of the domains lets the distilled model represent each source within its own domain, rather than forcing multiple source distributions into a single domain. Per-domain probes then act as a conditional prior, eliciting behavior from whichever source contributed that domain. These results suggest source detectability can be preserved even in source mixing by ensuring each source contributes distinct capabilities. We use this conditioning intuition to test a variant of the classifier method more robust to same-domain source mixing in Appendix C.4.

4.3 How the Target of Distillation Affects Detectability

Distillation is applied to a variety of target (student) models, which differ along factors such as their prior training lineage and scale. Since a target model's own priors could plausibly interact with the distilled signal, we ask whether detectability is an artifact of a particular target choice or holds more generally across targets. We study this along two axes: *initialization* and *scale*.

Setup. *Initialization.* We vary the target model's organization and domain prior, distilling onto LLaMA3.2-1B-Instruct (from a different organization and thus independently constructed) and Qwen2.5-Coder-1.5B-Instruct (has in-domain priors for the code domain distillation). *Scale.* We vary target model size within a fixed family, additionally distilling onto the 3B, and 7B variants of Qwen2.5-Instruct. In both cases, models are trained identically to the Qwen-base (1.5B) distilled models. In both settings, same-target distilled models serve as each other's nulls during testing unless specified otherwise.

Detectability is largely unchanged by choice of target initialization or size. Figure 6 likewise shows that detectability remains largely unchanged across target sizes, especially under the classifier. Likelihood margin differences, however, vary more across sources for larger target sizes. Larger targets may reproduce source idiosyncrasies more faithfully, strengthening the signal, but larger nulls may also model the source's continuations better, narrowing the gap with the true distilled model. Together with the initialization results, this suggests that detectability depends more on the source than on the target.

Likelihood is sensitive to target mismatch, while the classifier remains robust. We additionally consider two mismatch settings, where an auditor does not know the distilled model's (1) base model or (2) size, and so uses nulls built on a different base or size. Figure 6 shows that classifier margins remain largely unchanged in both cases. Likelihood margins, in contrast, shift substantially. They drop when the distilled model's base is weaker than that of the nulls (Llama → Qwen), and they inflate when it is stronger (Qwen → Llama), risking false positives. For different size settings, where the distilled model is smaller than the nulls, detectability decreases further as the size gap grows (e.g., 1.5b vs 3b: -0.042; 1.5b vs. 7b: -0.107). We suspect that likelihood entangles detectability with capability, making it sensitive to any capability gap between the distilled model and its nulls.

4.4 Multi-Stage Distillation Detection

Modern post-training pipelines often involve multiple stages, raising the question of whether later distillation weakens signals introduced earlier in training. We study this in a two-stage setting where the first SFT stage matches our previous experiments and each model is then further tuned using on-policy distillation (OPD). We restrict this analysis to OPD as the second stage, rather than making claims about multi-stage post-training more broadly, and ask whether the first-stage source remains detectable after additional training.



Figure 7: Effect of later-stage OPD on first-stage source detectability. Bars show the average change in $\widehat{\Delta}$ relative to the single-stage *baseline*; dots show change per source. Blue tests with matched two-stage nulls; red uses single-stage nulls. Negative values indicate weaker detectability.

Setup. We take the first-stage SFT child models over Qwen2.5-7B-Instruct from §4.3, selected for their training stability, and further tune them with a second stage of OPD using two secondary parents, Qwen3.5-27B and Qwen3.8-27B. This yields two sets of two-stage models, one per secondary parent (further training details in Appendix A.5). For each two-stage model, nulls consist of the other two-stage models that share its second-stage OPD parent but differ in first-stage SFT parent, unless stated otherwise.

Further on-policy distillation largely preserves earlier stage source detectability. Figure 7 shows that further OPD has little systematic effect on first-stage source detectability under both likelihood and classifier methods, regardless of the OPD teacher. This may be partly explained by the on-policy nature of the second stage: OPD trains on trajectories generated by the student itself, with the teacher providing supervision at states the student already visits. Rather than having the student imitate teacher-generated trajectories, the second-stage teacher refines behavior along the student's existing distribution Li et al. (2026a). This may allow characteristics inherited from the first-stage source to persist through an additional distillation stage. Still, the mean margin difference is consistently negative, if small. Because the distilled model and its nulls share the same OPD teacher, the second stage may pull them toward a common distribution, slightly narrowing the gap between them. It therefore remains open whether longer OPD or additional reinforcement learning (RL) stages would eventually erode detectability, or whether it would plateau.

Early-stage detectability is generally robust to auditing with single-stage nulls. In practice, it may be impractical to build nulls that replicate a distilled model's full multi-stage training pipeline. We therefore repeat the audit using single-stage SFT nulls. Margin differences again remain small. Classifier changes vary more across sources but stay centered near zero. The small rise in likelihood may reflect the OPD models' modest capability advantage over the single-stage nulls.

5 Conclusion

We present the first systematic study of how realistic post-training decisions affect distillation detectability. Overall, detectability holds up surprisingly well across domain mixtures, target models, and an additional stage of on-policy distillation. The clearest exception is mixing multiple sources within the same domain, which generally weakens detectability, while mixing sources across domains largely preserves it. We also find that likelihood-based signals become sensitive to capability and data-cardinality mismatch when reference models do not closely match the audited model, whereas classifier signals are more stable. Taken together, our results suggest that detectability depends not only on the post-training pipeline, but also on how distinctive the source is: specialized sources can leave stronger signals, while shared lineages may reduce separation. This motivates more robust detection methods and further study of model distinctiveness. We release our testbed, models, data, and code to support extensions to larger models, broader source sets, and more complex pipelines. As more realistic training histories become available for study, we can move from asking whether distillation is detectable to understanding when and why it remains detectable.

AI USE STATEMENT

Since we study distillation detectability, generative models are a core component of our experimental setup: source models generate the synthetic data used for distillation, and additional models are used throughout the post-training and evaluation pipelines described in the paper.

We also used generative AI tools in a supporting role during manuscript preparation, primarily to polish grammar, wording, and presentation, as well as for routine coding assistance. The research questions, hypotheses, experimental design, analysis, and scientific conclusions were developed by the authors. All AI-assisted material was reviewed and verified by the authors, who take responsibility for the final content of the paper.

ETHICS STATEMENT

This work studies methods for detecting whether a model was distilled from the outputs of another model. We do not conduct experiments with human participants or collect new personal data; our experiments use model-generated data and publicly available model and prompt resources described in the paper.

Distillation detection can support transparency, provenance auditing, and investigation of potentially undisclosed model dependencies. At the same time, characterizing when detection succeeds or fails could inform attempts to make distillation harder to detect. We therefore present our results as measurements of detectability under specific experimental assumptions, rather than as definitive evidence of ownership, policy violations, or misuse. We release our models, data, and code to support reproducible research on more reliable attribution methods and a better understanding of how post-training affects model provenance.

REPRODUCIBILITY STATEMENT

We give a high-level outline of our experiment construction and details concerning experimental setup in §4. We place all specific model training, data generation, classifier construction and probe selection details in Appendix A. We also will provide links to our project artifacts and codebase.

REFERENCES

Anthropic. Detecting and preventing distillation attacks. https://www.anthropic.com/news/detecting-and-preventing-distillation-attacks, February 2026a. Accessed: 2026-09-24.

Anthropic. Detecting and countering misuse of AI: September 2026. Technical report, Anthropic, September 2026b. URL https://www.anthropic.com/threat-intelligence-report-september-2026. Published September 10, 2026.

Kyunghoon Bae, Eunbi Choi, Kibong Choi, Stanley Jungkyu Choi, Yemuk Choi, Kyubeen Han, Seokhee Hong, Junwon Hwang, Taewan Hwang, Joonwon Jang, Hyojin Jeon, Kijeong Jeon, Gerrard Jeongwon Jo, Hyunjik Jo, Jiyeon Jung, Euisoon Kim, Hyosang Kim, Jihoon Kim, Joonkee Kim, Seonghwan Kim, Soyeon Kim, Sunkyoung Kim, Yireun Kim, Yongil Kim, Youchul Kim, Edward Hwayoung Lee, Gwangho Lee, Haeju Lee, Honglak Lee, Jinsik Lee, Kyungmin Lee, Sangha Park, Young Min Paik, Yongmin Park, Youngyong Park, Sanghyun Seo, Sihoon Yang, Heuiyeen Yeen, Sihyuk Yi, and Hyeongu Yun. EXAONE 4.0: Unified large language models integrating non-reasoning and reasoning modes. *arXiv preprint arXiv:2507.11407*, 2025.

Federico Barbero, Xiangming Gu, Christopher A. Choquette-Choo, Chawin Sitawarin, Matthew Jagielski, Itay Yona, Petar Veličković, Ilia Shumailov, and Jamie Hayes. Extracting alignment data in open models, 2025. URL https://arxiv.org/abs/2510.18554.

Akhiad Bercovich, Itay Levy, Izik Golan, Mohammad Dabbah, Ran El-Yaniv, Omri Puny, Ido Galil, Zach Moshe, Tomer Ronen, Najeeb Nabwani, Ido Shahaf, Oren Tropp, Ehud Karpas, Ran Zilberstein, Jiaqi Zeng, Soumye Singhal, Alexander Bukharin, Yian Zhang, Tugrul Konuk, Gerald Shen, Ameya Sunil Mahabaleshwarkar, Bilal Kartal, Yoshi Suhara, Olivier Delalleau, Zijia Chen,

Zhilin Wang, David Mosallanezhad, Adi Renduchintala, Haifeng Qian, Dima Rekesh, Fei Jia, Somshubra Majumdar, Vahid Noroozi, Wasi Uddin Ahmad, Sean Narenthiran, Aleksander Ficek, Mehrzad Samadi, Jocelyn Huang, Siddhartha Jain, Igor Gitman, Ivan Moshkov, Wei Du, Shubham Toshniwal, George Armstrong, Branislav Kisacanin, Matvei Novikov, Daria Gitman, Evelina Bakhturina, Prasoon Varshney, Makesh Narsimhan, Jane Polak Scowcroft, John Kamalu, Dan Su, Kezhi Kong, Markus Kliegl, Rabeeh Karimi Mahabadi, Ying Lin, Sanjeev Satheesh, Jupinder Parmar, Pritam Gundecha, Brandon Norick, Joseph Jennings, Shrimai Prabhumoye, Syeda Nahida Akter, Mostofa Patwary, Abhinav Khattar, Deepak Narayanan, Roger Waleffe, Jimmy Zhang, Bor-Yiing Su, Guyue Huang, Terry Kong, Parth Chadha, Sahil Jain, Christine Harvey, Elad Segal, Jining Huang, Sergey Kashirsky, Robert McQueen, Izzy Putterman, George Lam, Arun Venkatesan, Sherry Wu, Vinh Nguyen, Manoj Kilaru, Andrew Wang, Anna Warno, Abhilash Somasamudramath, Sandip Bhaskar, Maka Dong, Nave Assaf, Shahar Mor, Omer Ullman Argov, Scot Junkin, Oleksandr Romanenko, Pedro Larroy, Monika Katariya, Marco Rovinelli, Viji Balas, Nicholas Edelman, Anahita Bhiwandiwalla, Muthu Subramaniam, Smita Ithape, Karthik Ramamoorthy, Yuting Wu, Suguna Varshini Velury, Omri Almog, Joyjit Daw, Denys Fridman, Erick Galinkin, Michael Evans, Shaona Ghosh, Katherine Luna, Leon Derczynski, Nikki Pope, Eileen Long, Seth Schneider, Guillermo Siman, Tomasz Grzegorzek, Pablo Ribalta, Monika Katariya, Chris Alexiuk, Joey Conway, Trisha Saar, Ann Guan, Krzysztof Pawelec, Shyamala Prayaga, Oleksii Kuchaiev, Boris Ginsburg, Oluwatobi Olabiyi, Kari Briski, Jonathan Cohen, Bryan Catanzaro, Jonah Alben, Yonatan Geifman, and Eric Chung. Llama-nemotron: Efficient reasoning models, 2025a. URL https://arxiv.org/abs/2505.00949.

Akhiad Bercovich, Itay Levy, Izik Golan, Mohammad Dabbah, Ran El-Yaniv, Omri Puny, Ido Galil, Zach Moshe, Tomer Ronen, Najeeb Nabwani, Ido Shahaf, Oren Tropp, Ehud Karpas, Ran Zilberstein, Jiaqi Zeng, Soumye Singhal, Alexander Bukharin, Yian Zhang, Tugrul Konuk, Gerald Shen, Ameya Sunil Mahabaleshwarkar, Bilal Kartal, Yoshi Suhara, Olivier Delalleau, Zijia Chen, Zhilin Wang, David Mosallanezhad, Adi Renduchintala, Haifeng Qian, Dima Rekesh, Fei Jia, Somshubra Majumdar, Vahid Noroozi, Wasi Uddin Ahmad, Sean Narenthiran, Aleksander Ficek, Mehrzad Samadi, Jocelyn Huang, Siddhartha Jain, Igor Gitman, Ivan Moshkov, Wei Du, Shubham Toshniwal, George Armstrong, Branislav Kisacanin, Matvei Novikov, Daria Gitman, Evelina Bakhturina, Prasoon Varshney, Makesh Narsimhan, Jane Polak Scowcroft, John Kamalu, Dan Su, Kezhi Kong, Markus Kliegl, Rabeeh Karimi Mahabadi, Ying Lin, Sanjeev Satheesh, Jupinder Parmar, Pritam Gundecha, Brandon Norick, Joseph Jennings, Shrimai Prabhumoye, Syeda Nahida Akter, Mostofa Patwary, Abhinav Khattar, Deepak Narayanan, Roger Waleffe, Jimmy Zhang, Bor-Yiing Su, Guyue Huang, Terry Kong, Parth Chadha, Sahil Jain, Christine Harvey, Elad Segal, Jining Huang, Sergey Kashirsky, Robert McQueen, Izzy Putterman, George Lam, Arun Venkatesan, Sherry Wu, Vinh Nguyen, Manoj Kilaru, Andrew Wang, Anna Warno, Abhilash Somasamudramath, Sandip Bhaskar, Maka Dong, Nave Assaf, Shahar Mor, Omer Ullman Argov, Scot Junkin, Oleksandr Romanenko, Pedro Larroy, Monika Katariya, Marco Rovinelli, Viji Balas, Nicholas Edelman, Anahita Bhiwandiwalla, Muthu Subramaniam, Smita Ithape, Karthik Ramamoorthy, Yuting Wu, Suguna Varshini Velury, Omri Almog, Joyjit Daw, Denys Fridman, Erick Galinkin, Michael Evans, Shaona Ghosh, Katherine Luna, Leon Derczynski, Nikki Pope, Eileen Long, Seth Schneider, Guillermo Siman, Tomasz Grzegorzek, Pablo Ribalta, Monika Katariya, Chris Alexiuk, Joey Conway, Trisha Saar, Ann Guan, Krzysztof Pawelec, Shyamala Prayaga, Oleksii Kuchaiev, Boris Ginsburg, Oluwatobi Olabiyi, Kari Briski, Jonathan Cohen, Bryan Catanzaro, Jonah Alben, Yonatan Geifman, and Eric Chung. Llama-Nemotron: Efficient reasoning models. *arXiv preprint arXiv:2505.00949*, 2025b.

Nicholas Carlini, Steve Chien, Milad Nasr, Shuang Song, Andreas Terzis, and Florian Tramer. Membership inference attacks from first principles, 2022. URL https://arxiv.org/abs/2112.03570.

Junying Chen, Zhenyang Cai, Ke Ji, Xidong Wang, Wanlong Liu, Rongsheng Wang, Jianye Hou, and Benyou Wang. HuatuoGPT-o1, towards medical complex reasoning with LLMs. *arXiv preprint arXiv:2412.18925*, 2024.

DeepSeek-AI. DeepSeek-r1: Incentivizing reasoning capability in LLMs via reinforcement learning. *arXiv preprint arXiv:2501.12948*, 2025a.

DeepSeek-AI. DeepSeek-v3.2: Pushing the frontier of open large language models. *arXiv preprint arXiv:2512.02556*, 2025b.

Wei Du, Shubham Toshniwal, Branislav Kisacanin, Sadegh Mahdavi, Ivan Moshkov, George Armstrong, Stephen Ge, Edgar Minasyan, Feng Chen, and Igor Gitman. Nemotron-math: Efficient long-context distillation of mathematical reasoning from multi-mode supervision. *arXiv preprint arXiv:2512.15489*, 2025.

Tao Ge, Xin Chan, Xiaoyang Wang, Dian Yu, Haitao Mi, and Dong Yu. Scaling synthetic data creation with 1,000,000,000 personas. *arXiv preprint arXiv:2406.20094*, 2024.

GLM-4.5 Team. GLM-4.5: Agentic, reasoning, and coding (ARC) foundation models. *arXiv preprint arXiv:2508.06471*, 2025.

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

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

Etash Guha, Ryan Marten, Sedrick Keh, Negin Raoof, Georgios Smyrnis, Hritik Bansal, Marianna Nezhurina, Jean Mercat, Trung Vu, Zayne Sprague, Ashima Suvarna, Benjamin Feuer, Liangyu Chen, Zaid Khan, Eric Frankel, Sachin Grover, Caroline Choi, Niklas Muennighoff, Shiye Su, Wanjia Zhao, John Yang, Shreyas Pimpalgaonkar, Kartik Sharma, Charlie Cheng-Jie Ji, Yichuan Deng, Sarah Pratt, Vivek Ramanujan, Jon Saad-Falcon, Jeffrey Li, Achal Dave, Alon Albalak, Kushal Arora, Blake Wulfe, Chinmay Hegde, Greg Durrett, Sewoong Oh, Mohit Bansal, Saadia Gabriel, Aditya Grover, Kai-Wei Chang, Vaishaal Shankar, Aaron Gokaslan, Mike A. Merrill, Tatsunori Hashimoto, Yejin Choi, Jenia Jitsev, Reinhard Heckel, Maheswaran Sathiamoorthy, Alexandros G. Dimakis, and Ludwig Schmidt. Openthoughts: Data recipes for reasoning models, 2025a. URL https://arxiv.org/abs/2506.04178.

Etash Guha, Ryan Marten, Sedrick Keh, Negin Raoof, Georgios Smyrnis, Hritik Bansal, Marianna Nezhurina, Jean Mercat, Trung Vu, Zayne Sprague, Ashima Suvarna, Benjamin Feuer, Liangyu Chen, Zaid Khan, Eric Frankel, Sachin Grover, Caroline Choi, Niklas Muennighoff, Shiye Su,

Wanjia Zhao, John Yang, Shreyas Pimpalgaonkar, Kartik Sharma, Charlie Cheng-Jie Ji, Yichuan Deng, Sarah Pratt, Vivek Ramanujan, Jon Saad-Falcon, Jeffrey Li, Achal Dave, Alon Albalak, Kushal Arora, Blake Wulfe, Chinmay Hegde, Greg Durrett, Sewoong Oh, Mohit Bansal, Saadia Gabriel, Aditya Grover, Kai-Wei Chang, Vaishaal Shankar, Aaron Gokaslan, Mike A. Merrill, Tatsunori Hashimoto, Yejin Choi, Jenia Jitsev, Reinhard Heckel, Maheswaran Sathiamoorthy, Alexandros G. Dimakis, and Ludwig Schmidt. OpenThoughts: Data recipes for reasoning models. *arXiv preprint arXiv:2506.04178*, 2025b.

Daya Guo, Dejian Yang, Haowei Zhang, Junxiao Song, Peiyi Wang, Qihao Zhu, Runxin Xu, Ruoyu Zhang, Shirong Ma, Xiao Bi, Xiaokang Zhang, Xingkai Yu, Yu Wu, Z. F. Wu, Zhibin Gou, Zhihong Shao, Zhuoshu Li, Ziyi Gao, Aixin Liu, Bing Xue, Bingxuan Wang, Bochao Wu, Bei Feng, Chengda Lu, Chenggang Zhao, Chengqi Deng, Chong Ruan, Damai Dai, Deli Chen, Dongjie Ji, Erhang Li, Fangyun Lin, Fucong Dai, Fuli Luo, Guangbo Hao, Guanting Chen, Guowei Li, H. Zhang, Hanwei Xu, Honghui Ding, Huazuo Gao, Hui Qu, Hui Li, Jianzhong Guo, Jiashi Li, Jingchang Chen, Jingyang Yuan, Jinhao Tu, Junjie Qiu, Junlong Li, J. L. Cai, Jiaqi Ni, Jian Liang, Jin Chen, Kai Dong, Kai Hu, Kaichao You, Kaige Gao, Kang Guan, Kexin Huang, Kuai Yu, Lean Wang, Lecong Zhang, Liang Zhao, Litong Wang, Liyue Zhang, Lei Xu, Leyi Xia, Mingchuan Zhang, Minghua Zhang, Minghui Tang, Mingxu Zhou, Meng Li, Miaojun Wang, Mingming Li, Ning Tian, Panpan Huang, Peng Zhang, Qiancheng Wang, Qinyu Chen, Qiushi Du, Ruiqi Ge, Ruisong Zhang, Ruizhe Pan, Runji Wang, R. J. Chen, R. L. Jin, Ruyi Chen, Shanghao Lu, Shangyan Zhou, Shanhuang Chen, Shengfeng Ye, Shiyu Wang, Shuiping Yu, Shunfeng Zhou, Shuting Pan, S. S. Li, Shuang Zhou, Shaoqing Wu, Tao Yun, Tian Pei, Tianyu Sun, T. Wang, Wangding Zeng, Wen Liu, Wenfeng Liang, Wenjun Gao, Wenqin Yu, Wentao Zhang, W. L. Xiao, Wei An, Xiaodong Liu, Xiaohan Wang, Xiaokang Chen, Xiaotao Nie, Xin Cheng, Xin Liu, Xin Xie, Xingchao Liu, Xinyu Yang, Xinyuan Li, Xuecheng Su, Xuheng Lin, X. Q. Li, Xiangyue Jin, Xiaojin Shen, Xiaosha Chen, Xiaowen Sun, Xiaoxiang Wang, Xinnan Song, Xinyi Zhou, Xianzu Wang, Xinxia Shan, Y. K. Li, Y. Q. Wang, Y. X. Wei, Yang Zhang, Yanhong Xu, Yao Li, Yao Zhao, Yaofeng Sun, Yaohui Wang, Yi Yu, Yichao Zhang, Yifan Shi, Yiliang Xiong, Ying He, Yishi Piao, Yisong Wang, Yixuan Tan, Yiyang Ma, Yiyuan Liu, Yongqiang Guo, Yuan Ou, Yuduan Wang, Yue Gong, Yuheng Zou, Yujia He, Yunfan Xiong, Yuxiang Luo, Yuxiang You, Yuxuan Liu, Yuyang Zhou, Y. X. Zhu, Yanping Huang, Yaohui Li, Yi Zheng, Yuchen Zhu, Yunxian Ma, Ying Tang, Yukun Zha, Yuting Yan, Z. Z. Ren, Zehui Ren, Zhangli Sha, Zhe Fu, Zhean Xu, Zhenda Xie, Zhengyan Zhang, Zhewen Hao, Zhicheng Ma, Zhigang Yan, Zhiyu Wu, Zihui Gu, Zijia Zhu, Zijun Liu, Zilin Li, Ziwei Xie, Ziyang Song, Zizheng Pan, Zhen Huang, Zhipeng Xu, Zhongyu Zhang, and Zhen Zhang. Deepseek-r1 incentivizes reasoning in llms through reinforcement learning. *Nature*, 645(8081):633–638, 2025. ISSN 1476-4687. doi: 10.1038/s41586-025-09422-z. URL http://dx.doi.org/10.1038/s41586-025-09422-z.

David Hall, Ahmed Ahmed, Christopher Chou, Abhinav Garg, Rohith Kuditipudi, Will Held, Nikil Ravi, Herumb Shandilya, Jason Wang, Jason Bolton, Siddharth Karamcheti, Suhas Kotha, Tony Lee, Nelson Liu, Joel Niklaus, Ashwin Ramaswami, Kamyar Salahi, Kaiyue Wen, Chi Heem Wong, Sherry Yang, Ivan Zhou, and Percy Liang. Introducing Marin: An open lab for building foundation models. Marin Community Blog, May 2025. URL https://marin.community/blog/2025/05/19/announcement/. Blog post.

Li Hao, He CAO, Bin Feng, Daniel Shao, Xiangru Tang, Zhiyuan Yan, Yonghong Tian, Li Yuan, and Yu Li. Beyond chemical QA: Evaluating LLM's chemical reasoning with modular chemical operations. In *The Thirty-ninth Annual Conference on Neural Information Processing Systems Datasets and Benchmarks Track*, 2026. URL https://openreview.net/forum?id=mvahXacGWg.

Jesse Hemerik and Jelle Goeman. Exact testing with random permutations. *Test*, 27(4):811–825, 2018.

Binyuan Hui, Jian Yang, Zeyu Cui, Jiaxi Yang, Dayiheng Liu, Lei Zhang, Tianyu Liu, Jiajun Zhang, Bowen Yu, Keming Lu, Kai Dang, Yang Fan, Yichang Zhang, An Yang, Rui Men, Fei Huang, Bo Zheng, Yibo Miao, Shanghaoran Quan, Yunlong Feng, Xingzhang Ren, Xuancheng Ren, Jingren Zhou, and Junyang Lin. Qwen2.5-coder technical report. *arXiv preprint arXiv:2409.12186*, 2024.

Intelligent Internet. II-Medical-Reasoning: Medical reasoning dataset. https://huggingface.co/datasets/Intelligent-Internet/II-Medical-Reasoning-SFT, 2025.

Liwei Jiang, Yuanjun Chai, Margaret Li, Mickel Liu, Raymond Fok, Nouha Dziri, Yulia Tsvetkov, Maarten Sap, and Yejin Choi. Artificial hivemind: The open-ended homogeneity of language models (and beyond). In *The Thirty-ninth Annual Conference on Neural Information Processing Systems Datasets and Benchmarks Track*, 2026. URL https://openreview.net/forum?id=saDOrrnNTz.

Jaehun Jung, Seungju Han, Ximing Lu, Skyler Hallinan, David Acuna, Shrimai Prabhumoye, Mostafa Patwary, Mohammad Shoeybi, Bryan Catanzaro, and Yejin Choi. Prismatic synthesis: Gradient-based data diversification boosts generalization in LLM reasoning. *arXiv preprint arXiv:2505.20161*, 2025.

Rohith Kuditipudi, Jing Huang, Sally Zhu, Diyi Yang, Christopher Potts, and Percy Liang. Blackbox model provenance via palimpsestic membership inference, 2025. URL https://arxiv.org/abs/2510.19796.

Nathan Lambert, Jacob Morrison, Valentina Pyatkin, Shengyi Huang, Hamish Ivison, Faeze Brahman, Lester James V. Miranda, Alisa Liu, Nouha Dziri, Shane Lyu, Yuling Gu, Saumya Malik, Victoria Graf, Jena D. Hwang, Jiangjiang Yang, Ronan Le Bras, Oyvind Tafjord, Chris Wilhelm, Luca Soldaini, Noah A. Smith, Yizhong Wang, Pradeep Dasigi, and Hannaneh Hajishirzi. Tulu 3: Pushing frontiers in open language model post-training. *arXiv preprint arXiv:2411.15124*, 2024.

Nathan Lambert, Jacob Morrison, Valentina Pyatkin, Shengyi Huang, Hamish Ivison, Faeze Brahman, Lester James V. Miranda, Alisa Liu, Nouha Dziri, Shane Lyu, Yuling Gu, Saumya Malik, Victoria Graf, Jena D. Hwang, Jiangjiang Yang, Ronan Le Bras, Oyvind Tafjord, Chris Wilhelm, Luca Soldaini, Noah A. Smith, Yizhong Wang, Pradeep Dasigi, and Hannaneh Hajishirzi. Tulu 3: Pushing frontiers in open language model post-training, 2025. URL https://arxiv.org/abs/2411.15124.

Jia LI, Edward Beeching, Lewis Tunstall, Ben Lipkin, Roman Soletskyi, Shengyi Costa Huang, Kashif Rasul, Longhui Yu, Albert Jiang, Ziju Shen, Zihan Qin, Bin Dong, Li Zhou, Yann Fleureau, Guillaume Lample, and Stanislas Polu. Numinamath. [https://huggingface.co/AI-MO/NuminaMath-CoT](https://github.com/project-numina/aimo-progress-prize/blob/main/report/numina_dataset.pdf), 2024.

Jijie Li, Li Du, Hanyu Zhao, Bo wen Zhang, Liangdong Wang, Boyan Gao, Guang Liu, and Yonghua Lin. Infinity instruct: Scaling instruction selection and synthesis to enhance language models, 2025a. URL https://arxiv.org/abs/2506.11116.

Pingzhi Li, Zhen Tan, Mohan Zhang, Huaizhi Qu, Huan Liu, and Tianlong Chen. Doge: Defensive output generation for llm protection against knowledge distillation, 2025b. URL https://arxiv.org/abs/2505.19504.

Yaxuan Li, Yuxin Zuo, Bingxiang He, Jinqian Zhang, Chaojun Xiao, Cheng Qian, Tianyu Yu, Huan ang Gao, Wenkai Yang, Zhiyuan Liu, and Ning Ding. Rethinking on-policy distillation of large language models: Phenomenology, mechanism, and recipe, 2026a. URL https://arxiv.org/abs/2604.13016.

Yaxuan Li, Yuxin Zuo, Bingxiang He, Jinqian Zhang, Chaojun Xiao, Cheng Qian, Tianyu Yu, Huanang Gao, Wenkai Yang, Zhiyuan Liu, and Ning Ding. Rethinking on-policy distillation of large language models: Phenomenology, mechanism, and recipe. *arXiv preprint arXiv:2604.13016*, 2026b.

Yuetai Li, Xiang Yue, Zhangchen Xu, Fengqing Jiang, Luyao Niu, Bill Yuchen Lin, Bhaskar Ramasubramanian, and Radha Poovendran. Small models struggle to learn from strong reasoners, 2025c. URL https://arxiv.org/abs/2502.12143.

Kaiyuan Liu, Shaotian Yan, Rui Miao, Bing Wang, Chen Shen, Jun Zhang, and Jieping Ye. Where did this sentence come from? tracing provenance in llm reasoning distillation, 2026. URL https://arxiv.org/abs/2512.20908.

Kevin Lu and Thinking Machines Lab. On-policy distillation. *Thinking Machines Lab: Connectionism*, 2025. doi: 10.64434/tml.20251026. https://thinkingmachines.ai/blog/on-policy-distillation.

Thang Luong, Dawsen Hwang, Hoang H. Nguyen, Golnaz Ghiasi, Yuri Chervonyi, Insuk Seo, Junsu Kim, Garrett Bingham, Jonathan Lee, Swaroop Mishra, Alex Zhai, Clara Huiyi Hu, Henryk Michalewski, Jimin Kim, Jeonghyun Ahn, Junhwi Bae, Xingyou Song, Trieu H. Trinh, Quoc V. Le, and Junehyuk Jung. Towards robust mathematical reasoning. In *Proceedings of the 2025 Conference on Empirical Methods in Natural Language Processing*, 2025. URL https://aclanthology.org/2025.emnlp-main.1794/.

Pratyush Maini, Hengrui Jia, Nicolas Papernot, and Adam Dziedzic. Llm dataset inference: Did you train on my dataset?, 2024. URL https://arxiv.org/abs/2406.06443.

Niklas Muennighoff, Zitong Yang, Weijia Shi, Xiang Lisa Li, Li Fei-Fei, Hannaneh Hajishirzi, Luke Zettlemoyer, Percy Liang, Emmanuel Candès, and Tatsunori Hashimoto. s1: Simple test-time scaling, 2025. URL https://arxiv.org/abs/2501.19393.

Dhruv Nathawani, Shuoyang Ding, Vitaly Lavrukhin, Igor Gitman, Somshubra Majumdar, Evelina Bakhturina, Boris Ginsburg, and Jane Polak Scowcroft. Nemotron-Post-Training-Dataset-v2, August 2025. URL https://huggingface.co/datasets/nvidia/Nemotron-Post-Training-Dataset-v2.

Ivica Nikolic, Teodora Baluta, and Prateek Saxena. Model provenance testing for large language models, 2025. URL https://arxiv.org/abs/2502.00706.

NVIDIA. Nemo gym: An open source library for scaling reinforcement learning environments for llm. https://github.com/NVIDIA-NeMo/Gym, 2025. GitHub repository.

NVIDIA. Nemotron-Instruction-Following-Chat-v1. https://huggingface.co/datasets/nvidia/Nemotron-Instruction-Following-Chat-v1, 2025a.

NVIDIA. Nemotron-Math-Proofs-v1. https://huggingface.co/datasets/nvidia/Nemotron-Math-Proofs-v1, 2025b.

NVIDIA. Nemotron-RL-coding-competitive_coding. https://huggingface.co/datasets/nvidia/Nemotron-RL-coding-competitive_coding, 2025c.

NVIDIA. Nemotron-Science-v1. https://huggingface.co/datasets/nvidia/Nemotron-Science-v1, 2025d.

OpenAI. gpt-oss-120b & gpt-oss-20b model card. *arXiv preprint arXiv:2508.10925*, 2025.

Leyi Pan, Aiwei Liu, Shiyu Huang, Yijian Lu, Xuming Hu, Lijie Wen, Irwin King, and Philip S. Yu. Can llm watermarks robustly prevent unauthorized knowledge distillation?, 2025. URL https://arxiv.org/abs/2502.11598.

Alexander Panfilov, David Schmotz, Ilia Shumailov, Luca Beurer-Kellner, Joachim Schaeffer, Ameya Prabhu, Jonas Geiping, and Maksym Andriushchenko. Stealing reasoning traces from proprietary llm apis, 2026. URL https://arxiv.org/abs/2608.09867.

Abhishek Panigrahi, Bingbin Liu, Sadhika Malladi, Sham Kakade, and Surbhi Goel. In good graces: Principled teacher selection for knowledge distillation, 2025. URL https://arxiv.org/abs/2511.02833.

Carlos Miguel Patiño, Kashif Rasul, Quentin Gallouédec, Ben Burtenshaw, Sergio Paniego, Vaibhav Srivastav, Thibaud Frere, Ed Beeching, Lewis Tunstall, Leandro von Werra, and Thomas Wolf. Unlocking on-policy distillation for any model family. https://huggingface.co/spaces/HuggingFaceH4/on-policy-distillation, 2025.

Prime Intellect. SYNTHETIC-1. https://huggingface.co/collections/PrimeIntellect/synthetic-1-67a2c399cfdd6c9f7fae0c37, 2025.

Valentina Pyatkin, Saumya Malik, Victoria Graf, Hamish Ivison, Shengyi Huang, Pradeep Dasigi, Nathan Lambert, and Hannaneh Hajishirzi. Generalizing verifiable instruction following. *arXiv preprint arXiv:2507.02833*, 2025.

Xiaoqi Qiu, Hao Zeng, Zhiyu Hou, and Hongxin Wei. Provable model provenance set for large language models, 2026. URL https://arxiv.org/abs/2602.00772.

Qwen Team. Qwen2.5 technical report. *arXiv preprint arXiv:2412.15115*, 2024.

Qwen Team. QwQ-32B: Embracing the power of reinforcement learning. https://qwenlm.github.io/blog/qwq-32b/, 2025.

Qwen Team. Qwen3.5: Towards native multimodal agents. https://qwen.ai/blog?id=qwen3.5, 2026a.

Qwen Team. Qwen3.8-Max: A new bar for coding and cowork. https://qwen.ai/blog?id=qwen3.8, 2026b.

Rajat Rawat, Sizhe Chen, Akshay Anand, Michael Duan, Bob Rotsted, and Sewon Min. Reference-based distillation detection in llms, 2026. URL https://arxiv.org/abs/2607.09692.

Tom Sander, Pierre Fernandez, Alain Durmus, Matthijs Douze, and Teddy Furon. Watermarking makes language models radioactive, 2024. URL https://arxiv.org/abs/2402.14904.

Weijia Shi, Anirudh Ajith, Mengzhou Xia, Yangsibo Huang, Daogao Liu, Terra Blevins, Danqi Chen, and Luke Zettlemoyer. Detecting pretraining data from large language models, 2024. URL https://arxiv.org/abs/2310.16789.

Mingjie Sun, Yida Yin, Zhiqiu Xu, J. Zico Kolter, and Zhuang Liu. Idiosyncrasies in large language models, 2025. URL https://arxiv.org/abs/2502.12150.

5 Team, Aohan Zeng, Xin Lv, Qinkai Zheng, Zhenyu Hou, Bin Chen, Chengxing Xie, Cunxiang Wang, Da Yin, Hao Zeng, Jiajie Zhang, Kedong Wang, Lucen Zhong, Mingdao Liu, Rui Lu, Shulin Cao, Xiaohan Zhang, Xuancheng Huang, Yao Wei, Yean Cheng, Yifan An, Yilin Niu, Yuanhao Wen, Yushi Bai, Zhengxiao Du, Zihan Wang, Zilin Zhu, Bohan Zhang, Bosi Wen, Bowen Wu, Bowen Xu, Can Huang, Casey Zhao, Changpeng Cai, Chao Yu, Chen Li, Chendi Ge, Chenghua Huang, Chenhui Zhang, Chenxi Xu, Chenzheng Zhu, Chuang Li, Congfeng Yin, Daoyan Lin, Dayong Yang, Dazhi Jiang, Ding Ai, Erle Zhu, Fei Wang, Gengzheng Pan, Guo Wang, Hailong Sun, Haitao Li, Haiyang Li, Haiyi Hu, Hanyu Zhang, Hao Peng, Hao Tai, Haoke Zhang, Haoran Wang, Haoyu Yang, He Liu, He Zhao, Hongwei Liu, Hongxi Yan, Huan Liu, Huilong Chen, Ji Li, Jiajing Zhao, Jiamin Ren, Jian Jiao, Jiani Zhao, Jianyang Yan, Jiaqi Wang, Jiayi Gui, Jiayue Zhao, Jie Liu, Jijie Li, Jing Li, Jing Lu, Jingsen Wang, Jingwei Yuan, Jingxuan Li, Jingzhao Du, Jinhua Du, Jinxin Liu, Junkai Zhi, Junli Gao, Ke Wang, Lekang Yang, Liang Xu, Lin Fan, Lindong Wu, Lintao Ding, Lu Wang, Man Zhang, Minghao Li, Minghuan Xu, Mingming Zhao, Mingshu Zhai, Pengfan Du, Qian Dong, Shangde Lei, Shangqing Tu, Shangtong Yang, Shaoyou Lu, Shijie Li, Shuang Li, Shuang-Li, Shuxun Yang, Sibo Yi, Tianshu Yu, Wei Tian, Weihan Wang, Wenbo Yu, Weng Lam Tam, Wenjie Liang, Wentao Liu, Xiao Wang, Xiaohan Jia, Xiaotao Gu, Xiaoying Ling, Xin Wang, Xing Fan, Xingru Pan, Xinyuan Zhang, Xinze Zhang, Xiuqing Fu, Xunkai Zhang, Yabo Xu, Yandong Wu, Yida Lu, Yidong Wang, Yilin Zhou, Yiming Pan, Ying Zhang, Yingli Wang, Yingru Li, Yinpei Su, Yipeng Geng, Yitong Zhu, Yongkun Yang, Yuhang Li, Yuhao Wu, Yujiang Li, Yunan Liu, Yunqing Wang, Yuntao Li, Yuxuan Zhang, Zezhen Liu, Zhen Yang, Zhengda Zhou, Zhongpei Qiao, Zhuoer Feng, Zhuorui Liu, Zichen Zhang, Zihan Wang, Zijun Yao, Zikang Wang, Ziqiang Liu, Ziwei Chai, Zixuan Li, Zuodong Zhao, Wenguang Chen, Jidong Zhai, Bin Xu, Minlie Huang, Hongning Wang, Juanzi Li, Yuxiao Dong, and Jie Tang. Glm-4.5: Agentic, reasoning, and coding (arc) foundation models, 2025. URL https://arxiv.org/abs/2508.06471.

Team Olmo. Olmo 3. *arXiv preprint arXiv:2512.13961*, 2025.

Xavier Theimer-Lienhard, Mushtaha El-Amin, Fay Elhassan, Sahaj Vaidya, Victor Cartier-Negadi, David Sasu, Lars Klein, and Mary-Anne Hartley. Fully open meditron: An auditable pipeline for clinical llms, 2026. URL https://arxiv.org/abs/2605.16215.

Somin Wadhwa, Chantal Shaib, Silvio Amir, and Byron C. Wallace. Who taught you that? tracing teachers in model distillation, 2025. URL https://arxiv.org/abs/2502.06659.

Alexander Wan, Kevin Klyman, Sayash Kapoor, Nestor Maslej, Shayne Longpre, Betty Xiong, Percy Liang, and Rishi Bommasani. The 2025 foundation model transparency index, 2025. URL https://arxiv.org/abs/2512.10169.

Boxin Wang, Chankyu Lee, Nayeon Lee, Sheng-Chieh Lin, Wenliang Dai, Yang Chen, Yangyi Chen, Zhuolin Yang, Zihan Liu, Mohammad Shoeybi, Bryan Catanzaro, and Wei Ping. Nemotron-cascade: Scaling cascaded reinforcement learning for general-purpose reasoning models, 2026a. URL https://arxiv.org/abs/2512.13607.

Miles Wang, Robi Lin, Kat Hu, Joy Jiao, Neil Chowdhury, Ethan Chang, and Tejal Patwardhan. Frontierscience: Evaluating ai's ability to perform expert-level scientific tasks, 2026b. URL https://arxiv.org/abs/2601.21165.

Yuxiang Wei, Zhe Wang, Jiawei Liu, Yifeng Ding, and Lingming Zhang. Magicoder: Empowering code generation with OSS-instruct. In *Proceedings of the 41st International Conference on Machine Learning*, volume 235 of *Proceedings of Machine Learning Research*, pp. 52632–52657. PMLR, 21–27 Jul 2024. URL https://proceedings.mlr.press/v235/wei24h.html.

Matt Wesney. ToT-Biology. https://huggingface.co/datasets/mattwesney/ToT-Biology, 2025.

Zhaomin Wu, Haodong Zhao, Ziyang Wang, Jizhou Guo, Qian Wang, and Bingsheng He. Llm dna: Tracing model evolution via functional representations, 2026. URL https://arxiv.org/abs/2509.24496.

Zhangchen Xu, Fengqing Jiang, Luyao Niu, Yuntian Deng, Radha Poovendran, Yejin Choi, and Bill Yuchen Lin. Magpie: Alignment data synthesis from scratch by prompting aligned LLMs with nothing. *arXiv preprint arXiv:2406.08464*, 2024.

An Yang, Anfeng Li, Baosong Yang, Beichen Zhang, Binyuan Hui, Bo Zheng, Bowen Yu, Chang Gao, Chengen Huang, Chenxu Lv, Chujie Zheng, Dayiheng Liu, Fan Zhou, Fei Huang, Feng Hu, Hao Ge, Haoran Wei, Huan Lin, Jialong Tang, Jian Yang, Jianhong Tu, Jianwei Zhang, Jianxin Yang, Jiaxi Yang, Jing Zhou, Jingren Zhou, Junyang Lin, Kai Dang, Keqin Bao, Kexin Yang, Le Yu, Lianghao Deng, Mei Li, Mingfeng Xue, Mingze Li, Pei Zhang, Peng Wang, Qin Zhu, Rui Men, Ruize Gao, Shixuan Liu, Shuang Luo, Tianhao Li, Tianyi Tang, Wenbiao Yin, Xingzhang Ren, Xinyu Wang, Xinyu Zhang, Xuancheng Ren, Yang Fan, Yang Su, Yichang Zhang, Yinger Zhang, Yu Wan, Yuqiong Liu, Zekun Wang, Zeyu Cui, Zhenru Zhang, Zhipeng Zhou, and Zihan Qiu. Qwen3 technical report, 2025a. URL https://arxiv.org/abs/2505.09388.

An Yang, Anfeng Li, Baosong Yang, Beichen Zhang, Binyuan Hui, Bo Zheng, Bowen Yu, Chang Gao, Chengen Huang, Chenxu Lv, Chujie Zheng, Dayiheng Liu, Fan Zhou, Fei Huang, Feng Hu, Hao Ge, Haoran Wei, Huan Lin, Jialong Tang, Jian Yang, Jianhong Tu, Jianwei Zhang, Jianxin Yang, Jiaxi Yang, Jing Zhou, Jingren Zhou, Junyang Lin, Kai Dang, Keqin Bao, Kexin Yang, Le Yu, Lianghao Deng, Mei Li, Mingfeng Xue, Mingze Li, Pei Zhang, Peng Wang, Qin Zhu, Rui Men, Ruize Gao, Shixuan Liu, Shuang Luo, Tianhao Li, Tianyi Tang, Wenbiao Yin, Xingzhang Ren, Xinyu Wang, Xinyu Zhang, Xuancheng Ren, Yang Fan, Yang Su, Yichang Zhang, Yinger Zhang, Yu Wan, Yuqiong Liu, Zekun Wang, Zeyu Cui, Zhenru Zhang, Zhipeng Zhou, and Zihan Qiu. Qwen3 technical report. *arXiv preprint arXiv:2505.09388*, 2025b.

Nicolas Yax, Pierre-Yves Oudeyer, and Stefano Palminteri. Phylolm : Inferring the phylogeny of large language models and predicting their performances in benchmarks, 2025. URL https://arxiv.org/abs/2404.04671.

Qiying Yu, Zheng Zhang, Ruofei Zhu, Yufeng Yuan, Xiaochen Zuo, Yu Yue, Weinan Dai, Tiantian Fan, Gaohong Liu, Lingjun Liu, Xin Liu, Haibin Lin, Zhiqi Lin, Bole Ma, Guangming Sheng, Yuxuan Tong, Chi Zhang, Mofan Zhang, Wang Zhang, Hang Zhu, Jinhua Zhu, Jiaze Chen, Jiangjie Chen, Chengyi Wang, Hongli Yu, Yuxuan Song, Xiangpeng Wei, Hao Zhou, Jingjing Liu, Wei-Ying Ma, Ya-Qin Zhang, Lin Yan, Mu Qiao, Yonghui Wu, and Mingxuan Wang. Dapo: An open-source llm reinforcement learning system at scale, 2025. URL https://arxiv.org/abs/2503.14476.

Z.ai. GLM-4.7. https://z.ai/blog/glm-4.7, 2025.

Yanzhao Zhang, Mingxin Li, Dingkun Long, Xin Zhang, Huan Lin, Baosong Yang, Pengjun Xie, An Yang, Dayiheng Liu, Junyang Lin, Fei Huang, and Jingren Zhou. Qwen3 embedding: Advancing text embedding and reranking through foundation models, 2025. URL https://arxiv.org/abs/2506.05176.

Wenting Zhao, Xiang Ren, Jack Hessel, Claire Cardie, Yejin Choi, and Yuntian Deng. WildChat: 1m ChatGPT interaction logs in the wild. *arXiv preprint arXiv:2405.01470*, 2024.

Lianmin Zheng, Wei-Lin Chiang, Ying Sheng, Tianle Li, Siyuan Zhuang, Zhanghao Wu, Yonghao Zhuang, Zhuohan Li, Zi Lin, Eric P. Xing, Joseph E. Gonzalez, Ion Stoica, and Hao Zhang. Lmsys-chat-1m: A large-scale real-world llm conversation dataset, 2024. URL https://arxiv.org/abs/2309.11998.

Jeffrey Zhou, Tianjian Lu, Swaroop Mishra, Siddhartha Brahma, Sujoy Basu, Yi Luan, Denny Zhou, and Le Hou. Instruction-following evaluation for large language models. *arXiv preprint arXiv:2311.07911*, 2023.

Terry Yue Zhuo, Xiaolong Jin, Hange Liu, Juyong Jiang, Tianyang Liu, Chen Gong, Bhupesh Bishnoi, Vaisakhi Mishra, Marek Suppa, Noah Ziems, Saiteja Utpala, Ming Xu, Guangyu Song, Kaixin Li, Yuhan Cao, Bo Liu, Zheng Liu, Sabina Abdurakhmanova, Wenhao Yu, Mengzhao Jia, Jihan Yao, Kenneth Hamilton, Kumar Shridhar, Minh Chien Vu, Dingmin Wang, Jiawei Liu, Zijian Wang, Qian Liu, Binyuan Hui, Meg Risdal, Ahsen Khaliq, Atin Sood, Zhenchang Xing, Wasi Uddin Ahmad, John Grundy, David Lo, Banghua Zhu, Xiaoning Du, Torsten Scholak, and Leandro von Werra. Bigcodearena: Unveiling more reliable human preferences in code generation via execution, 2025. URL https://arxiv.org/abs/2510.08697.

A Data and Training Construction

A.1 Source Models and Generation Setup

We distill from ten open-weight reasoning models released by six organizations (Table 1). Each source model is queried through its own chat template with thinking mode enabled, and all generation is performed with vLLM. We add no system prompt of our own; if a prompt dataset includes a system prompt, we keep it and pass it to the source model unchanged.

Source model	Abbrev.	Checkpoint	Organization
GPT-OSS-120B (OpenAI, 2025)	Oss-120b	`openai/gpt-oss-120b`	OpenAI
GPT-OSS-20B (OpenAI, 2025)	Oss-20b	`openai/gpt-oss-20b`	OpenAI
Qwen3-235B-A22B (Yang et al., 2025b)	Qwen3	`Qwen/Qwen3-235B-A22B` (thinking mode)	Qwen
QwQ-32B (Qwen Team, 2025)	Qwq	`Qwen/QwQ-32B`	Qwen
DeepSeek-V3.2 (DeepSeek-AI, 2025b)	Ds-v3.2	`deepseek-ai/DeepSeek-V3.2`	DeepSeek
DeepSeek-R1-Distill-Qwen2.5-32B (DeepSeek-AI, 2025a)	R1-Qwen	`deepseek-ai/DeepSeek-R1-Distill-Qwen-32B`	DeepSeek
DeepSeek-R1-Distill-Llama3-70B (DeepSeek-AI, 2025a)	R1-Llama	`deepseek-ai/DeepSeek-R1-Distill-Llama-70B`	DeepSeek
OLMo-3.1-Think-32B (Team Olmo, 2025)	Olmo	`allenai/Olmo-3.1-32B-Think`	AI2
GLM-4.7-Flash (Z.ai, 2025; GLM-4.5 Team, 2025)	Glm	`zai-org/GLM-4.7-Flash`	Z.ai
EXAONE-4.0-32B (Bae et al., 2025)	Exaone	`LGAI-EXAONE/EXAONE-4.0-32B`	LG AI Research

Table 1: Source models. The organization column defines the six groups used for the cross-organization null sets and the organization-level classifiers.

Sampling. Each source model generates three independent responses per prompt, so every prompt contributes three training examples. We sample with temperature 0.7, top-p 0.95 and min-p 0, with at most $16{,}384$ new tokens and a context length of $32{,}768$ tokens. These models are served with tensor parallelism across two H100 GPUs and prefix caching. For GLM, we additionally use its native multi-token-prediction speculative decoding (one draft token) and its reasoning parser; speculative decoding affects throughput but leaves the sampling distribution unchanged.

Trace normalization. Before training, we convert every response to a common format consisting of a thinking block followed by a final answer, `<think>\n{reasoning}\n</think>\n\n{answer}`. When a model returns its reasoning in a separate field, we reinsert it between the think tags. For the Oss models, which use the Harmony response format, the analysis channel becomes the reasoning and the final channel becomes the answer; tool-call headers are rewritten as plain text. Some traces have thinking that doesn't finish within the max token budget; however, there aren't many cases so we leave them in the dataset.

A.2 Prompt Sources and Data Splits

Distillation prompts. For each domain, we mix five or six public prompt datasets (Table 2) to capture a diverse distribution of tasks within that domain. Each dataset is shuffled with seed 2025 and truncated to a fixed per-dataset limit, giving exactly 6,000 prompts per domain. With three responses per prompt, each source model contributes 18k training examples per domain, or 72k across all four domains. All source models answer the same prompts, so distilled models within a matched comparison differ only in the source model that produced their training responses.

We apply the same filters to every dataset. We keep only the first user turn and drop multi-turn conversations so that no other model's assistant turns enter the prompt. We drop prompts shorter than 100 characters or longer than 25,000 characters, and remove exact duplicates across all datasets within a domain.

Domain	Dataset (subset / split)	Prompts
Code	`nvidia/Nemotron-RL-coding-competitive_coding` (NVIDIA, 2025c)	1,200
	`PrimeIntellect/verifiable-coding-problems` (Prime Intellect, 2025)	1,200
	`open-thoughts/OpenThoughts3-1.2M` (domain = code) (Guha et al., 2025b)	1,200
	`nvidia/Llama-Nemotron-Post-Training-Dataset` (SFT / code) (Bercovich et al., 2025b)	1,200
	`PrimeIntellect/real-world-swe-problems` (Prime Intellect, 2025)	1,200
Math	`nvidia/Nemotron-PrismMath` (Jung et al., 2025)	1,200
	`allenai/tulu-3-sft-personas-math` (Lambert et al., 2024; Ge et al., 2024)	1,200
	`open-thoughts/OpenThoughts3-1.2M` (domain = math) (Guha et al., 2025b)	1,200
	`nvidia/Llama-Nemotron-Post-Training-Dataset` (SFT / math) (Bercovich et al., 2025b)	1,200
	`nvidia/Nemotron-Math-Proofs-v1` (lean) (NVIDIA, 2025b)	1,200
Science	`FreedomIntelligence/medical-o1-reasoning-SFT` (en) (Chen et al., 2024)	1,200
	`Intelligent-Internet/II-Medical-Reasoning-SFT` (Intelligent Internet, 2025)	1,200
	`mattwesney/ToT-Biology` (Wesney, 2025)	1,200
	`open-thoughts/OpenThoughts3-1.2M` (domain = science) (Guha et al., 2025b)	1,200
	`nvidia/Nemotron-Science-v1` (MCQ) (NVIDIA, 2025d)	600
	`nvidia/Nemotron-Science-v1` (RQA) (NVIDIA, 2025d)	600
IF / Chat	`argilla/ifeval-like-data` (Xu et al., 2024; Zhou et al., 2023)	1,000
	`allenai/tulu-3-sft-personas-instruction-following` (Lambert et al., 2024; Ge et al., 2024)	1,000
	`allenai/IF_multi_constraints_upto5` (Pyatkin et al., 2025)	1,000
	`allenai/WildChat-1M` (Zhao et al., 2024)	1,000
	`nvidia/Llama-Nemotron-Post-Training-Dataset` (SFT / chat) (Bercovich et al., 2025b)	1,000
	`nvidia/Nemotron-Instruction-Following-Chat-v1` (chat_if) (NVIDIA, 2025a)	1,000

Table 2: Distillation prompt sources. Limits are applied after shuffling with seed 2025.

Classifier and probe prompts. Classifier training data and probes are drawn from a second prompt pool, built from prompt datasets disjoint from the distillation prompt pool. As a result, no distilled model is probed on prompts it was trained on. All ten source models answer every prompt in this pool once. For each source and domain, we split the resulting responses $90/10$ into a *train* split, used only to fit the source-specific classifiers (§B.1.2), and a *test* split, from which probes are selected (§B.2). Every source answers the same prompts, so the split sizes are identical across sources (Table 3).

Split	Code	Math	Science	IF / Chat
Train (classifier training)	40,257	37,260	37,481	36,990
Test (probe candidates)	4,474	4,140	4,165	4,110

Table 3: Responses per source model in each split of the classifier and probe pool.

A.3 Supervised Fine-Tuning

We train each distilled model by full-parameter SFT of an instruction-tuned target model on responses from a single source model, or from a mixture of sources or domains (§A.4). We use TRL's `SFTTrainer` with the target model's chat template, compute loss only on completion tokens, and do not pack sequences. Table 4 lists the shared hyperparameters.

Hyperparameter	Value
Optimizer	AdamW (fused), weight decay 0
Learning rate	1×10^{-4}, linear decay, 10% warmup
Epochs	3
Effective batch size	128 (1 per device $\times$ 16 accumulation $\times$ 8 GPUs)
Max sequence length	32,768 tokens
Gradient clipping	1.0
Precision	bf16, FlashAttention-2
Seed	2026
Hardware	8×H100 80GB (DDP / DeepSpeed ZeRO-2)

Table 4: Shared SFT hyperparameters.

Target models. Our default target model, Qwen2.5-1.5B-Instruct (Qwen Team, 2024), and Llama-3.2-1B-Instruct (Grattafiori et al., 2024) are each distilled from all ten source models in all four domains, with 18k examples per source and domain. Qwen2.5-Coder-1.5B-Instruct (Hui et al., 2024), Qwen2.5-3B-Instruct and Qwen2.5-7B-Instruct are distilled from all ten source models on code only. Because of memory constraints, the 3B and 7B models use an effective batch size of 64 (8 accumulation steps); the 3B models use ZeRO-2, and the 7B models use ZeRO-3 with CPU optimizer offloading and Liger kernels.

Learning-rate selection. We swept learning rates in $\{5 \times 10^{-6}, 10^{-5}, 10^{-4}\}$ on Ds-v3.2 math responses, holding out 10% of the data (16,200/1,800 examples). For both Qwen2.5-1.5B-Instruct and Llama-3.2-1B-Instruct, 10^{-4} yields the lowest held-out loss, reached near the end of the third epoch (Llama 0.775, Qwen 0.670). We train for three epochs in all settings for uniformity. All reported distilled models are trained on their full data without a held-out split.

A.4 Mixture Construction

We construct mixtures along two axes: source identity and domain identity. All mixture models use Qwen2.5-1.5B-Instruct as the target model and follow the SFT recipe in §A.3. We concatenate examples from the relevant components, shuffle the resulting dataset, and then train a single distilled model. For each mixture type, we distinguish between two dataset-size regimes. In the *fixed-cardinality* setting, we downsample the mixture to 18k examples, matching the size of a single-source, single-domain dataset. In the *variable-cardinality* setting, each component retains all 18k examples, so the total training set grows with the number of components.

Multi-source mixtures. We first vary the source while holding the domain fixed to code. We combine responses from three source models, either using 6k examples from each source (`3s-18k`) or all 18k examples from each source (`3s-54k`). For `3s-18k`, we downsample by only select one of the three continuations per prompt, which holds the total amount of training data fixed while reducing each source's contribution by a factor of three relative to its corresponding single-source model. We additionally train a five-source mixture using all 18k examples per source (`5s-90k`).

Mixed-domain mixtures. We next hold the source fixed and mix its responses across all four domains. In the fixed-cardinality setting, we sample 4.5k examples per domain for 18k examples total (`4d-18k`). In the variable-cardinality setting, we retain all 18k examples per domain for 72k examples total (`4d-72k`). We construct both variants independently for each of the ten source models.

Multi-source, cross-domain mixtures. Finally, we vary both source and domain simultaneously. For each three-source configuration, we assign each source a distinct domain and include all 18k examples from each source–domain pair, yielding 54k examples in total. Each source therefore contributes through only one domain, allowing us to distinguish mixing multiple sources within a domain from mixing sources whose contributions come from different domains.

Configurations. We sample the three-source configurations at random. Table 5 lists the sampled source combinations and, for the cross-domain mixtures, their corresponding domain assignments.

Sources	Sizes	Domain assignment
Oss-120b, Ds-v3.2, Qwen3	18k / 54k	code / math / science
R1-Qwen, Ds-v3.2, Qwen3	18k / 54k	math / code / science
Oss-120b, Glm, R1-Llama	18k / 54k	IF/Chat / math / code
Oss-20b, Qwq, Glm	18k / 54k	IF/Chat / science / code
Olmo, Exaone, Ds-v3.2	18k / 54k	IF/Chat / math / science
Oss-120b, Ds-v3.2, Qwen3, Exaone, Glm	90k	Code

Table 5: Multi-source configurations. The last column lists, in order, the domain each source contributes in the one-domain-per-source setting.

A.5 On-Policy Distillation

For the multi-stage experiments, we start from the ten Qwen2.5-7B-Instruct models distilled on code in §A.3, which we use because this target trained reliably across all source models. We then apply a second stage of on-policy distillation (OPD) using a fixed teacher. Following Li et al. (2026b), the student generates its own trajectories and the frozen teacher provides dense, token-level supervision on those trajectories; we use no task reward or verifier.We run separate experiments with Qwen3.5-27B (Qwen Team, 2026a) and Qwen3.8-27B (Qwen Team, 2026b) as the second-stage teacher.

Objective. Because the teacher and Qwen2.5 student use different tokenizers, we perform distillation across vocabularies. We align teacher and student tokens by byte offset and aggregate the teacher's next-token distributions over the span of each student token. We map teacher token ids into the student vocabulary by surface-string matching, covering approximately 87% of the student vocabulary. At each student position, we compute a generalized Jensen–Shannon divergence ($\beta = 0.5$) over the union of the teacher's and student's top-16 tokens (Patiño et al., 2025). We exclude a small window around the `</think>` boundary from the loss.

Prompts and rollouts. We use the first 1,000 problems from `nvidia/Nemotron-RL-coding-competitive_coding` (NVIDIA, 2025c), retaining only the problem statements. We divide training into ten stages of 100 prompts. At each stage, the current student generates one rollout per prompt using temperature 0.3, top-p 0.95, repetition penalty 1.15, and presence penalty 1.0. We allow up to 4,096 reasoning tokens, then close the thinking block and allow up to 2,048 answer tokens. The teacher scores these rollouts, we update the student on them, and the next stage generates new trajectories from the updated student. This iterative procedure keeps the training data close to the student's current policy throughout OPD. We use the checkpoint after the final stage as the two-stage distilled model.

Optimization. We update all student parameters with AdamW in bf16 and use gradient checkpointing, a learning rate of 10^{-6}, gradient clipping at 1.0, and three epochs over each stage's rollouts. The effective batch size is 8 (4 GPUs $\times$ 2 gradient-accumulation steps), and the maximum sequence length is 9,000 tokens. Training runs on a single 8×H100 node: the student occupies four GPUs, while vLLM serves the teacher on the remaining four with tensor parallelism 4.

B METHOD AND EXPERIMENTAL DETAILS

B.1 SIMILARITY SCORE IMPLEMENTATIONS

We follow the notation of Section 3. For a source model S, a comparison model $q \in \{d_S\} \cup \Omega_S$, and a probe x, let $y^S \sim S(\cdot \mid x)$ denote the source's pre-generated response and $y^q \sim q(\cdot \mid x)$ the response generated by q. For each probe, we sample one response from q at temperature 0.7, with a maximum of $1{,}024$ new tokens. We also truncate y^S to its first $1{,}024$ tokens under S's tokenizer so that both responses use the same length budget. We compute each similarity score independently for every probe and pass the resulting scores to Algorithm 1.

B.1.1 REFERENCE LIKELIHOOD

We define $\sigma_{\mathrm{LL}}(S, q, x)$ (Eq. 3) as the mean per-token log-likelihood that q assigns to the source response y^S. We render the prompt with q's chat template, enabling both the generation prompt and thinking. If the rendered prompt already ends with a think tag and y^S begins with one, we remove the duplicate tag. We obtain token log-probabilities with a single vLLM prefill pass, identify the prompt–continuation boundary from tokenizer offsets, and average only over continuation tokens. We use one source response per probe and report the raw mean log-likelihood without further normalization against the target model.

B.1.2 SOURCE-SPECIFIC CLASSIFIER

For each source S and domain, we train a one-vs-rest classifier h_S to distinguish responses generated by S from responses generated by the other nine source models. We train each classifier only on the training split of the shared classifier/probe pool (§A.2, Table 3) and draw all evaluation probes from the test split. Thus, the classifier never sees the source response associated with any evaluation probe during training. We define $\sigma_{\mathrm{clf}}(S, q, x) = h_S(y^q)$ (Eq. 4) as the predicted probability that q's response was generated by S.

Features. We lowercase each response and remove digits and all punctuation except hyphens. For each source, we rank n-grams ($n \in \{1, 2, 3\}$) by their TF-IDF scores relative to responses from the other sources and retain the top $10{,}000$. We then take the union of these source-specific lists across all ten sources to form a shared vocabulary and represent each response as a TF-IDF vector over that vocabulary.

Training. We treat responses from S as positives and pool responses from the other nine sources as negatives. We balance the two classes by subsampling without replacement to the size of the smaller class, then fit an ℓ_2-regularized logistic regression with scikit-learn defaults, `max_iter`$= 1000$, and seed 42.

Organization-level classifiers. For the cross-organization analysis, we train analogous one-vs-rest classifiers over the six organizations in Table 1. We use the same feature representation and classifier, but balance representation across organizations before training. Each organization contributes $10{,}000$ code responses from the same training split, divided evenly across its constituent source models so that organizations represented by multiple models do not receive greater weight. For example, we divide DeepSeek's $10{,}000$ responses evenly across its three source models. Each classifier treats one organization as the positive class and pools responses from the remaining five organizations as negatives, balancing the two classes as above.

B.1.3 EMBEDDING SIMILARITY

Raw cosine similarity We use `Qwen/Qwen3-Embedding-8B` (Zhang et al., 2025) to encode model responses. Embedding similarity is defined as:

$$\sigma_{\mathrm{emb}}(S, q, x) = \cos\left(E(y^S), E(y^q)\right), \qquad y^S \sim S(\cdot \mid x),\ y^q \sim q(\cdot \mid x), \tag{5}$$

where E is a fixed text embedding model.

Centered cosine similarity Responses to the same prompt can share semantic content and task requirements. This common component can obscure differences between models in raw cosine similarity. We therefore apply prompt-wise centering.

Let $\mathcal{M}$ contain all candidate sources and all evaluated models in the current distilled-model family, including the null models. For each probe, we compute the mean embedding across these models:

$$\boldsymbol{\mu}(x) = \frac{1}{|\mathcal{M}|} \sum_{M \in \mathcal{M}} E(x). \tag{6}$$

We subtract this common mean from each embedding:

$$\widetilde{E}(x) = E(x) - \boldsymbol{\mu}(x). \tag{7}$$

We then compute $\sigma_{\text{emb}}(S, q, x)$ using Eq. 5, replacing each original embedding with its centered version.

To see if centering might be effective, we first tested it on source similarities. We select 300 shared prompts from the data used to train the distilled models in each of four domains (Code, Math, Science, and IF/Chat), using the corresponding response from each of the ten source models, yielding 12000 responses to 1200 prompts.

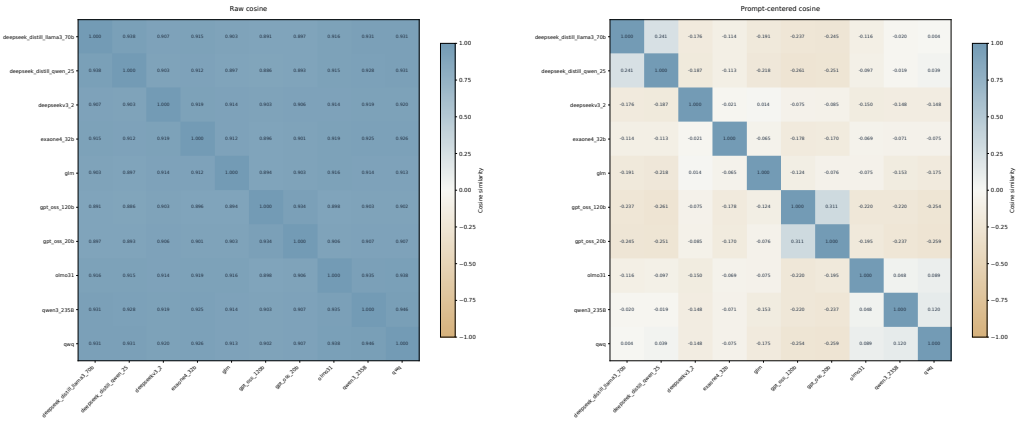


Figure 8: Pairwise cosine similarities between source models before (left) and after (right) prompt-wise centering. Each grid is the cosine similarity between two source' responses to the same prompt, averaged over 1200 prompts.

Figure 8 shows uniformly high raw cosine similarities across source pairs. After centering, similarities vary more substantially between pairs, making differences between source representations more apparent. This observation motivates evaluating centered cosine similarity for source attribution.

Principal Component Analysis Even after prompt-wise centering, the embeddings may contain variation unrelated to source identity. We therefore examine whether retaining their dominant directions of variation improves source attribution.

For each prompt and distilled model and nulls, we first ℓ_2-normalize the centered embeddings and fit a shared PCA to the pooled vectors from all probes and all models in $\mathcal{M}$. We retain the leading 100 (PCA–100) or 15 (PCA–15) principal components, normalize the projected vectors to unit length, and compute cosine similarity between them. Both source and distilled model embeddings use the same fitted projection. We use randomized SVD with random seed zero and no whitening.

Margin standardization While the sign-flip tests use per-prompt margins computed from the cosine similarity in Eq. 5, we additionally standardize the margins when comparing the four embedding-based methods, because centering and PCA can change how much they vary across prompts. For each source-distilled model pair, we first average the margins over prompts, then divide this average by their standard deviation. We do this separately for raw cosine, centered cosine, centered PCA–100 cosine and centered PCA–15 cosine.

Let Δ_j be the true distilled model's similarity minus the largest null;s similarity on probe j, and let $\overline{\Delta}$ be its average over the N probes. We report

$$m_{\text{std}} = \frac{\overline{\Delta}}{\sigma_\Delta}, \qquad \sigma_\Delta = \sqrt{\frac{1}{N}\sum_{j=1}^{N}(\Delta_j - \overline{\Delta})^2}. \tag{8}$$

Results We evaluate single-source, single-domain settings in Code, Math, Science, and IF/Chat, with ten source and two targets: Qwen2.5-1.5B and Llama-3.2-1B. This gives 80 comparisons for each embedding method.

Figure 9 shows that most mean margins are negative and most sign-flip tests do not reach $p < 0.05$ across all four methods. Raw cosine, centered cosine, centered PCA-100, and centered PCA-15 yield 2, 9, 8, and 5 significant comparisons out of 80. These results provide limited evidence for source attribution under the comparative distillation test.

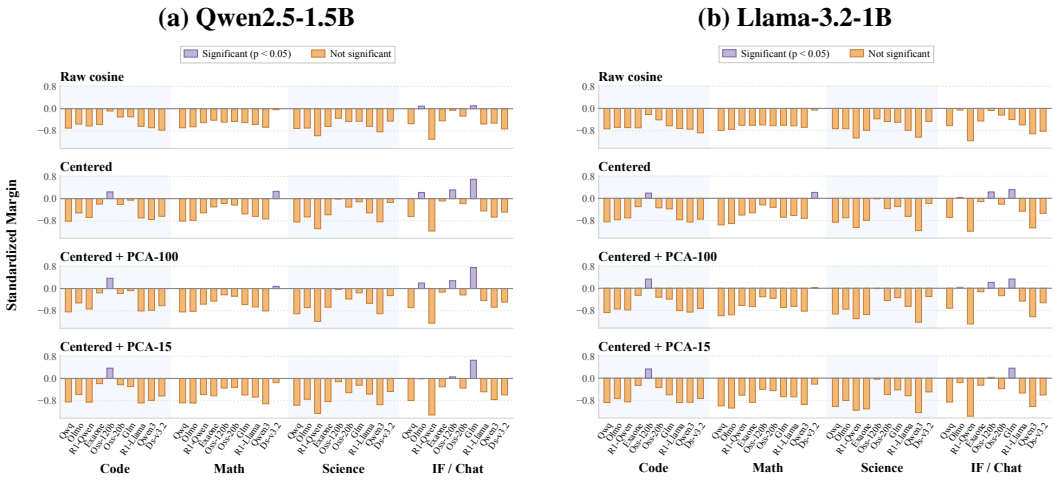


Figure 9: Standardized margins for Qwen2.5-1.5B targets (left) and Llama-3.2-1B targets (right). Rows, from top to bottom, show raw cosine, centered cosine, centered PCA-100, and centered PCA-15. Each bar summarizes one source comparison over 1000 prompts. Purple indicates $p < 0.05$; orange indicates $p \geq 0.05$.

Figure 10 shows that centering increases the mean standardized margin relative to raw cosine in all four domains for the two targets. Adding PCA-100 largely preserves this improvement in half the cases but reduces it in the others. PCA-15 yields lower mean standardized margins than centering alone in all eight. Thus, centering improves the result relative to raw cosine, but PCA does not provide a consistent additional benefit.

One possible explanation is that prompt-wise centering may make differences related to parent–child attribution easier to see, but may also make unrelated differences easier to see. Since PCA keeps the directions in which the embeddings differ the most, it may retain some of these unrelated differences. This may also explain the larger decline with PCA-15. Retaining only 15 components may keep unrelated variation while discarding smaller differences that are useful for source attribution. Retaining 100 components may preserve more of these useful differences.

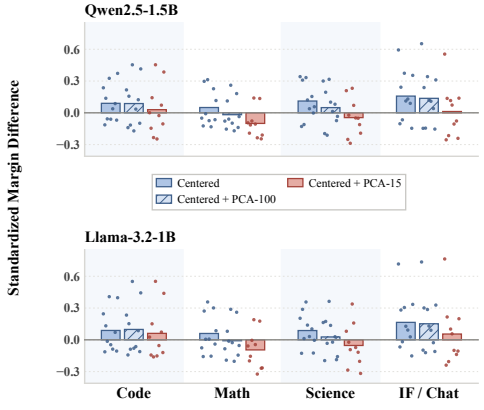


Figure 10: Changes in standardized margin relative to raw cosine, for Qwen2.5-1.5B (top) and Llama-3.2-1B (bottom). Positive values indicate higher standardized margins than raw cosine.

B.2 Probe Construction

B.2.1 Candidate Probe Pools

The classifier and probe pool of §A.2 contains 41k–45k prompts per domain, drawn from datasets disjoint from the distillation data (Table 6). Each dataset is shuffled with seed 2025 and truncated to the listed limit, after which we apply the same filters as for the distillation prompts (first user turn only, length limits, and exact-match deduplication). The candidate probes for each domain are the prompts in the 10% test split (4.1k–4.5k per domain; Table 3), and the source responses to these prompts serve as y^S during scoring.

Domain	Dataset	Limit
Code	Nemotron-Post-Training-v2 (code) (Nathawani et al., 2025)	10,000
	Magicoder-Evol-Instruct-110K (Wei et al., 2024)	10,000
	tulu-3-sft-personas-code (Lambert et al., 2024)	10,000
	bigcodearena-preference-5k (Zhuo et al., 2025)	5,000
	Nemotron-Cascade-SFT-SWE (Wang et al., 2026a)	10,000
Math	Nemotron-Post-Training-v2 (math) (Nathawani et al., 2025)	10,000
	s1K-1.1 (Muennighoff et al., 2025)	1,000
	NuminaMath-1.5 (LI et al., 2024)	10,000
	IMO-AnswerBench (Luong et al., 2025)	400
	Nemotron-Math-v2 (Du et al., 2025)	10,000
	DAPO-Math-17k (Yu et al., 2025)	10,000
Science	Nemotron-Post-Training-v2 (stem) (Nathawani et al., 2025)	10,000
	fully-open-meditron (+ synthetic_moove) (Theimer-Lienhard et al., 2026)	7,500 + 2,500
	hero_run_4_science (Hall et al., 2025)	10,000
	ChemCoTBench (Hao et al., 2026)	1,500
	FrontierScience (Wang et al., 2026b)	160
	Nemotron-SFT-Science-v2 (rqa / so / syn_mcq / vendor) (NVIDIA, 2025d)	$4 \times 2{,}500$
IF / Chat	Nemotron-Post-Training-v2 (chat) (Nathawani et al., 2025)	10,000
	Infinity-Instruct (7M_core) (Li et al., 2025a)	10,000
	infinite-chats-eval (Jiang et al., 2026)	100
	chatbot_arena_conversations (Zheng et al., 2024)	10,000
	arena-human-preference-140k (Zheng et al., 2024)	10,000
	Nemotron-RL-IF-Adversarial (NVIDIA, 2025)	1,000

Table 6: Prompt datasets for the classifier and probe pool.

B.2.2 Heuristic Probe Selection

We rank candidate prompts by how strongly they elicit source-specific behavior, as measured by how well the source-specific classifiers separate the ten sources' responses to that prompt. Letting $\mathcal{S}$ denote the set of source models, we define

$$\rho(x) = \frac{1}{|\mathcal{S}|} \sum_{S \in \mathcal{S}} \frac{1}{|\mathcal{S}|} \sum_{S' \in \mathcal{S}} \begin{cases} h_{S'}(y^S) & S' = S, \\ 1 - h_{S'}(y^S) & S' \neq S. \end{cases} \tag{9}$$

A high $\rho(x)$ indicates that each source's response to x is claimed by its own classifier and rejected by all others. We keep the 1,000 highest-ranked prompts per domain. Selection depends only on source responses and never on any distilled model, so it cannot favor the true distilled model in the test.

B.2.3 Random Probe Baseline

As a baseline, we draw 1,000 prompts per domain uniformly at random (seed 2025) from the same test split, matching the query budget of heuristic selection. Both settings share generation, truncation and scoring, and thus differ only in which prompts are used as probes. To assess sensitivity to the draw, we repeat the random baseline on the single-source and multi-source configurations.

B.2.4 PROBE CHOICE: HEURISTIC VS. RANDOM

Figure 11 compares heuristically selected probes with randomly sampled probes in the single-source setting. Heuristic selection increases the mean margin for nearly every source–domain pair, while the few decreases are negligible. The effect is strongest for the classifier score, where gains reach roughly 0.25 and remain consistent across all four domains. This is partly expected because the heuristic itself ranks prompts using the same classifiers. For likelihood, gains are modest in code, math, and science, but substantially larger in IF/Chat, where open-ended prompts allow more variation in source-specific continuations. We therefore use heuristically selected probes by default.

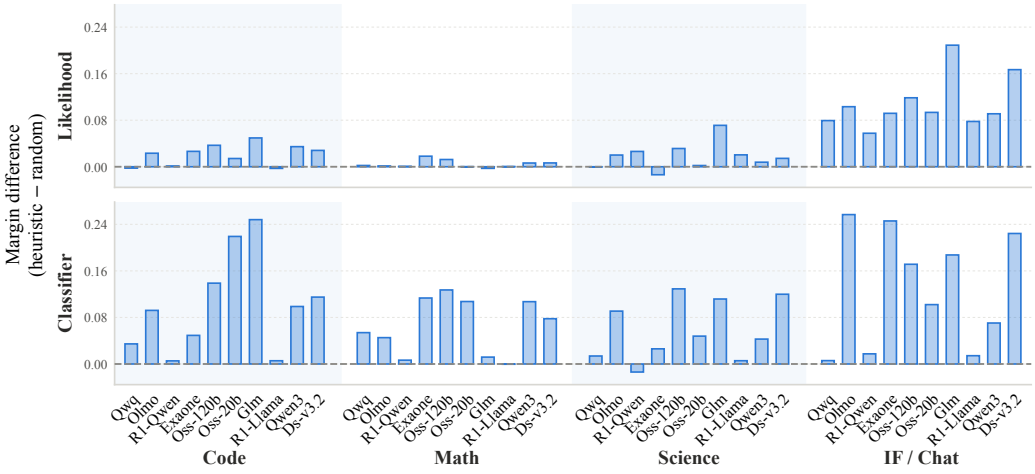


Figure 11: **Effect of probe selection on distillation detectability.** For each single-source distilled model, bars show the change in mean per-probe margin $\widehat{\Delta}$ from using heuristically selected rather than randomly sampled probes (heuristic − random), shown separately for likelihood and classifier scores across the four domains. Both probe sets contain 1,000 prompts per domain drawn from the same candidate pool. Positive values indicate stronger detectability with heuristic selection. Overall, heuristic selection improves detectability for nearly all source–domain pairs, with the largest gains for classifier scores and in IF/Chat.

B.3 TESTING FRAMEWORK

We formalize our oracle comparative distillation test in Algorithm 1.

Algorithm 1 Comparative Distillation Test

Input: Parent P, child c_P, null children Ω, probes $X = \{x_j\}_{j=1}^m$
Parameters: Similarity score s, number of randomizations Π
Output: Mean margin $\widehat{\Delta}$, p-value p

for $j \in [m]$ **do**
  $\Delta_j \leftarrow s(P, c_P, x_j) - \max_{\omega \in \Omega} s(P, \omega, x_j)$
end for
$\widehat{\Delta} \leftarrow m^{-1} \sum_{j=1}^m \Delta_j$
for $\pi \in [\Pi]$ **do**
  $\mathbf{r}^{(\pi)} \sim \mathrm{Unif}(\{-1, +1\}^m)$; $\quad \widehat{\Delta}^{(\pi)} \leftarrow m^{-1} \sum_{j=1}^m r_j^{(\pi)} \Delta_j$
end for
$p \leftarrow \dfrac{1 + \sum_{\pi=1}^{\Pi} \mathbf{1}\{\widehat{\Delta}^{(\pi)} \geq \widehat{\Delta}\}}{1 + \Pi}$
return $\widehat{\Delta}, p$

C Additional Results

C.1 Out-of-Domain Probes

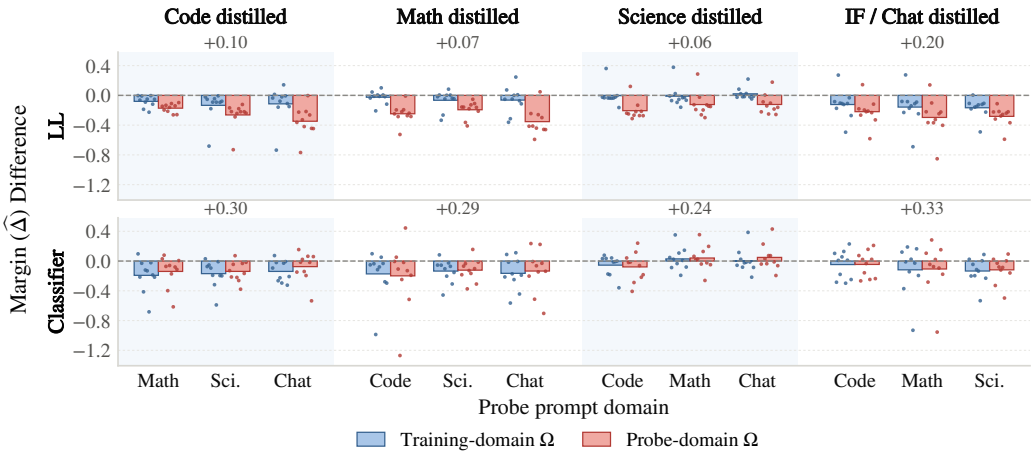


Figure 12: Effect of auditing without knowledge of distillation domain. Bars show the average change in $\widehat{\Delta}$ from the in-domain *baseline* matching the domain of the distilled model; dots show individual sources. Numbers over the column groups represent default $\widehat{\Delta}$ under the *baseline*.

We study how distillation detection is further impacted when the probes used are out-of-domain (OOD) to the distillation domain of the distilled model.

Setup. *OOD probes, same-domain nulls*. We first consider an audit where the domain of distillation is unknown, leading to the use of probes that do not match the distilled models' domain. For example, if the domain of distillation is math, we probe with the code, science, and IF/Chat probes, respectively. We repeat this for each domain. The nulls are the same-domain as the distilled model.

OOD probes, OOD nulls. If the domain of distillation is unknown, a more realistic assumption is that an auditor would use reference models that match the domain of the probes rather than the unknown domain of the distilled model. We repeat the same experiment as the *OOD probes, same-domain nulls* setting, but use nulls that match the domain of the probes per audit.

Detectability generally decreases when using OOD probes. Figure 12 shows that detectability typically decreases in all mismatched domain settings. Under likelihood, when the distilled model and nulls have the same domain (blue bars), detectability drops only a little, with the biggest drops in the IF/Chat distilled models. We suspect that, because the probes are OOD to both the distilled model and nulls, both get worse at modeling source continuations in other domains. The STEM domains may have more capability generalization between themselves, hence the smaller degradations, unlike IF/Chat. On the other hand, when the nulls match the domain of the probes (red bars), detectability degrades much more. Intuitively, the nulls now are more capable of modeling the continuations that match their domain, unlike the distilled source, which must try to find source idiosyncrasies to model in an unfamiliar domain. Classifiers generally decrease in detectability in both settings; it seems that whether the nulls are OOD are not, the classifier can typically determine that the nulls are not carrying source lexical signals. There appears to be a few cases where the classifiers can assign even lower scores to null models when they are in the same domain as the prompt, which appears as slightly less detectability degradation. However, we treat these as noisy results.

In general, auditing with probes from the wrong domain can produce false negatives. These results suggest that a negative result in a domain that may not match the true distillation domain still offers some evidence that the model was not distilled in that domain rather than not distilled from the source at all. However, if the concern is distillation of a particular domain or capability, these results

may still be useful. For example, a model owner with a strong agentic coding or terminal-use model may want to verify that others are not distilling these specific high-value capabilities.

C.2 Cross-Organization Auditing

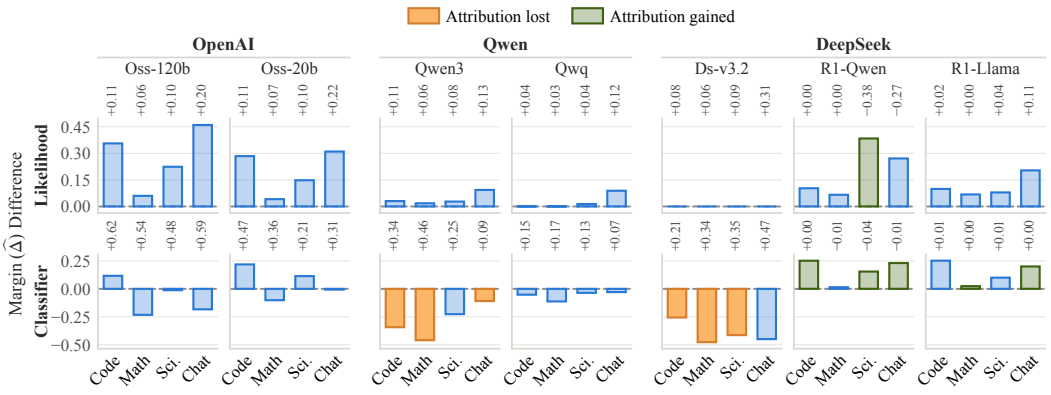


Figure 13: Effect of cross-organization auditing. For each source from an organization with multiple models (grouped by organization), bars show the change in $\widehat{\Delta}$ relative to the canonical *baseline* per audited domain, when same-organization sources are removed from the null set and the source-specific classifier is replaced by an organization-level classifier (§B.1.2; likelihood is unchanged). Orange indicates detection became insignificant; olive indicates detection became significant. Numbers above each bar represent the default $\widehat{\Delta}$ under the *baseline*. Y-axis scales differ between rows.

Our source set includes multiple models from the same organization: Oss-20b and Oss-120b from OpenAI; R1-Qwen, R1-Llama, and Ds-v3.2 from DeepSeek; and Qwq and Qwen3 from Qwen. Our main experiments treat these as distinct sources, requiring attribution to a specific model. We additionally consider an organization-level audit, where the goal is instead to determine whether a model from a candidate organization contributed to distillation. This setting also lets us study how similarity between related source models affects detectability.

Setup. For the seven sources with same-organization counterparts, we repeat the single-source experiments on Qwen2.5-1.5B-Instruct across all four domains. For likelihood, we remove same-organization models from Ω_S, leaving eight nulls for OpenAI and Qwen sources and seven for DeepSeek sources. The similarity score and 1,000 heuristic probes remain unchanged. For the classifier, we additionally replace each source-specific classifier with the corresponding organization-level classifier from §B.1.2, which distinguishes generations from any model in that organization from those of the other organizations. Figure 13 reports the change in $\widehat{\Delta}$ relative to the model-level audit.

Same-organization models can act as particularly strong nulls under likelihood. Figure 13 shows that likelihood detectability generally increases once models from the same organization are removed from the null set, but the effect depends strongly on the source. The largest gains occur for the Oss models and the two R1 distills, where a same-organization model is very often the closest null. This suggests that these models assign relatively similar likelihoods to one another's outputs, making model-level attribution harder even when organization-level attribution remains easy. In contrast, Ds-v3.2 is essentially unchanged because neither R1 distill is typically its closest null, while the Qwen models see only small gains. Overall, removing same-organization nulls raises the mean likelihood margin from 0.059 to 0.182, but changes significance in only one audit. Thus, shared organizational lineage can substantially narrow the margin between related sources without necessarily eliminating detectability outright.

Organization-level classifiers recover shared signals, but can obscure distinctive models within an organization. The classifier results show a different tradeoff. Moving from source-specific

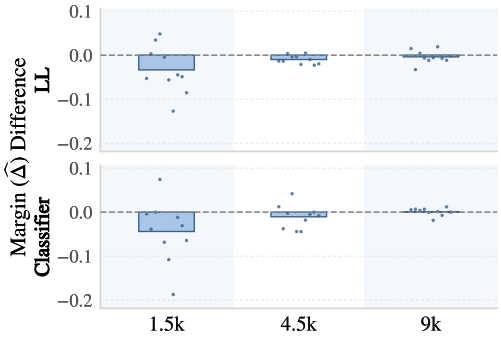


Figure 14: Effect of source cardinality. Bars show the average change in $\widehat{\Delta}$ from the 18k-cardinality *baseline*; dots show individual sources. Per cardinality, distilled models are match to same-cardinality models. **Detectability generally increases with greater cardinality, but saturates quickly.**

to organization-level classifiers substantially improves detectability for R1-Qwen and R1-Llama, which share DeepSeek-R1 supervision and are among the hardest sources to distinguish individually. At the same time, detectability drops sharply for Ds-v3.2 and Qwen3. This suggests that the organization-level classifier learns features that are common across an organization's models rather than simply making attribution easier. When several models share a strong common style, as with the R1 distills, pooling them strengthens that signal; when one model is comparatively distinct from its siblings, as with Ds-v3.2, those organization-level features can become less representative of it. As a result, organization-level attribution is not uniformly easier than model-level attribution: it helps when models within an organization resemble one another, but can hurt the organization's outliers.

Shared lineage can make source-level attribution harder. Taken together, the R1 models provide the clearest example of this effect. R1-Qwen and R1-Llama share final-stage DeepSeek-R1 supervision, are among the least detectable sources in our single-source experiments, and become substantially easier to detect once they no longer need to be distinguished from other DeepSeek models. The organization-level classifier similarly recovers signal that their source-specific classifiers miss. This suggests that shared training lineage can reduce the source-specific differences on which attribution relies. As models increasingly reuse common synthetic data and post-training pipelines, source-level attribution may therefore become harder even when broader organization-level signals remain detectable.

C.3 How Cardinality of Source Data Affects Detectability

Distillation occurs at all scales (Muennighoff et al., 2025; Guha et al., 2025a). We study how cardinality of source data impacts detectability by varying the amount of data our distilled models are trained on.

Setup. We test distillation detectability at four cardinalities: 1.5k, 4.5k, 9k, and 18k. Our experiments match that of the single-source, code *baseline* in §4.1, repeated across each data cardinality. Subsets for each dataset size are randomly sampled from the original 18k code traces per source. We ensure random subsets across each source contain the same sampled prompts. Nulls are the other same-cardinality models unless specified otherwise.

Detectability generally increases with cardinality but saturates quickly. As Figure 14 shows, distilled models trained on more source data are more detectable, likely because the target model has more opportunity to absorb each source's idiosyncrasies. However, the gains saturate quickly, with models distilled on only 4.5k traces generally only slightly less detectable than those distilled on 18k. At 1.5k, the change in margin also varies far more across sources than at 4.5k or 9k, suggesting

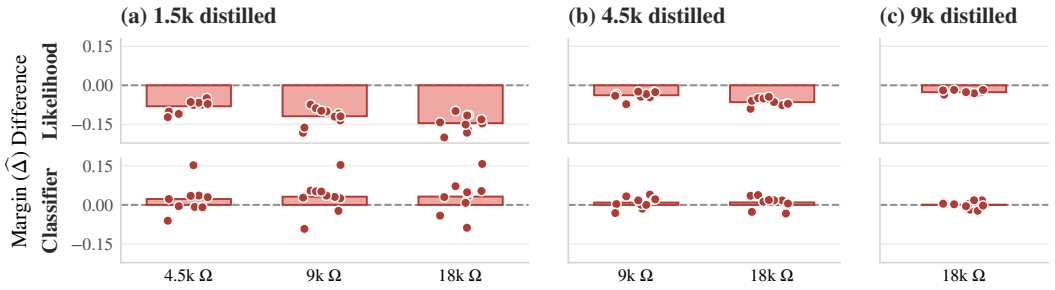


Figure 15: Effect of mismatched source cardinality. Bars show the average change in $\widehat{\Delta}$ from the 18k-cardinality *baseline*; dots show individual sources. Each of (a), (b), and (c) specify the data cardinality of the distilled model; the x-axis denotes the data cardinality of the nulls. **For likelihood, detectability generally decreases the larger the difference in cardinality between distilled model and nulls. Classifier, by constrast, remains robust to cardinality mismatch.**

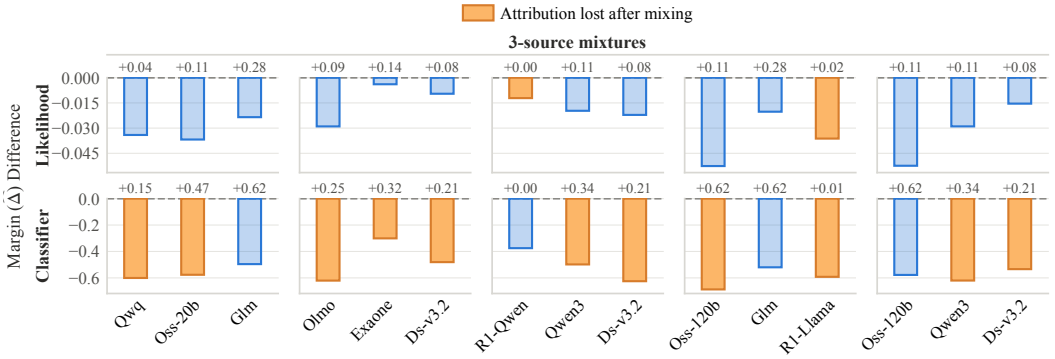


Figure 16: Effect of mixing sources (3s-18k) in the same domain (code). Bars show the change in $\widehat{\Delta}$ relative to the corresponding single-source *baseline* per indicated source audit. Orange indicates the mixing caused detection to become insignificant. Numbers above each bar represent the default $\widehat{\Delta}$ under the respective single-source *baseline*.

that with very little data, how much source signal transfers depends heavily on the particular source and prompt selection.

Likelihood is sensitive to cardinality mismatches, while classifiers remain robust. Figure 15 shows that, under likelihood, detectability generally decreases as the nulls are trained on increasingly more data than the distilled model. We attribute this to the nulls' greater general modeling ability: models trained on more reasoning traces better predict reasoning text in general, including the source's, which narrows the margin. This gap shrinks, however, as the distilled model's cardinality grows. For example, a 9k distilled model loses only 0.027 against 18k nulls, whereas a 1.5k distilled model loses 0.081 against 4.5k nulls, despite the smaller absolute gap. One explanation is that once the distilled model has seen enough source data, it captures source-specific features that additional non-source data cannot give the nulls. Classifier signals, by contrast, remain largely unchanged under cardinality mismatch, varying most when the distilled model is trained on only 1.5k traces. When the amount of a source's data used in distillation is unknown, classifiers may therefore be the more robust auditing method.

For likelihood, cardinality mismatch contributes to detectability degradation when mixing different sources and domains despite fixing total data cardinality. For example, in Figure 16, we see that the 3s-18k distilled models consistently decrease in detectability. When inspecting the

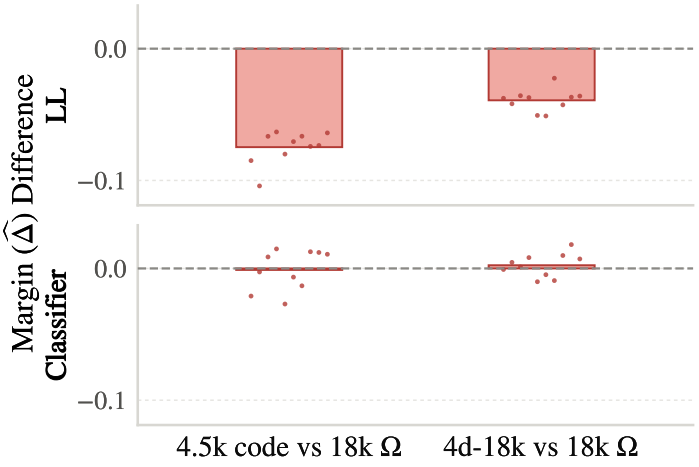


Figure 17: Comparison between the 4.5k vs. 18k single domain cardinality mismatch setting and the 4d-18k vs 18k single domain setting. Both audits occur in the code domain. Bars show the average change in $\widehat{\Delta}$ from the 18k-cardinality code *baseline*; dots show individual sources.

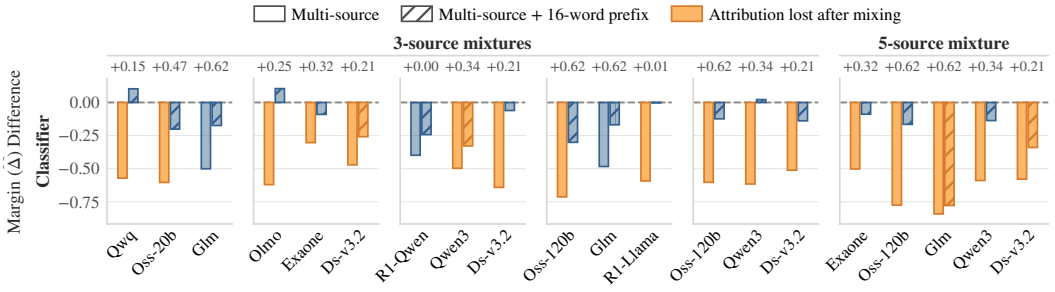


Figure 18: Comparison between standard probing and probing combined with source prefix on the classifier signal in the multi-source code-domain setting. Bars show the change in $\widehat{\Delta}$ from the single-source code *baseline* for each audited source. Numbers over each pair of bars represents the default $\widehat{\Delta}$ from the single-source code *baseline*. **In general, probing with a source prefix results in less detectability degradation.**

cardinality of the source data in the distilled models, we see the effective cardinality comparison is 6k vs. 18k. Ultimately, this cardinality mismatch swallows any change in detectability or capability as a result of mixing with the other sources. The classifier signal's general robustness to cardiinality mismatch also explains why it sees the same massive degradation under the 3s-18k setting as the 3s-54k settings (Figure 3).

A similar effect for likelihood is observed in the 4d-18k setting we studied for mixing domains. When auditing in a specific domain, because detectability is relative to that source's respective single-domain detectability, the effective in-domain cardinality under the 4d-18k models is 4.5k vs the 18k nulls. However, as illustrated in Figure 17, the other 13.5k traces from other domains do provide a weak amplifying effect due to capability increase from seeing more data, hence the margin difference relative to the single-domain 18k *baseline* being smaller than the single domain 4.5k vs 18k comparison setting. Classifier signals again see little change due to cardinality mismatch robustness and lack of influence from other domains.

C.4 Improving Classifier Robustness to Single-Domain Multi-Source Distillation

From the *multi-source, one domain per source* experiments in §4.2, we hypothesized that the in-domain prompts acted as domain priors eliciting learned behaviors from the source contributing that

respective domain. We apply this idea when auditing using classifier signals in the multi-source, code domain setting. For each code probe x and a given source S, we take the continuation $S(x)$ and truncate it to a 16 word prefix $S(x)$[:16]. We run the same test described in §3.1 using the classifier signal, but with $x \parallel S(x)$[:16] for all $x \in X$ as the probes.

Figure 18 demonstrates that, compared to the original classifier approach, adding even a short source prior results in a notable decrease in detectability degradation. There still exists a few settings where degradation approaches that of the classifier strategy without any source prefix (e.g., Glm in the 5s-90k mixture and Qwen3 in the $\{$R1-Qwen, Qwen3, Ds-v3.2$\}$ configurations). We speculate this could again be due to having to learn distinct source distributions under a limited capacity code space as with likelihood, causing less source lexical behaviors to be absorbed. However, we note that the sources per multi-source configuration that drop in detectability under likelihood (Figure 3) do not match those under classifier with source prefixing. Another possibility is that a longer source prefix is needed to elicit specific source behaviors, though we leave that ablation to future work due to compute limitations.

We note that this prefill strategy has similarly been employed in works such as Panfilov et al. (2026); Barbero et al. (2025), where they similarly use prefills as priors to guide distilled models to elicit specific source behaviors. We reinforce the effectiveness of the strategy, but also illuminate possible limitations due to source mixing that it by itself may not be able to overcome.