Beyond Clean or Contaminated: A Regime-Dependent Survey and Empirical Audit of Data Contamination in Large Language Models

Anonymous ACL submission

Abstract

Benchmark contamination threatens the validity of language model evaluation when test instances, answers, derived variants, or benchmark metadata enter model development pipelines. Prior work has studied this problem from many angles, including corpus overlap, semantic paraphrases, preference learning, multimodal shortcuts, and detector design, but the resulting landscape remains fragmented. The prevailing binary view of "clean" versus "contaminated" models also fails to capture how contamination actually behaves. Our survey presents data contamination as a regime-dependent phenomenon rather than a single condition. We organize the problem along three axes: phase (when contamination enters the lifecycle), granularity (what benchmark information is exposed), and modality (which signal carries it). We then synthesize evidence that contamination effects vary with phase, task domain, model scale, decoding protocol, and language. We reorganize detection methods into statistical and causal families, and distinguish detection of exposure from estimation of score inflation. To ground the survey, we conduct a small controlled empirical audit of representative detectors, which shows that separability depends jointly on exposure form, benchmark domain, and detector family. Overall, contamination detection should be read as setting-specific evidence, not a universal guarantee. Future evaluations should report not just whether contamination is suspected, but also which form is considered and whether the signal reflects actual score inflation.

1 Introduction

Recent progress in large language models (LLMs) has made benchmark-based evaluation central to comparing systems (Grattafiori et al., 2024; OpenAI et al., 2024; DeepSeek-AI et al., 2025). Yet modern development pipelines blur the boundary between training and evaluation (Balloccu et al., 2024; Chang et al., 2024). When benchmark instances, answers, or derived variants appear in training, the reported scores on those benchmarks may no longer reflect genuine generalization. This is the problem of data contamination.

A key difficulty is that contamination is not a single condition. The same surface overlap can have very different consequences depending on when it occurs, what form it takes, and how the model is evaluated. Exposure during pre-training, instruction tuning, or preference learning can produce different observed effects. Direct answer leakage, paraphrastic variants, and benchmark metadata are not interchangeable. Score inflation, if it occurs at all, depends on phase, granularity, task type, model scale, and decoding protocol. The prevailing binary view, which classifies models as simply "clean" or "contaminated" based on surface overlap, fails to capture this variation. Crucially, not all overlap produces detectable risk, and not all risk produces measurable score inflation.

Our paper surveys data contamination as a regime-dependent phenomenon. We organize the problem along three axes: when contamination enters the model lifecycle (phase), what benchmark information is exposed (granularity), and which modality carries the signal. We synthesize empirical characteristics that determine whether contamination is observable, retained, and translated into benchmark-score inflation. Finally, we review detection methods and ground the survey with a small controlled empirical audit that illustrates how detector behavior depends on regime.

Distinguishing this work from prior surveys (Appendix A), we expand the contamination scope beyond pre-training to include alignment, retrieval assets, and synthetic variants, and we reorganize detection into complementary statistical and causal paradigms (Deng et al., 2024; Fu et al., 2024). We advance the field by: (1) reframing existing

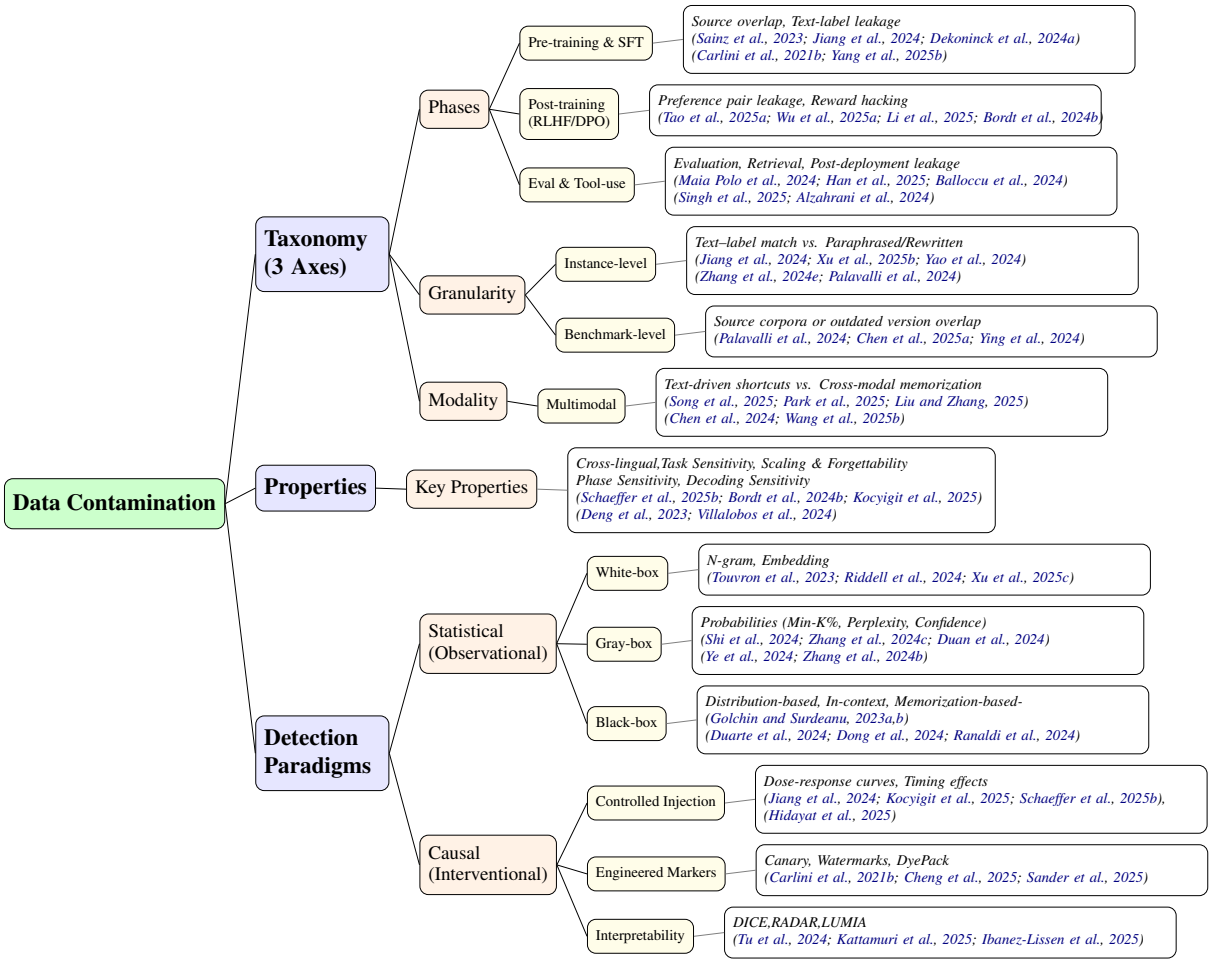

Figure 1: Structure of this paper

contamination categories as a regime-aware, three-dimensional framework that links contamination phase, exposure granularity, and signal modality; (2) characterizing how contamination properties depend on regime rather than treating them as fixed; and (3) using a controlled case study to illustrate that detector separability is setting-specific, not a universal guarantee.

Section 2 introduces the taxonomy. Section 3 synthesizes regime-dependent properties, covering phase sensitivity, task sensitivity, language sensitivity, scale, and memorization. Section 4 reorganizes detection into statistical and causal families. Section 5 presents the empirical audit. Section 6 outlines open directions. Mitigation strategies are reviewed in Appendix D. Our code is available at `https://anonymous.4open.science/r/Data-Contamination-Survey-4F1C`.

2 Taxonomy

Existing studies on data contamination remain fragmented, lacking systematic origins and forms. To address this, we introduce a taxonomy along three axes, *lifecycle phase*, *granularity*, and *modality*.

Data Contamination Let $\mathcal{B}$ denote a benchmark, where each instance $x = (u, y)$ consists of an input u and a target output y. Let $\mathcal{A}_{\text{dev}}$ denote all resources accessible during model development, including pre-training data, supervised fine-tuning data, alignment data, prompts, metadata, and retrieval stores. We say that x is contaminated if there exists $z \in \mathcal{A}_{\text{dev}}$ such that $z \sim_{\text{task}} x$, where $\sim_{\text{task}}$ indicates that z preserves task-relevant information sufficient to identify the instance, infer the target output, or reconstruct the input–output association.

2.1 Contamination Phases

Pre-training Contamination Contamination during pre-training manifests in two phases. First, *initial pre-training* leakage occurs when benchmark instances permeate large-scale web crawls, causing memorization prior to task adaptation (Sainz et al., 2023). Second, risks persist

during *continued pre-training*: integrating newer or domain-specific corpora can unintentionally assimilate evaluation sets, particularly when data sources overlap with benchmark construction (Ke et al., 2023; Chen et al., 2025a).

Supervised Fine-tuning Contamination Instruction datasets may inadvertently include benchmark questions or their templated variants, often via the reformatting of overlapping QA sources (Dekoninck et al., 2024b). Such leakage can promote format memorization and inflate benchmark scores. Moreover, advanced augmentation like Chain-of-Thought (CoT) can make surface overlap harder to detect (Samuel et al., 2024).

Preference Learning Contamination Benchmark contamination can enter preference learning through annotator exposure to test items or the recycling of benchmark responses as preference pairs (Tao et al., 2025a; Wu et al., 2025a). Unlike SFT contamination, this setting affects the preference or reward signal rather than only making benchmark text more likely under the model. As a result, contaminated examples may not exhibit conspicuous anomalies in perplexity (PPL) or likelihood scores, yet they can still bias the reward model or policy toward rigid, reward-hacking behaviors during inference. Appendix B.1 provides the objective-level view.

Evaluation-time External-Resource Contamination For systems with access to tools or search, benchmark related information may be retrieved during evaluation, including solutions, discussions, or near duplicate instances from external sources (Han et al., 2025). This can be viewed as non-parametric contamination.

2.2 Contamination Granularity

Granularity describes what type of benchmark information enters the development pipeline.

2.2.1 Instance Level

Input-only Contamination Input-only contamination (Jiang et al., 2024) occurs when the input or prompt portion of an evaluation example appears in the training corpus, while the corresponding answer or solution is absent. This form exposes the model to benchmark wording, format, or entities, but not to the gold output.

Input-output Contamination Input-output contamination (Jiang et al., 2024; Li et al., 2024e) occurs when both the input and the answer, solution, or task label are present in the training data. This form is more severe because the model may memorize not only the benchmark prompt but also the expected response. It can be further divided into **unpaired contamination**, where inputs and outputs appear separately, and **paired contamination**, where input-output pairs appear explicitly together. The latter is generally more damaging because it directly exposes the mapping evaluated at test time (Yang et al., 2025b). We provide a comparative analysis of text-only and text-label contamination in Appendix B.3.

Semantic Contamination Semantic contamination (Xu et al., 2025b) extends beyond exact text overlap. It occurs when training data contains paraphrases, translations, masked versions, equivalent reasoning problems, or other semantic-preserving transformations. This form can evade surface matching while still transferring the underlying task identity or solution structure (Yao et al., 2024; Palavalli et al., 2024). Building on this view, Xu et al. (2025b) distinguish between **entity contamination**, involving named entities and their implicit biases, and **factual contamination**, involving concrete events, contexts, and detailed assertions.

2.2.2 Benchmark Level

Information-level Contamination Information-level contamination (Xu et al., 2025a) captures exposure to benchmark meta-information rather than exact instances. Examples include label distributions, temporal metadata, annotation rubrics, benchmark construction notes, or external reviews. Such information can shape model priors or decoding behavior, introducing statistical biases even when no individual test item is memorized. We conduct an impact analysis of information-level contamination in Appendix B.4.

Construction-level Contamination Construction level contamination (Palavalli et al., 2024) occurs when the training data includes materials from which a benchmark is built, such as source corpora, earlier benchmark versions or problem generators. Even without exact test instances, the model may learn the benchmark's patterns, templates, or generation rules.

2.3 Contamination Modality

In multimodal evaluation, contamination can appear in two representative forms:

Text-only Contamination When text prompts, labels, or text–label pairs overlap with the training corpus, multimodal models may exploit textual shortcuts that effectively reconstruct test instances. This is consistent with evidence that some VLM benchmarks remain partially solvable even when visual inputs are removed (Chen et al., 2024).

Cross-modal Contamination Cross-modal contamination occurs when training data contains benchmark related image text pairs or image text label triplets, rather than generic cross-modal supervision. Such exposure may allow models to memorize the visual textual associations evaluated at test time, especially when the training pairs duplicate or closely resemble evaluation instances (Song et al., 2025). For reasoning MLLMs, Liu and Zhang (2025) held the visual input fixed while probing a family of related tasks, helping expose contamination driven overfitting that can inflate benchmarks.

3 Properties

Phase Sensitivity Data contamination can manifest throughout the model lifecycle (Balloccu et al., 2024). Kocyigit et al. (2025) found that the training phase at which contamination occurs plays a crucial role in its impact. Early contamination leads to a sharp initial performance increase, but this effect gradually diminishes as training progresses. Later contamination ultimately causes a larger performance inflation. Uniform contamination (across the entire training process) produces the most lasting effects without sharp performance surges.

Decoding Sensitivity The effects of contamination also depend on inference parameters. Schaeffer et al. (2025b) demonstrated that higher sampling temperatures can weaken these contamination driven score boosts. However, this relationship between temperature and memorized output is not always monotonic (Hayes et al., 2025).

Task Sensitivity The impact of data contamination depends on the task, rather than causing uniform score inflation (Golchin and Surdeanu, 2023b; Jiang et al., 2024). For reasoning tasks, contamination can appear as a gap between final answers and reasoning consistency (Ishikawa, 2025). For generation tasks such as summarization, performance gains may depend on whether the leaked content is copied exactly or only reformatted (Palavalli et al., 2024). Moreover, the effect of contamination depends on data availability.

Language Sensitivity Cross-lingual contamination (as an example of semantic contamination in section 2.2.1) can arise when translated versions of benchmark items appear in training data. Such exposure can improve performance on the original benchmark even without direct exposure to the English test set, while also evading surface-level detection methods (Yao et al., 2024; Zhang et al., 2024a). However, its effect depends on language coverage: Kocyigit et al. (2025) find that contamination has little measurable impact for resource-scarce languages. This suggests that multilingual evaluations should audit both original and translated benchmark variants.

Scale and Memorization Model scale and later training both affect whether contamination persists. Larger models often show stronger contamination effects, likely because they are more capable of memorizing and reproducing exposed instances (Kocyigit et al., 2025; Schaeffer et al., 2025b; Riddell et al., 2024). Recent work further suggests that contamination is not a binary state, but depends on exposure magnitude and timing. Schaeffer et al. (2025a) model benchmark inflation as a dose-response effect, while Bordt et al. (2024b) show that later training in data-rich regimes can dilute some memorized traces.

4 Detection Methods

Prior work detects contamination via two paradigms: *statistical* approaches, which infer leakage from observed signals (e.g., overlap, memorization) without training control; and *causal* approaches, which intervene by injecting controlled data or markers to estimate effects.

4.1 Statistical Approaches

We categorize statistical detectors by the level of model access required: white-box (train data), gray-box (probabilities), and black-box (outputs).

4.1.1 White-box

White-box methods use training data or model internals to directly detect contamination.

N-gram based Exact n-gram overlap serves as the prevalent baseline for contamination detection, explicitly tracked in major LLM reports (Achiam et al., 2023). Addressing the computational costs of this method, Xu et al. (2025c) proposed INFINI-GRAM, an FM-index–based engine that facilitates

memory-efficient exact substring search across trillion-token corpora.

Embedding-based To capture semantic contamination beyond lexical matching (Reimers, 2019), embedding similarity was employed for semantic deduplication in instruction tuning (Lee et al., 2023). Crucially, it underpinned the *LLM Decontaminator*, a hybrid pipeline that couples embedding retrieval with LLM verification to detect rephrased leakage (Yang et al., 2023).

4.1.2 Gray-box

Gray-box methods assume partial access to outputs (e.g., token probabilities) and derive contamination signals from perplexity or confidence statistics (in Table 5). These methods are typically efficient. In 2022, Carlini et al. (2022a) established the foundational approach from a membership-inference perspective, showing that thresholding per-example negative log-likelihood (perplexity) serves as a principled baseline. By 2023, reference models were introduced to refine detection: Li (2023) benchmarked perplexity against specific references, while Wei et al. (2023) utilized style-matched sets (e.g., GSM8K) to identify anomalies. Entering 2024, the field expanded into more granular probability analysis. While Duan et al. (2024) critically analyzed why traditional MIAs underperform on LLMs due to fuzzy boundaries, Shi et al. (2024) introduced Min-$K\%$, shifting focus to aggregating low-probability token outliers. This method was rapidly refined: Min-$K\%$++ (Zhang et al., 2024c) added local probability maxima, PAC (Ye et al., 2024) introduced input perturbations, and DC-PDD (Zhang et al., 2024d) incorporated corpus frequency divergences. Parallel to these probability-centric improvements, PaCoST (Zhang et al., 2024b) proposed a threshold-free approach by statistically comparing confidence on original items vs. semantically equivalent paraphrases. Overall, token-based aggregation is cheaper than paraphrase-based generation. Tao et al. (2025b) extends contamination detection to the RL post-training stage and shows that reinforcement learning can induce more confident and lower-entropy output that reveal contaminated data.

4.1.3 Black-box

Black-box methods operate without access to training data or internals and rely solely on model outputs and behavioral assumptions in Appendix C.1.2.

Memorization-based Prompt-and-completion protocols test whether a model can reconstruct masked benchmark content or consistently prefer original instances over perturbed alternatives, suggesting memorization: Golchin and Surdeanu (2023b) propose guided completion-based detection; DCQ (Golchin and Surdeanu, 2023a) used multiple-choice questions with synonym perturbations; Chang et al. (2023) studied memorization via cloze tasks and data archaeology; TS-Guessing (Deng et al., 2023) probed reconstruction of masked elements; DE-COP (Duarte et al., 2024) targets copyrighted content via verbal vs. paraphrased probing; and (Ranaldi et al., 2024) highlighted contamination signals in Text-to-SQL via masked schema recovery. Building on prior taxonomy, recent work has developed modality-aware techniques, including multimodal semantic perturbations that elicit contamination-sensitive behaviors (Park et al., 2025) and dynamic evaluation protocols that generate benchmark variants to reduce pre-training overlap (Yang et al., 2025a; Wang et al., 2025b). Khan et al. (2025) proposed Natural-DaCoDe, combining code naturalness with token-level completion accuracy via a lightweight classifier (SVM) to detect contamination.

Output Distributions and In-context Sensitivity Some methods look for contamination-induced changes in output distributions or contextual responsiveness rather than direct reconstruction. Dong et al. (2024) proposed CDD based on the peakedness of output distributions and pair it with TED for mitigation while preserving evaluation validity. Zawalski et al. (2025) proposed CoDeC, a black-box contamination detection method that identifies memorized examples by measuring how in-context demonstrations influence model predictions. When leakage is rare, Kaneko and Baldwin (2025) showed that few-shot self-detection (in-context labeled leaked vs. non-leaked examples prompting explicit classification) can improve robustness in low-leakage regimes.

Quantifying Contamination Recent work aims to *quantify* contamination intensity and calibrate reported scores. We say a benchmark B is contaminated when the training corpus D_{train} contains benchmark information, either directly or through transformations. Rather than treating contamination as only a binary event, its intensity can be viewed as the fraction of benchmark test in-

stances whose information appears in the training data. Xu et al. (2025a) proposed DCR, an *efficient* framework that aggregates multi-level contamination signals into a unified risk factor and converts raw accuracy into a contamination-adjusted performance estimate for fairer model comparisons. From a representation-change perspective, Choi et al. (2025) introduced a Kernel Divergence Score that quantifies leakage by comparing the kernel similarity structure of sample embeddings before vs. after fine-tuning.

4.2 Causal Approaches

Controlled Injection Causal studies quantify contamination by explicitly injecting benchmark content into training corpora at varying doses (e.g., frequency, timing) to map *dose–response* relationships (Jiang et al., 2024; Yao et al., 2024; Kocyigit et al., 2025). While requiring substantial control over training pipelines, this design offers direct causal attribution of score inflation compared to observational diagnostics (Schaeffer et al., 2025a; Bordt et al., 2024b). For instance, Hidayat et al. (2025) utilized this framework to benchmark detection strategies under controlled leakage, ultimately validating simple n-gram/ROUGE-L matching as more reliable than complex permutation methods (Ni et al., 2025).

Interpretability White-box signals also include mechanistic and activation-level evidence: Tu et al. (2024) proposed DICE, training layer-wise contamination classifiers on intermediate activations during fine-tuning and showing that such signals correlate with performance inflation. Kattamuri et al. (2025) proposed RADAR, a mechanistic interpretability–based detector that separates recall from reasoning using forward-pass features (e.g., confidence convergence and attention/circuit dynamics), surfacing memorization-like behavior without requiring the training corpus. From a related membership-inference perspective, LUMIA (Ibanez-Lissen et al., 2025) trained linear probes over hidden states to discriminate *member* vs. *non-member* samples, providing layer-localized leakage evidence. Zhu et al. (2025) further argued contamination can manifest as shortcut neurons; they proposed identifying such neurons via causal analyses and suppressing them via patching to obtain more trustworthy evaluation behavior.

Marker-Based Verification This approach engineered detectable signals into benchmarks *pre-release* to enable verifiable black-box testing. Techniques included Canary insertion (Carlini et al., 2021a) (measuring recall of synthetic sequences), DyePack (Cheng et al., 2025) (identifying learned backdoor behaviors with provable false-positive control), and Watermarking (Sander et al., 2025) (detecting signal "radioactivity"). Unlike observational methods, these protocols transform detection into a statistically grounded verification procedure with explicit error guarantees.

5 Empirical Audit

The preceding sections argue that contamination detection depends on regime: phase, granularity and task. To illustrate these interactions concretely, we conduct a small controlled audit of representative detectors under fine-tuning contamination.

5.1 Setup

We vary four factors: model scale, task domain, exposure form, and detector. The study uses three Qwen3 models (1.7B, 4B, and 8B), two benchmarks (OlymMATH and MBPP), four contamination conditions, and eight detectors. For each model–benchmark pair, we train separate adapters under matched settings.

Datasets and Contamination Conditions OlymMATH represents mathematical reasoning and MBPP represents code generation. For OlymMATH, we use 100 English easy problems (60 contamination pool, 40 holdout). For MBPP, we use 100 contaminated and 100 holdout examples, with a separate clean pool for background fine-tuning. We compare four conditions: *clean* (no contamination-pool examples), *text-only* (prompt injected without gold answer), *text+label* (prompt and answer injected), and *paraphrase* (prompt rewritten while preserving the task).

Model Fine-tuning. All adapters are trained for three epochs with matched QLoRA and optimization settings detailed in Appendix E.

Detectors and Metrics. We evaluate eight detectors: Min-K%, Min-K++, Zlib, Neighbor Attack, Perplexity Gap, Memorization Score, RefPPLRatio, and CoDeC. Likelihood-based methods score only the benchmark prompt, focusing on prompt membership signals. We report AUROC as the primary metric.

5.2 Results

Table 1 reports AUROC by benchmark, model, and condition. We retain the full detector layout to show how different detector families respond to each regime.

Clean Baseline Clean adapters remain near random on both benchmarks (best AUROC 0.542–0.553 on OlymMATH, 0.509–0.534 on MBPP), confirming that detectors do not trivially separate the two pools when exposure is absent.

OlymMATH Analysis Direct prompt and prompt+answer exposure are both highly detectable across model scales (best AUROC 0.976–0.998 and 0.869–0.958, respectively). Paraphrastic exposure is substantially harder to detect, with best AUROC dropping to 0.652–0.688. Semantic rewriting appears to weaken prompt-level membership signals in mathematical reasoning tasks even when direct exposure is clearly visible.

MBPP Analysis On MBPP, all three contaminated conditions remain separable from holdout examples, and paraphrase is not consistently harder than text-only or text+label. Zlib is the strongest detector in most MBPP contaminated settings (AUROC 0.718–0.873), suggesting that surface or compression-related signals may remain recoverable after paraphrastic rewriting in code-generation prompts.

Model Scale Scale does not show a monotonic relationship with separability. Qwen3-4B often achieves the strongest results, while Qwen3-8B does not consistently improve. We treat model scale as a secondary factor in this setting.

Ablation and Robustness Checks We keep the main text focused on the controlled Qwen3 audit and report supporting analyses in Appendix E. These include downstream evaluation on the original benchmark examples, measured by pass@1 or exact-answer accuracy, a Llama-3.1-8B supplement, a full fine-tuning comparison, an epoch sweep over contamination dose, and bootstrap confidence checks. Together, these analyses provide robustness evidence while keeping the main results centered on detector separability.

5.3 Implications and Practitioner Guide

This audit highlights two broader challenges for contamination detection. First, detectability depends on how benchmark information is exposed during training. Direct prompt exposure and prompt–answer exposure produce strong prompt-level signals, whereas paraphrastic exposure can substantially weaken them. Second, detector reliability is task-dependent. The same paraphrastic condition is more separable on MBPP than on OlymMATH, suggesting that contamination observability depends jointly on the detector, the domain, and the structure of the task.

These observations are consistent with recent evidence that contamination detectors remain fragile in modern LLMs. Existing methods often rely on likelihood, membership, or output-consistency signals, but widely used detectors can exhibit non-monotonic sensitivity and weak cross-method agreement (Dekoninck et al., 2024b; Fu et al., 2024). This issue becomes more severe when leakage is rare, since a small number of contaminated examples may not induce a detectable shift in the overall likelihood distribution. Reasoning-oriented training, including RL and CoT-style augmentation, can further obscure otherwise detectable memorization signals (Wang et al., 2025a; Kocyigit and Yildirim, 2026). Lan et al. (2026) proposes Zero-CoT truncation to reveal evasive contamination, while Janicki et al. (2026) finds that many black-box detectors perform close to random guessing on modern LLMs.

A further limitation is the gap between exposure detection and benchmark score inflation. Strong membership signals may indicate contamination, but the resulting accuracy gain can be small when the affected examples are easy or already solvable. Conversely, contamination may inflate scores through subtle distributional shortcuts, such as format familiarity or label-prior alignment, without producing clear prompt-level membership signals.

Overall, contamination detection remains challenging because memorization signals are shaped by exposure form, leakage frequency, task structure, and training procedures. Future work should couple contamination detection with controlled exposure experiments that measure the resulting change in benchmark performance, while transparently reporting the detectors, hyperparameters, task types, languages, and training methods used in the evaluation.

6 Future Directions

From Qualitative Detection to Contamination Scaling Laws Current research predominantly

Model	Condition	Min-K	Min-K++	Zlib	Neighbor	PPL	Mem	RefPPL	CoDeC
				OlymMATH					
Qwen3-1.7B	Clean	0.528	0.454	0.388	0.470	0.535	0.474	**0.553**	0.509
Qwen3-1.7B	Text only	0.844	0.903	0.931	0.601	0.847	0.658	**0.976**	0.837
Qwen3-1.7B	Text+label	0.760	**0.869**	0.841	0.591	0.772	0.630	0.853	0.825
Qwen3-1.7B	Paraphrase	0.614	0.634	0.576	0.470	0.616	0.416	**0.674**	0.530
Qwen3-4B	Clean	**0.542**	0.490	0.441	0.453	0.541	0.468	0.499	0.525
Qwen3-4B	Text only	0.955	0.972	0.986	0.693	0.957	0.770	**0.998**	0.954
Qwen3-4B	Text+label	0.880	0.932	0.934	0.653	0.881	0.711	0.953	**0.958**
Qwen3-4B	Paraphrase	0.657	0.673	0.654	0.470	0.656	0.472	**0.688**	0.646
Qwen3-8B	Clean	**0.548**	0.446	0.405	0.426	0.546	0.478	**0.548**	0.458
Qwen3-8B	Text only	0.950	0.953	**0.982**	0.654	0.939	0.750	0.939	0.926
Qwen3-8B	Text+label	0.862	0.894	**0.937**	0.667	0.864	0.679	0.903	0.903
Qwen3-8B	Paraphrase	0.649	0.573	0.609	0.502	0.647	0.457	**0.652**	0.547
				MBPP					
Qwen3-1.7B	Clean	0.521	0.528	**0.534**	0.505	0.519	0.484	0.461	0.515
Qwen3-1.7B	Text only	0.603	0.659	**0.718**	0.656	0.630	0.469	0.600	0.592
Qwen3-1.7B	Text+label	0.598	0.656	**0.724**	0.694	0.630	0.472	0.604	0.620
Qwen3-1.7B	Paraphrase	0.603	0.644	**0.723**	0.658	0.635	0.491	0.602	0.587
Qwen3-4B	Clean	0.503	0.505	**0.516**	0.475	0.509	0.454	0.425	0.458
Qwen3-4B	Text only	0.700	0.716	**0.873**	0.721	0.792	0.512	0.737	0.565
Qwen3-4B	Text+label	0.690	0.698	**0.853**	0.692	0.772	0.578	0.710	0.606
Qwen3-4B	Paraphrase	0.702	0.707	**0.873**	0.704	0.795	0.493	0.720	0.512
Qwen3-8B	Clean	0.502	**0.509**	0.505	0.448	0.496	0.474	0.405	0.441
Qwen3-8B	Text only	0.699	0.767	**0.869**	0.700	0.776	0.472	0.740	0.428
Qwen3-8B	Text+label	0.689	0.739	**0.839**	0.735	0.755	0.466	0.723	0.484
Qwen3-8B	Paraphrase	0.686	0.738	**0.852**	0.722	0.765	0.486	NA	0.472

Table 1: Detector AUROC under controlled contamination. Rows grouped by benchmark and model. Bold marks the best detector per row. Min-K and Min-K++ report the best value over K $\in \{10\%, 20\%, 30\%\}$. Contamination pool is positive; holdout pool is negative.

treats contamination as a qualitative binary, identifying *existence* rather than quantifying *impact*. We advocate for formalizing *contamination scaling laws* analogous to scaling laws (Kaplan et al., 2020). Future work should derive functional relationships (e.g., power-law dynamics $\Delta P \propto \alpha \cdot C^{\beta}$) that map leakage density (C) and model scale to benchmark inflation (ΔP). Developing such laws could help move the field from post-hoc forensic analysis toward more predictive contamination analysis, enabling approximate forecasts of score inflation under controlled assumptions and supporting more principled score calibration.

Synthetic Data Contamination A critical frontier is ensuring evaluation robustness amidst *synthetic saturation*. We propose three directions: (i) developing *provenance and quantification* tools to track synthetic exposure and recursive loops in training corpora; (ii) designing *dynamic* reasoning benchmarks with continuously renewable instances to prevent memorization of finite datasets; and (iii) aligning evaluation with *training-time mitigation* strategies (e.g., data curation policies) to prevent model collapse (Shumailov et al., 2024), ensuring benchmarks remain predictive as the web becomes increasingly model-generated.

Data Contamination in MoE Models Many recent models use Mixture-of-Experts (MoE) architectures, where only a few experts are activated for each input. This makes contamination different from that in dense models, since memorized benchmark examples may be concentrated in specific experts or routing modules. Future work should investigate whether contamination induces distinctive routing behaviors and whether expert-level interventions can mitigate its effects.

7 Conclusion

This survey presents data contamination as a threat to benchmark validity that depends on regime, not a single binary condition. We organize the problem along three axes (phase, granularity, modality) and show that contamination effects depend on exposure form, dose, task domain, model scale, decoding protocol, and language. These properties explain why the same contamination source can be visible under one evaluation protocol yet hidden under another, and why detecting overlap or membership signals does not amount to estimating score inflation. Future evaluations should comprehensively report whether contamination is suspected, specify the form and source of the potential contamination, clarify the assumptions and limitations of the detectors used, and assess whether the observed contamination signals translate into measurable score inflation.

8 Limitations

While we extensively cover various forms of data contamination, it is possible that new contamination mechanisms or models may not be fully captured in our analysis. Additionally, our focus is primarily on data contamination within the context of LLMs, and we may not have fully incorporated previous research on data contamination in other areas of machine learning. Given the proliferation of benchmarks, we select representative examples in Appendix to illustrate data construction methods. Additionally, as this survey focuses on LLM data contamination, we may not cover all related areas such as membership inference attacks (MIA), machine unlearning, and LLM memorization. Our study is limited by the under-exploration of dynamic benchmarks and a predominantly technical focus that restricts the assessment of societal risks arising from contamination across specific languages, domains, or demographics.

References

Josh Achiam, Steven Adler, Sandhini Agarwal, Lama Ahmad, Ilge Akkaya, Florencia Leoni Aleman, Diogo Almeida, Janko Altenschmidt, Sam Altman, Shyamal Anadkat, and 1 others. 2023. Gpt-4 technical report. *arXiv preprint arXiv:2303.08774*.

Norah Alzahrani, Hisham Abdullah Alyahya, Yazeed Alnumay, Sultan Alrashed, Shaykhah Alsubaie, Yusef Almushaykeh, Faisal Mirza, Nouf Alotaibi, Nora Altwairesh, Areeb Alowisheq, M. Saiful Bari, and Haidar Khan. 2024. When benchmarks are targets: Revealing the sensitivity of large language model leaderboards. In *ACL*.

Yushi Bai, Jiahao Ying, Yixin Cao, Xin Lv, Yuze He, Xiaozhi Wang, Jifan Yu, Kaisheng Zeng, Yijia Xiao, Haozhe Lyu, and 1 others. 2024. Benchmarking foundation models with language-model-as-an-examiner. *Advances in Neural Information Processing Systems*, 36.

Simone Balloccu, Patrícia Schmidtová, Mateusz Lango, and Ondrej Dusek. 2024. Leak, cheat, repeat: Data contamination and evaluation malpractices in closed-source LLMs. In *Proceedings of the 18th Conference of the European Chapter of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 67–93.

Sebastian Bordt, Harsha Nori, and Rich Caruana. 2023. Elephants never forget: Testing language models for memorization of tabular data. In *NeurIPS 2023 Second Table Representation Learning Workshop*.

Sebastian Bordt, Harsha Nori, Vanessa Rodrigues, Besmira Nushi, and Rich Caruana. 2024a. Elephants never forget: Memorization and learning of tabular data in large language models. In *Conference on Language Modeling (COLM)*.

Sebastian Bordt, Suraj Srinivas, Valentyn Boreiko, and Ulrike von Luxburg. 2024b. How much can we forget about data contamination? *arXiv preprint arXiv:2410.03249*.

Boxi Cao, Mengjie Ren, Hongyu Lin, Xianpei Han, Feng Zhang, Junfeng Zhan, and Le Sun. 2024. Structeval: Deepen and broaden large language model assessment via structured evaluation. *Preprint*, arXiv:2408.03281.

Nicholas Carlini, Steve Chien, Milad Nasr, Shuang Song, Andreas Terzis, and Florian Tramer. 2022a. Membership inference attacks from first principles. In *2022 IEEE symposium on security and privacy (SP)*, pages 1897–1914. IEEE.

Nicholas Carlini, Daphne Ippolito, Matthew Jagielski, Katherine Lee, Florian Tramer, and Chiyuan Zhang. 2022b. Quantifying memorization across neural language models. *arXiv preprint arXiv:2202.07646*.

Nicholas Carlini, Florian Tramer, Eric Wallace, Matthew Jagielski, Ariel Herbert-Voss, Katherine Lee, Adam Roberts, Tom Brown, Dawn Song, Ulfar Erlingsson, Alina Oprea, and Colin Raffel. 2021a. Extracting training data from large language models. *Preprint*, arXiv:2012.07805.

Nicholas Carlini, Florian Tramer, Eric Wallace, Matthew Jagielski, Ariel Herbert-Voss, Katherine Lee, Adam Roberts, Tom Brown, Dawn Song, Ulfar Erlingsson, and 1 others. 2021b. Extracting training data from large language models. In *30th USENIX Security Symposium (USENIX Security 21)*, pages 2633–2650.

Kent Chang, Mackenzie Cramer, Sandeep Soni, and David Bamman. 2023. Speak, memory: An archaeology of books known to ChatGPT/GPT-4. In *Proceedings of the 2023 Conference on Empirical Methods in Natural Language Processing*, pages 7312–7327, Singapore. Association for Computational Linguistics.

Yupeng Chang, Xu Wang, Jindong Wang, Yuan Wu, Linyi Yang, Kaijie Zhu, Hao Chen, Xiaoyuan Yi, Cunxiang Wang, Yidong Wang, and 1 others. 2024. A survey on evaluation of large language models. *ACM Transactions on Intelligent Systems and Technology*, 15(3):1–45.

Lin Chen, Jinsong Li, Xiaoyi Dong, Pan Zhang, Yuhang Zang, Zehui Chen, Haodong Duan, Jiaqi Wang, Yu Qiao, Dahua Lin, and Feng Zhao. 2024. Are we on the right way for evaluating large vision-language models? In *Advances in Neural Information Processing Systems*.

Pin-Er Chen, Da-Chen Lian, Shu-Kai Hsieh, Sieh-Chuen Huang, Hsuan-Lei Shao, Jun-Wei Chiu, Yang-Hsien Lin, Zih-Ching Chen, Cheng-Kuang, Eddie TC

Huang, and Simon See. 2025a. Continual pretraining is (not) what you need in domain adaption. *Preprint*, arXiv:2504.13603.

Simin Chen, Yiming Chen, Zexin Li, Yifan Jiang, Zhongwei Wan, Yixin He, Dezhi Ran, Tianle Gu, Haizhou Li, Tao Xie, and Baishakhi Ray. 2025b. Recent advances in large langauge model benchmarks against data contamination: From static to dynamic evaluation. *Preprint*, arXiv:2502.17521.

Simin Chen, Pranav Pusarla, and Baishakhi Ray. 2025c. Dynamic benchmarking of reasoning capabilities in code large language models under data contamination. *Preprint*, arXiv:2503.04149.

Daixuan Cheng, Yuxian Gu, Shaohan Huang, Junyu Bi, Minlie Huang, and Furu Wei. 2024. Instruction pretraining: Language models are supervised multitask learners. *arXiv preprint arXiv:2406.14491*.

Yize Cheng, Wenxiao Wang, Mazda Moayeri, and Soheil Feizi. 2025. DyePack: Provably flagging test set contamination in LLMs using backdoors. In *Proceedings of the 2025 Conference on Empirical Methods in Natural Language Processing*, pages 15367–15384, Suzhou, China. Association for Computational Linguistics.

Hyeong Kyu Choi, Maxim Khanov, Hongxin Wei, and Yixuan Li. 2025. How contaminated is your benchmark? quantifying dataset leakage in large language models with kernel divergence. *Preprint*, arXiv:2502.00678.

Karl Cobbe, Vineet Kosaraju, Mohammad Bavarian, Mark Chen, Heewoo Jun, Lukasz Kaiser, Matthias Plappert, Jerry Tworek, Jacob Hilton, Reiichiro Nakano, Christopher Hesse, and John Schulman. 2021. Training verifiers to solve math word problems. *arXiv preprint arXiv:2110.14168*.

DeepSeek-AI, Daya Guo, Dejian Yang, Haowei Zhang, Junxiao Song, Ruoyu Zhang, Runxin Xu, Qihao Zhu, Shirong Ma, Peiyi Wang, Xiao Bi, Xiaokang Zhang, Xingkai Yu, Yu Wu, Z. F. Wu, Zhibin Gou, Zhihong Shao, Zhuoshu Li, Ziyi Gao, and 181 others. 2025. Deepseek-r1: Incentivizing reasoning capability in llms via reinforcement learning. *arXiv preprint arXiv:2501.12948*.

Jasper Dekoninck, Mark Niklas Müller, and Martin Vechev. 2024a. Constat: Performance-based contamination detection in large language models. *arXiv preprint arXiv:2405.16281*.

Jasper Dekoninck, Mark Niklas Müller, Maximilian Baader, Marc Fischer, and Martin Vechev. 2024b. Evading data contamination detection for language models is (too) easy. *Preprint*, arXiv:2402.02823.

Chunyuan Deng, Yilun Zhao, Yuzhao Heng, Yitong Li, Jiannan Cao, Xiangru Tang, and Arman Cohan. 2024. Unveiling the spectrum of data contamination in language model: A survey from detection to remediation. In *Findings of the Association for Computational Linguistics: ACL 2024*, pages 16078–16092, Bangkok, Thailand. Association for Computational Linguistics.

Chunyuan Deng, Yilun Zhao, Xiangru Tang, Mark Gerstein, and Arman Cohan. 2023. Investigating data contamination in modern benchmarks for large language models. *arXiv preprint arXiv:2311.09783*.

Yihong Dong, Xue Jiang, Huanyu Liu, Zhi Jin, Bin Gu, Mengfei Yang, and Ge Li. 2024. Generalization or memorization: Data contamination and trustworthy evaluation for large language models. In *Findings of the Association for Computational Linguistics: ACL 2024*, pages 12039–12050.

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

André V. Duarte, Xuandong Zhao, Arlindo L. Oliveira, and Lei Li. 2024. De-cop: Detecting copyrighted content in language models training data. *arXiv preprint arXiv:2402.09910*.

Lizhou Fan, Wenyue Hua, Xiang Li, Kaijie Zhu, Mingyu Jin, Lingyao Li, Haoyang Ling, Jinkui Chi, Jindong Wang, Xin Ma, and Yongfeng Zhang. 2024. Nphardeval4v: A dynamic reasoning benchmark of multimodal large language models. *arXiv preprint arXiv:2403.01777*.

Yujuan Fu, Ozlem Uzuner, Meliha Yetisgen, and Fei Xia. 2024. Does data contamination detection work (well) for llms? a survey and evaluation on detection assumptions. *arXiv preprint arXiv:2410.18966*.

Shahriar Golchin and Mihai Surdeanu. 2023a. Data contamination quiz: A tool to detect and estimate contamination in large language models. *arXiv preprint arXiv:2311.06233*.

Shahriar Golchin and Mihai Surdeanu. 2023b. Time travel in llms: Tracing data contamination in large language models. *arXiv preprint arXiv:2308.08493*.

Aaron Grattafiori, Abhimanyu Dubey, Abhinav Jauhri, Abhinav Pandey, Abhishek Kadian, Ahmad Al-Dahle, Aiesha Letman, Akhil Mathur, Alan Schelten, Alex Vaughan, and 1 others. 2024. The llama 3 herd of models. *arXiv preprint arXiv:2407.21783*.

Ziwen Han, Meher Mankikar, Julian Michael, and Zifan Wang. 2025. Search-time data contamination. *arXiv preprint arXiv:2508.13180*.

Jamie Hayes, Marika Swanberg, Harsh Chaudhari, Itay Yona, Ilia Shumailov, Milad Nasr, Christopher A. Choquette-Choo, Katherine Lee, and A. Feder Cooper. 2025. Measuring memorization in language models via probabilistic extraction. In *Proceedings of the 2025 Conference of the Nations of the Americas*

Chapter of the Association for Computational Linguistics: Human Language Technologies (Volume 1: Long Papers), pages 9266–9291, Albuquerque, New Mexico. Association for Computational Linguistics.

Naila Shafirni Hidayat, Muhammad Dehan Al Kautsar, Alfan Farizki Wicaksono, and Fajri Koto. 2025. Simulating training data leakage in multiple-choice benchmarks for llm evaluation. *Preprint*, arXiv:2505.24263.

Ruijie Hou, Yueyang Jiao, Hanxu Hu, Yingming Li, Wai Lam, Huajian Zhang, and Hongyuan Lu. 2025. Lne-blocking: An efficient framework for contamination mitigation evaluation on large language models. *Preprint*, arXiv:2509.15218.

Luis Ibanez-Lissen, Lorena Gonzalez-Manzano, Jose Maria de Fuentes, Nicolas Anciaux, and Joaquin Garcia-Alfaro. 2025. Lumia: Linear probing for unimodal and multimodal membership inference attacks leveraging internal llm states. *Preprint*, arXiv:2411.19876.

Yui Ishikawa. 2025. Data contamination or genuine generalization? disentangling llm performance on benchmarks. *Academic Journal of Natural Science*, 2(2):16–22.

Naman Jain, King Han, Alex Gu, Wen-Ding Li, Fanjia Yan, Tianjun Zhang, Sida Wang, Armando Solar-Lezama, Koushik Sen, and Ion Stoica. 2024. Livecodebench: Holistic and contamination free evaluation of large language models for code. *arXiv preprint arXiv:2403.07974*.

Juliusz Janicki, Savvas Chamezopoulos, Evangelos Kanoulas, and Georgios Tsatsaronis. 2026. Detecting data contamination in large language models. *arXiv preprint arXiv:2604.19561*.

Minhao Jiang, Ken Ziyu Liu, Ming Zhong, Rylan Schaeffer, Siru Ouyang, Jiawei Han, and Sanmi Koyejo. 2024. Investigating data contamination for pre-training language models. *arXiv preprint arXiv:2401.06059*.

Masahiro Kaneko and Timothy Baldwin. 2025. Investigating how pre-training data leakage affects models' reproduction and detection capabilities. In *Proceedings of the 2025 Conference on Empirical Methods in Natural Language Processing*, pages 23556–23566, Suzhou, China. Association for Computational Linguistics.

Jared Kaplan, Sam McCandlish, Tom Henighan, Tom B. Brown, Benjamin Chess, Rewon Child, Scott Gray, Alec Radford, Jeffrey Wu, and Dario Amodei. 2020. Scaling laws for neural language models. *Preprint*, arXiv:2001.08361.

Antonia Karamolegkou, Jiaang Li, Li Zhou, and Anders Søgaard. 2023. Copyright violations and large language models. In *Proceedings of the 2023 Conference on Empirical Methods in Natural Language Processing*, pages 7403–7412, Singapore. Association for Computational Linguistics.

Ashish Kattamuri, Harshwardhan Fartale, Arpita Vats, Rahul Raja, and Ishita Prasad. 2025. Radar: Mechanistic pathways for detecting data contamination in llm evaluation. *arXiv preprint arXiv:2510.08931*.

Zixuan Ke, Yijia Shao, Haowei Lin, Tatsuya Konishi, Gyuhak Kim, and Bing Liu. 2023. Continual pre-training of language models. *Preprint*, arXiv:2302.03241.

Haris Ali Khan, Yanjie Jiang, Qasim Umer, Yuxia Zhang, Waseem Akram, and Hui Liu. 2025. Has my code been stolen for model training? a naturalness based approach to code contamination detection. *Proc. ACM Softw. Eng.*, 2(FSE).

Muhammed Yusuf Kocyigit, Eleftheria Briakou, Daniel Deutsch, Jiaming Luo, Colin Cherry, and Markus Freitag. 2025. Overestimation in llm evaluation: A controlled large-scale study on data contamination's impact on machine translation. *Preprint*, arXiv:2501.18771.

Muhammed Yusuf Kocyigit and Caglar Yildirim. 2026. The impact of post-training on data contamination. *arXiv preprint arXiv:2601.06103*.

Yifan Lan, Yuanpu Cao, Hanyu Wang, Lu Lin, and Jinghui Chen. 2026. The illusion of reasoning: Exposing evasive data contamination in llms via zero-cot truncation. *Preprint*, arXiv:2605.21856.

Ariel N Lee, Cole J Hunter, and Nataniel Ruiz. 2023. Platypus: Quick, cheap, and powerful refinement of llms. *arXiv preprint arXiv:2308.07317*.

Fangyu Lei, Qian Liu, Yiming Huang, Shizhu He, Jun Zhao, and Kang Liu. 2024. S3Eval: A synthetic, scalable, systematic evaluation suite for large language model. In *Proceedings of the 2024 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies (Volume 1: Long Papers)*, pages 1259–1286.

Dawei Li, Renliang Sun, Yue Huang, Ming Zhong, Bohan Jiang, Jiawei Han, Xiangliang Zhang, Wei Wang, and Huan Liu. 2025. Preference leakage: A contamination problem in llm-as-a-judge. *Preprint*, arXiv:2502.01534.

Jia Li, Ge Li, Xuanming Zhang, Yihong Dong, and Zhi Jin. 2024a. Evocodebench: An evolving code generation benchmark aligned with real-world code repositories. *arXiv preprint arXiv:2404.00599*.

Xiang Li, Yunshi Lan, and Chao Yang. 2024b. Treeeval: Benchmark-free evaluation of large language models through tree planning. *arXiv preprint arXiv:2402.13125*.

Yanyang Li. 2024. Awesome data contamination. `https://github.com/lyy1994/awesome-data-contamination`.

Yanyang Li, Tin Long Wong, Cheung To Hung, Jianqiao Zhao, Duo Zheng, Ka Wai Liu, Michael R. Lyu, and Liwei Wang. 2024c. C^2leva: Toward comprehensive and contamination-free language model evaluation. *arXiv preprint arXiv:2412.04947*.

Yanyang Li, Jianqiao Zhao, Duo Zheng, Zi-Yuan Hu, Zhi Chen, Xiaohui Su, Yongfeng Huang, Shijia Huang, Dahua Lin, Michael Lyu, and Liwei Wang. 2023. CLEVA: Chinese language models EVAluation platform. In *Proceedings of the 2023 Conference on Empirical Methods in Natural Language Processing: System Demonstrations*, pages 186–217.

Yucheng Li. 2023. Estimating contamination via perplexity: Quantifying memorisation in language model evaluation. *arXiv preprint arXiv:2309.10677*.

Yucheng Li, Frank Guerin, and Chenghua Lin. 2024d. Latesteval: Addressing data contamination in language model evaluation through dynamic and time-sensitive test construction. *arXiv preprint arXiv:2312.12343*.

Yucheng Li, Yunhao Guo, Frank Guerin, and Chenghua Lin. 2024e. An open-source data contamination report for large language models. In *Findings of the Association for Computational Linguistics: EMNLP 2024*, pages 528–541, Miami, Florida, USA. Association for Computational Linguistics.

Ming Liu and Wensheng Zhang. 2025. Reasoning multimodal large language model: Data contamination and dynamic evaluation. *Preprint*, arXiv:2506.07202.

Sadegh Mahdavi, Muchen Li, Kaiwen Liu, Christos Thrampoulidis, Leonid Sigal, and Renjie Liao. 2025. Leveraging online olympiad-level math problems for llms training and contamination-resistant evaluation. *Preprint*, arXiv:2501.14275.

Felipe Maia Polo, Ronald Xu, Lucas Weber, Mírian Silva, Onkar Bhardwaj, Leshem Choshen, Allysson Flavio Melo de Oliveira, Yuekai Sun, and Mikhail Yurochkin. 2024. Efficient multi-prompt evaluation of LLMs.

Iman Mirzadeh, Keivan Alizadeh, Hooman Shahrokhi, Oncel Tuzel, Samy Bengio, and Mehrdad Farajtabar. 2025. Gsm-symbolic: Understanding the limitations of mathematical reasoning in large language models. *Preprint*, arXiv:2410.05229.

Shiwen Ni, Xiangtao Kong, Chengming Li, Xiping Hu, Ruifeng Xu, Jia Zhu, and Min Yang. 2025. Training on the benchmark is not all you need. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 39, pages 24948–24956.

OpenAI, Josh Achiam, Steven Adler, Sandhini Agarwal, Lama Ahmad, Ilge Akkaya, Florencia Leoni Aleman, Diogo Almeida, Janko Altenschmidt, Sam Altman, Shyamal Anadkat, Red Avila, Igor Babuschkin, Suchir Balaji, Valerie Balcom, Paul Baltescu, Haiming Bao, Mohammad Bavarian, Jeff Belgum, and 262 others. 2024. Gpt-4 technical report. *arXiv preprint arXiv:2303.08774*.

Medha Palavalli, Amanda Bertsch, and Matthew Gormley. 2024. A taxonomy for data contamination in large language models. In *Proceedings of the 1st Workshop on Data Contamination (CONDA)*, pages 22–40, Bangkok, Thailand. Association for Computational Linguistics.

Jaden Park, Mu Cai, Feng Yao, Jingbo Shang, Soochahn Lee, and Yong Jae Lee. 2025. Contamination detection for vlms using multi-modal semantic perturbation. *arXiv preprint arXiv:2511.03774*.

Long Phan, Alice Gatti, Ziwen Han, and et al. 2025. Humanity's last exam. *Preprint*, arXiv:2501.14249.

Kun Qian, Shunji Wan, Claudia Tang, Youzhi Wang, Xuanming Zhang, Maximillian Chen, and Zhou Yu. 2024. Varbench: Robust language model benchmarking through dynamic variable perturbation. *arXiv preprint arXiv:2406.17681*.

Federico Ranaldi, Elena Sofia Ruzzetti, Dario Onorati, Leonardo Ranaldi, Cristina Giannone, Andrea Favalli, Raniero Romagnoli, and Fabio Massimo Zanzotto. 2024. Investigating the impact of data contamination of large language models in text-to-SQL translation. In *Findings of the Association for Computational Linguistics: ACL 2024*, pages 13909–13920, Bangkok, Thailand. Association for Computational Linguistics.

Mathieu Ravaut, Bosheng Ding, Fangkai Jiao, Hailin Chen, Xingxuan Li, Ruochen Zhao, Chengwei Qin, Caiming Xiong, and Shafiq Joty. 2024. How much are llms contaminated? a comprehensive survey and the llmsanitize library. *arXiv preprint arXiv:2404.00699*.

N Reimers. 2019. Sentence-bert: Sentence embeddings using siamese bert-networks. *arXiv preprint arXiv:1908.10084*.

Martin Riddell, Ansong Ni, and Arman Cohan. 2024. Quantifying contamination in evaluating code generation capabilities of language models. *arXiv preprint arXiv:2403.04811*.

Oscar Sainz, Jon Campos, Iker García-Ferrero, Julen Etxaniz, Oier Lopez de Lacalle, and Eneko Agirre. 2023. NLP evaluation in trouble: On the need to measure LLM data contamination for each benchmark. In *Findings of the Association for Computational Linguistics: EMNLP 2023*, pages 10776–10787.

Vinay Samuel, Yue Zhou, and Henry Peng Zou. 2024. Towards data contamination detection for modern large language models: Limitations, inconsistencies, and oracle challenges. *Preprint*, arXiv:2409.09927.

Tom Sander, Pierre Fernandez, Saeed Mahloujifar, Alain Durmus, and Chuan Guo. 2025. Detecting benchmark contamination through watermarking. *Preprint*, arXiv:2502.17259.

Rylan Schaeffer, Ken Liu, Brando Miranda, Ahmed M Ahmed, Niloofar Mireshghallah, and Sanmi Koyejo. 2025a. The contamination paradox: Why test set leakage can be both potent and negligible. In *NeurIPS 2025 Workshop on Evaluating the Evolving LLM Lifecycle: Benchmarks, Emergent Abilities, and Scaling*.

Rylan Schaeffer, Brando Miranda, Joshua Kazdan, Ken Liu, Ahmed M Ahmed, Niloofar Mireshghallah, and Sanmi Koyejo. 2025b. Causally quantifying the effect of test set contamination on generative benchmarks. In *NeurIPS 2025 Workshop on Evaluating the Evolving LLM Lifecycle: Benchmarks, Emergent Abilities, and Scaling*.

Avi Schwarzschild, Zhili Feng, Pratyush Maini, Zachary C. Lipton, and J. Zico Kolter. 2024. Rethinking llm memorization through the lens of adversarial compression. *arXiv preprint arXiv:2404.15146*.

Zhihong Shao, Peiyi Wang, Qihao Zhu, Runxin Xu, Junxiao Song, Xiao Bi, Haowei Zhang, Mingchuan Zhang, Y. K. Li, Y. Wu, and Daya Guo. 2024. Deepseekmath: Pushing the limits of mathematical reasoning in open language models. *Preprint*, arXiv:2402.03300.

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

Ilia Shumailov, Zakhar Shumaylov, Yiren Zhao, Nicolas Papernot, Ross Anderson, and Yarin Gal. 2024. Ai models collapse when trained on recursively generated data. *Nature*, 631(8022):755–759.

Shivalika Singh, Yiyang Nan, Alex Wang, Daniel D'Souza, Sayash Kapoor, Ahmet Üstün, Sanmi Koyejo, Yuntian Deng, Shayne Longpre, Noah Smith, Beyza Ermis, Marzieh Fadaee, and Sara Hooker. 2025. The leaderboard illusion.

Dingjie Song, Sicheng Lai, Mingxuan Wang, Shunian Chen, Lichao Sun, and Benyou Wang. 2025. Both text and images leaked! a systematic analysis of data contamination in multimodal LLM. In *Findings of the Association for Computational Linguistics: EMNLP 2025*, pages 10527–10542, Suzhou, China. Association for Computational Linguistics.

Saurabh Srivastava, Anto PV, Shashank Menon, Ajay Sukumar, Alan Philipose, Stevin Prince, Sooraj Thomas, and 1 others. 2024. Functional benchmarks for robust evaluation of reasoning performance, and the reasoning gap. *arXiv preprint arXiv:2402.19450*.

Yongding Tao, Tian Wang, Yihong Dong, Huanyu Liu, Kechi Zhang, Xiaolong Hu, and Ge Li. 2025a. Detecting data contamination from reinforcement learning post-training for large language models. *arXiv preprint arXiv:2510.09259*.

Yongding Tao, Tian Wang, Yihong Dong, Huanyu Liu, Kechi Zhang, Xiaolong Hu, and Ge Li. 2025b. Detecting data contamination from reinforcement learning post-training for large language models. *arXiv preprint arXiv:2510.09259*.

Hugo Touvron, Thibaut Lavril, Gautier Izacard, Xavier Martinet, Marie-Anne Lachaux, Timothée Lacroix, Baptiste Rozière, Naman Goyal, Eric Hambro, Faisal Azhar, and 1 others. 2023. Llama: Open and efficient foundation language models. *arXiv preprint arXiv:2302.13971*.

Shangqing Tu, Kejian Zhu, Yushi Bai, Zijun Yao, Lei Hou, and Juanzi Li. 2024. Dice: Detecting in-distribution contamination in llm's fine-tuning phase for math reasoning. *arXiv preprint arXiv:2406.04197*.

Pablo Villalobos, Anson Ho, Jaime Sevilla, Tamay Besiroglu, Lennart Heim, and Marius Hobbhahn. 2024. Will we run out of data? limits of llm scaling based on human-generated data. *Preprint*, arXiv:2211.04325.

Han Wang, Haoyu Li, Brian Ko, and Huan Zhang. 2025a. On the fragility of benchmark contamination detection in reasoning models. *arXiv preprint arXiv:2510.02386*.

Hanqing Wang, Yuan Tian, Mingyu Liu, Zhenhao Zhang, and Xiangyang Zhu. 2025b. Sdeval: Safety dynamic evaluation for multimodal large language models. *arXiv preprint arXiv:2508.06142*.

Siyuan Wang, Zhuohan Long, Zhihao Fan, Zhongyu Wei, and Xuanjing Huang. 2024. Benchmark self-evolving: A multi-agent framework for dynamic llm evaluation. *arXiv preprint arXiv:2402.11443*.

Jason Wei, Nguyen Karina, Hyung Won Chung, Yunxin Joy Jiao, Spencer Papay, Amelia Glaese, John Schulman, and William Fedus. 2024. Measuring short-form factuality in large language models. *Preprint*, arXiv:2411.04368.

Tianwen Wei, Liang Zhao, Lichang Zhang, Bo Zhu, Lijie Wang, Haihua Yang, Biye Li, Cheng Cheng, Weiwei Lü, Rui Hu, Chenxia Li, Liu Yang, Xilin Luo, Xuejie Wu, Lunan Liu, Wenjun Cheng, Peng Cheng, Jianhao Zhang, Xiaoyu Zhang, and 11 others. 2023. Skywork: A more open bilingual foundation model. *Preprint*, arXiv:2310.19341.

Colin White, Samuel Dooley, Manley Roberts, Arka Pal, Ben Feuer, Siddhartha Jain, Ravid Shwartz-Ziv, Neel Jain, Khalid Saifullah, Siddartha Naidu, and 1 others. 2024. Livebench: A challenging, contamination-free llm benchmark. *arXiv preprint arXiv:2406.19314*.

Hengyu Wu and Yang Cao. 2025. Membership inference attacks on large-scale models: A survey. *arXiv preprint arXiv:2503.19338*.

Mingqi Wu, Zhihao Zhang, Qiaole Dong, Zhiheng Xi, Jun Zhao, Senjie Jin, Xiaoran Fan, Yuhao Zhou, Huijie Lv, Ming Zhang, and 1 others. 2025a. Reasoning

or memorization? unreliable results of reinforcement learning due to data contamination. *arXiv preprint arXiv:2507.10532*.

Xiaobao Wu, Liangming Pan, Yuxi Xie, Ruiwen Zhou, Shuai Zhao, Yubo Ma, Mingzhe Du, Rui Mao, Anh Tuan Luu, and William Yang Wang. 2025b. Antileakbench: Preventing data contamination by automatically constructing benchmarks with updated real-world knowledge. *Preprint*, arXiv:2412.13670.

Yunjia Xi, Jianghao Lin, Yongzhao Xiao, Zheli Zhou, Rong Shan, Te Gao, Jiachen Zhu, Weiwen Liu, Yong Yu, and Weinan Zhang. 2025. A survey of llm-based deep search agents: Paradigm, optimization, evaluation, and challenges. *Preprint*, arXiv:2508.05668.

Cheng Xu, Shuhao Guan, Derek Greene, M Kechadi, and 1 others. 2024. Benchmark data contamination of large language models: A survey. *arXiv preprint arXiv:2406.04244*.

Cheng Xu, Nan Yan, Shuhao Guan, Changhong Jin, Yuke Mei, Yibing Guo, and Tahar Kechadi. 2025a. DCR: Quantifying data contamination in LLMs evaluation. In *Proceedings of the 2025 Conference on Empirical Methods in Natural Language Processing*, pages 23013–23031, Suzhou, China. Association for Computational Linguistics.

Cheng Xu, Nan Yan, Shuhao Guan, Yuke Mei, and Tahar Kechadi. 2025b. SSA: Semantic contamination of LLM-driven fake news detection. In *Proceedings of the 2025 Conference on Empirical Methods in Natural Language Processing*, pages 14748–14762, Suzhou, China. Association for Computational Linguistics.

Hao Xu, Jiacheng Liu, Yejin Choi, Noah A. Smith, and Hannaneh Hajishirzi. 2025c. Infini-gram mini: Exact n-gram search at the Internet scale with FM-index. In *Proceedings of the 2025 Conference on Empirical Methods in Natural Language Processing*, pages 24955–24980, Suzhou, China. Association for Computational Linguistics.

Xin Xu, Jiaxin Zhang, Tianhao Chen, Zitong Chao, Jishan Hu, and Can Yang. 2025d. Ugmathbench: A diverse and dynamic benchmark for undergraduate-level mathematical reasoning with large language models. *Preprint*, arXiv:2501.13766.

Vishnu Vardhan Yadoji. 2025. Data centric guard (dc-guard)-a framework for trustworthy llm evaluation. In *NeurIPS 2025 Workshop on Evaluating the Evolving LLM Lifecycle: Benchmarks, Emergent Abilities, and Scaling*.

Shuo Yang, Wei-Lin Chiang, Lianmin Zheng, Joseph E. Gonzalez, and Ion Stoica. 2023. Rethinking benchmark and contamination for language models with rephrased samples. *arXiv preprint arXiv:2311.04850*.

Yue Yang, Shuibo Zhang, Kaipeng Zhang, Yi Bin, Yu Wang, Ping Luo, and Wenqi Shao. 2025a. Dynamic multimodal evaluation with flexible complexity by vision-language bootstrapping. In *International Conference on Learning Representations (ICLR)*. Oral.

Zhen Yang, Hongyi Lin, Yifan He, Jie Xu, Zeyu Sun, Shuo Liu, Pengpeng Wang, Zhongxing Yu, and Qingyuan Liang. 2025b. Rethinking the effects of data contamination in code intelligence. *Preprint*, arXiv:2506.02791.

Feng Yao, Yufan Zhuang, Zihao Sun, Sunan Xu, Animesh Kumar, and Jingbo Shang. 2024. Data contamination can cross language barriers. In *Proceedings of the 2024 Conference on Empirical Methods in Natural Language Processing*, pages 17864–17875.

Wentao Ye, Jiaqi Hu, Liyao Li, Haobo Wang, Gang Chen, and Junbo Zhao. 2024. Data contamination calibration for black-box LLMs. In *Findings of the Association for Computational Linguistics: ACL 2024*, pages 10845–10861.

Jiahao Ying, Yixin Cao, Yushi Bai, Qianru Sun, Bo Wang, Wei Tang, Zhaojun Ding, Yizhe Yang, Xuanjing Huang, and YAN Shuicheng. 2024. Automating dataset updates towards reliable and timely evaluation of large language models. In *The Thirty-eight Conference on Neural Information Processing Systems Datasets and Benchmarks Track*.

Zhuohao Yu, Chang Gao, Wenjin Yao, Yidong Wang, Wei Ye, Jindong Wang, Xing Xie, Yue Zhang, and Shikun Zhang. 2024. KIEval: A knowledge-grounded interactive evaluation framework for large language models. In *Proceedings of the 62nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 5967–5985.

Yang Yue, Zhiqi Chen, Rui Lu, Andrew Zhao, Zhaokai Wang, Yang Yue, Shiji Song, and Gao Huang. 2025. Does reinforcement learning really incentivize reasoning capacity in llms beyond the base model? *Preprint*, arXiv:2504.13837.

Michał Zawalski, Meriem Boubdir, Klaudia Bałazy, Besmira Nushi, and Pablo Ribalta. 2025. Detecting data contamination in llms via in-context learning. *arXiv preprint arXiv:2510.27055*.

Hugh Zhang, Jeff Da, Dean Lee, Vaughn Robinson, Catherine Wu, Will Song, Tiffany Zhao, Pranav Raja, Dylan Slack, Qin Lyu, and 1 others. 2024a. A careful examination of large language model performance on grade school arithmetic. *arXiv preprint arXiv:2405.00332*.

Huixuan Zhang, Yun Lin, and Xiaojun Wan. 2024b. PaCoST: Paired confidence significance testing for benchmark contamination detection in large language models. In *Findings of the Association for Computational Linguistics: EMNLP 2024*, pages 1794–1809.

Jingyang Zhang, Jingwei Sun, Eric Yeats, Yang Ouyang, Martin Kuo, Jianyi Zhang, Hao Frank Yang, and Hai Li. 2024c. Min-k *arXiv preprint arXiv:2404.02936*.

Linghao Zhang, Shilin He, Chaoyun Zhang, Yu Kang, Bowen Li, Chengxing Xie, Junhao Wang, Maoquan Wang, Yufan Huang, Shengyu Fu, Elsie Nallipogu, Qingwei Lin, Yingnong Dang, Saravan Rajmohan, and Dongmei Zhang. 2025. Swe-bench goes live! *Preprint*, arXiv:2505.23419.

Weichao Zhang, Ruqing Zhang, Jiafeng Guo, Maarten de Rijke, Yixing Fan, and Xueqi Cheng. 2024d. Pretraining data detection for large language models: A divergence-based calibration method. In *Proceedings of the 2024 Conference on Empirical Methods in Natural Language Processing*, pages 5263–5274.

Zhehao Zhang, Jiaao Chen, and Diyi Yang. 2024e. Darg: Dynamic evaluation of large language models via adaptive reasoning graph. *arXiv preprint arXiv:2406.17271*.

Qihao Zhao, Yangyu Huang, Tengchao Lv, Lei Cui, Qinzheng Sun, Shaoguang Mao, Xin Zhang, Ying Xin, Qiufeng Yin, Scarlett Li, and Furu Wei. 2024. Mmlu-cf: A contamination-free multi-task language understanding benchmark. *Preprint*, arXiv:2412.15194.

Yang Zhou, Hongyi Liu, Zhuoming Chen, Yuandong Tian, and Beidi Chen. 2025. Gsm-infinite: How do your llms behave over infinitely increasing context length and reasoning complexity? *Preprint*, arXiv:2502.05252.

Kaijie Zhu, Jiaao Chen, Jindong Wang, Neil Zhenqiang Gong, Diyi Yang, and Xing Xie. 2024a. Dyval: Dynamic evaluation of large language models for reasoning tasks. *arXiv preprint arXiv:2309.17167*.

Kaijie Zhu, Jindong Wang, Qinlin Zhao, Ruochen Xu, and Xing Xie. 2024b. Dynamic evaluation of large language models by meta probing agents. *arXiv preprint arXiv:2402.14865*.

Kejian Zhu, Shangqing Tu, Zhuoran Jin, Lei Hou, Juanzi Li, and Jun Zhao. 2025. Establishing trustworthy llm evaluation via shortcut neuron analysis. *Preprint*, arXiv:2506.04142.

Qin Zhu, Qingyuan Cheng, Runyu Peng, Xiaonan Li, Tengxiao Liu, Ru Peng, Xipeng Qiu, and Xuanjing Huang. 2024c. Inference-time decontamination: Reusing leaked benchmarks for large language model evaluation. *arXiv preprint arXiv:2406.13990*.

Wenhong Zhu, Hongkun Hao, Zhiwei He, Yun-Ze Song, Jiao Yueyang, Yumeng Zhang, Hanxu Hu, Yiran Wei, Rui Wang, and Hongyuan Lu. 2024d. CLEAN–EVAL: Clean evaluation on contaminated large language models. In *Findings of the Association for Computational Linguistics: NAACL 2024*, pages 835–847.

A Prior Surveys

Prior research on data contamination primarily focuses on three main areas: definition, detection, and mitigation. Xu et al. (2024) and Deng et al. (2024) provide comprehensive surveys that thoroughly examine data contamination in large language models, covering conceptual definitions, detection methodologies, and mitigation strategies with similar classification frameworks for detection methods (matching-based and comparison-based). However, they differ significantly in their conceptualization of contamination types. The first paper primarily distinguishes between task-level contamination and language-level contamination, providing a function-oriented taxonomy. In contrast, the second paper presents a more granular severity-based hierarchy with four distinct levels: semantic-level (topical overlap), information-level (metadata and distributions), data-level (content without labels), and label-level contamination (complete exposure including ground truth). Their approaches to mitigation strategies also diverge notably. While the first paper emphasizes evaluation guidelines and procedural recommendations, the second paper offers a more structured framework categorized into three comprehensive strategies: data curation, data refactoring, and benchmark-free evaluation.

The remaining three studies focus on specialized subdomains of data contamination. (Fu et al., 2024) focuses on black-box detection assumptions and shows these methods fail under certain conditions, raising concerns about their fundamental reliability. Ravaut et al. (2024) investigates the critical issue of contamination in LLMs, categorizing it into data contamination and model contamination , while further distinguishing between input-only and input-label contamination scenarios. The authors systematically review state-of-the-art detection methods, including string matching, embedding similarity analysis, likelihood-based techniques, and novel LLM-driven approaches, highlighting their strengths and limitations. Chen et al. (2025b) conduct an in-depth analysis of existing static to dynamic benchmark aimed at reducing data contamination risks. Based on this, they propose a series of optimal design principles for dynamic benchmarking and analyze the limitations of existing dynamic benchmarks.

Wu and Cao (2025) systematically reviews *data contamination* in large language models (LLMs), i.e., unintended train–test interactions that can in-

Survey	Definition	Detection	Mitigation	Core Content and Innovation
Deng et al. (2024)	●	●	●	Comprehensive strategies for detection and mitigation.
Ravaut et al. (2024)	○	●	○	Focuses on detection methods.
Xu et al. (2024)	●	●	●	Unified framework for detection and mitigation.
Fu et al. (2024)	○	◐	○	Detection methods assumptions.
Chen et al. (2025b)	○	○	◐	Overview of dynamic benchmarks.
Wu and Cao (2025)	○	●	○	Focuses on detection(MIA) methods.

Table 2: Summary of Recent Surveys on Data Contamination. ● indicates full coverage, ○ means the aspect is not the main focus, and ◐ refers to sub-domain coverage. More details are available in Appendix A.

flate benchmark scores and misrepresent true generalization. It adopts a unified pipeline perspective to clarify definitions/taxonomies and contamination sources across pre-training, SFT, and preference training, and analyzes downstream impacts on evaluation reliability, model comparison, fairness, and safety claims. It further synthesizes detection (white-/gray-/black-box, including MIA-inspired signals) and mitigation strategies (deduplication/filtering, locked protocols to avoid prompt-on-test, and dynamic/contamination-labeled benchmarks), while outlining open challenges such as post-training and multi-modal leakage and the need for standardized reporting.

B Taxonomy

B.1 RL post-training contamination

Recent advancements in Large Language Models (LLMs) have increasingly relied on Reinforcement Learning (RL) post-training, particularly Reinforcement Learning with Verifiable Rewards (RLVR), to enhance reasoning capabilities. Consequently, the RL phase has emerged as a critical, yet often overlooked, source of data contamination.

B.1.1 The Shift from Likelihood to Reward Maximization

The fundamental challenge in characterizing RL contamination stems from the shift in training objectives. Pre-training and SFT are governed by Maximum Likelihood Estimation (MLE), where the model minimizes the negative log-likelihood loss to maximize the probability of observed data sequences. This process naturally imprints strong likelihood-based signals, such as unusually low perplexity for memorized samples, which traditional detectors rely upon.

In stark contrast, RL post-training decouples the model's optimization from simple likelihood metrics. The objective function in RL (e.g., GRPO (Shao et al., 2024)) maximizes the expected reward $\mathcal{R}$ from generated outputs:

$$\mathcal{J}_{\text{RL}}(\theta) = \mathbb{E}_{q\sim D_{\text{RL}},\{o_i\}\sim\pi_{\theta_{\text{old}}}}\left[f(\mathcal{R}(o_i),\pi_\theta)\right] \quad (1)$$

Here, the optimization is driven by external, often sparse reward signals (e.g., 1 for a correct answer, 0 otherwise) rather than the token-level probabilities of a ground-truth response. As a result, contaminated samples in the RL phase do not necessarily exhibit the perplexity anomalies found in SFT contamination, rendering likelihood-based detectors ineffective.

B.1.2 Policy Collapse and Path Dependence

Instead of verbatim memorization, RL contamination manifests as *policy collapse (Yue et al., 2025)*. When a model is trained on a benchmark sample during the RL phase, the policy converges to a narrow, high-reward reasoning path to maximize the metric (e.g., pass@1 accuracy).

This phenomenon creates a distinct signature in the model's output entropy. For contaminated samples, the token-level entropy distribution becomes highly sparse, indicating that the model has become nearly deterministic along a specific trajectory. While general RL training can induce sparsity, contaminated samples exhibit a rigid *path dependence*: they struggle to deviate from the memorized reasoning trace even when explicitly instructed to provide an alternative solution. This rigidity serves as the primary signal for distinguishing RL-contaminated data from clean data.

B.2 Retrieval-time (Tool-use) Contamination

Han et al. (2025) identify a novel leakage modality termed *Search-Time Data Contamination* (STC), which specifically undermines the evaluation of search-based LLM agents (Xi et al., 2025). Unlike traditional contamination where test data leaks into the training corpus, STC occurs during the inference retrieval step when an agent discovers the evaluation dataset itself (e.g., hosted on public platforms like HuggingFace) containing ground-truth

labels. This allows the agent to bypass reasoning by directly extracting answers from the retrieved context. Empirical analysis across benchmarks such as Humanity's Last Exam (HLE) (Phan et al., 2025) and SimpleQA (Wei et al., 2024) reveals that agents successfully retrieve the answer key for approximately 3% of queries. This leakage artificially inflates performance; blocking access to the contaminated sources results in an accuracy drop of approximately 15% on the affected subsets, necessitating new guardrails such as source filtering for reliable agent evaluation.

B.3 Text contamination vs. Text-label contamination

(Li et al., 2024e) demonstrated through controlled experiments that while ground-truth contamination significantly inflates performance on benchmarks such as ARC and HellaSwag, text contamination often yields negligible or even detrimental effects. This distinction is further corroborated by findings in machine translation, where models exposed to ground-truth labels exhibited drastic score inflation (e.g., +30 BLEU) compared to those exposed only to source texts, confirming that the leakage of ground-truth labels is the primary driver of artificial performance gains rather than mere exposure to evaluation text.

B.4 Impact Analysis of Information-Level Contamination

Experiments reveal that Information-Level contamination—exposure to metadata or label distributions without instance leakage—can significantly inflate performance via statistical priors. For instance, InstructLM-1.3B (Cheng et al., 2024)'s accuracy on SST-2 surged from 29.17% to 41.14% solely due to such exposure, suggesting models exploit class imbalances to "guess" answers. This contamination is often implicit and stealthy; models (e.g., Qwen2.5) may internalize metadata from academic papers or online discussions during pre-training. However, sensitivity varies by domain: while classification tasks are highly vulnerable to such biases, reasoning benchmarks like GSM8K (Cobbe et al., 2021) remain resilient. Consequently, the DCR framework explicitly penalizes this risk in Adjusted Accuracy (Acc_{adj}) to distinguish genuine zero-shot reasoning from reliance on prior meta-knowledge.

C Detection

C.1 Definition of Assumptions

C.1.1 Verbatim Memorization

In the context of LLMs, verbatim memorization (Carlini et al., 2021b, 2022b) refers to the phenomenon where a model recalls exact sequences of text, often from the data it has been trained on. This occurs when a model has seen a specific passage or piece of information during its training process and is able to reproduce it exactly when prompted. Verbatim memorization can lead to issues of data contamination, where the model unintentionally outputs copyrighted or sensitive material verbatim, causing concerns regarding privacy, intellectual property, and validity in analytical tasks.

Instance-level contamination (Fu et al., 2024) does not always lead to verbatim memorization. Utilizing instance generation (Carlini et al., 2022b; Karamolegkou et al., 2023), demonstrates that verbatim memorization requires repeated exposures to this instance x during training. Indeed, future research on contamination should place more emphasis on LLMs' memorization. Schwarzschild et al. (2024) proposed that strings can be considered memorized if they can be reproduced using a shorter prompt, while Karamolegkou et al. (2023) investigated verbatim memorization, particularly in the context of copyrighted materials.

C.1.2 Black-Box Method Assumption

Golchin and Surdeanu (2023a) has assumed that when a model has memorized instances from the original dataset, it will prefer selecting options containing the original instance over semantically similar perturbations. Additionally, LLMs may exhibit positional biases, where certain positions in multiple-choice options are more likely to be chosen, leading to potential overestimation or underestimation of contamination levels.

Golchin and Surdeanu (2023b) gave the assumption that by providing a "guided instruction" with dataset name, partition information, and part of the reference instance, LLMs can generate the complete version of the data instance. This allows for calculating overlap between generated completions and reference instances, helping to infer whether the dataset partition is contaminated.

Duarte et al. (2024) assumed that LLMs may memorize specific copyrighted content, such as books or academic papers, during training. When encountering similar content, they can distinguish

Literature	Contamination Granularity Investigated				
	Text	Text and Label	Semantic	Information	Benchmark
Dekoninck et al. (2024b)		✓			✓
Wu and Cao (2025)		✓			
Chen et al. (2025b)		✓	✓		
Palavalli et al. (2024)		✓	✓		✓
Jiang et al. (2024)	✓	✓			
Li et al. (2024e)	✓	✓			
Xu et al. (2024)	✓	✓	✓	✓	
Xu et al. (2025a)	✓	✓	✓	✓	
Xu et al. (2025b)		✓	✓		
Fu et al. (2024)		✓	✓		✓

Table 3: Comparison of contamination types covered in recent studies. The types correspond to: Text (Input-only contamination), Text+Label (Ground-truth contamination), Semantic (Paraphrased or variation-based leakage), Infomation (Information-level or metadata leakage), and Benchmark (Specific test set structure or membership inference).

whether they've seen it before. DE-COP exploits this by designing multiple-choice questions to test if the model can accurately identify original copyrighted content from paraphrased versions. Additionally, model selection biases can affect copyright detection results, and DE-COP introduces a calibration method to minimize such biases.

In (Dong et al., 2024), it is assumed that contaminated training data significantly affects the output distribution of large language models. Specifically, when trained on contaminated data, the model's output distribution becomes more peaked, causing it to produce more consistent outputs on contaminated data, favoring outputs strongly correlated with the training data.

Deng et al. (2023) assumed that if an LLM can accurately guess missing parts of a test set, such as keywords or answer options, without external assistance, it suggests that the model has encountered the corresponding benchmark data during training. This indicates memorization-based contamination. The TS-Guessing protocol tests whether the model has memorized benchmark data by having it guess hidden information.

Ranaldi et al. (2024) assumed that data contamination can be detected solely by analyzing the inputs and outputs of LLMs. For example, unusually high accuracy on tasks from datasets like Spider indicates that the model may have been exposed to this dataset during training, leading to memorization rather than genuine understanding. Additionally, data contamination may lead to inflated performance on zero-shot tasks when the model encounters potentially contaminated data during training.

Chang et al. (2023) assumed that LLMs may memorize portions of text from their training data, especially when evaluation datasets contain known texts. This memorization can lead to inflated performance on tasks such as code generation. Moreover, data repetition on the web—through search engines and open datasets—encourages memorization, which improves accuracy on tasks involving familiar content.

C.2 Additional Methods

In the domain of code benchmarks, Riddell et al. (2024) assess contamination through both surface-level and semantic-level matching. Li et al. (2024e) employ a two-stage detection pipeline: (1) retrieving potential verbatim samples via Bing Search and Common Crawl, and (2) quantifying textual overlap using the METEOR recall score with a threshold of $\tau > 0.75$. More recently, Ni et al. (2025) proposed a gray-box method based on *option shuffling*. For any given n-choice multiple-choice item, they generate all $n!$ permutations of the answer options to obtain a set of derived instances. They then compute the corresponding log-likelihood scores, $P = \{\log p_1, \ldots, \log p_{n!}\}$. By identifying statistically significant outliers—either through a maximum-score check or an Isolation Forest—they interpret these anomalies as evidence of benchmark leakage in the LLM's pre-training data. ConStat (Dekoninck et al., 2024a) reframes

contamination as non-generalizable score inflation on a target benchmark and detects (and quantifies) it by statistically comparing performance on the target benchmark versus a reference benchmark under calibration with a set of reference models.

C.3 Related Tools

C.3.1 Data Contamination Detector

Li et al. (2024e) present Contamination Detector to check whether test examples appear on the internet via Bing search and Common Crawl index. The tool is available at: https://github.com/liyucheng09/Contamination_Detector.

Ravaut et al. (2024) presented an open-source library for contamination detection in NLP datasets and LLMs. The library combines multiple methods for contamination detection and is available at: https://github.com/ntunlp/LLMSanitize.

Overlapy is a Python package developed to evaluate textual overlap (N-Grams) between two volumes of text. This tool can be accessed at: https://github.com/nlx-group/overlapy.

Yao et al. (2024) introduced Deep Contam, a method that detects cross-lingual contamination, which inflates LLMs' benchmark performance while evading existing detection methods. An effective detection method is provided in the repository, accessible at: https://github.com/ShangDataLab/Deep-Contam.

Tu et al. (2024) discussed the detection of in-distribution data contamination using LLM's internal state. The tool is available at: https://github.com/THU-KEG/DICE.

Bordt et al. (2023, 2024a) presented Tabmemcheck, an open-source Python library designed to test language models for memorization of tabular datasets. The package includes four different tests for verbatim memorization of a tabular dataset (header test, row completion test, feature completion test, first token test). It also provides additional heuristics to test what an LLM knows about a tabular dataset, such as feature names test, feature values test, dataset name test, and sampling. The package can be found at: https://github.com/interpretml/LLM-Tabular-Memorization-Checker.

Yang et al. (2023) provided a package that includes the LLM decontaminator, which quantifies a dataset's rephrased samples relative to a benchmark. Based on the detection results, the contamination of rephrased samples in the dataset can be estimated and removed from the training set. This tool is available at: https://github.com/lm-sys/llm-decontaminator.

C.3.2 Benchmarks for Detection Methods

In this section, we enumerate existing benchmarks utilized for quantifying data contamination. These datasets consistently comprise textual content paired with corresponding labels. The annotation of labels within these benchmarks is systematically conducted based on release dates, establishing a temporal framework for analysis. WikiMIA(Shi et al., 2024) datasets serve as a benchmark designed to evaluate membership inference attack (MIA) methods, specifically in detecting pretraining data from extensive large language models. BookMIA(Shi et al., 2024) utilizes book data to evaluate detection methods. Duan et al. (2024) introduce MIMIR, a Python package for evaluating memorization in LLMs, which presents greater challenges than WikiMIA. Zhang et al. (2024d) introduce PatentMIA, specifically designed for Chinese-language pre-training data detection. Ye et al. (2024) propose a StackMIAsub dataset which supports most white- and black-box models, to evaluate detection methods.

C.4 Evidence Collection Platforms

Several initiatives are currently collecting evidence on data contamination. Below are key platforms and resources involved in this effort:

- **The Language Model Contamination Index (LM Contamination Index)**: This is a database used to track and record evidence of language model contamination. For more information, visit: https://hitz-zentroa.github.io/lm-contamination/.
- **CONDA-Workshop Data Contamination Database**: This is a community-driven project focused on the centralized collection of data contamination evidence. The goal is to help the community understand the extent of the problem and assist researchers in avoiding previous mistakes. Detailed information can be found at: https://huggingface.co/spaces/CONDA-Workshop/Data-Contamination-Database.

D Mitigation and Trustworthy Evaluation

This section discusses methods to avoid data contamination in evaluation. First, to reduce risks,

Paradigm	Access needed	Representative methods	Core signal / idea	Strengths / caveats
White-box	Training corpus or internals	N-gram overlap Embedding similarity Layer/activation probes	Directly measure train–test overlap (lexical/semantic) or train classifiers on "sensitive" internal representations to infer membership.	High precision for *developers* who can audit data/internals; may miss heavily transformed leakage without strong semantic tooling; not usable for closed models.
Gray-box	Token probabilities(no weights)	Min-K%, Min-K%++ PAC (perturbation-based) DC-PDD (divergence) PaCoST (paraphrase) Perplexity	Membership inference from probability "outliers" (often via thresholds), sometimes enhanced by perturbations, frequency divergence, or paraphrase consistency tests.	Often efficient (token-based methods need only a constant number of forward passes + light algebra); however sensitivity to k/thresholds and transformation robustness can be an issue; paraphrasing adds compute.
Black-box	API outputs (No token probabilities)	Guided completion DCQ (perturbations) TS-Guessing DE-COP (copyright) CDD (distribution) Canary insertion Natural-DaCoDe	Probe memorization via masked-span reconstruction / completion overlap or MCQ selection of the original instance; estimate contamination from output distribution characteristics; use signals to measure recall	Most practical for *closed* models; but relies on heuristic assumptions and prompt/task design, can be brittle across settings and may be affected by safety filters;

Table 4: Summary of contamination detection paradigms (Section 4): white-/gray-/black-box methods differ by required access, detection signals, and robustness/efficiency trade-offs.

benchmarks are often constructed following three strategies: data updating-based methods, data rewriting-based methods, and prevention-based methods. Second, dynamic evaluation generates adaptive samples using techniques like rule-based methods or agent-based methods. Finally, LLM-as-a-judge eliminates contamination risks, making it a key for contamination-free evaluation. However, there may be issues of preference contamination (Li et al., 2025).

Data Contamination Detection Data contamination detection can be regarded as a specific instance of membership inference attacks (MIA), which were initially introduced to measure memorization and privacy risks (Wu and Cao, 2025).Prior work on data contamination detection in LLMs has mainly focused on the pre-training and Supervised Fine-Tuning stages

D.1 Contamination-free Benchmark Construction Methods

Contamination-free benchmarking strategies ensure datasets stay up-to-date, preventing models from using outdated data. Data rewriting-based methods demonstrate the practical efficacy of paraphrasing techniques in contamination mitigation. Preventive measures involve technical defenses like encryption, access control, and de-contamination during inference to guarantee the reliability and fairness of LLM evaluation.

D.1.1 Data Updating-based Methods

Using the most recent data is intuitive for constructing contamination-free benchmarks, and some studies have proposed automatically collecting recent data to build questions. Meanwhile, recent data also need to maintain the stability of difficulty. LatestEval proposed an automated pipeline to dynamically generate contamination-free test sets from recent materials (Li et al., 2024d). White et al. (2024) introduced LiveBench, a dynamically updated benchmark that integrates tasks across math, coding, and reasoning with automated scoring to mitigate data contamination. Similarly, Jain et al. (2024) introduced LiveCodeBench, a code-generation benchmark that extends prior methodologies by dynamically evaluating self-repair capabilities and maintaining update cycles. Fan et al. (2024) introduced NPHardEval4V-a dynamically updated benchmark to assess reasoning capabilities of MLLMs. In code evaluation, EvoCodeBench (Li et al., 2024a) was proposed to dynamically align with recent code repositories for fair evaluation. Wu et al. (2025b) proposed AntiLeak-Bench, an automated anti-leakage benchmarking framework that constructs samples with explicitly new knowl-

Method	Thresholds	Perturbation	Efficiency	Low-Leakage
Perplexity (Carlini et al., 2022a)	**High** (Requires baseline)	No	**High** (One forward pass)	**Low** Degrades when leakage is rare.
Reference Models (Li, 2023; Wei et al., 2023)	**Medium** (Comparison based)	No	**Medium** (Requires ref model)	**Medium** Improved calibration over raw PPL.
Min-K% (Shi et al., 2024)	**High** (Sensitive to k)	No	**High** (Cheaper than generation)	**Low** Degrades when leakage is rare.
Min-K%++ (Zhang et al., 2024c)	**High**	No	**High**	**Medium** Improved accuracy over Min-K%.
PAC (Ye et al., 2024)	**Medium**	**Yes** (Input perturbation)	**Medium**	**Medium** Robustness via perturbation analysis.
DC-PDD (Zhang et al., 2024d)	**Medium**	No	**High**	**Medium** Focuses on calibration.
PaCoST (Zhang et al., 2024b)	**None** (Threshold-free)	**Yes** (Paraphrasing)	**Low** (Computationally expensive)	**High** Robust(threshold-free design).

Table 5: Summary of Gray-box Data Contamination Detection Methods. We compare methods based on their reliance on thresholds, use of perturbations, computational efficiency, and performance on low-leakage data.

edge not in LLMs' training sets and features a fully automated workflow to build and update the benchmark. Zhang et al. (2025) presents SWE-bench-Live, a live-updatable benchmark for evaluating LLMs in issue-resolving tasks, an automated curation pipeline to enable continuous updates. Mahdavi et al. (2025) presents an automated pipeline to create the AoPS-Instruct dataset, and introduces LiveAoPSBench, an evolving contamination-resistant evaluation set with timestamps.

D.1.2 Data Rewriting-based Methods

This type of methods use data augmentation to remove contamination from benchmarks, with LLMs' superior rephrasing and verifying capabilities. Zhu et al. (2024d) proposed Clean-Eval to purify contaminated benchmarks by paraphrasing and back-translating data into semantically equivalent but lexically distinct forms. Zhao et al. (2024) proposed the MMLU-CF dataset, which is constructed by collecting diverse questions, cleaning data, sampling difficulty reasonably, checking data integrity with LLMs, and applying rewriting methods such as rephrasing questions and shuffling options to ensure the dataset remains contamination-free. CLEVA (Li et al., 2023) employed non-repetitive sampling and multi-strategy data rewriting for robust evaluation. Ying et al. (2024) updated benchmarks with two strategies: style-preserving mimicry with LLMs and cognitive-level expansion using Bloom's taxonomy. Mirzadeh et al. (2025) introduced GSM-Symbolic, an improved benchmark generated from symbolic templates revealing that their mathematical reasoning performance declines when numerical values or clause numbers in questions are altered, indicating they may replicate training data rather than perform reasoning.

D.1.3 Prevention-based Methods

Preventive measures focus on safeguarding test data integrity through technical and procedural controls. Core strategies include encrypting public test data with public-key cryptography, enforcing strict access permissions, and prohibiting derivative data creation. Zhu et al. (2024c) introduced Inference-Time Decontamination (ITD), a novel technique that identifies and rewrites potentially memorized responses during model inference. Li et al. (2024c) introduced C^2LEVA, a comprehensive bilingual benchmark with systematic contamination prevention mechanisms, which implements proactive measures such as test data rotation.

D.2 Dynamic Evaluation

Rule-based Dynamic approaches address data contamination by leveraging adaptive assessment

frameworks. Zhu et al. (2024a) introduced DYVAL, a graph-based system that generates evaluation samples through algorithmic composition, constraint application, and functional descriptions. Its directed acyclic graph (DAG) architecture facilitates multi-step reasoning tasks with precisely controlled complexity. Lei et al. (2024) developed S3EVAL, a framework for SQL evaluation that utilizes randomized table-query pairs. This synthetic approach allows for customizable task lengths and difficulty levels, while systematically assessing long-context reasoning capabilities. Zhang et al. (2024e) proposed the DARG method, which dynamically generates evaluation samples with adjustable complexity and diversity using adaptive reasoning graphs. Srivastava et al. (2024) introduced functionalization, a technique that transforms static question-answer pairs into parameterized code, enabling the generation of infinite test variants. Qian et al. (2024) further extended dynamic evaluation by perturbing key variables in questions, allowing for the dynamic generation of datasets with controlled variations.

Agent-based Zhu et al. (2024b) proposed Multi-Principle Assessment (MPA), which utilizes LLM-based agents to automatically transform questions into new ones. Wang et al. (2024) introduced a multi-agent framework to implement self-evolving benchmarks, which dynamically mutates question contexts and structures to update benchmarks. Chen et al. (2025c) proposes DyCodeEval, a novel benchmarking suite that uses multiple agents to generate semantically equivalent variations of seed programming problems without changing core logic, dynamically evaluating Code LLMs under potential data contamination risks across diverse problem sets to ensure reliable reasoning capability assessments. Cao et al. (2024) proposed StructEval that conducts evaluations across multiple cognitive levels and critical concepts starting from an atomic test objective, offering comprehensive, robust, and consistent LLM evaluations that resist data contamination risks. Xu et al. (2025d) introduced UGMathBench, a dynamic benchmark for evaluating undergraduate-level mathematical reasoning in LLMs and proposes metrics effective accuracy (EAcc) and reasoning gap, revealing through evaluation of 23 LLMs. Zhou et al. (2025) developd a grade school math problem generator to create the GSM-Infinite benchmark, which enables fine-grained control over problem difficulty and context length.

D.3 LLM-as-a-Judge

Next-generation evaluation leverages LLMs themselves as assessment tools. They can serve the roles of scoring, ranking, and selection. Bai et al. (2024) presented the "LM-as-Examiner" framework, generating questions and evaluating responses through reference-free analysis. Yu et al. (2024) deployed LLMs as "Interactors" in structured multi-turn dialogues that probe model capabilities while minimizing contamination risks. Li et al. (2024b) proposed TreeEval, a benchmark-free system where LLMs generate hierarchical question trees. This adaptive approach adjusts difficulty based on model performance, creating unique assessment paths that prevent data contamination.

But Li et al. (2025) identified systematic bias in LLM-as-a-judge evaluations, where models trained on synthetic data from architecturally similar foundations receive unfair preference, compromising evaluation fairness.

D.4 Trustworthy Evaluation Pipeline

Contamination mitigation evaluation aims to estimate an LLM's *non-memorized* capability on potentially leaked benchmarks without the need to reconstruct "clean" datasets. To this end, Hou et al. (2025) propose LNE-BLOCKING, a two-stage *detection–disruption* framework. It first uses length-normalized entropy (LNE) via greedy decoding to estimate per-instance contamination strength. Subsequently, it adaptively applies a token-level *blocking* operation—suppressing top-probability tokens when contamination signals are strong—to steer generation away from memorized paths. This approach recovers contamination-aware scores more efficiently and stably than sampling-heavy filters such as TED.

Yadoji (2025) introduce *DC-Guard*, a unified data-centric framework that reframes benchmarking as an ecological audit. By jointly assessing memorization (MCI), benchmark representativeness (BES), and contamination-aware score correction (CRMA), DC-Guard aims to establish more transparent, reproducible, and trustworthy LLM evaluations.

Disentangling Contamination from Generalization. A fundamental challenge in trustworthy evaluation lies in resolving the ambiguity between rote memorization and genuine reasoning.

(Ishikawa, 2025) systematically address this by proposing a three-tier framework combining n-gram alignment, canary insertion, and perturbation testing. Crucially, they posit that performance on out-of-distribution (OOD) data serves as the true litmus test for generalization. Future evaluation protocols must therefore prioritize strict OOD validation to rigorously distinguish latent reasoning capabilities from surface-level data leakage.

E Additional Experimental Details

This appendix supports the controlled empirical audit in Section 5 along three axes: *reproducibility* (training and detector settings), *robustness* (training method, contamination dose, supplementary model, and bootstrap confidence intervals), and *scope clarification* (behavior-level benchmark evaluation and auxiliary detector screening).

E.1 Training and Data Formatting Details

Table 6 lists the main training settings. The main Qwen3 and MBPP experiments use the same QLoRA configuration across all contamination conditions, so that the exposure form is the main controlled difference. The full fine-tuning ablation keeps the effective batch size matched to QLoRA, but uses a lower learning rate and gradient checkpointing for memory stability.

Data formatting. For OlymMATH, the text only condition is trained as causal language modeling text, while clean, text+label, and paraphrase use chat-style examples. For MBPP, all four conditions use chat-style instruction data. This matches the setup in Section 5.1.

E.2 Detector Settings

Table 7 summarizes detector-side settings. All likelihood-based detectors score only the benchmark prompt unless otherwise noted. AUROC treats the contamination pool as the positive class and the holdout pool as the negative class.

E.3 Behavior-Level Benchmark Evaluation

Table 8 summarizes benchmark-level behavior on contamination-pool and held-out examples. MBPP uses pass@1 and OlymMATH uses exact-answer accuracy. These results complement the prompt-side detector analysis in the main text and reinforce a consistent caution: detector separability and downstream score inflation are related, but they are not the same quantity. On MBPP, text only contamination is often detectable while producing near-zero or negative pass@1 gaps, whereas text+label and paraphrase yield much larger behavioral gains. On OlymMATH, the benchmark-level accuracy gaps remain modest even when prompt-side AUROC is high.

E.4 Ablation on Training Method

Table 10 compares QLoRA and full fine tuning on Qwen3-1.7B for OlymMATH. Full fine tuning does not consistently increase detector separability in this controlled setting. It improves the clean sanity-check row and is similar on paraphrase, but QLoRA yields stronger separability for direct prompt exposure and prompt+answer exposure. Since QLoRA yields comparable or stronger separability in the main direct-exposure settings while keeping the training setup consistent across models, we use QLoRA as the primary setting and treat full fine tuning as a robustness check.

E.5 Ablation on Contamination Dose

We use the number of fine-tuning epochs as a proxy for contamination dose: more epochs over the same contamination pool correspond to stronger exposure. Table 11 reports the resulting sweep on Qwen3-4B for OlymMATH text+label contamination. Detector AUROC generally increases with dose, which supports the interpretation that the measured signal reflects increased benchmark exposure rather than a single-run artifact. CoDeC follows the same upward trend after its rerun with the target-alone versus context-conditioned implementation.

E.6 Supplementary Llama-3.1-8B Results

Table 12 summarizes the full fast-detector sweep on Llama-3.1-8B for OlymMATH, reporting the best detector and its AUROC for each condition. The Llama-3.1-8B results provide a model-side robustness check on the Qwen3 main study. Two observations are worth noting. First, RefPPLRatio is the strongest detector on every contaminated condition, including paraphrase, where Qwen3 detectors weaken substantially. Second, the earlier CoDeC-only snapshot understates separability on this model: when the full detector set is run, paraphrase remains highly detectable for Llama-3.1-8B. Representative bootstrap confidence intervals for these settings are reported in Appendix E.7.

Hyperparameter	Main QLoRA experiments	Full fine-tuning ablation
Models	Qwen3-1.7B, 4B, 8B	Qwen3-1.7B
Benchmarks	OlymMATH, MBPP	OlymMATH
Quantization	4-bit NF4	None
LoRA rank / alpha / dropout	64 / 128 / 0.05	NA
Epochs	3	3
Effective batch size	16	16
Learning rate	2×10^{-4}	2×10^{-5}
Precision	bf16	bf16
Maximum sequence length	2048	2048

Table 6: Main fine-tuning hyperparameters. The contamination-dose ablation uses the same QLoRA setup and varies only the number of epochs. We use cosine scheduling with a 0.05 warmup ratio and seed 42 for all runs.

Detector	Setting
Min-K% Prob	We evaluate K in {10%, 20%, 30%}; main tables report the best AUROC over these K values.
Min-K++	We evaluate K in {10%, 20%, 30%}; main tables report the best AUROC over these K values.
Zlib	Scores use compression-normalized likelihood following the prompt-only scoring protocol.
Neighbor Attack	Scores compare the original prompt with perturbed neighbor prompts.
Perplexity Gap	Scores use the difference between prompt likelihood and perturbed or reference likelihood.
Memorization Score	Scores estimate prompt-level memorization using model likelihood statistics.
RefPPLRatio	Scores compare target model perplexity against the reference model perplexity.
CoDeC	We use one context example from the same split, excluding the target itself, and token range (10, -1). The continuous AUROC score is the mean target log probability without context minus the mean target log probability with context.

Table 7: Detector hyperparameters and score conventions

E.7 Bootstrap Confidence Intervals

Table 9 reports representative 95% bootstrap confidence intervals for the best detector AUROC in settings where complete CI reruns are available. We summarize Qwen3-8B and Llama-3.1-8B on OlymMATH, together with Qwen3-8B on MBPP. The clean rows have wider intervals centered near chance, while the strongly contaminated settings remain well separated after resampling. This stability check supports the qualitative conclusions in the main text: direct prompt exposure and prompt+answer exposure are robustly detectable, while paraphrase remains the weaker OlymMATH setting for Qwen3-8B, even though the Llama-3.1-8B robustness check stays highly separable.

Benchmark	Model	Condition	Contam	Held-out	Gap
MBPP pass@1					
MBPP	Qwen3-1.7B	Clean	0.070	0.050	+0.020
MBPP	Qwen3-1.7B	Text only	0.060	0.030	+0.030
MBPP	Qwen3-1.7B	Text+label	0.180	0.040	+0.140
MBPP	Qwen3-1.7B	Paraphrase	0.190	0.070	+0.120
MBPP	Qwen3-4B	Clean	0.110	0.110	+0.000
MBPP	Qwen3-4B	Text only	0.080	0.110	-0.030
MBPP	Qwen3-4B	Text+label	0.340	0.080	+0.260
MBPP	Qwen3-4B	Paraphrase	0.450	0.120	+0.330
MBPP	Qwen3-8B	Clean	0.100	0.150	-0.050
MBPP	Qwen3-8B	Text only	0.060	0.130	-0.070
MBPP	Qwen3-8B	Text+label	0.490	0.100	+0.390
MBPP	Qwen3-8B	Paraphrase	0.560	0.110	+0.450
OlymMATH accuracy					
OlymMATH	Qwen3-1.7B	Clean	0.000	0.000	+0.000
OlymMATH	Qwen3-1.7B	Text only	0.017	0.050	-0.033
OlymMATH	Qwen3-1.7B	Text+label	0.083	0.025	+0.058
OlymMATH	Qwen3-1.7B	Paraphrase	0.067	0.000	+0.067
OlymMATH	Qwen3-4B	Clean	0.050	0.000	+0.050
OlymMATH	Qwen3-4B	Text only	0.067	0.025	+0.042
OlymMATH	Qwen3-4B	Text+label	0.133	0.050	+0.083
OlymMATH	Qwen3-4B	Paraphrase	0.183	0.050	+0.133
OlymMATH	Qwen3-8B	Clean	0.050	0.000	+0.050
OlymMATH	Qwen3-8B	Text only	0.067	0.050	+0.017
OlymMATH	Qwen3-8B	Text+label	0.117	0.050	+0.067
OlymMATH	Qwen3-8B	Paraphrase	0.133	0.025	+0.108

Table 8: Behavior-level benchmark evaluation on contamination-pool and held-out examples. MBPP uses pass@1 and OlymMATH uses exact-answer accuracy.

Benchmark	Model	Condition	Best detector	AUROC	95% bootstrap CI
OlymMATH					
OlymMATH	Qwen3-8B	Clean	Min-K%(0.3)	0.548	[0.432, 0.666]
OlymMATH	Qwen3-8B	Text only	Zlib	0.982	[0.949, 0.999]
OlymMATH	Qwen3-8B	Text+label	Zlib	0.937	[0.881, 0.978]
OlymMATH	Qwen3-8B	Paraphrase	Min-K%(0.3)	0.649	[0.537, 0.755]
OlymMATH	Llama-3.1-8B	Clean	CoDeC	0.628	[0.522, 0.733]
OlymMATH	Llama-3.1-8B	Text only	RefPPLRatio	0.996	[0.988, 1.000]
OlymMATH	Llama-3.1-8B	Text+label	RefPPLRatio	1.000	[1.000, 1.000]
OlymMATH	Llama-3.1-8B	Paraphrase	RefPPLRatio	0.967	[0.933, 0.994]
MBPP					
MBPP	Qwen3-8B	Clean	Min-K++(0.1)	0.509	[0.470, 0.553]
MBPP	Qwen3-8B	Text only	Zlib	0.869	[0.816, 0.914]
MBPP	Qwen3-8B	Text+label	Zlib	0.839	[0.780, 0.887]
MBPP	Qwen3-8B	Paraphrase	Zlib	0.852	[0.795, 0.899]

Table 9: Representative bootstrap confidence intervals for the best detector AUROC. Confidence intervals use the percentile bootstrap with 1,000 resamples.

Condition	QLoRA best	Full FT best	Full - QLoRA
Clean	0.553	0.615	+0.062
Text only	0.976	0.814	-0.162
Text+label	0.869	0.819	-0.050
Paraphrase	0.674	0.678	+0.005

Table 10: Training method ablation on Qwen3-1.7B and OlymMATH. Each value is the best detector AUROC for that condition.

Detector	Epoch 1	Epoch 3	Epoch 5	Monotonic
Min-K%(0.3)	0.613	0.882	0.977	Yes
Min-K++(0.3)	0.586	0.946	1.000	Yes
Zlib	0.580	0.935	0.997	Yes
Perplexity Gap	0.609	0.885	0.979	Yes
RefPPLRatio	0.602	0.955	1.000	Yes
CoDeC	0.651	0.938	0.946	Yes

Table 11: Contamination dose ablation for Qwen3-4B on OlymMATH text+label contamination. The number of fine-tuning epochs serves as a proxy for exposure strength. Values are AUROC.

Condition	Best detector	AUROC
Clean	CoDeC	0.628
Text only	RefPPLRatio	0.996
Text+label	RefPPLRatio	1.000
Paraphrase	RefPPLRatio	0.967

Table 12: Llama-3.1-8B fast-detector summary on OlymMATH. Each row reports the best detector and its AUROC for that condition.