A Survey on Data Contamination for Large Language Models

Anonymous ACL submission

Abstract

Recent advancements in Large Language Models (LLMs) have demonstrated significant progress in various areas, such as text generation and code synthesis. However, the reliability of performance evaluation has come under scrutiny due to data contamination—the unintended overlap between training and test datasets. This overlap has the potential to artificially inflate model performance, as LLMs are typically trained on extensive datasets scraped from publicly available sources. These datasets often inadvertently overlap with the benchmarks used for evaluation, leading to an overestimation of the models' true generalization capabilities. In this paper, we first examine the definition and impacts of data contamination. Secondly, we review methods for contamination-free evaluation, focusing on three strategies: data updating-based methods, data rewriting-based methods, and prevention-based methods. Specifically, we highlight dynamic benchmarks and LLM-driven evaluation methods. Finally, we categorize contamination detecting methods based on model information dependency: White-Box, Gray-Box, and Black-Box detection approaches. Our survey highlights the requirements for more rigorous evaluation protocols and proposes future directions for addressing data contamination challenges.

1 Introduction

Recent breakthroughs in Large Language Models (LLMs) have demonstrated remarkable capabilities in text generation, code synthesis, and mathematical reasoning (Zhao et al., 2023; OpenAI et al., 2024; DeepSeek-AI et al., 2025). However, the reliability of LLM evaluation is increasingly questioned due to data contamination-the unintended overlap between training and test data sets (Balloccu et al., 2024; Chang et al., 2024). This is especially problematic as LLMs use large web-scraped datasets that are prone to overlap with testing benchmarks. (Xu et al., 2024b) analyzed 31 LLMs in the context of mathematical reasoning, uncovering widespread data contamination. LLMs are known to memorize portions of their training data, and under certain prompts, they can reproduce this data verbatim (Carlini et al., 2022). As highlighted by (Sainz et al., 2023), a critical consequence of data contamination is that scientific studies relying on contaminated LLMs may produce erroneous conclusions, potentially invalidating valid hypotheses. To underscore the importance of addressing data contamination in both the development and evaluation of LLMs, we present a comprehensive review of data contamination.

In section 2, we define data contamination as the inclusion of data from the testing set during the pre-training phase, which artificially inflates model performance. Recent studies extend this definition along two dimensions: phase-based contamination in LLMs' lifecycle and benchmark-based contamination in LLMs' evaluation. For phase-based analysis, contamination mechanisms include pre-training phase leakage, fine-tuning biases, cross-modal leakage (Yao et al., 2024), and indirect human interactions (Palavalli et al., 2024). Meanwhile, we divide benchmark-based contamination into four types: Text contamination, Text-label contamination, Augmentation-based contamination, and Benchmark-level contamination. We provide a fine-grained analysis of its impacts, including Evidence collection, Non-Contamination scenarios, Quantification of contamination, and Characteristics of data contamination.

In section 3, we discuss how to achieve contamination-free evaluation. For static benchmarks, current research focuses on three key contamination-free strategies: automatically updating datasets using the most recent data, rewriting existing data, and implementing proactive risk

*Corresponding authors

prevention mechanisms. Meanwhile, dynamic evaluation frameworks(Zhu et al., 2024a; Lei et al., 2024; Zhang et al., 2024e; Ying et al., 2024) generate test samples using techniques like combinatorial optimization, graph-based reasoning, and controlled randomization, creating an evolving evaluation system. Additionally, the LLM-as-a-judge paradigm(Bai et al., 2024) turns LLMs into meta-evaluators, enabling intelligent assessments independent of static benchmarks.

In section 4, we explore methodologies for detecting data contamination in LLMs. We categorize data contamination detection approaches into three distinct paradigms: white-box detection, which relies on full access to model architectures or training data to achieve high precision, employing techniques such as N-gram overlap (Brown et al., 2020) or embedding similarity (Reimers, 2019); gray-box detection, which leverages partial model information, such as token probabilities, to identify contamination; and black-box detection, which operates without access to internal model details, relying instead on heuristic rules (the details are outlined in Appendix B). Together, these approaches illustrate the evolving and multifaceted landscape of data contamination detection methods, each offering unique advantages and challenges.

The organization of this paper is as follows, as shown in figure 1. In Section 2, we discuss existing work on the definition and impacts of data contamination. Section 3 summarizes current methods for constructing contamination-free datasets and dynamic evaluation approaches. Section 4 discusses how to detect data contamination. Finally, in Section 5, we present several significant future challenges in this area.

Difference with previous survey Our paper systematically summarizes the definitions of data contamination across different scenarios and provides a fine-grained analysis of its impacts, particularly focusing on the characteristics of data contamination in Section 2.2.4. Additionally, we enumerate several benchmarks utilized for quantifying data contamination in Section 3.4.

Survey	Definition	Detection	Mitigation
Ravaut et al.	×	✓	×
Xu et al.	✓	✓	✓
Fu et al.	×	Partial	×
Chen et al.	×	×	Partial
Deng et al.	✓	✓	✓
Ours	Comprehensive	✓	✓

Table 1: Summary of Prior Surveys, ✓ means full coverage and × indicates it is not the main focus. Details are in Appendix D.

2 What is Data Contamination

2.1 Definition

In recent years, a growing body of research has emerged to address the issue of data contamination in LLMs. However, the field lacks a standardized methodology to comprehensively summarize data contamination. Simply, let $\mathcal{D}_{\text{train}}$ denote the training dataset and $\mathcal{D}_{\text{test}}$ the evaluation dataset. Data contamination occurs when: $\mathcal{D}_{\text{train}} \cap \mathcal{D}_{\text{test}} \neq \emptyset$. Building on this definition, our research extends the framework into two significant directions: (1) examining vulnerabilities across the entire lifecycle of LLMs, including pre-training, fine-tuning, and post-deployment contamination, and (2) addressing risks to benchmark integrity, such as data manipulation and potential label leakage.

2.1.1 Phase-based Contamination

Contamination during pre-training (Sainz et al., 2023) During initial training, web-scraped data often contains unwanted content (e.g., benchmark datasets like GLUE) because of imperfect filtering and deduplication (Lee et al., 2022). While complete prevention is impractical, transparency about pre-training data helps avoid biased evaluations (Dodge et al., 2021).

Contamination during fine-tuning (Sainz et al., 2023) Directly fine-tuning models on benchmark data constitutes a significant form of data contamination. This practice is relatively uncommon in industrial settings (Dekoninck et al., 2024b), as developers prioritize maintaining model generalizability. However, in academic contexts, researchers deliberately employ this approach to create contaminated models for controlled experimental analysis.

Contamination after deployment (Balloccu et al., 2024) Post-deployment contamination introduces the notion of *indirect data leakage*, where human interactions during model operation may inadvertently expose benchmark.

Multi-modal contamination (Song et al., 2024) Contamination in multi-modal models can occur in two primary ways. First, when text-label pairs or just the text inputs appear in the training corpus, creating direct overlap with test examples. Second, when triplets containing text, image, and labels

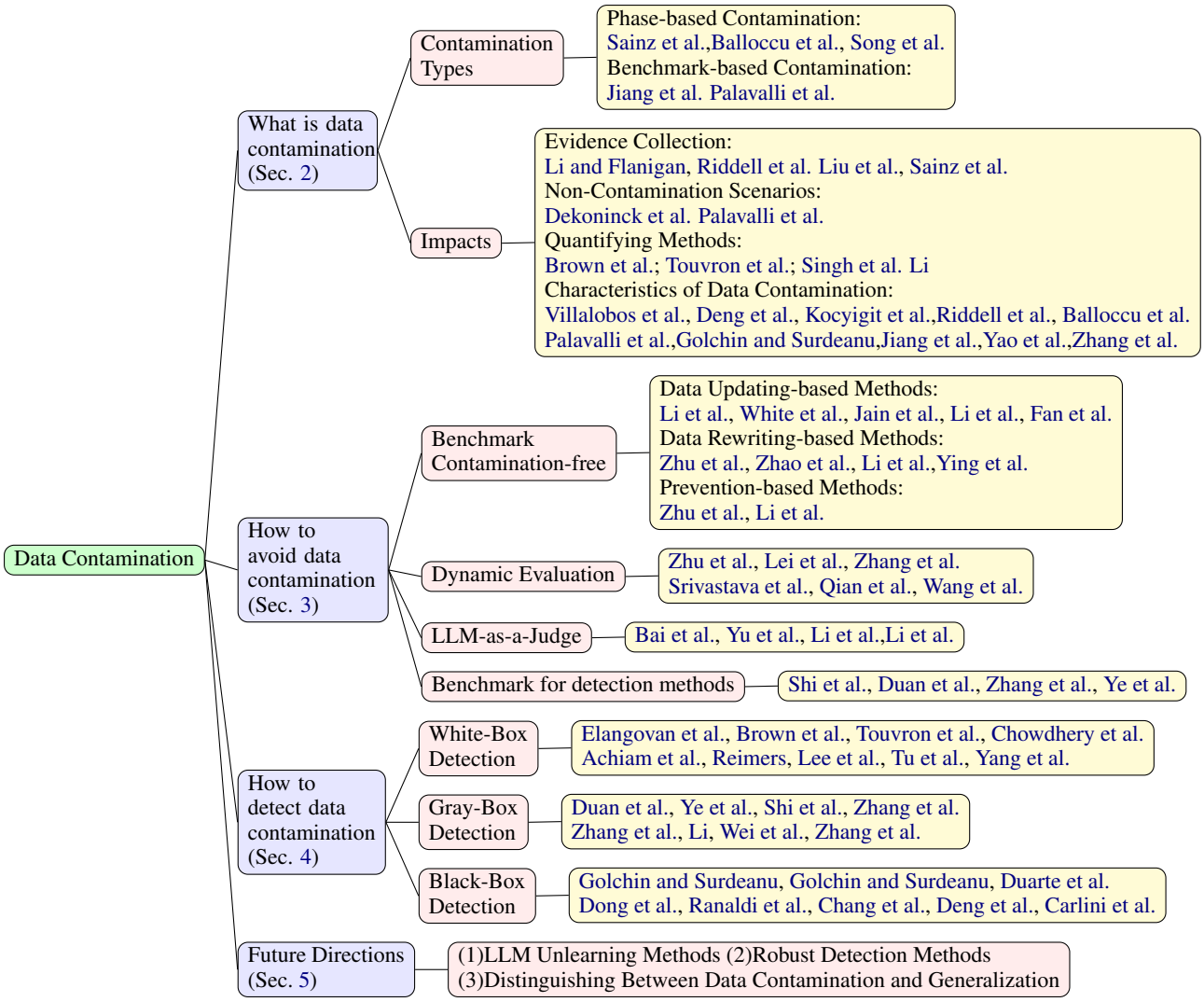

Figure 1: Structure of this paper

appear in the training corpus, allowing models to memorize text-image-label triplet rather than learning generalizable capabilities.

2.1.2 Benchmark-based Contamination

Text contamination (Jiang et al., 2024) 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) Text-label contamination refers to cases where the pre-training corpus contains not only input text but also prompts (task instructions) and corresponding labels or answers from evaluation samples, effectively exposing the model to both questions and correct answers prior to testing.

Augmentation-based contamination (Palavalli et al., 2024) Augmentation-based contamination refers to data contamination caused by means such as sample masking (deleting key input/output content), noise injection (rewriting/replacing labels/alternative answers), or adversarial augmentation (adding distracting options/irrelevant context).

Benchmark-level contamination (Palavalli et al., 2024) Benchmark-level contamination arises when the model incorporates partial source corpora of benchmark datasets or outdated versions of these benchmarks during the training process.

2.2 Impacts

Data contamination critically undermines evaluation reliability and research validity. As (Sainz et al., 2023; Riddell et al., 2024) demonstrated, benchmark overfitting can artificially inflate model performance and compromise scientific conclu-

sions in NLP studies. For a comprehensive understanding of data contamination, we need to collect evidence, clarify non-contamination scenarios, quantify contamination and identify characteristics of data contamination.

2.2.1 Evidence Collection

Some papers reproduce results from previous work by attributing them to data contamination. Li and Flanigan (2024) proposed evaluating models on training datasets with membership inference attacks, revealing contamination effects on performance. Riddell et al. (2024) quantitatively analyzes the overlap between popular code generation benchmarks and pretraining corpora and reveals data contamination. Liu et al. (2024) revealed that Chinese LLMs exhibit superficial knowledge despite extensive training, a phenomenon that may partially stem from data contamination. For a broad perspective, Sainz et al. (2023) highlighted that current evidence on contamination remains fragmented across publications and informal channels, suggesting that the prevalence of contamination may be significantly underestimated.

2.2.2 Non-Contamination Scenarios

In this section, we explore non-contamination scenarios, where the overlap between training and testing data does not lead to performance improvement. Dekoninck et al. (2024a) established a causal relationship between model performance improvement and data contamination, explicitly defining cases where such overlap exists but does not enhance performance as non-contamination. Furthermore, Palavalli et al. (2024) clarified several phenomena that improve performance on downstream tasks without being influenced by contamination. These include language understanding, prior task understanding, and transductive learning. These phenomena enhance empirical results while preserving the integrity of both the task and the model, distinguishing them from data contamination.

2.2.3 Quantifying Contamination

For model developers, the scenario for quantifying contamination focuses on using the N-grams algorithm to measure the overlap between the training and test data. Contamination scoring mechanisms classify evaluation samples through threshold-based indices. For instance, Brown et al. (2020) used N-grams to evaluate contamination by checking whether each token in the tested sample appears in an n-gram from the pre-training corpus. In contrast, Touvron et al. (2023) introduced a method to align extensions between the testing samples and pre-training corpus, allowing mismatches in certain token positions using a "skip_budget" hyperparameter. Singh et al. (2024) further extended this method, focusing on the longest contaminated token span rather than all potential matches. In cases where this corpus is unavailable, search engines can be utilized to identify relevant problems and their corresponding solutions. If we can find it, it has likely already been used as training data by large language models (Li, 2023b). Besides N-grams, (Cao et al., 2024) proposed that current MIA-related metrics such as perplexity and Zlib compression entropy cannot effectively distinguish contaminated data from cleansed data, and there is a need for new metrics to quantify data contamination. Singh et al. (2024) proposed a new contamination evaluation protocol, ConTAM, to explore how data contamination affects the evaluation results of LLMs, and provided a method to quantify it.

2.2.4 Characteristics of Data Contamination

In this section, we summarize five key characteristics of data contamination in large language models.

The inevitability of data contamination As large language models (LLMs) continue to scale up, the size of their training datasets expands 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.

Scaling laws Larger models exhibit stronger contamination effects than smaller ones (Kocyigit et al., 2025). 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).

Cross-stage Characteristics Data contamination can occur during model pre-training, post-training, and deployment (Balloccu et al., 2024). Kocyigit et al. (2025) found that the training stage 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-stage contamination ultimately causes a larger perfor-

mance gap. Uniform contamination (spread across the entire training process) produces the most lasting effects, with no significant spikes.

Task-specific Characteristics Recent research indicates that data contamination impact varied across different tasks. Comparative experiments by Palavalli et al. (2024) further reveal significant differences in the performance gains contributed by verbatim versus reformatted contamination in summarization tasks. Moreover, Golchin and Surdeanu (2023b), using an prompt-guided detection approach, demonstrated that the efficacy of the same contamination type varies markedly across different tasks during pretraining. Jiang et al. (2024) confirmed that the order and distribution of contaminated data moderate performance across different tasks, underscoring the need for evaluation protocols to be differentially designed for each task.

Cross-lingual Characteristics LLMs are overfitted to translated versions of benchmark test sets in non-English languages. This practice inflates 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). Kocyigit et al. (2025) finds that contamination requires sufficient language representation to produce measurable effects: for resource-scarce languages, contamination has almost no impact on performance. These findings reveal the complexity and threshold effects of data contamination in multilingual environments.

3 How to Avoid Data Contamination

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).

3.1 Benchmark Contamination-free Strategies

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

3.1.1 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) is proposed to dynamically align with recent code repositories for fair evaluation.

3.1.2 Data Rewriting-based Methods

This type of methods use data augmentation to remove contamination from benchmarks, with LLMs' superior rephrasing and verifying capabilities. Zhu et al. (2024d) proposed Clean-Eval to purify contaminated benchmarks by paraphrasing and back-translating data into semantically equivalent but lexically distinct forms. Zhao et al. (2024) proposed the MMLU-CF dataset, which is constructed by collecting diverse questions, cleaning data, sampling difficulty reasonably, checking data integrity with LLMs, and applying rewriting methods such as rephrasing questions and shuffling options to ensure the dataset remains contamination-free. CLEVA (Li et al., 2023) employs 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.

3.1.3 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 and enhanced encryption.

3.2 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.

3.3 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.

3.4 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.

4 How to Detect Data Contamination

Data contamination detection involves identifying whether a text or dataset has been included in a model's training corpus. We categorize detection methods into three paradigms based on model access: white-box, gray-box, and black-box and analyse robustness in section 5.2. For practical imple-

mentation guidance, we list some detection tools in Appendix C.

4.1 White-Box Detection

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

N-gram based The n-gram overlap method, known for its effectiveness and simplicity, is widely used for model developer contamination detection. Leading LLMs like LLaMA2 (Touvron et al., 2023), PaLM (Chowdhery et al., 2023), and GPT-4 (Achiam et al., 2023) stress the importance of detecting train/test overlaps. Riddell et al. (2024) quantifies the contamination levels between code generation benchmarks and pretraining corpora through surface-level and semantic-level methods involving n-gram matching.

Embeddings similarity based Embeddings similarity compares texts via cosine similarity of their embeddings (with a threshold value), capturing semantic relationships beyond lexical variations (Reimers, 2019). Lee et al. (2023) used a similarity exclusion method based on embeddings, reducing dataset redundancy and filtering out duplicate data to ensure clean training data. To address sophisticated contamination forms, Yang et al. (2023) introduced a hybrid approach combining embedding similarity search with GPT-4 powered semantic analysis. This detects paraphrased samples, enabling proactive benchmark decontamination.

Layer-specific Tu et al. (2024) proposed DICE to detect in-distribution contamination during fine-tuning by analyzing layer-specific activation patterns. This method trains contamination classifiers on sensitive intermediate layers, demonstrating a strong correlation between detection signals and performance inflation across multiple LLMs.

4.2 Gray-Box Detection

Gray-box approaches utilize partial model information, such as token probabilities, to compute perplexity or confidence, which can help detect data contamination. These methods mostly make a binary decision by comparing the score against a threshold value. Duan et al. (2024) systematically investigated the underwhelming MIA performance on LLMs, identifying three primary contributing factors: the massive scale of training datasets that complicates memorization patterns, the limited number of training iterations that reduce model overfitting, and the inherently fuzzy decision boundaries between member and non-member samples. To address these shortcomings, the Min-K% method established token-based effective methods using outlier token probabilities for pretraining data detection (Shi et al., 2024). However, the effectiveness of this approach heavily depends on pre-designed K values and threshold parameters. Zhang et al. (2024c) subsequently proposed Min-K%++, theoretically grounding detection in local probability maxima identification and moved beyond heuristic-based methods. Similarly, based on the Min-K% method, Ye et al. (2024) introduced PAC, an MIA method that calculates polarization distances (based on Max k_1% and Min k_2%) through input perturbations. Zhang et al. (2024d) proposed DC-PDD to employ corpus frequency divergences to reduce false positives. Without a threshold value, Zhang et al. (2024b) introduced PaCoST, a method that detects data contamination by statistically comparing confidence scores between original test items and their semantically equivalent paraphrased counterparts. Alternative perplexity-based methods, Li (2023a) compare perplexity on benchmark samples against contaminated and clean baselines to show data contamination.Similar to this, (Wei et al., 2023) uses GPT-4 to generate data that is stylistically similar to the original GSM8K. The authors then compute the perplexity on the GSM8K training set (train), GSM8K test set (test), and GSM8K reference set (ref).

Efficiency analysis Token-based methods(Min-K% (Shi et al., 2024), PAC (Ye et al., 2024), Min-K%++ (Zhang et al., 2024c), DC-PDD (Zhang et al., 2024d)) consist of a constant number of LLM forward passes and then some basic algebraic operations, making them efficient. And PaCoST (Zhang et al., 2024b) requires computational resources for sample paraphrasing.

4.3 Black-Box Detection

Black-box methods operate without access to model internals, training corpus, primarily relies on the model's outputs for decision-making. Specifically, these methods heavily rely on certain assumptions shown in Appendix B.

Memorization-based methods Masked and completions are key practices in black-box methods. Golchin and Surdeanu (2023b) first proposed a guided prompt-based detection method, which

effectively identifies contamination in datasets through instance completion. Golchin and Surdeanu (2023a) introduced DCQ, a multiple-choice question framework where each question presents an original instance alongside three perturbed versions (in which words are replaced with contextually relevant synonyms) and one invalid option. If the LLM consistently selects the original instance, this behavior may indicate data contamination. Similarly, Chang et al. (2023) introduced a challenging cloze task and used data archaeology to investigate the memorization of passages from 571 novels by LLMs. Deng et al. (2023) proposed TS-Guessing, a protocol designed to test a model's ability to reconstruct masked elements of test data. Duarte et al. (2024) developed DE-COP, a copyright detection framework that employs verbal versus paraphrased multiple-choice probing. This approach uncovers subtle contamination in major benchmarks. As highlighted by (Ranaldi et al., 2024), the Text-to-SQL task with GPT-3.5 involves data contamination, where the model is tasked with reconstructing masked column names using the table name, the remaining column names, and contextual information.

In contrast to previous studies, Dong et al. (2024) introduced CDD to identify contamination by analyzing the peakedness of output distributions. When paired with the TED mitigation technique, the CDD approach effectively addresses both explicit and implicit forms of contamination while preserving the validity of model evaluations. Canary insertion (Carlini et al., 2021a) involves retraining open models on synthetic benchmark-mimicking examples ("canaries") and measuring recall rates, with higher recall rates indicating a greater propensity for memorization.

5 Future Directions

5.1 LLM Unlearning Methods

Unlearning techniques offer the potential to mitigate LLM privacy risks by erasing specific data elements. For language models, (Jang et al., 2023) demonstrates that performing gradient ascent on target token sequences is an effective method for forgetting them. Eldan and Russinovich (2023) is the first paper to present an effective technique for unlearning in large language models. Future research should explore integrating contamination mitigation through unlearning methods. This emerging field shows promise and fundamental challenges. For instance, Shumailov et al. (2024) claims such data erasure may be fundamentally unachievable in current architecture. Shi et al. (2024) uses Min-K% to audit unlearning methods and they've found that some content still remain.

5.2 Robust Detection Methods

Current detection methods confront several challenges. Existing black-box contamination detection approaches rely on heuristic rules that Fu et al. (2024) showed fail under certain conditions, raising concerns about their fundamental reliability. Safety mechanisms within LLMs further complicate detection efforts, as methods that directly prompt for contaminated content often trigger filters that mask contamination indicators. Meanwhile, traditional methods may lack effectiveness in detecting augmentation-based contamination(Dekoninck et al., 2024b). Future work should focus on developing detection methods with more robust assumptions that remain valid across a variety of scenarios and transformation strategies.

5.3 Distinguishing Between Data Contamination and Generalization

The ambiguity between contamination and generalization remains unresolved. (Ishikawa, 2025) systematically distinguishes data contamination from generalization in LLM benchmarks through a three-tier framework combining n-gram alignment, canary insertion, and perturbation testing. In this context, out-of-distribution (OOD) data performance is considered a true form of generalization. Building on this, future work should emphasize the distinctions between data contamination and generalization.

6 Conclusion

Our paper examines three fundamental perspectives in data contamination research: (1) defining data contamination through the lenses of phases and benchmarks; (2) exploring methodologies for conducting contamination-free evaluations, with a particular focus on dynamic evaluation and LLM-based assessment techniques; and (3) investigating methods for detecting data contamination, offering a comprehensive analysis of existing techniques and their limitations. Furthermore, we provide actionable recommendations for enhancing contamination-aware evaluation systems, aiming to foster more reliable LLM development practices.

7 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. And there are so many static benchmarks that we only list some to demonstrate a contamination-free benchmark construction method. 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.

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*.

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. 2024. Elephants never forget: Memorization and learning of tabular data in large language models. In *Conference on Language Modeling (COLM)*.

Tom Brown, Benjamin Mann, Nick Ryder, Melanie Subbiah, Jared D Kaplan, Prafulla Dhariwal, Arvind Neelakantan, Pranav Shyam, Girish Sastry, Amanda Askell, and 1 others. 2020. Language models are few-shot learners. *NeurIPS*, 33:1877–1901.

Jialun Cao, Wuqi Zhang, and Shing-Chi Cheung. 2024. Concerned with data contamination? assessing countermeasures in code language model. *arXiv preprint arXiv:2403.16898*.

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

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

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

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

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

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

Aakanksha Chowdhery, Sharan Narang, Jacob Devlin, Maarten Bosma, Gaurav Mishra, Adam Roberts, Paul Barham, Hyung Won Chung, Charles Sutton, Sebastian Gehrmann, and 1 others. 2023. Palm: Scaling language modeling with pathways. *Journal of Machine Learning Research*, 24(240):1–113.

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

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

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

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

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

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

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

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

Aparna Elangovan, Jiayuan He, and Karin Verspoor. 2021. Memorization vs. generalization : Quantifying data leakage in NLP performance evaluation. In *Proceedings of the 16th Conference of the European Chapter of the Association for Computational Linguistics: Main Volume*, pages 1325–1335, Online. Association for Computational Linguistics.

Ronen Eldan and Mark Russinovich. 2023. Who's harry potter? approximate unlearning in llms. *Preprint*, arXiv:2310.02238.

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*.

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*.

Joel Jang, Dongkeun Yoon, Sohee Yang, Sungmin Cha, Moontae Lee, Lajanugen Logeswaran, and Minjoon Seo. 2023. Knowledge unlearning for mitigating privacy risks in language models. In *Proceedings of the 61st Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 14389–14408, Toronto, Canada. Association for Computational Linguistics.

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*.

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.

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

Yucheng Li. 2023b. An open source data contamination report for large language models. *arXiv preprint arXiv:2310.17589*.

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*.

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.

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.

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.

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*.

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, Jamie Hayes, Eleni Triantafillou, Guillermo Ortiz-Jimenez, Nicolas Papernot, Matthew Jagielski, Itay Yona, Heidi Howard, and Eugene Bagdasaryan. 2024. Ununlearning: Unlearning is not sufficient for content regulation in advanced generative ai. *arXiv preprint arXiv:2407.00106*.

Aaditya K Singh, Muhammed Yusuf Kocyigit, Andrew Poulton, David Esiobu, Maria Lomeli, Gergely Szilvasy, and Dieuwke Hupkes. 2024. Evaluation data contamination in llms: how do we measure it and (when) does it matter? *arXiv preprint arXiv:2411.03923*.

Dingjie Song, Sicheng Lai, Shunian Chen, Lichao Sun, and Benyou Wang. 2024. Both text and images

leaked! a systematic analysis of multimodal llm data contamination. *arXiv preprint arXiv:2411.03823*.

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*.

Hugo Touvron, Louis Martin, Kevin Stone, Peter Albert, Amjad Almahairi, Yasmine Babaei, Nikolay Bashlykov, Soumya Batra, Prajjwal Bhargava, Shruti Bhosale, and 1 others. 2023. Llama 2: Open foundation and fine-tuned chat models. *arXiv preprint arXiv:2307.09288*.

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.

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*.

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*.

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

Ruijie Xu, Zengzhi Wang, Run-Ze Fan, and Pengfei Liu. 2024b. Benchmarking benchmark leakage in large language models. *Preprint*, arXiv:2404.18824.

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*.

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.

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*.

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 1 others. 2024. Mmlu-cf: A contamination-free multi-task language understanding benchmark. *arXiv preprint arXiv:2412.15194*.

Wayne Xin Zhao, Kun Zhou, Junyi Li, Tianyi Tang, Xiaolei Wang, Yupeng Hou, Yingqian Min, Beichen Zhang, Junjie Zhang, Zican Dong, and 1 others. 2023. A survey of large language models. *arXiv preprint arXiv:2303.18223*.

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*.

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 Data Contamination Evidence Collection Efforts

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.

B Definition of Assumptions

B.1 Verbatim Memorization

In the context of LLMs, verbatim memorization (Carlini et al., 2021b, 2022) 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.

B.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.

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.

B.3 Memorization and Data Contamination

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

C Data Contamination Detector

Li (2023b) 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/liyucheng09/Contamination_Detector.

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

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

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

Bordt et al. (2023, 2024) 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 Prior Surveys

Prior research on data contamination primarily focuses on three main areas: definition, detection, and mitigation. Xu et al. (2024a) and Deng et al. (2024) provide comprehensive surveys that thoroughly examine data contamination in large language models, covering conceptual definitions, detection methodologies, and mitigation strategies with similar classification frameworks for detection methods (matching-based and comparison-based). However, they differ significantly in their conceptualization of contamination types. The first paper primarily distinguishes between task-level contamination and language-level contamination, providing a function-oriented taxonomy. In contrast, the second paper presents a more granular severity-

based hierarchy with four distinct levels: semantic-level (topical overlap), information-level (metadata and distributions), data-level (content without labels), and label-level contamination (complete exposure including ground truth). Their approaches to mitigation strategies also diverge notably. While the first paper emphasizes evaluation guidelines and procedural recommendations, the second paper offers a more structured framework categorized into three comprehensive strategies: data curation, data refactoring, and benchmark-free evaluation.

The remaining three studies focus on specialized subdomains of data contamination. (Fu et al., 2024) focuses on black-box detection assumptions and shows these methods fail under certain conditions, raising concerns about their fundamental reliability. Ravaut et al. (2024) investigates the critical issue of contamination in LLMs, categorizing it into data contamination and model contamination , while further distinguishing between input-only and input-label contamination scenarios. The authors systematically review state-of-the-art detection methods, including string matching, embedding similarity analysis, likelihood-based techniques, and novel LLM-driven approaches, highlighting their strengths and limitations. Chen et al. (2025) 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.