Beyond Clean and Contaminated: A Survey on the Fundamental Properties of Data Contamination in Large Language Models

Anonymous ACL submission

Abstract

Benchmark-based evaluation remains the primary mechanism for comparing large language models (LLMs), yet modern development pipelines increasingly blur the boundary between training and testing. Beyond direct train–test overlap, contamination leaks through pathways such as post-training, evaluation-time "test-set fitting," and retrieval-enabled tool use. In this paper, we frame data contamination as an evaluation-validity failure mode and propose a three-dimensional taxonomy based on *phase*, *granularity*, and *modality*. We argue that contamination is regime-dependent rather than binary, summarizing key properties such as inevitability under web-scale collection, scaling effects, and forgettability. Building on these insights, we reorganize detection methods into two complementary paradigms: *statistical* approaches (quantifying inflation via observational signals) and *causal* approaches (verifying via controlled injection). Finally, we provide a critical discussion of these detection methodologies.

1 Introduction

Recent breakthroughs in Large Language Models (LLMs) have demonstrated remarkable capabilities in text generation, code synthesis, and mathematical reasoning (Grattafiori et al., 2024; OpenAI et al., 2024; DeepSeek-AI et al., 2025). Benchmark-based evaluation is central to comparing LLMs, yet modern development pipelines blur the boundary between training and evaluation (Balloccu et al., 2024): prompt libraries (Han et al., 2025), retrieval indices and tool configurations (Han et al., 2025) create additional pathways for benchmark information to sneak into models' knowledge inadvertently. Consequently, the reliability of LLM evaluation is increasingly questioned due to data contamination-the unintended overlap between training and test datasets (Chang et al., 2024; Sainz et al., 2023).

Crucially, the prevailing binary perspective—classifying models as simply "clean" or "contaminated" based on surface-level pre-training overlap—fails to capture the nuance of modern development. Leakage in current pipelines is diverse: it can occur implicitly through semantic equivalents in instruction tuning (Dekoninck et al., 2024b), propagate via preference pairs during alignment (Tao et al., 2025), or emerge dynamically through retrieval mechanisms. Furthermore, the magnitude of model performance inflation is explicitly modulated by both the timing of contamination (Kocyigit et al., 2025) and the nature of the downstream task (Ishikawa, 2025). This leads to a 'consistency crisis', exposing the inconsistency and fragility of current detection paradigms (Dekoninck et al., 2024b).

Despite this evolving landscape, existing surveys predominantly focus on compiling detection techniques and mitigation methods, largely overlooking the phase propagation of risks and failing to systematize the intrinsic characteristics (Deng et al., 2024; Fu et al., 2024). Distinguishing our work from prior surveys (Appendix A), we expand the definition of contamination to a cross-phase scope—where $\mathcal{D}_{\text{train}}$ includes alignment, synthetic variants, and retrieval assets—formalizing it as any metric-inflating overlap as defined in Sec 2. We advance the field by: (1) establishing a unified *three-dimensional taxonomy* (phase, granularity, and modality) that explicitly covers underexplored risks in preference learning and retrieval pipelines (Tao et al., 2025; Han et al., 2025); (2) characterizing intrinsic properties such as forgettability and scaling effects; and (3) reorganizing detection into complementary *statistical* (observational) and *causal* (interventional) paradigms to better navigate the current consistency crisis.

Section 3 characterizes contamination along three axes: *phase*, *granularity*, and *modality*. Regarding *phase*, we highlight risks beyond pre-

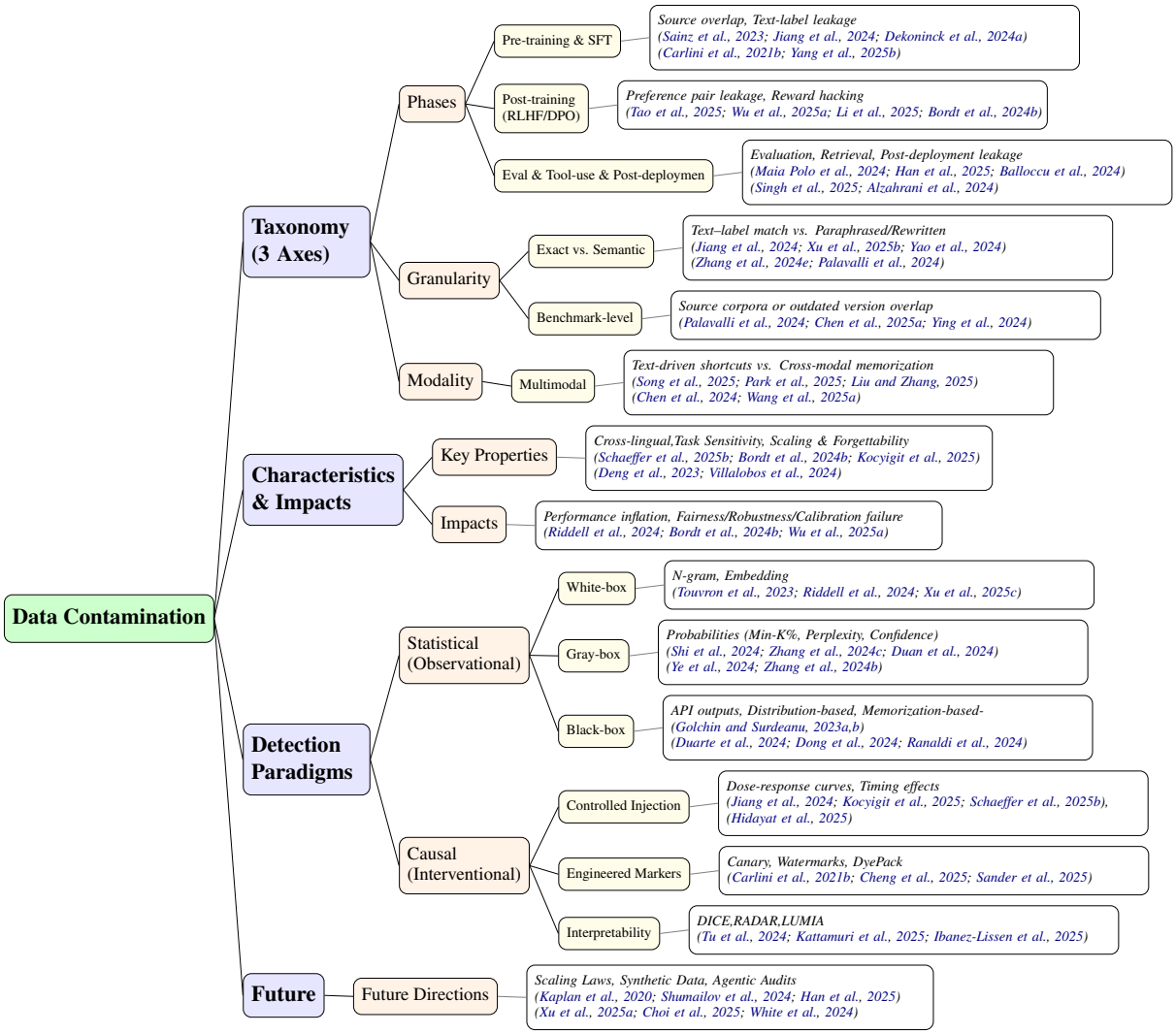

Figure 1: Structure of this paper

training, covering RL post-training (Tao et al., 2025), evaluation-time fitting (Maia Polo et al., 2024), and retrieval-time leakage (Han et al., 2025). Different from prior works in table 2, we systematize *granularity* from instance-level surface or semantic overlap (Xu et al., 2025b) to broad benchmark-level inclusion (Palavalli et al., 2024). Finally, for *modality*, we distinguish between text-driven shortcuts and cross-modal memorization (Song et al., 2025).

We synthesize the impacts of data contamination in section 4. While performance inflation is the most visible risk—modulated significantly by leakage granularity and phase—contamination fundamentally undermines evaluation integrity. This inflation masks critical flaws in fairness and robustness, degrading calibration and compromising the scientific validity of conclusions (Bordt et al., 2024b).

In section 5, we argue contamination is not a binary condition but a regime-dependent phenomenon shaped by multiple interacting factors. It distills recurring characteristics including (i) *inevitability* under web-scale data collection (Deng et al., 2023), (ii) *decoding* dependence (Schaeffer et al., 2025b), (iii) *cross-phase dependence* on when leakage occurs (Kocyigit et al., 2025), (iv) *task and format sensitivity* (Kocyigit et al., 2025), (v) *cross-lingual* effects via translated benchmarks (Yao et al., 2024), and (vi) *scaling and forgettability (Bordt et al., 2024b)*, where timing and exposure magnitude govern whether memorized traces persist or are diluted by subsequent training (Bordt et al., 2024b).

Section 6 reorganizes detection into two paradigms: *statistical* methods, which infer leakage from observational signals and increasingly aim to *quantify* inflation (Xu et al., 2025a); and

causal methods, which use controlled injection to verify dose–response effects (Jiang et al., 2024). We conclude by critically assessing their robustness and applicability under varying constraints.

Finally, to move beyond reactive detection toward a predictive science, we highlight three future directions in Section 7: quantitative modeling, synthetic data saturation, and generative evaluation. Due to space constraints, we provide a review of mitigation strategies in appendix E.

2 Preliminaries

Data Contamination Let $\mathcal{D}_{\text{train}}$ encompass all development assets: pre-training and alignment corpora (including paraphrased or synthetic variants), as well as auxiliary components like retrieval stores and prompts. Let $\mathcal{D}_{\text{test}}$ denote the benchmark instances. We define *data contamination* as the presence of overlap between the test set $\mathcal{D}_{\text{test}}$ and $\mathcal{D}_{\text{train}}$ ($\mathcal{D}_{\text{train}} \cap \mathcal{D}_{\text{test}} \neq \emptyset$). Such leakage artificially inflates metrics through memorization rather than reflecting genuine generalization.

3 A Taxonomy of Data Contamination

Existing studies on data contamination remain fragmented, lacking a systematic characterization of its origins and forms. To address this, we introduce a taxonomy along three axes—*lifecycle phase*, *granularity*, and *modality*—providing a structured framework to guide detection and mitigation.

3.1 Contamination Phases

Pre-training Phase 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). Crucially, this phase can re-contaminate the model, even after rigorous initial sanitization.

SFT and Instruction 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 promotes format-specific memorization, leading to significant score inflation. Moreover, advanced augmentation like Chain-of-Thought (CoT) can obfuscate these traces, complicating detection (Samuel et al., 2024).

Preference Learning Contamination Benchmark leakage occurs via annotator exposure to test items or the recycling of benchmark responses as preference pairs (Tao et al., 2025; Wu et al., 2025a) (detailed in appendix B.1). This directly amplifies reward-driven overfitting, solidifying the model's reliance on memorized patterns. RL-phase contamination poses a greater detection challenge than leakage in Supervised Fine-Tuning (SFT) because the optimization objective shifts from matching the data distribution via $\max_\theta \mathbb{E}[\log \pi_\theta(y|x)]$ to maximizing the expected reward $J(\theta) = \mathbb{E}_{\tau\sim\pi_\theta}[R(\tau)]$. Consequently, contaminated examples may not exhibit conspicuous anomalies in perplexity (PPL) or likelihood scores, yet they can covertly induce the policy π_θ to converge toward rigid, reward-hacking behaviors during inference.

Evaluation-pipeline Contamination Contamination can occur *without* changing training data, e.g., (i) repeatedly tuning prompts on the test set (Maia Polo et al., 2024), (ii) selecting few-shot examples that overlap with the benchmark (Singh et al., 2025), or (iii) iterating on public leaderboards for model selection (Alzahrani et al., 2024).

Retrieval-time (Tool-use) Contamination For tool-augmented or search-enabled systems, the model may retrieve solutions during evaluation (detailed in appendix B.2), yielding high scores without genuine generalization (Han et al., 2025).

Post-deployment Contamination After deployment, human interactions and logging pipelines may inadvertently expose benchmark content, which can later be collected into training data (directly or indirectly), creating a data flywheel that leaks evaluation items back into future model versions (Balloccu et al., 2024). This phase is especially relevant for continual learning and production data reuse.

3.2 Contamination Granularity

Text Contamination (Jiang et al., 2024; Li et al., 2024e) Text contamination occurs when the input text components of evaluation samples appear in the pre-training corpus, creating an overlap between test and training data that may artificially inflate model performance metrics.

Text-label Contamination (Jiang et al., 2024; Li et al., 2024e) Text-label contamination refers to scenarios where the pre-training corpus contains not only the input text but also the corresponding prompts (task instructions) and labels from evaluation samples. This effectively exposes the model to both questions and correct answers prior to testing. Specifically, this phenomenon can be categorized into two types: (1) **unpaired contamination**, where both the input and output appear in the corpus but are not collocated or linked directly; and (2) **paired contamination**, where input-output pairs appear explicitly together. The latter is considered more severe (Yang et al., 2025b). We provide a comparative analysis of the impacts of Text Contamination and Text-label Contamination in Appendix B.3.

Semantic Contamination (Xu et al., 2025b) Semantic contamination refers to contamination that extends beyond exact matches of benchmark instances, arising instead from semantic-preserving transformations. Common mechanisms include input/output masking, noise injection, and adversarial or distractor-based augmentation. Unlike overlap-centric definitions, this perspective encompasses a broader spectrum of contamination capable of evading surface-level detection while artificially inflating benchmark performance (Yao et al., 2024; Palavalli et al., 2024). Building on this, Xu et al. (2025b) distinguished between **entity contamination** (involving named entities and their implicit biases) and **factual contamination** (covering concrete events, context, and detailed assertions).

Information-Level Contamination (Xu et al., 2025a) This category captures the leakage of benchmark-related meta-information—such as label distributions, temporal metadata, or external reviews—rather than exact test instances. Such exposure can implicitly shape model priors, introducing statistical biases that inadvertently skew the evaluation process by aligning the model with the test set's structural characteristics. And we conduct an impact analysis of information-level contamination in appendix B.4.

Benchmark-level Contamination (Palavalli et al., 2024) Benchmark-level contamination occurs when a model is trained on the source corpora underlying a benchmark (or on outdated versions of the benchmark), such that instances or their variants inadvertently leak into the training data.

3.3 Contamination Modality

Contamination in multimodal models typically arises through two major pathways.

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

Cross-modal Memorization. When image–text pairs (or image–text–label triplets) are present during training, models can memorize cross-modal correspondences instead of learning generalizable vision–language representations (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 static benchmarks.

4 Impacts

In this part, we synthesize the literature from two complementary angles: (i) the *impacts* of contamination on reported performance and reliability-related properties, (ii) *empirical evidence* used to substantiate leakage in practice.

4.1 Impacts

Data contamination critically undermines evaluation integrity. (Riddell et al., 2024) revealed that contamination not only artificially inflates performance but also invalidates scientific conclusions in NLP. Moreover, these detrimental effects transcended superficial metrics like accuracy, degrading core model properties and ultimately compromising safety (Bordt et al., 2024b).

Performance Inflation While performance inflation represents the most critical risk of data contamination, it is not an inevitable outcome; its magnitude is modulated by factors such as the phase, granularity, and task. Supporting this nuance, Yang et al. (2025b) found that LLMs exhibit substantial performance inflation under *text-label* contamination, while text, label-only, and unpaired contamination have largely negligible effects for coding tasks.

Fairness Contaminated data often encode inherent biases, such as the over-representation of specific demographics or viewpoints. During training,

models may disproportionately amplify these biases and propagate them into their generated outputs, leading to a degradation in fairness (Dodge et al., 2021).

Robustness Contamination creates an illusion of competence, driving high scores through memorization rather than genuine generalization. This masks inherent **brittleness**, where models fail precipitously on slight input perturbations or out-of-distribution (OOD) data, rendering their deployed robustness severely compromised.

Calibration Contamination severely distorts confidence calibration, causing models to assign excessive confidence to contaminated instances. This spurious certainty compromises downstream reliability: it can blind uncertainty-based hallucination detectors to confident errors (Manakul et al., 2023), and destabilize Reinforcement Learning (RL) alignment (Casper et al., 2023), which relies on accurate probability estimates for effective reward optimization and self-correction.

4.2 Empirical Evidence of Contamination

Empirical evidence of contamination is primarily established by identifying unexpectedly strong benchmark performance and attributing it to prior training exposure. Early surveys noted that while evidence was emerging, it remained fragmented (Sainz et al., 2023). By 2024, studies began to employ more granular detection methods, such as Riddell et al. (2024), who quantified the specific overlap between benchmarks and training corpora for code models. Most recently, the scope of evidence has expanded to include diverse signals. For instance, Sendyka et al. (2025) demonstrated anomalous robustness, where models could solve code-generation tasks even under extreme prompt redaction, indicating verbatim memorization. Concurrently, Wu et al. (2025a) investigated the Qwen2.5 model family within reinforcement learning contexts, revealing potential leakage issues in mathematical benchmarks. These findings collectively suggest that contamination has permeated specialized domains beyond general text, including code and mathematics. Moreover, this evidence reveals a systemic failure manifesting as *abnormal robustness*, where models exhibit capabilities even under severely degraded conditions. Additional evidence will be discussed in Appendix C.

5 Characteristics of Data Contamination

In this section, we summarize 6 characteristics of data contamination in LLMs.

Inevitability As LLMs continue to scale up, the size of their training datasets expanded correspondingly (Villalobos et al., 2024). These datasets are often sourced from extensive web crawls, which may inadvertently overlap with evaluation benchmarks, leading to data contamination (Deng et al., 2023). This process is currently inevitable.

Injection 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. Late-phase contamination ultimately causes a larger performance inflation. Uniform contamination (spread across the entire training process) produces the most lasting effects, with no significant spikes.

Decoding Sensitivity Manifestations of contamination are also modulated by inference parameters. Schaeffer et al. (2025b) demonstrated that higher sampling temperatures can dampen these contamination-driven score boosts, whereas greedy decoding tends to amplify the retrieval of memorized patterns. However, this relationship between temperature and extraction is not always monotonic (Hayes et al., 2025).

Task Sensitivity The impact of data contamination is fundamentally task-dependent, rather than a uniform inflation of performance scores (Golchin and Surdeanu, 2023b; Jiang et al., 2024). For reasoning tasks, contamination manifested as a discrepancy between direct answers and logical consistency (Ishikawa, 2025), whereas in generative tasks like summarization, the gains were sensitive to whether the leakage was verbatim or reformatted (Palavalli et al., 2024). Moreover, this sensitivity was mediated by data resource characteristics; for instance, contamination yields disproportionately larger performance gains in low-resource settings—such as rare language pairs in machine translation—compared to high-resource ones (Kocyigit et al., 2025). Thus, contamination is not monolithic; it demands task-specific evaluation.

Cross-lingual Characteristics LLMs are overfitted to translated versions of benchmark test sets in non-English languages. This practice inflated model performance on the original English benchmarks without direct exposure to them (Yao et al., 2024), while evading existing detection methods (Zhang et al., 2024a). Building on this Kocyigit et al. (2025) found that contamination requires sufficient language representation to produce measurable effects: for resource-scarce languages, contamination has almost no impact on performance. Cross-lingual contamination can exhibit a representation threshold, and multilingual claims should report audits or controlled tests of exposure to translated benchmark variants.

Scaling & Forgettability Larger models exhibit stronger contamination effects than smaller ones (Kocyigit et al., 2025; Schaeffer et al., 2025b). As LLMs' memorization ability grows significantly with their model size, we argue it becomes even easier for them to reproduce training data instances (Riddell et al., 2024). Recent studies suggest that benchmark contamination is not a binary state but a dynamic phenomenon governed by exposure magnitude and timing. Schaeffer et al. (2025a) resolved the "contamination paradox" by modeling performance gains as a dose–response function, where benchmark inflation depends on the fraction of leaked tokens relative to model capacity and the incentive to memorize. Consistent with this view, Bordt et al. (2024b) observed that while even minor leakage causes measurable overfitting in Chinchilla-optimal (Hoffmann et al., 2022) setups, such effects are regime-dependent: in data-rich environments (e.g., $> 5\times$ Chinchilla-optimal tokens), continued training can effectively *wash out* memorized traces via dilution. Consequently, late-phase contamination is significantly more detrimental than early-stage exposure.

6 Data Contamination Detection

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. Despite this, current methods face a *consistency crisis*, often yielding contradictory diagnoses and remaining vulnerable to evasion via instruction tuning.

6.1 Statistical Approaches

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

6.1.1 White-box

White-box methods use training data or model internals to directly audit overlap.

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).

6.1.2 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 4). 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.

6.1.3 Black-box

Black-box methods operate without access to training data or internals and rely solely on model outputs and behavioral assumptions in appendix D.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., 2025a).

Distributions-based Dong et al. (2024) proposed CDD based on the peakedness of output distributions and pair it with TED for mitigation while preserving evaluation validity.

Binary Judgments Khan et al. (2025) proposed NATURAL-DACODE, combining code naturalness with token-level completion accuracy via a lightweight classifier (SVM) to detect contamination. 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 Impact Beyond Binary Judgments Beyond binary "contaminated vs. clean" decisions, 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 (directly or via transformations), i.e., when $D_{\text{train}}^{\text{test}} \cap B^{\text{test}} \neq \emptyset$; the contamination risk is

$$\text{Score} = \frac{\left|D_{\text{train}}^{\text{test}} \cap B^{\text{test}}\right|}{\left|B^{\text{test}}\right|}. \quad (1)$$

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, motivated by the intuition that seen instances are perturbed differently than unseen ones.

6.2 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.

6.3 Detection Methods Discussion

Fragility and Inconsistency Under a ground-truth contamination setting (involving instruction fine-tuning with CoT-style augmentation), widely used detectors exhibit non-monotonic sensitivity and weak cross-method agreement (low Spearman correlation), resulting in contradictory judgments even for frontier LLMs (Dekoninck et al., 2024b; Fu et al., 2024). And when leakage is *rare*, detectors based purely on scoring signals (PPL, Min-$K\%$) can degrade;

Decoding Sensitivity Black-box detectors relying on model outputs (e.g., TS-Guessing) are susceptible to decoding parameters; specifically, high-temperature sampling can mask memorization signatures that are otherwise visible under greedy decoding (Schaeffer et al., 2025b).

7 Future Directions

From Qualitative Detection to Contamination Scaling Laws Current research predominantly 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). Establishing these laws would transition the field from post-hoc forensics to predictive science, enabling developers to forecast metric gains from leakage rates and perform mathematically grounded 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.

Generative Benchmarks Contamination As discussed in Section 5, the transition toward generative and agentic evaluation necessitates that contamination research account for stochastic decoding and output diversity. We advocate for a multi-faceted approach: (i) *Robust Protocols*: shifting from text matching to execution-based scoring and semantic equivalence, distinguishing metric inflation from genuine reasoning; (ii) *Causal Dynamics*: analyzing contamination via dose–response studies that account for sampling strategies (e.g., temperature) and prompt sensitivity; and (iii) *Agentic Governance*: treating interaction traces and tool logs as leakage vectors, necessitating sandboxed environments and auditable provenance. Standardized reporting remains essential to ensure comparability amidst these shifts (Schaeffer et al., 2025b).

8 Conclusion

Data contamination fundamentally threatens the trustworthiness of LLM evaluation. This survey systematizes the field by establishing a comprehensive *taxonomy* of leakage pathways, analyzing the *regime-dependent characteristics* of contamination (e.g., scale and forgettability), and contrasting observational *detection* with causal interventions. Beyond merely inflating metrics, we highlight that contamination compromises fairness and scientific validity. Consequently, reliable benchmarking requires treating contamination as a *first-class variable* through rigorous audits and transparent reporting. Future progress hinges on developing quantitative contamination scaling laws and addressing *synthetic data feedback loops* to sustain evaluation validity in an increasingly model-generated web.

9 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.

Stephen Casper, Xander Davies, Claudia Shi, Thomas Krendl Gilbert, Jérémy Scheurer, Javier Rando, Rachel Freedman, Tomasz Korbak, David Lindner, Pedro Freire, Tony Wang, Samuel Marks, Charbel-Raphaël Segerie, Micah Carroll, Andi Peng, Phillip Christoffersen, Mehul Damani, Stewart Slocum, Usman Anwar, and 13 others. 2023. Open problems and fundamental limitations of reinforcement learning from human feedback. *Preprint*, arXiv:2307.15217.

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 pre-training 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 pre-training: 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*.

Jesse Dodge, Maarten Sap, Ana Marasović, William Agnew, Gabriel Ilharco, Dirk Groeneveld, Margaret Mitchell, and Matt Gardner. 2021. Documenting large webtext corpora: A case study on the colossal clean crawled corpus. *Preprint*, arXiv:2104.08758.

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.

Jordan Hoffmann, Sebastian Borgeaud, Arthur Mensch, Elena Buchatskaya, Trevor Cai, Eliza Rutherford, Diego de Las Casas, Lisa Anne Hendricks, Johannes Welbl, Aidan Clark, Tom Hennigan, Eric Noland, Katie Millican, George van den Driessche, Bogdan Damoc, Aurelia Guy, Simon Osindero, Karen Simonyan, Erich Elsen, and 3 others. 2022. Training compute-optimal large language models. *Preprint*, arXiv:2203.15556.

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*.

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.

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.

Changmao Li and Jeffrey Flanigan. 2024. Task contamination: Language models may not be few-shot anymore. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 38, pages 18471–18480.

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.

Chuang Liu, Renren Jin, Mark Steedman, and Deyi Xiong. 2024. Evaluating Chinese large language models on discipline knowledge acquisition via memorization and robustness assessment. In *Proceedings of the 1st Workshop on Data Contamination (CONDA)*, pages 1–12, Bangkok, Thailand. 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.

Potsawee Manakul, Adian Liusie, and Mark J. F. Gales. 2023. Selfcheckgpt: Zero-resource black-box hallucination detection for generative large language models. *Preprint*, arXiv:2303.08896.

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*.

Radzim Sendyka, Christian Cabrera, Andrei Paleyes, Diana Robinson, and Neil Lawrence. 2025. Llm performance for code generation on noisy tasks. *Preprint*, arXiv:2505.23598.

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. 2025. 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*.

Yuan Tseng, Titouan Parcollet, Rogier van Dalen, Shucong Zhang, and Sourav Bhattacharya. 2025. Evaluation of llms in speech is often flawed: Test set contamination in large language models for speech recognition. *Preprint*, arXiv:2505.22251.

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.

Hanqing Wang, Yuan Tian, Mingyu Liu, Zhenhao Zhang, and Xiangyang Zhu. 2025a. 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*.

Zeng Wang, Minghao Shao, Jitendra Bhandari, Likhitha Mankali, Ramesh Karri, Ozgur Sinanoglu, Muhammad Shafique, and Johann Knechtel. 2025b. Vericontaminated: Assessing llm-driven verilog coding for data contamination. *Preprint*, arXiv:2503.13572.

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.

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

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 1: 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.

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.

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

$$\mathcal{J}_{\text{RL}}(\theta) = \mathbb{E}_{q\sim D_{\text{RL}},\{o_i\}\sim\pi_{\theta_{\text{old}}}}\left[f(\mathcal{R}(o_i),\pi_\theta)\right] \quad (2)$$

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

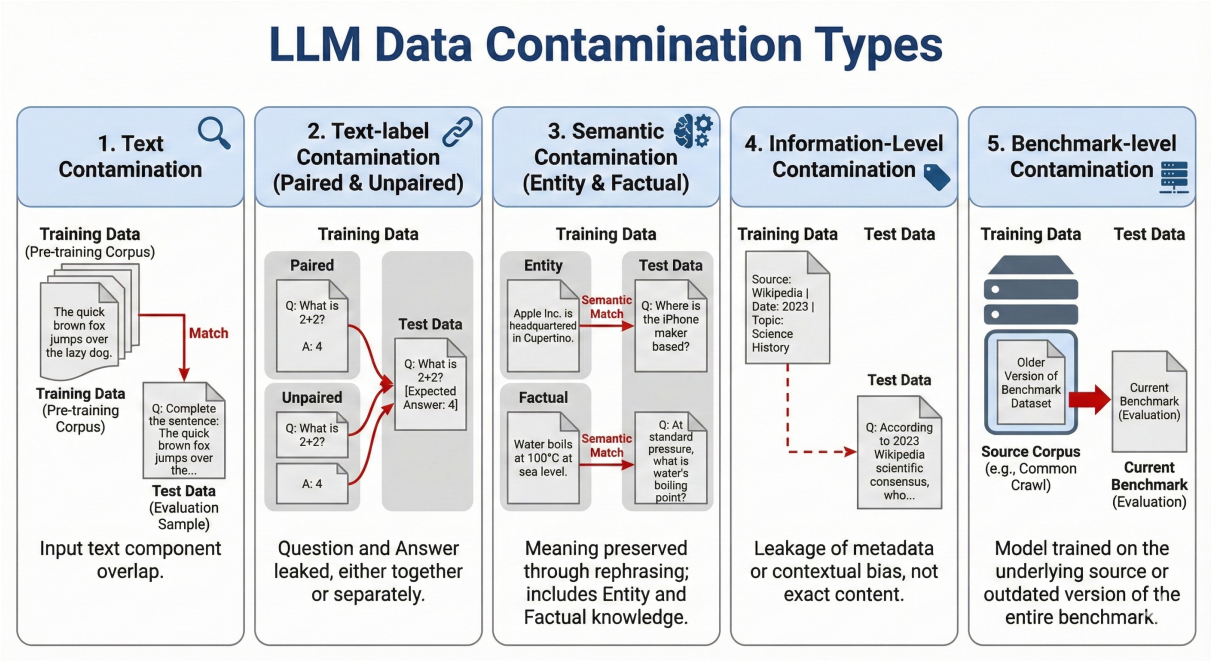

Figure 2: contamination granularity

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 Impacts

More evidence of data contamination Beyond these primary signals, other studies have highlighted diverse manifestations of leakage across different scopes. Li and Flanigan (2024) utilized membership inference as a detection mechanism, while Liu et al. (2024) observed discrepancies in Chinese LLMs where high benchmark scores masked superficial knowledge rather than genuine comprehension. Furthermore, contamination risks have been identified in specialized domains beyond standard text and software code: Tseng et al. (2025) revealed that transcript leakage significantly distorts metrics in the audio domain, and Wang et al. (2025b) reported similar severity in hardware description languages like Verilog, emphasizing the need for rigorous scrutiny across all modalities.

D Detection

D.1 Definition of Assumptions

D.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.

D.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 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.

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 2: 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).

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.

D.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.

D.3 Related Tools

D.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 contamina-

tion, 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.

D.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.

D.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.

E Mitigation and Trustworthy Evaluation

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

E.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

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 3: Summary of contamination detection paradigms (Section 6): white-/gray-/black-box methods differ by required access, detection signals, and robustness/efficiency trade-offs.

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.

E.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 knowledge 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.

E.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 op-

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 4: 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.

tions 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.

E.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.

E.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.

E.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.

E.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.