Benchmarking Large Language Models Under Data Contamination: A Survey from Static to Dynamic Evaluation

Anonymous ACL submission

Abstract

In the era of evaluating large language models (LLMs), data contamination has become an increasingly prominent concern. To address this risk, LLM benchmarking has evolved from a *static* to a *dynamic* paradigm. In this work, we conduct an in-depth analysis of existing *static* and *dynamic* benchmarks for evaluating LLMs. We first examine methods that enhance *static* benchmarks and identify their inherent limitations. We then highlight a critical gap—the lack of standardized criteria for evaluating *dynamic* benchmarks. Based on this observation, we propose a series of optimal design principles for *dynamic* benchmarking and analyze the limitations of existing *dynamic* benchmarks. This survey provides a concise yet comprehensive overview of recent advancements in data contamination research, offering valuable insights and a clear guide for future research efforts. We maintain a GitHub repository to continuously collect both static and dynamic benchmarking methods for LLMs. The repository can be found at this link[1].

1 Introduction

The field of natural language processing (NLP) has advanced rapidly in recent years, fueled by breakthroughs in Large Language Models (LLMs) such as GPT-4, Claude3, and DeepSeek (Achiam et al., 2023; Liu et al., 2024; Wan et al., 2023). Trained on vast amounts of Internet-sourced data, these models have demonstrated remarkable capabilities across various applications, including code generation, text summarization, and mathematical reasoning (Codeforces, 2025; Hu et al., 2024).

To develop and enhance LLMs, beyond advancements in model architectures and training algorithms, a crucial area of research focuses on effectively evaluating their intelligence. Traditionally, LLM evaluation has relied on *static* benchmarking, which involves using carefully curated human-crafted datasets and assessing model performance with appropriate metrics (Wang, 2018; Achiam et al., 2023; Gunasekar et al., 2023).

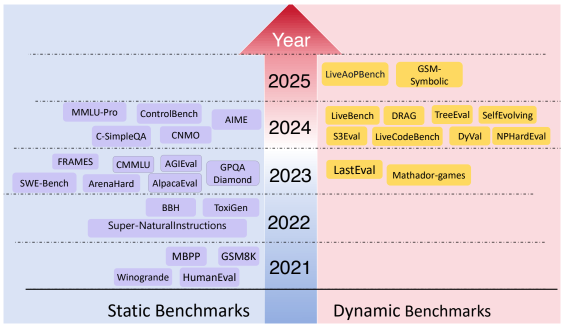

Figure 1: The progress of benchmarking LLM

However, because these *static* benchmarks are released on the Internet for transparent evaluation, and LLMs gather as much data as possible from the Internet for training, potential data contamination is unavoidable (Magar and Schwartz, 2022; Deng et al., 2024c; Li et al., 2024d; Sainz et al., 2024; Balloccu et al., 2024a). Data contamination occurs when benchmark data is inadvertently included in the training phase of language models, leading to inflated and misleading performance assessments. Although this issue has long been recognized—rooted in the fundamental machine learning principle of separating training and test sets—it has become more critical with the rise of LLMs, which often scrape vast amounts of publicly available Internet data (Achiam et al., 2023), increasing the risk of contamination.

To mitigate the risk of data contamination in LLM benchmarking, researchers have proposed several enhancements to static evaluation methods, including data encryption (Jacovi et al., 2023) and post-hoc contamination detection (Shi et al., 2024). However, due to the inherent limitations of static approaches—such as unverifiable data exposure—these enhancements have seen limited adoption. As a result, researchers have shifted toward

[1] Static-to-Dynamic-LLMEval GitHub Repository

new *dynamic* benchmarking paradigms, as illustrated in Fig. 1. Dynamic methods aim to reduce contamination risk either by continuously updating benchmark datasets based on LLM training timestamps (White et al., 2024; Jain et al., 2024), or by regenerating test data to reconstruct and replace original benchmarks (Chen et al., 2024; Zhou et al., 2025; Mirzadeh et al., 2025).

Although many dynamic benchmarking methods have been proposed to promote fair and transparent evaluation of LLMs, most existing work primarily highlights the advantages of these dynamic benchmarks (White et al., 2024). However, the question remains: *What are the potential trade-offs of using dynamic benchmarks to evaluate LLMs?* The limitations of dynamic benchmarking—such as the computational overhead of continuous updates, and the need for reliable timestamp metadata—are not yet fully explored.

Moreover, existing surveys on LLM data contamination have mainly focused on post-hoc detection techniques (Deng et al., 2024b; Ravaut et al., 2024; Xu et al., 2024a; Dong et al., 2024; Balloccu et al., 2024b), offering little attention to the emerging landscape of dynamic benchmarking strategies. Considering the growing importance and adoption of dynamic benchmarking methods, it is essential to assess their effectiveness and limitations. Unfortunately, our empirical survey of existing dynamic benchmarking approaches reveals that their evaluations are highly fragmented. To date, there is no systematic work that defines clear evaluation criteria for dynamic benchmarks themselves. Moreover, existing reviews often overlook a detailed comparison of the strengths and weaknesses of different dynamic methods, leaving a gap in understanding their practical trade-offs and applicability.

To bridge this gap, we first conduct a systematic survey of benchmarking methods for LLMs designed to mitigate the risk of data contamination, covering both *static* and *dynamic* benchmarks. We summarize state-of-the-art methods and provide an in-depth discussion of their strengths and limitations. Furthermore, we are the first to summarize and abstract a set of criteria for evaluating *dynamic* benchmarks. Our study reveals that existing *dynamic* benchmarks do not fully satisfy these proposed criteria, implying the imperfection of current design. We hope that our criteria will provide valuable insights for the future design and standardization of *dynamic* benchmarking methods.

The paper is organized as shown in Fig. 2. We first review the background on data contamination (§2), then survey *static* benchmarks and their improvements (§3). Next, we introduce key principles and existing approaches for *dynamic* benchmarking (§4). Finally, we discuss open challenges and future directions (§5).

2 Background

2.1 Data Contamination

Data contamination arises when LLM training data $\mathcal{D}_{\text{train}}$ improperly overlaps with evaluation data $\mathcal{D}_{\text{test}}$, undermining performance validity. We review existing work and formalize the definition.

Exact contamination occurs when there is any exact duplicate in the benchmark dataset

$$\exists\, d \quad \text{s.t.} \quad d \in \mathcal{D}_{\text{train}} \text{ and } d \in \mathcal{D}_{\text{test}}$$

In other word, there exist a data point d that both in $\mathcal{D}_{\text{train}}$ and $\mathcal{D}_{\text{test}}$. Common cases include verbatim test examples appearing in training corpora, code snippets from benchmark implementations, or documentation leaks.

Syntactic contamination occurs when a test data point could be found in the training dataset after a syntactic transformation, such that

$$\exists\, d \quad \text{s.t.} \quad \mathcal{F}_{\text{syntactic}}(d) \in \mathcal{D}_{\text{train}} \text{ and } d \in \mathcal{D}_{\text{test}}$$

where $\mathcal{F}_{\text{syntactic}}$ denotes syntactic transformations like punctuation normalization, whitespace modification, synonym substitution, morphological variations, or syntactic paraphrasing while preserving lexical meaning.

Examples of each contamination We provide contamination examples in Table 1. Syntactic contamination occurs when test data is rephrased from training data using a prefix. Whether this constitutes true contamination is debated, as it's difficult to separate memorization from reasoning. In this work, we treat such transformations as contamination, since some NLP tasks rely heavily on syntax.

Significance of Contamination Data contamination poses a serious threat to the integrity of LLM benchmarking, particularly as models grow in scale and are trained on vast, publicly available corpora. Without proper safeguards, evaluations may inadvertently test models on data they have seen during training, leading to inflated performance metrics and misleading claims about generalization and robustness. Recent studies underscore this concern: Schaeffer (2023) demonstrate that

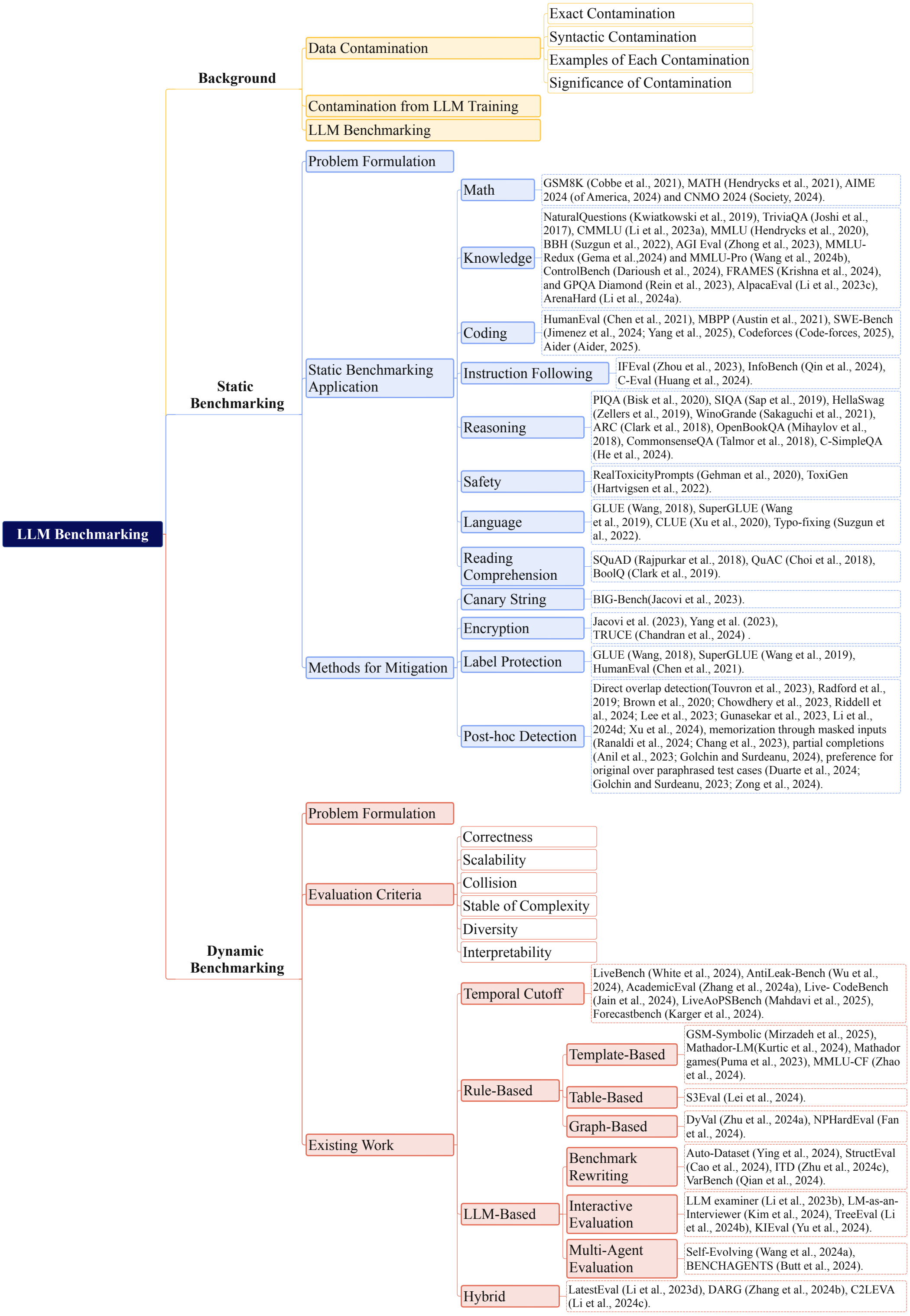

Figure 2: Taxonomy of research on benchmarking LLMs

Contamination Type	Training Data	Testing Data
Exact Contamination	`Write a Python function to check if a number is prime.`	`Write a Python function to check if a number is prime.`
Syntactic Contamination	`Write a Python function to check if a number is prime.`	`You are a helpful code assistant for Python. Write a Python function to check if a number is prime.`

Table 1: Examples of Data Contamination in LLMs

pretraining on test data can significantly distort evaluation outcomes; Balloccu et al. (2024b) reveal how easily data contamination and evaluation malpractices can occur in closed-source LLMs; Xu et al. (2024b) propose methods to quantify such contamination; and Deng et al. (2024a) provide a comprehensive survey of existing risks and mitigation strategies. The issue gained public attention when Meta's LLaMA 4 faced allegations of using a non-public version fine-tuned for benchmark gains (Babic, 2025), raising concerns about evaluation transparency—despite Meta's denial of test set exposure. Such cases underscore the need for contamination-aware benchmarking to accurately assess LLM performance on truly unseen data. We also present a proof-of-concept evaluation in §A to highlight the impact of data contamination.

2.2 Contamination Source

Data contamination can occur during the pre-training, post-training, or fine-tuning phases of LLM development. Unlike traditional models with clear separations between training and evaluation data, LLMs are pre-trained on massive, diverse datasets—often scraped from the web (e.g., FineWeb (Penedo et al., 2024))—which increases the risk of evaluation data overlap. In the post-training phase, models are further fine-tuned on large human-annotated (Mukherjee et al., 2023; Kim et al., 2023) or synthetic datasets (Ding et al., 2023; Teknium, 2023; Wang et al., 2023) that may resemble evaluation tasks, further compounding contamination risks. Although retrieval-based detection methods (Team et al., 2024; Achiam et al., 2023) exist, the sheer scale and complexity of training corpora make it difficult to entirely exclude evaluation data. Additionally, many LLMs keep their training data proprietary (Dubey et al., 2024; Yang et al., 2024), complicating the accurate assessment of their true performance and highlighting the need for fair and reliable benchmarks. This opacity further exacerbates data contamination, as it impedes the community's ability to verify and mitigate potential overlaps between training and evaluation data.

2.3 LLM Benchmarking

As LLMs evolve into general-purpose task solvers, it is crucial to develop benchmarks that provide a holistic view of their performance. To this end, significant human effort has been dedicated to building comprehensive benchmarks that assess various aspects of model performance. For example, instruction-following tasks evaluate a model's ability to interpret and execute commands (Zhou et al., 2023; Qin et al., 2024; Huang et al., 2024), while coding tasks assess its capability to generate and understand programming code (Chen et al., 2021; Austin et al., 2021; Jimenez et al., 2024; Codeforces, 2025; Aider, 2025). Despite their usefulness, static benchmarks face challenges as LLMs evolve rapidly and continue training on all available data (Villalobos et al., 2022). Over time, unchanging benchmarks may become too easy for stronger LLMs or introduce data contamination issues. Recognizing this critical problem, contamination detectors have been developed to quantify contamination risks, and dynamic benchmarks have been proposed to mitigate these issues.

3 Static Benchmarking

3.1 Problem Formulation

A static benchmark is given by $\mathcal{D} = (\mathcal{X}, \mathcal{Y}, \mathcal{S}(.))$, where $\mathcal{D}$ represents the seed dataset, consisting of input prompts $\mathcal{X}$, expected outputs $\mathcal{Y}$, and a scoring function $\mathcal{S}(\cdot)$ that evaluates the quality of an LLM's outputs by comparing them against $\mathcal{Y}$.

3.2 Static Benchmark Application

Math Math benchmarks evaluate a model's ability to solve multi-step math problems. Datasets such as GSM8K (Cobbe et al., 2021) and MATH (Hendrycks et al., 2021) require models to work through complex problems. Recent challenges like AIME 2024 (of America, 2024) and CNMO 2024 (Society, 2024) further test a model's capacity to tackle diverse and intricate math tasks.

Coding Coding benchmarks measure a model's ability to generate and debug code. HumanEval (Chen et al., 2021) and MBPP (Austin et al., 2021) test code synthesis and debugging, whereas SWE-Bench (Jimenez et al., 2024; Yang et al., 2025) addresses more advanced challenges. Competitive platforms like Codeforces (Codeforces, 2025) and datasets such as Aider (Aider, 2025) further probe dynamic problem solving.

Instruction Following Instruction benchmarks evaluate a model's ability to comprehend and execute detailed directives. Datasets like IFEval (Zhou et al., 2023) and InfoBench (Qin et al., 2024) simulate real-world scenarios requiring clear, step-by-step guidance, with C-Eval (Huang et al., 2024) focusing on Chinese instructions.

Other Applications We provide a detailed introduction to other applications in Appendix B, along with a further analysis on enhancing static benchmarks in Appendix C.

4 Dynamic Benchmarking

4.1 Problem Formulation

A dynamic benchmark is defined as $\mathcal{B}_{\text{dynamic}} = (\mathcal{D}, T(\cdot)), \quad \mathcal{D} = (\mathcal{X}, \mathcal{Y}, \mathcal{S}(\cdot))$ where $\mathcal{D}$ represents the static benchmark dataset. The transformation function $T(\cdot)$ modifies the data set during the benchmarking to avoid possible data contamination. The dynamic dataset for the evaluation of an LLM can then be expressed as $\mathcal{D}_t = T_t(\mathcal{D}), \quad \forall t \in \{1, \ldots, N\}$ where $\mathcal{D}_t$ represents the evaluation data set at the timestamp t, and N is the total timestamp number, which could be finite or infinite. If the seed dataset $\mathcal{D}$ is empty, the dynamic benchmarking dataset will be created from scratch.

4.2 Criteria Summarization and Abstraction

While many dynamic benchmarking methods have been proposed to evaluate LLMs, the criteria for evaluating these benchmarks themselves remain non-standardized. To address this gap, we analyze existing evaluation practices and abstract them into a unified framework. We review over 50 dynamic benchmarking papers, focusing specifically on how they evaluate their own benchmarks. Although many of these works include some form of self-evaluation, the approaches are often inconsistent, incomplete, or lack depth. For example, DyVal2 evaluates benchmark complexity and correctness, but does not address the interpretability of the benchmark construction process.

To systematize this landscape, we identify a unified set of evaluation criteria and present them in Table 2. We then assess whether each dynamic benchmark fully supports, partially supports, or does not support each criterion. For instance, in the case of correctness: benchmarks with built-in guarantees—such as those using temporal cutoffs or rule-based generation—are marked as "supported". Benchmarks generated using LLMs are marked as "partially supported" if they include validation (e.g., human or automated checks); otherwise, they are labeled "not supported." More guidance for classify each dynamic benchmarks could be found in §D.

4.3 Summarized Evaluation Criteria

4.3.1 Correctness

The first criterion for evaluating the quality of dynamic benchmarking is $\mathsf{Correctness}$. If the correctness of the generated dataset cannot be guaranteed, the benchmark may provide a false sense of reliability when applied to benchmarking LLMs, leading to misleading evaluations. We quantify the correctness of dynamic benchmarks as:

$$\mathsf{Correctness} = \mathbb{E}_{i=1}^{N} \mathcal{S}(\mathcal{Y}_i, \mathcal{G}(\mathcal{X}_i))$$

where $\mathcal{X}_i$ and $\mathcal{Y}_i$ represent the input and output of the i^{th} transformation, respectively. The function $\mathcal{G}(\cdot)$ is an oracle that returns the ground truth of its input, ensuring an objective reference for correctness evaluation. For example, the function $\mathcal{G}(\cdot)$ could be a domain-specific annotator. This equation can be interpreted as the expected alignment between the outputs of the transformed data set and their corresponding ground truth values, measured using the scoring function $\mathcal{S}(\cdot)$. A higher correctness score indicates that the dynamic benchmark maintains correctness to the ground truth.

4.3.2 Scalability

The next evaluation criterion is scalability, which measures the ability of dynamic benchmarking methods to generate large-scale benchmark datasets. A smaller dataset can introduce more statistical errors during the benchmarking process. Therefore, an optimal dynamic benchmark should generate a larger dataset while minimizing associated costs. The scalability of a dynamic benchmark is quantified as:

$$\mathsf{Scalability} = \mathbb{E}_{i=1}^{N} \left[\frac{\|T_i(\mathcal{D})\|}{\|\mathcal{D}\| \times \mathsf{Cost}(T_i)} \right]$$

This represents the expectation over the entire transformation space, where $\|T_i(\mathcal{D})\|$ is the size of the transformed dataset, and $\|\mathcal{D}\|$ is the size of the original dataset. The function $\mathsf{Cost}(\cdot)$ measures the cost associated with the transformation process, which could include monetary cost, time spent, or manual effort according to the detailed scenarios. This equation could be interpreted as the proportion of data that can be generated per unit cost.

4.3.3 Collision

One of the main motivations for dynamic benchmarking is to address the challenge of balancing transparent benchmarking with the risk of data contamination. Since the benchmarking algorithm is

Dynamic Mechnisms	Bechmark Name	Evaluation Creteria Correctness	Scalability	Collision	Stable of Complexity	Diversity	Interpretability
Temporal Cutoff	LiveBench (White et al., 2024)	●	◐	◐	○	○	●
	AcademicEval (Zhang et al., 2024a)	●	◐	◐	○	○	●
	LiveCodeBench (Jain et al., 2024)	●	◐	◐	○	○	●
	LiveAoPSBench (Mahdavi et al., 2025)	●	◐	◐	○	○	●
	AntiLeak-Bench (Wu et al., 2024)	●	●	◐	●	○	●
Rule-Based	S3Eval (Lei et al., 2024)	●	●	◐	●	◐	●
	DyVal (Zhu et al., 2024a)	●	●	●	●	◐	●
	MMLU-CF (Zhao et al., 2024)	●	◐	○	●	◐	◐
	NPHardEval (Fan et al., 2024)	●	●	●	●	◐	●
	GSM-Symbolic (Mirzadeh et al., 2025)	●	○	◐	●	◐	●
	PPM (Chen et al., 2024)	●	●	●	◐	◐	●
	GSM-Infinite (Zhou et al., 2025)	●	●	●	●	◐	●
LLM-Based	Auto-Dataset (Ying et al., 2024)	◐	●	◐	●	●	○
	LLM-as-an-Interviewer (Kim et al., 2024)	◐	●	◐	◐	●	○
	TreeEval (Li et al., 2024b)	◐	●	◐	◐	◐	○
	BeyondStatic (Li et al., 2023b)	◐	●	◐	○	●	○
	StructEval (Cao et al., 2024)	◐	●	●	●	●	○
	Dynabench (Kiela et al., 2021)	◐	◐	◐	◐	●	○
	Self-Evolving (Wang et al., 2024a)	◐	●	◐	●	●	○
Hybrid	DARG (Zhang et al., 2024b)	◐	●	◐	●	●	◐
	LatestEval (Li et al., 2023d)	●	◐	◐	○	○	●
	C2LEVA (Li et al., 2024c)	◐	●	◐	◐	●	◐

Table 2: Existing dynamic benchmarks and their quality on our summarized criteria. ● represents support, ◐ represents partial support, and ○ represents no support

publicly available, an important concern arises: *If these benchmarks are used to train LLM, can they still reliably reflect the true capabilities of LLMs?* To evaluate the robustness of a dynamic benchmark against this challenge, we introduce the concept of *collision* in dynamic benchmarking. Collision refers to the extent to which different transformations of the benchmark dataset produce overlapping data, potentially limiting the benchmark's ability to generate novel and diverse test cases. To quantify this, we propose the following metrics:

$$\textsf{Collision Rate} = \mathbb{E}_{i,j=1,\, i\neq j}^{N}\left[\frac{\|\mathcal{D}_i \cap \mathcal{D}_j\|}{\|\mathcal{D}\|}\right]$$
$$\textsf{Repeat} = \mathbb{E}_{i=1}^{N}\left[k \mid k = \min\left\{\bigcup_{j=1}^{k}\mathcal{D}_j \supseteq \mathcal{D}_i\right\}\right]$$

Collision Rate measures the percentage of overlap between two independently transformed versions of the benchmark dataset, indicating how much potential contamination among two trials. Repeat Trials quantifies the expected number of transformation trials required to fully regenerate an existing transformed dataset $T_i(\mathcal{D})$, providing insight into the benchmark's ability to produce novel variations. These metrics help assess whether a dynamic benchmark remains effective in evaluating LLM capabilities, even when exposed to potential training data contamination.

4.3.4 Stable of Complexity

Dynamic benchmarks must also account for complexity to help users determine whether a performance drop in an LLM on the transformed dataset is due to potential data contamination or an increase in task complexity. If a dynamic transformation increases the complexity of the seed dataset, a performance drop is expected, even without data contamination. However, accurately measuring the complexity of a benchmark dataset remains a challenging task. Existing work has proposed various complexity metrics, but these are often domain-specific and do not generalize well across different applications. For example, DyVal (Zhu et al., 2024a) proposes applying graph complexity to evaluate the complexity of reasoning problems. Formally, given a complexity measurement function $\Psi(\cdot)$, the stability can be formulated as:

$$\textsf{Stability} = \textsf{Var}(\Psi(D_i))$$

This equation can be interpreted as the variance in complexity across different trials, where high variance indicates that the dynamic benchmarking method is not stable.

4.3.5 Diversity

The diversity metric can be categorized into two components: external diversity and internal diversity: External diversity measures the variation between the transformed dataset and the seed dataset. Internal diversity quantifies the differences between two transformation trials.

$$\textsf{External Diversity} = \mathbb{E}_{i=1}^{N}\Theta(\mathcal{D}_i, \mathcal{D})$$
$$\textsf{Internal Diversity} = \mathbb{E}_{i,j=1,i\neq j}^{N}\Theta(\mathcal{D}_i, \mathcal{D}_j)$$

where $\Theta(\cdot)$ is a function that measures the diversity between two datasets. For example, it could be the N-gram metrics or the reference based metrics, such as BLEU scores.

4.3.6 Interpretability

Dynamic benchmarking generates large volumes of transformed data, making manual verification costly and challenging. To ensure correctness, the transformation process must be interpretable. Interpretable transformations reduce the need for extensive manual validation, lowering costs. Rule-based or manually crafted transformations are inherently interpretable, while LLM-assisted transformations depend on the model's transparency and traceability. In such cases, additional mechanisms like explainability tools, or human-in-the-loop validation may be needed to ensure reliability and correctness.

4.4 Existing Work

Table 4 summarizes recent dynamic benchmarks. Dynamic benchmarks can be categorized into four types: temporal cutoff, rule-based generation, LLM-based generation, and hybrid approaches.

4.4.1 Temporal Cutoff

Since LLMs typically have a knowledge cutoff date, using data collected after this cutoff to construct dataset can help evaluate the model while mitigating data contamination. This approach has been widely adopted to construct reliable benchmarks that prevent contamination. LiveBench (White et al., 2024) collects questions based on the latest information source, e.g., math competitions from the past 12 months, with new questions added and updated every few months. AntiLeak-Bench (Wu et al., 2024) generates queries about newly emerged knowledge that was unknown before the model's knowledge cutoff date to eliminate potential data contamination. AcademicEval (Zhang et al., 2024a) designs academic writing tasks on latest arXiv papers. LiveCodeBench (Jain et al., 2024) continuously collects new human-written coding problems from online coding competition platforms like LeetCode. LiveAoPS-Bench (Mahdavi et al., 2025) collects live math problems from the Art of Problem Solving forum. Forecastbench (Karger et al., 2024) updates new forecasting questions on a daily basis from different data sources, e.g., prediction markets.

Limitations The collection process typically requires significant human effort (White et al., 2024; Jain et al., 2024), and continuous updates demand ongoing human involvement. Despite the popularity of temporal cutoffs, using recent information from competitions to evaluate LLMs can still lead to data contamination, as these problems are likely to be reused in future competitions (Wu et al., 2024). Verification is often overlooked in these live benchmarks (White et al., 2024).

4.4.2 Rule-Based Generation

This method synthesizes new test cases based on predefined rules, featuring an extremely low collision probability (Zhu et al., 2024a).

Template-Based GSM-Symbolic (Mirzadeh et al., 2025) creates dynamic math benchmarks by using query templates with placeholder variables, which are randomly filled to generate diverse problem instances. Mathador-LM(Kurtic et al., 2024) generates evaluation queries by adhering to the rules of Mathador games(Puma et al., 2023) and varying input numbers. MMLU-CF (Zhao et al., 2024) follows the template of multiple-choice questions and generates novel samples by shuffling answer choices and randomly replacing incorrect options with "None of the other choices."

Table-Based S3Eval (Lei et al., 2024) evaluates the reasoning ability of LLMs by assessing their accuracy in executing random SQL queries on randomly generated SQL tables.

Graph-Based In this category, LLMs are evaluated with randomly generated graphs. For instance, DyVal (Zhu et al., 2024a) assesses the reasoning capabilities of LLMs using randomly generated directed acyclic graphs (DAGs). The framework first constructs DAGs with varying numbers of nodes and edges to control task difficulty. These DAGs are then transformed into natural language descriptions through rule-based conversion. Finally, the LLM is evaluated by querying it for the value of the root node. Similarly, NPHardEval (Fan et al., 2024) evaluates the reasoning ability of LLMs on well-known P and NP problems, such as the Traveling Salesman Problem (TSP). Random graphs of varying sizes are synthesized as inputs for TSP to assess the LLM's performance. Xie et al. (2024) automatically constructs Knights and Knaves puzzles with random reasoning graph.

Limitations The pre-defined rules may limit sample diversity, and publicly available rule-generated data may increase the risk of in-distribution contamination during training (Tu et al., 2024).

4.4.3 LLM-Based Generation

Benchmark Rewriting In this category, LLMs are employed to rewrite samples from existing static benchmarks, which may be contaminated. Auto-Dataset (Ying et al., 2024) prompts LLMs to generate two types of new samples: one that retains the stylistics and essential knowledge of the original, and another that presents related questions at different cognitive levels (Bloom et al., 1956). StructEval (Cao et al., 2024) expands on examined concepts from the original benchmark by using LLMs and knowledge graphs to develop a series of extended questions. ITD (Zhu et al., 2024c) utilizes a contamination detector (Shi et al., 2024) to identify contaminated samples in static benchmarks and then prompts an LLM to rewrite them while preserving their difficulty levels. VarBench (Qian et al., 2024) prompts LLMs to generate new ones.

Interactive Evaluation In this category, inspired by the human interview process, LLMs are evaluated through multi-round interactions with an LLM (Li et al., 2023b). LLM-as-an-Interviewer (Kim et al., 2024) employs an interviewer LLM that first paraphrases queries from existing static benchmarks and then conducts a multi-turn evaluation by posing follow-up questions or providing feedback on the examined LLM's responses. TreeEval (Li et al., 2024b) begins by generating an initial question on a given topic using an LLM. Based on the previous topic and the examined LLM's response, it then generates follow-up subtopics and corresponding questions to further assess the model. KIEval (Yu et al., 2024) generates follow-up questions based on the evaluated model's response to an initial question from a static benchmark.

Multi-Agent Evaluation Inspired by the recent success of multi-agents systems (Guo et al., 2024), multi-agent collaborations are used to construct dynamic benchmarks. Benchmark Self-Evolving (Wang et al., 2024a) employs a multi-agent framework to dynamically extend existing static benchmarks, showcasing the potential of agent-based methods. Given a task description, BENCHAGENTS (Butt et al., 2024) leverages a multi-agent framework for automated benchmark creation. It splits the process into planning, generation, verification, and evaluation—each handled by a specialized LLM agent. This coordinated approach, with human-in-the-loop feedback, yields scalable, diverse, and high-quality benchmarks.

Limitations The quality of LLM-generated samples is often uncertain. For instance, human annotation in LatestEval (Li et al., 2023d) reveals that 10% of samples lack faithfulness or answerability. In interactive settings, reliability further depends on the interviewer LLM.

4.4.4 Hybrid Generation

LatestEval (Li et al., 2023d) combines temporal cutoff and LLM-based generation to automatically generate reading comprehension datasets using LLMs on real-time content from sources such as BBC. DARG (Zhang et al., 2024b) integrates LLM-based and graph-based generation. It first extracts reasoning graphs from existing benchmarks and then perturbs them into new samples using predefined rules. C^2LEVA (Li et al., 2024c) incorporates all three contamination-free construction methods to build a contamination-free bilingual evaluation.

5 Discussions

Current Challenges. Benchmarking LLMs is essential for evaluating model performance, but traditional static benchmarks risk data contamination. Dynamic benchmarks address this by updating or regenerating test data, aiming to maintain integrity. However, current dynamic methods often lack standardized evaluation criteria, suffer from limited scalability, and offer little interpretability. Many also fail to systematically assess trade-offs like computational overhead and robustness.

Future Directions. Future work should establish standardized evaluation frameworks with criteria such as correctness, diversity, and scalability. Contamination-resilient benchmarks—using temporal filtering, synthetic data, or rule-based generation—can further improve reliability. Dynamic benchmarks should also support continual updates, cross-model applicability, and human-in-the-loop validation. Public update logs and improved interpretability will enhance transparency and trust in LLM evaluation.

6 Conclusion

This survey reviews the literature on data contamination in LLM benchmarking, analyzing both static and dynamic approaches. We find that static methods, though consistent, become more vulnerable to contamination as training datasets grow. While dynamic approaches show promise, they face challenges in reliability and reproducibility. Future research should focus on standardized dynamic evaluation, and practical mitigation tools.

Limitations

While this survey provides a comprehensive overview of static and dynamic benchmarking methods for LLMs, there are several limitations to consider. First, due to the rapidly evolving nature of LLM development and benchmarking techniques, some recent methods or tools may not have been fully covered. As benchmarking practices are still emerging, the methods discussed may not yet account for all potential challenges or innovations in the field. Additionally, our proposed criteria for dynamic benchmarking are a first step and may need further refinement and validation in real-world applications. Lastly, this survey focuses primarily on high-level concepts and may not delve into all the fine-grained technical details of specific methods, which may limit its applicability to practitioners seeking in-depth implementation guidelines.

Ethical Considerations

Our work is rooted in the goal of enhancing the transparency and fairness of LLM evaluations, which can help mitigate the risks of bias and contamination in AI systems. However, ethical concerns arise when considering the use of both static and dynamic benchmarks. Static benchmarks, if not carefully constructed, can inadvertently perpetuate biases, especially if they rely on outdated or biased data sources. Dynamic benchmarks, while offering a more adaptive approach, raise privacy and security concerns regarding the continual collection and updating of data. Moreover, transparency and the potential for misuse of benchmarking results, such as artificially inflating model performance or selecting biased evaluation criteria, must be carefully managed. It is essential that benchmarking frameworks are designed with fairness, accountability, and privacy in mind, ensuring they do not inadvertently harm or disadvantage certain user groups or research domains. Lastly, we encourage further exploration of ethical guidelines surrounding data usage, model transparency, and the broader societal impact of AI benchmarks.

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*.

Aider. 2025. Aider. https://aider.chat. Accessed: 2025-02-06.

Rohan Anil, Andrew M Dai, Orhan Firat, Melvin Johnson, Dmitry Lepikhin, Alexandre Passos, Siamak Shakeri, Emanuel Taropa, Paige Bailey, Zhifeng Chen, and 1 others. 2023. Palm 2 technical report. *arXiv preprint arXiv:2305.10403*.

Jacob Austin, Augustus Odena, Maxwell Nye, Maarten Bosma, Henryk Michalewski, David Dohan, Ellen Jiang, Carrie Cai, Michael Terry, Quoc Le, and 1 others. 2021. Program synthesis with large language models. *arXiv preprint arXiv:2108.07732*.

John Babic. 2025. Meta's LLaMA 4: Advancing AI Amid Benchmark Controversies. Accessed: 2025-05-19.

Simone Balloccu, Patrícia Schmidtová, Mateusz Lango, and Ondrej Dusek. 2024a. 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, St. Julian's, Malta. Association for Computational Linguistics.

Simone Balloccu, Patrícia Schmidtová, Mateusz Lango, and Ondřej Dušek. 2024b. Leak, cheat, repeat: Data contamination and evaluation malpractices in closed-source llms. *arXiv preprint arXiv:2402.03927*.

Farima Fatahi Bayat, Lechen Zhang, Sheza Munir, and Lu Wang. 2024. Factbench: A dynamic benchmark for in-the-wild language model factuality evaluation. *arXiv preprint arXiv:2410.22257*.

Yonatan Bisk, Rowan Zellers, Jianfeng Gao, Yejin Choi, and 1 others. 2020. Piqa: Reasoning about physical commonsense in natural language. In *Proceedings of the AAAI conference on artificial intelligence*, volume 34, pages 7432–7439.

Benjamin S Bloom, Max D Engelhart, EJ Furst, Walker H Hill, and David R Krathwohl. 1956. Handbook i: cognitive domain. *New York: David McKay*, pages 483–498.

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. *Advances in neural information processing systems*, 33:1877–1901.

Natasha Butt, Varun Chandrasekaran, Neel Joshi, Besmira Nushi, and Vidhisha Balachandran. 2024. Benchagents: Automated benchmark creation with agent interaction. *arXiv preprint arXiv:2410.22584*.

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. In *Findings of the Association for Computational Linguistics:*

ACL 2024, pages 5300–5318, Bangkok, Thailand. Association for Computational Linguistics.

Nishanth Chandran, Sunayana Sitaram, Divya Gupta, Rahul Sharma, Kashish Mittal, and Manohar Swaminathan. 2024. Private benchmarking to prevent contamination and improve comparative evaluation of llms. *arXiv preprint arXiv:2403.00393*.

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.

Mark Chen, Jerry Tworek, Heewoo Jun, Qiming Yuan, Henrique Ponde De Oliveira Pinto, Jared Kaplan, Harri Edwards, Yuri Burda, Nicholas Joseph, Greg Brockman, and 1 others. 2021. Evaluating large language models trained on code. *arXiv preprint arXiv:2107.03374*.

Simin Chen, Xiaoning Feng, Xiaohong Han, Cong Liu, and Wei Yang. 2024. Ppm: Automated generation of diverse programming problems for benchmarking code generation models. *Proceedings of the ACM on Software Engineering*, 1(FSE):1194–1215.

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

Eunsol Choi, He He, Mohit Iyyer, Mark Yatskar, Wentau Yih, Yejin Choi, Percy Liang, and Luke Zettlemoyer. 2018. Quac: Question answering in context. *arXiv preprint arXiv:1808.07036*.

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.

Christopher Clark, Kenton Lee, Ming-Wei Chang, Tom Kwiatkowski, Michael Collins, and Kristina Toutanova. 2019. Boolq: Exploring the surprising difficulty of natural yes/no questions. *arXiv preprint arXiv:1905.10044*.

Peter Clark, Isaac Cowhey, Oren Etzioni, Tushar Khot, Ashish Sabharwal, Carissa Schoenick, and Oyvind Tafjord. 2018. Think you have solved question answering? try arc, the ai2 reasoning challenge. *arXiv preprint arXiv:1803.05457*.

Karl Cobbe, Vineet Kosaraju, Mohammad Bavarian, Mark Chen, Heewoo Jun, Lukasz Kaiser, Matthias Plappert, Jerry Tworek, Jacob Hilton, Reiichiro Nakano, and 1 others. 2021. Training verifiers to solve math word problems. *arXiv preprint arXiv:2110.14168*.

Codeforces. 2025. Codeforces: Competitive programming platform. `https://codeforces.com`. Accessed: 2025-02-06.

Kevian Darioush, Syed Usman, Guo Xingang, Havens Aaron, Dullerud Geir, Seiler Peter, Qin Lianhui, and Hu Bin. 2024. Capabilities of large language models in control engineering: A benchmark study on gpt-4, claude 3 opus, and gemini 1.0 ultra. *arXiv preprint arXiv:2404.03647*.

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

Chunyuan Deng, Yilun Zhao, Yuzhao Heng, Yitong Li, Jiannan Cao, Xiangru Tang, and Arman Cohan. 2024a. 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, Yuzhao Heng, Yitong Li, Jiannan Cao, Xiangru Tang, and Arman Cohan. 2024b. Unveiling the spectrum of data contamination in language models: A survey from detection to remediation. *arXiv preprint arXiv:2406.14644*.

Chunyuan Deng, Yilun Zhao, Xiangru Tang, Mark Gerstein, and Arman Cohan. 2024c. Investigating data contamination in modern benchmarks for large language models. 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 8698–8711.

Ning Ding, Yulin Chen, Bokai Xu, Yujia Qin, Zhi Zheng, Shengding Hu, Zhiyuan Liu, Maosong Sun, and Bowen Zhou. 2023. Enhancing chat language models by scaling high-quality instructional conversations. *arXiv preprint arXiv:2305.14233*.

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. *arXiv preprint arXiv:2402.15938*.

André Vicente Duarte, Xuandong Zhao, Arlindo L Oliveira, and Lei Li. 2024. De-cop: Detecting copyrighted content in language models training data. In *Forty-first International Conference on Machine Learning*.

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

Lizhou Fan, Wenyue Hua, Lingyao Li, Haoyang Ling, and Yongfeng Zhang. 2024. NPHardEval: Dynamic benchmark on reasoning ability of large language models via complexity classes. In *Proceedings of the 62nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 4092–4114, Bangkok, Thailand. Association for Computational Linguistics.

Jia Feng, Jiachen Liu, Cuiyun Gao, Chun Yong Chong, Chaozheng Wang, Shan Gao, and Xin Xia. 2024. Complexcodeeval: A benchmark for evaluating large code models on more complex code. In *Proceedings of the 39th IEEE/ACM International Conference on Automated Software Engineering*, ASE '24, page 1895–1906, New York, NY, USA. Association for Computing Machinery.

Samuel Gehman, Suchin Gururangan, Maarten Sap, Yejin Choi, and Noah A. Smith. 2020. RealToxicityPrompts: Evaluating neural toxic degeneration in language models. In *Findings of the Association for Computational Linguistics: EMNLP 2020*, pages 3356–3369, Online. Association for Computational Linguistics.

Aryo Pradipta Gema, Joshua Ong Jun Leang, Giwon Hong, Alessio Devoto, Alberto Carlo Maria Mancino, Rohit Saxena, Xuanli He, Yu Zhao, Xiaotang Du, Mohammad Reza Ghasemi Madani, and 1 others. 2024. Are we done with mmlu? *arXiv preprint arXiv:2406.04127*.

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

Shahriar Golchin and Mihai Surdeanu. 2024. Time travel in LLMs: Tracing data contamination in large language models. In *The Twelfth International Conference on Learning Representations*.

Suriya Gunasekar, Yi Zhang, Jyoti Aneja, Caio César Teodoro Mendes, Allie Del Giorno, Sivakanth Gopi, Mojan Javaheripi, Piero Kauffmann, Gustavo de Rosa, Olli Saarikivi, and 1 others. 2023. Textbooks are all you need. *arXiv preprint arXiv:2306.11644*.

Taicheng Guo, Xiuying Chen, Yaqi Wang, Ruidi Chang, Shichao Pei, Nitesh V Chawla, Olaf Wiest, and Xiangliang Zhang. 2024. Large language model based multi-agents: A survey of progress and challenges. *arXiv preprint arXiv:2402.01680*.

Thomas Hartvigsen, Saadia Gabriel, Hamid Palangi, Maarten Sap, Dipankar Ray, and Ece Kamar. 2022. Toxigen: A large-scale machine-generated dataset for adversarial and implicit hate speech detection. *arXiv preprint arXiv:2203.09509*.

Yancheng He, Shilong Li, Jiaheng Liu, Yingshui Tan, Weixun Wang, Hui Huang, Xingyuan Bu, Hangyu Guo, Chengwei Hu, Boren Zheng, and 1 others. 2024. Chinese simpleqa: A chinese factuality evaluation for large language models. *arXiv preprint arXiv:2411.07140*.

Dan Hendrycks, Collin Burns, Steven Basart, Andy Zou, Mantas Mazeika, Dawn Song, and Jacob Steinhardt. 2020. Measuring massive multitask language understanding. *arXiv preprint arXiv:2009.03300*.

Dan Hendrycks, Collin Burns, Saurav Kadavath, Akul Arora, Steven Basart, Eric Tang, Dawn Song, and Jacob Steinhardt. 2021. Measuring mathematical problem solving with the math dataset. *arXiv preprint arXiv:2103.03874*.

Yebowen Hu, Kaiqiang Song, Sangwoo Cho, Xiaoyang Wang, Wenlin Yao, Hassan Foroosh, Dong Yu, and Fei Liu. 2024. When reasoning meets information aggregation: A case study with sports narratives. In *Proceedings of the 2024 Conference on Empirical Methods in Natural Language Processing*, pages 4293–4308, Miami, Florida, USA. Association for Computational Linguistics.

Yuzhen Huang, Yuzhuo Bai, Zhihao Zhu, Junlei Zhang, Jinghan Zhang, Tangjun Su, Junteng Liu, Chuancheng Lv, Yikai Zhang, Yao Fu, and 1 others. 2024. C-eval: A multi-level multi-discipline chinese evaluation suite for foundation models. *Advances in Neural Information Processing Systems*, 36.

Alon Jacovi, Avi Caciularu, Omer Goldman, and Yoav Goldberg. 2023. Stop uploading test data in plain text: Practical strategies for mitigating data contamination by evaluation benchmarks. In *Proceedings of the 2023 Conference on Empirical Methods in Natural Language Processing*, pages 5075–5084, Singapore. Association for Computational Linguistics.

Naman Jain, King Han, Alex Gu, Wen-Ding Li, Fanjia Yan, Tianjun Zhang, and 1 others. 2024. Livecodebench: Holistic and contamination free evaluation of large language models for code. *Preprint*, arXiv:2403.07974.

Carlos E Jimenez, John Yang, Alexander Wettig, Shunyu Yao, Kexin Pei, Ofir Press, and Karthik R Narasimhan. 2024. SWE-bench: Can language models resolve real-world github issues? In *The Twelfth International Conference on Learning Representations*.

Mandar Joshi, Eunsol Choi, Daniel S Weld, and Luke Zettlemoyer. 2017. Triviaqa: A large scale distantly supervised challenge dataset for reading comprehension. *arXiv preprint arXiv:1705.03551*.

Ezra Karger, Houtan Bastani, Chen Yueh-Han, Zachary Jacobs, Danny Halawi, Fred Zhang, and Philip E Tetlock. 2024. Forecastbench: A dynamic benchmark of ai forecasting capabilities. *arXiv preprint arXiv:2409.19839*.

Douwe Kiela, Max Bartolo, Yixin Nie, Divyansh Kaushik, Atticus Geiger, Zhengxuan Wu, Bertie Vidgen, Grusha Prasad, Amanpreet Singh, Pratik Ringshia, and 1 others. 2021. Dynabench: Rethinking benchmarking in nlp. *arXiv preprint arXiv:2104.14337*.

Eunsu Kim, Juyoung Suk, Seungone Kim, Niklas Muennighoff, Dongkwan Kim, and Alice Oh. 2024. Llm-as-an-interviewer: Beyond static testing through dynamic llm evaluation. *arXiv preprint arXiv:2412.10424*.

Seungone Kim, Se Joo, Doyoung Kim, Joel Jang, Seonghyeon Ye, Jamin Shin, and Minjoon Seo. 2023. The CoT collection: Improving zero-shot and few-shot learning of language models via chain-of-thought fine-tuning. In *Proceedings of the 2023 Conference on Empirical Methods in Natural Language Processing*, pages 12685–12708, Singapore. Association for Computational Linguistics.

Satyapriya Krishna, Kalpesh Krishna, Anhad Mohananey, Steven Schwarcz, Adam Stambler, Shyam Upadhyay, and Manaal Faruqui. 2024. Fact, fetch, and reason: A unified evaluation of retrieval-augmented generation. *arXiv preprint arXiv:2409.12941*.

Eldar Kurtic, Amir Moeini, and Dan Alistarh. 2024. Mathador-LM: A dynamic benchmark for mathematical reasoning on large language models. In *Proceedings of the 2024 Conference on Empirical Methods in Natural Language Processing*, pages 17020–17027, Miami, Florida, USA. Association for Computational Linguistics.

Tom Kwiatkowski, Jennimaria Palomaki, Olivia Redfield, Michael Collins, Ankur Parikh, Chris Alberti, Danielle Epstein, Illia Polosukhin, Jacob Devlin, Kenton Lee, and 1 others. 2019. Natural questions: a benchmark for question answering research. *Transactions of the Association for Computational Linguistics*, 7:453–466.

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, Mexico City, Mexico. Association for Computational Linguistics.

Haonan Li, Yixuan Zhang, Fajri Koto, Yifei Yang, Hai Zhao, Yeyun Gong, Nan Duan, and Timothy Baldwin. 2023a. Cmmlu: Measuring massive multitask language understanding in chinese. *arXiv preprint arXiv:2306.09212*.

Jiatong Li, Rui Li, and Qi Liu. 2023b. Beyond static datasets: A deep interaction approach to llm evaluation. *arXiv preprint arXiv:2309.04369*.

Tianle Li, Wei-Lin Chiang, Evan Frick, Lisa Dunlap, Tianhao Wu, Banghua Zhu, Joseph E Gonzalez, and Ion Stoica. 2024a. From crowdsourced data to high-quality benchmarks: Arena-hard and benchbuilder pipeline. *arXiv preprint arXiv:2406.11939*.

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

Xuechen Li, Tianyi Zhang, Yann Dubois, Rohan Taori, Ishaan Gulrajani, Carlos Guestrin, Percy Liang, and Tatsunori B. Hashimoto. 2023c. Alpacaeval: An automatic evaluator of instruction-following models. https://github.com/tatsu-lab/alpaca_eval.

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

Yucheng Li, Frank Geurin, and Chenghua Lin. 2023d. 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. 2024d. An open-source data contamination report for large language models. In *Findings of the Association for Computational Linguistics: EMNLP 2024*, pages 528–541.

Aixin Liu, Bei Feng, Bing Xue, Bingxuan Wang, Bochao Wu, Chengda Lu, Chenggang Zhao, Chengqi Deng, Chenyu Zhang, Chong Ruan, and 1 others. 2024. Deepseek-v3 technical report. *arXiv preprint arXiv:2412.19437*.

Inbal Magar and Roy Schwartz. 2022. Data contamination: From memorization to exploitation. In *Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers)*, pages 157–165.

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. *arXiv preprint arXiv:2501.14275*.

Todor Mihaylov, Peter Clark, Tushar Khot, and Ashish Sabharwal. 2018. Can a suit of armor conduct electricity? a new dataset for open book question answering. *arXiv preprint arXiv:1809.02789*.

Seyed 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. In *The Thirteenth International Conference on Learning Representations*.

Subhabrata Mukherjee, Arindam Mitra, Ganesh Jawahar, Sahaj Agarwal, Hamid Palangi, and Ahmed Awadallah. 2023. Orca: Progressive learning from complex explanation traces of gpt-4. *arXiv preprint arXiv:2306.02707*.

Mathematical Association of America. 2024. American invitational mathematics examination – aime. American Invitational Mathematics Examination – AIME 2024, February 2024.

Guilherme Penedo, Hynek Kydlíček, Loubna Ben allal, Anton Lozhkov, Margaret Mitchell, Colin Raffel, Leandro Von Werra, and Thomas Wolf. 2024. The fineweb datasets: Decanting the web for the finest text data at scale. In *The Thirty-eight Conference on Neural Information Processing Systems Datasets and Benchmarks Track*.

Sébastien Puma, Emmanuel Sander, Matthieu Saumard, Isabelle Barbet, and Aurélien Latouche. 2023. Reconsidering conceptual knowledge: Heterogeneity of its components. *Journal of Experimental Child Psychology*, 227:105587.

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*.

Yiwei Qin, Kaiqiang Song, Yebowen Hu, Wenlin Yao, Sangwoo Cho, Xiaoyang Wang, Xuansheng Wu, Fei Liu, Pengfei Liu, and Dong Yu. 2024. Infobench: Evaluating instruction following ability in large language models. *arXiv preprint arXiv:2401.03601*.

Alec Radford, Jeffrey Wu, Rewon Child, David Luan, Dario Amodei, Ilya Sutskever, and 1 others. 2019. Language models are unsupervised multitask learners. *OpenAI blog*, 1(8):9.

Pranav Rajpurkar, Robin Jia, and Percy Liang. 2018. Know what you don't know: Unanswerable questions for squad. *arXiv preprint arXiv:1806.03822*.

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. *arXiv preprint arXiv:2402.08100*.

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

David Rein, Betty Li Hou, Asa Cooper Stickland, Jackson Petty, Richard Yuanzhe Pang, Julien Dirani, Julian Michael, and Samuel R. Bowman. 2023. Gpqa: A graduate-level google-proof q&a benchmark. *Preprint*, arXiv:2311.12022.

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, Iker García Ferrero, Eneko Agirre, Jon Ander Campos, Alon Jacovi, Yanai Elazar, and Yoav Goldberg, editors. 2024. *Proceedings of the 1st Workshop on Data Contamination (CONDA)*. Association for Computational Linguistics, Bangkok, Thailand.

Keisuke Sakaguchi, Ronan Le Bras, Chandra Bhagavatula, and Yejin Choi. 2021. Winogrande: An adversarial winograd schema challenge at scale. *Communications of the ACM*, 64(9):99–106.

Maarten Sap, Hannah Rashkin, Derek Chen, Ronan LeBras, and Yejin Choi. 2019. Socialiqa: Commonsense reasoning about social interactions. *arXiv preprint arXiv:1904.09728*.

Rylan Schaeffer. 2023. Pretraining on the test set is all you need. *arXiv preprint arXiv:2309.08632*.

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. In *The Twelfth International Conference on Learning Representations*.

Chinese Mathematical Society. 2024. Chinese national high school mathematics olympiad (cnmo 2024). https://www.cms.org.cn/Home/comp/comp/cid/12.html. Accessed: 2025-02-06.

Liangtai Sun, Yang Han, Zihan Zhao, Da Ma, Zhennan Shen, Baocai Chen, Lu Chen, and Kai Yu. 2024. Scieval: A multi-level large language model evaluation benchmark for scientific research. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 38, pages 19053–19061.

Mirac Suzgun, Nathan Scales, Nathanael Schärli, Sebastian Gehrmann, Yi Tay, Hyung Won Chung, Aakanksha Chowdhery, Quoc V Le, Ed H Chi, Denny Zhou, and 1 others. 2022. Challenging big-bench tasks and whether chain-of-thought can solve them. *arXiv preprint arXiv:2210.09261*.

Alon Talmor, Jonathan Herzig, Nicholas Lourie, and Jonathan Berant. 2018. Commonsenseqa: A question answering challenge targeting commonsense knowledge. *arXiv preprint arXiv:1811.00937*.

Gemini Team, Petko Georgiev, Ving Ian Lei, Ryan Burnell, Libin Bai, Anmol Gulati, Garrett Tanzer, Damien Vincent, Zhufeng Pan, Shibo Wang, and 1 others. 2024. Gemini 1.5: Unlocking multimodal understanding across millions of tokens of context. *arXiv preprint arXiv:2403.05530*.

Teknium. 2023. Openhermes 2.5: An open dataset of synthetic data for generalist llm assistants.

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

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

Pablo Villalobos, Jaime Sevilla, Lennart Heim, Tamay Besiroglu, Marius Hobbhahn, and Anson Ho. 2022. Will we run out of data? an analysis of the limits of scaling datasets in machine learning. *arXiv preprint arXiv:2211.04325*, 1.

Zhongwei Wan, Xin Wang, Che Liu, Samiul Alam, Yu Zheng, Jiachen Liu, Zhongnan Qu, Shen Yan, Yi Zhu, Quanlu Zhang, and 1 others. 2023. Efficient large language models: A survey. *arXiv preprint arXiv:2312.03863*.

Alex Wang. 2018. Glue: A multi-task benchmark and analysis platform for natural language understanding. *arXiv preprint arXiv:1804.07461*.

Alex Wang, Yada Pruksachatkun, Nikita Nangia, Amanpreet Singh, Julian Michael, Felix Hill, Omer Levy, and Samuel Bowman. 2019. Superglue: A stickier benchmark for general-purpose language understanding systems. *Advances in neural information processing systems*, 32.

Guan Wang, Sijie Cheng, Xianyuan Zhan, Xiangang Li, Sen Song, and Yang Liu. 2023. Openchat: Advancing open-source language models with mixed-quality data. *arXiv preprint arXiv:2309.11235*.

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

Yubo Wang, Xueguang Ma, Ge Zhang, Yuansheng Ni, Abhranil Chandra, Shiguang Guo, Weiming Ren, Aaran Arulraj, Xuan He, Ziyan Jiang, and 1 others. 2024b. Mmlu-pro: A more robust and challenging multi-task language understanding benchmark. *arXiv preprint arXiv:2406.01574*.

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*.

Xiaobao Wu, Liangming Pan, Yuxi Xie, Ruiwen Zhou, Shuai Zhao, Yubo Ma, Mingzhe Du, Rui Mao, Anh Tuan Luu, and William Yang Wang. 2024. Antileak-bench: Preventing data contamination by automatically constructing benchmarks with updated real-world knowledge. *arXiv preprint arXiv:2412.13670*.

Chulin Xie, Yangsibo Huang, Chiyuan Zhang, Da Yu, Xinyun Chen, Bill Yuchen Lin, Bo Li, Badih Ghazi, and Ravi Kumar. 2024. On memorization of large language models in logical reasoning. *arXiv preprint arXiv:2410.23123*.

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*.

Liang Xu, Hai Hu, Xuanwei Zhang, Lu Li, Chenjie Cao, Yudong Li, Yechen Xu, Kai Sun, Dian Yu, Cong Yu, and 1 others. 2020. Clue: A chinese language understanding evaluation benchmark. *arXiv preprint arXiv:2004.05986*.

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

Xin Xu, Jiaxin ZHANG, Tianhao Chen, Zitong Chao, Jishan Hu, and Can Yang. 2025. UGMathbench: A diverse and dynamic benchmark for undergraduate-level mathematical reasoning with large language models. In *The Thirteenth International Conference on Learning Representations*.

An Yang, Baosong Yang, Beichen Zhang, Binyuan Hui, Bo Zheng, Bowen Yu, Chengyuan Li, Dayiheng Liu, Fei Huang, Haoran Wei, and 1 others. 2024. Qwen2.5 technical report. *arXiv preprint arXiv:2412.15115*.

John Yang, Carlos E. Jimenez, Alex L. Zhang, Kilian Lieret, Joyce Yang, Xindi Wu, Ori Press, Niklas Muennighoff, Gabriel Synnaeve, Karthik R. Narasimhan, Diyi Yang, Sida I. Wang, and Ofir Press. 2025. SWE-bench multimodal: Do ai systems generalize to visual software domains? In *The Thirteenth International Conference on Learning Representations*.

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*.

Jiahao Ying, Yixin Cao, Yushi Bai, Qianru Sun, Bo Wang, Wei Tang, Zhaojun Ding, Yizhe Yang, Xuanjing Huang, and Shuicheng YAN. 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, Bangkok, Thailand. Association for Computational Linguistics.

Rowan Zellers, Ari Holtzman, Yonatan Bisk, Ali Farhadi, and Yejin Choi. 2019. Hellaswag: Can a machine really finish your sentence? *arXiv preprint arXiv:1905.07830*.

Haozhen Zhang, Tao Feng, Pengrui Han, and Jiaxuan You. 2024a. Academiceval: Live long-context llm benchmark. In *ICLR 2025 (Withdrawn Submission)*. OpenReview record; CC BY 4.0.

Zhehao Zhang, Jiaao Chen, and Diyi Yang. 2024b. DARG: Dynamic evaluation of large language models via adaptive reasoning graph. In *The Thirty-eighth Annual Conference on Neural Information Processing Systems*.

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*.

Wanjun Zhong, Ruixiang Cui, Yiduo Guo, Yaobo Liang, Shuai Lu, Yanlin Wang, Amin Saied, Weizhu Chen, and Nan Duan. 2023. Agieval: A human-centric benchmark for evaluating foundation models. *arXiv preprint arXiv:2304.06364*.

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

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? *arXiv 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. In *The Twelfth International Conference on Learning Representations*.

Kaijie Zhu, Jindong Wang, Qinlin Zhao, Ruochen Xu, and Xing Xie. 2024b. Dynamic evaluation of large language models by meta probing agents. In *Proceedings of the 41st International Conference on Machine Learning*, ICML'24. JMLR.org.

Qin Zhu, Qinyuan Cheng, Runyu Peng, Xiaonan Li, Ru Peng, Tengxiao Liu, Xipeng Qiu, and Xuanjing Huang. 2024c. Inference-time decontamination: Reusing leaked benchmarks for large language model evaluation. In *Findings of the Association for Computational Linguistics: EMNLP 2024*, pages 9113–9129, Miami, Florida, USA. Association for Computational Linguistics.

Yongshuo Zong, Tingyang Yu, Ruchika Chavhan, Bingchen Zhao, and Timothy Hospedales. 2024. Fool your (vision and) language model with embarrassingly simple permutations. In *The 41st International Conference on Machine Learning*.

A Significance of Data Contamination

To demonstrate the effectiveness of dynamic benchmarks, we conduct a study using HumanEval and DyCodeEval (Chen et al., 2025) on three LLMs: Llama-3.2-1B, Llama-3.2-3B, and DeepSeek-Coder-1.3B. For each model, we simulate data contamination by intentionally leaking a portion of the benchmark dataset during fine-tuning.

For HumanEval: We directly include part of the benchmark dataset in fine-tuning. ForDyCodeEval: We run the benchmark twice on each seed dataset to generate two versions—one for training and one for evaluation. We experiment with contamination levels of 0%, 25%, 50%, 75%, and 100%, producing four distinct contaminated models.

The results show that for overfitted models, as the contamination level increases from 25% to 100%, accuracy on HumanEval also increases. This highlights the limitation of static benchmarks in detecting overfitting. However, on the dynamic DyCodeEval, even when a model is overfitted on one version, it maintains stable accuracy scores across different versions. This demonstrates the advantage of dynamic benchmarks in evaluating models under data contamination.

B Benchmark Applications

Knowledge Knowledge benchmarks evaluate LLM internal knowledge. NaturalQuestions (Kwiatkowski et al., 2019) and TriviaQA (Joshi et al., 2017) focus on retrieving real-world information, while multi-domain tasks are covered by MMLU (Hendrycks et al., 2020), BBH (Suzgun et al., 2022), and AGI Eval (Zhong et al., 2023). Recent extensions like MMLU-Redux (Gema et al., 2024) and MMLU-Pro (Wang et al., 2024b) refine these assessments further. Additionally, ControlBench (Darioush et al., 2024), FRAMES (Krishna et al., 2024), and GPQA Diamond (Rein et al., 2023) target technical and long-context challenges, with open-domain evaluations provided by AlpacaEval (Li et al., 2023c) and ArenaHard (Li et al., 2024a).

Reasoning Understanding and applying everyday knowledge is a key aspect of language comprehension. Benchmarks such as PIQA (Bisk et al., 2020), SIQA (Sap et al., 2019), HellaSwag (Zellers et al., 2019), and WinoGrande (Sakaguchi et al., 2021) are designed to assess a model's intuitive reasoning skills from multiple perspectives. In addition, academic challenge sets like ARC (Clark et al., 2018), OpenBookQA (Mihaylov et al., 2018), and CommonsenseQA (Talmor et al., 2018) push models further by requiring the integration of background knowledge with logical reasoning to arrive at plausible answers. C-SimpleQA (He et al., 2024) evaluates the factuality ability of language models to answer short questions in Chinese.

Safety Safety benchmarks are essential for evaluating the robustness of LLM's ability to generate non-toxic and ethically aligned content. Datasets such as RealToxicityPrompts (Gehman et al., 2020) and ToxiGen (Hartvigsen et al., 2022) assess resilience against producing harmful outputs.

Language Language benchmarks assess the LLMs' proficiency in specific languages. GLUE (Wang, 2018) and SuperGLUE (Wang et al., 2019) cover tasks from sentiment analysis to language inference, while CLUE (Xu et al., 2020) targets Chinese language. Typo-fixing (Suzgun et al., 2022) is also widely used.

Reading Comprehension Reading comprehension tasks test a model's ability to extract and infer information from text. Benchmarks like SQuAD (Rajpurkar et al., 2018), QuAC (Choi et al., 2018), and BoolQ (Clark et al., 2019) challenge models to understand passages and draw logical conclusions.

C Static Benchmark Enhancements

Because LLMs often train on publicly available data, static benchmarks risk being inadvertently included, leading to contamination. To mitigate this, several methods have been proposed to enhance *static* benchmarking.

C.0.1 Canary String

Canary strings are deliberately crafted, unique tokens embedded within a dataset to serve as markers for data contamination. When a model's output unexpectedly includes these tokens, it strongly indicates that the model has memorized portions of its training data rather than learning to generalize. For instance, the BIG-Bench dataset incorporates these strings so that model developers can identify and filter out such instances (Jacovi et al., 2023).

Limitations The effectiveness of canary strings depends on model trainers being aware of and responsive to these markers. If a developer aims

Leakage	HumanEval			DyCodeEval		
	Llama-3.2-1B	Llama-3.2-3B	DeepSeek-Coder-1.3b	Llama-3.2-1B	Llama-3.2-3B	DeepSeek-Coder-1.3b
0%	0.19	0.28	0.41	0.14	0.25	0.41
25%	0.29	0.32	0.47	0.08	0.18	0.13
50%	0.48	0.57	0.5	0.08	0.19	0.16
75%	0.68	0.71	0.59	0.07	0.21	0.14
100%	0.82	0.87	0.62	0.11	0.18	0.07

Table 3: A proof of concept experiment

Task	**Type**	**Benchmark**
Math	Static	GSM8K (Cobbe et al., 2021), MATH (Hendrycks et al., 2021), AIME 2024 (of America, 2024), CNMO 2024) (Society, 2024)
	Dynamic	LiveBench (White et al., 2024), UGMathBench (Xu et al., 2025), Mathador-LM (Kurtic et al., 2024)
Language	Static	GLUE (Wang, 2018), SuperGLUE (Wang et al., 2019), CLUE (Xu et al., 2020)
	Dynamic	LiveBench (White et al., 2024), C^2LEVA (Li et al., 2024c), ITD (Zhu et al., 2024c)
Coding	Static	HumanEval (Chen et al., 2021), MBPP (Austin et al., 2021), SWE-Bench (Jimenez et al., 2024; Yang et al., 2025), Codeforces (Codeforces, 2025), Aider (Aider, 2025)
	Dynamic	LiveBench (White et al., 2024), LiveCodeBench (Jain et al., 2024), ComplexCodeEval (Feng et al., 2024)
Reasoning	Static	PIQA (Bisk et al., 2020), SIQA (Sap et al., 2019), HellaSwag (Zellers et al., 2019), WinoGrande (Sakaguchi et al., 2021), ARC (Clark et al., 2018), OpenBookQA (Mihaylov et al., 2018), CommonsenseQA (Talmor et al., 2018), C-SimpleQA (He et al., 2024)
	Dynamic	LiveBench (White et al., 2024), DyVal (Zhu et al., 2024a), C^2LEVA (Li et al., 2024c), NPHardEval (Fan et al., 2024), S3Eval (Lei et al., 2024), DARG (Zhang et al., 2024b)
Knowledge	Static	NaturalQuestions (Kwiatkowski et al., 2019), TriviaQA (Joshi et al., 2017), CMMLU (Li et al., 2023a), MMLU (Hendrycks et al., 2020), BBH (Suzgun et al., 2022), AGI Eval (Zhong et al., 2023), MMLU-Redux (Gema et al., 2024), MMLU-Pro (Wang et al., 2024b), ControlBench (Darioush et al., 2024), FRAMES (Krishna et al., 2024), GPQA Diamond (Rein et al., 2023), AlpacaEval (Li et al., 2023c), ArenaHard (Li et al., 2024a)
	Dynamic	C^2LEVA (Li et al., 2024c), ITD (Zhu et al., 2024c), Auto-Dataset (Ying et al., 2024), DyVal2 (Zhu et al., 2024b), SciEval (Sun et al., 2024)
Safety	Static	RealToxicityPrompts (Gehman et al., 2020), ToxiGen (Hartvigsen et al., 2022)
	Dynamic	C^2LEVA (Li et al., 2024c), FactBench (Bayat et al., 2024)
Instruction	Static	IFEval (Zhou et al., 2023), InfoBench (Qin et al., 2024), C-Eval (Huang et al., 2024)
	Dynamic	LiveBench (White et al., 2024)
Comprehension	Static	SQuAD (Rajpurkar et al., 2018), QuAC (Choi et al., 2018), BoolQ (Clark et al., 2019)
	Dynamic	LatestEval (Li et al., 2023d), Antileak-bench (Wu et al., 2024)

Table 4: Summary of benchmarking applications.

to leak benchmarking data to boost scores, this method will not work.

C.0.2 Encryption

Encryption methods secure evaluation data by making it inaccessible to unauthorized parties, preventing its accidental inclusion in training sets. Jacovi et al. (2023) propose encrypting test data with a public key and a "No Derivatives" license to block automated crawling and reuse. Yang et al. (2023) show that even advanced decontamination methods can be defeated by minor text variations, emphasizing the need for robust encryption. Similarly, TRUCE (Chandran et al., 2024) leverages confidential computing and secure multi-party computation to enable private benchmarking, ensuring that test data and model parameters remain confidential.

Limitation While these methods effectively protect against data leakage, they depend on strong key management, they introduce extra computational overheads. These methods are vulnerable if encryption is compromised or private key is exposed.

C.0.3 Label Protection

Label protection involves keeping the true answers of a test set hidden from public access so that only an authorized evaluator can use them during model assessment. This approach is common in benchmarks such as GLUE (Wang, 2018), SuperGLUE (Wang et al., 2019), and OpenAI's HumanEval (Chen et al., 2021), etc., where the test labels are withheld to prevent models from learning or memorizing them during training. The key advantage of this method is its ability to maintain evaluation integrity by preventing model exposure to answers, thereby mitigating data contamination risks.

Limitations Label protection limits transparency and independent verification, and it forces researchers to rely on centralized evaluation systems for performance metrics, which can impede detailed error analysis and reproducibility.

C.0.4 Post-hoc Detection

Post-hoc detection mitigates data contamination by identifying overlaps between D_{train} and D_{test}. This is typically done through n-gram matching at various levels, such as tokens (Touvron et al., 2023) or words (Radford et al., 2019; Brown et al., 2020; Chowdhery et al., 2023). However, exact matching often leads to false negatives, prompting the use of more robust techniques like embedding-based similarity (Riddell et al., 2024; Lee et al., 2023; Gunasekar et al., 2023) and improved mapping metrics (Li et al., 2024d; Xu et al., 2024b).

Beyond direct overlap detection, post-hoc methods also analyze model behavior under different conditions, such as memorization through masked inputs (Ranaldi et al., 2024; Chang et al., 2023), partial completions (Anil et al., 2023; Golchin and Surdeanu, 2024), or preference for original over paraphrased test cases (Duarte et al., 2024; Golchin and Surdeanu, 2023; Zong et al., 2024). For instance, Dekoninck et al. (2024) propose CONSTAT, which detects contamination by comparing model performance across benchmarks.

Limitations Post-hot detection methods face several limitations. Full access to the training dataset is often restricted due to legal and privacy constraints, making overlap detection challenging. Additionally, assumptions about model behavior, such as higher memorization or lower perplexity for contaminated instances, may not hold across different models and tasks.

D Dynamic Benchmark Property Labeling Guidance

We label each dynamic benchmark as "supported," "partially supported," or "not supported" for each criterion based on the following guidelines:

Correctness:. Benchmarks with built-in guarantees (e.g., via temporal cutoffs or rule-based generation) are marked "supported." LLM-generated benchmarks are "partially supported" if validated (e.g., by humans or automation), and "not supported" otherwise.

Scalability:. Fully automated benchmarks are "supported." Those combining automation with human effort are "partially supported," while purely manual ones are "not supported."

Collision:. If a benchmark provides theoretical guarantees or formally analyzes collision rates, it is "supported." Empirical analysis without guarantees is "partial support," and absence of discussion results in "not supported."

Complexity Stability:. Benchmarks that define and control complexity are "supported." Those that define but do not control it receive "partial support." Lack of discussion results in "not supported."

Diversity:. Benchmarks that define and enforce diversity are "supported." Those that define but do not control it are "partially supported," and benchmarks that omit it are "not supported."

Interpretability:. Rule-based or human-designed benchmarks are "supported." Those combining rules with LLMs receive "partial support." Benchmarks relying entirely on LLMs without interpretability are "not supported."