Both Text and Images Leaked! A Systematic Analysis of Data Contamination in Multimodal LLM

Anonymous ACL submission

Abstract

The rapid progression of multimodal large language models (MLLMs) has demonstrated superior performance on various multimodal benchmarks. However, the issue of data contamination during training creates challenges in performance evaluation and comparison. While numerous methods exist for detecting models' contamination in large language models (LLMs), they are less effective for MLLMs due to their various modalities and multiple training phases. In this study, we introduce a multimodal data contamination detection framework, MM-Detect, designed for MLLMs. Our experimental results indicate that MM-Detect is quite effective and sensitive in identifying varying degrees of contamination, and can highlight significant performance improvements due to the leakage of multimodal benchmark training sets. Furthermore, we explore whether the contamination originates from the base LLMs used by MLLMs or the multimodal training phase, providing new insights into the stages at which contamination may be introduced.

1 Introduction

The development of MLLMs has exceeded expectations (Liu et al., 2023a; Lin et al., 2023), showcasing extraordinary performance on various multimodal benchmarks (Lu et al., 2022; Liu et al., 2023b; Song et al., 2024), even surpassing human performance. However, due to the partial obscurity associated with MLLMs training (OpenAI, 2023; Reid et al., 2024), it remains challenging to definitively ascertain the impact of training data on model performance, despite some works showing the employment of the training set of certain datasets (Liu et al., 2023a; Chen et al., 2023; Bai et al., 2023b). The issue of data contamination, occurring when training or test data of benchmarks is exposed during the model training phase (Xu et al., 2024), could potentially instigate inequitable performance comparisons among models. This not only creates a dilemma for users in model selection but also poses a significant hurdle to further advancements in this domain.

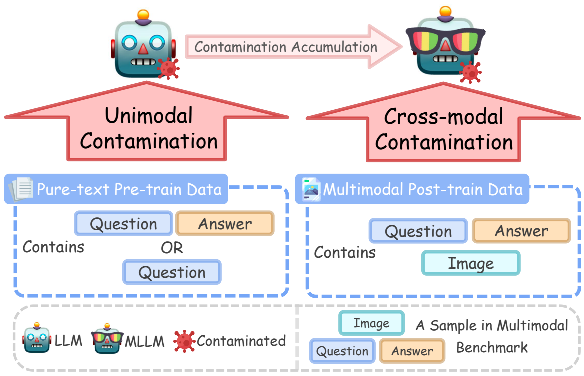

Figure 1: A description of Multimodal Data Contamination, which can originate from various training stages.

While numerous works in the field of LLMs have proposed methods for detecting data contamination (Yeom et al., 2018; Deng et al., 2024; Dong et al., 2024), MLLMs, due to their various modalities and multiple training phases (Liu et al., 2023a; Li et al., 2023), face limitations when applying these methods. Therefore, there is a pressing need for a multimodal contamination detection framework specifically tailored for MLLMs.

In this study, we carry out a systematic analysis of multimodal data contamination. Initially, we define **Multimodal Data Contamination**, as it pertains to the modality of data sources exposed to the MLLMs, into two categories: *Unimodal Contamination* and *Cross-modal Contamination*, as illustrated in Figure 1. Subsequently, we unveil a detection framework, **MM-Detect**, which incorporates two methods, *Option Order Sensitivity Test* and *Slot Guessing for Perturbation Caption*, designed to handle two common types of Visual Question Answering (VQA) tasks: multiple-choice and caption-based questions, respectively.

To corroborate the validity and sensitivity of our approach, we deliberately induce contamination in MLLMs, thus simulating feasible real-world scenarios. Experimental results indicate that **our method proves to be quite effective and sensitive** in identifying varying degrees of contamination. Interestingly, our findings reveal that **not only does leakage in the multimodal benchmark test set play a role, but the training set can also contribute significantly** to enhancing the model's performance. Then, applying MM-Detect on twelve widely-used MLLMs across five prevalent VQA datasets, we find that **both open-source and proprietary MLLMs exhibit contamination with varying degrees.**

To further delve into the stage where contamination is introduced, we employ a heuristic method. This method seeks to distinguish whether the contamination originates from the pre-training phase of LLMs or the multimodal training phase. Our findings suggest that **the contamination observed in some MLLMs may not necessarily stem from the multimodal training phase**. Instead, it could potentially be traced back to the pre-training stage of their respective LLMs.

To the best of our knowledge, our work is the first effort to systematically analyze multimodal data contamination. In conclusion, our research makes several important contributions:

- We formulate the definition for multimodal contamination and present the MM-Detect framework, comprising two innovative methods specifically designed for effective contamination detection in MLLMs.
- We demonstrate that leakage from multimodal benchmark data—whether from the training, validation, or test sets—can significantly enhance models' performance, with this performance gain intensifying as the degree of contamination increases.
- By employing a heuristic method, we pioneer the exploration into the stage at which contamination is introduced, revealing that it may stem not solely from the multimodal data but could also from the LLMs.

2 Preliminaries

We formally define the problem of multimodal data contamination and outline the unique challenges associated with its detection.

2.1 Definition of Multimodal Data Contamination

In contrast to single-modal contamination, multimodal contamination may arise from both unimodal and multimodal data sources, as depicted in Figure 1. The training data for MLLMs generally consists of pure text pre-training data D_{pretrain} and multimodal alignment or instruction-following data D_{vision}. Consider an instance (x, i, y) from a benchmark dataset D, where x represents the text input, i is the image input, and y is the label. Data contamination in MLLMs can be categorized into the following two cases:

- **Unimodal Contamination**: The pair (x, y) or the input x appears in D_{pretrain}.
- **Cross-modal Contamination**: The triplet (x, i, y) appears in D_{vision}.

In both cases, models trained on these data may gain an unfair advantage.

2.2 Challenges in Multimodal Detection

The challenges of multimodal contamination detection mainly arise from two aspects.

Challenge I: Inefficiency of Unimodal Methods. Despite the prevalence of unimodal detection methods, their application in multimodal scenarios often encounters difficulties. For example, **retrieval-based methods** (Brown et al., 2020; Touvron et al., 2023a) attempt to detect contamination by retrieving large-scale corpora used for model training. Yet, they struggle when retrieving multimodal information. Similarly, **logits-based methods** (Shi et al., 2024; Yeom et al., 2018) rely on observing the distribution of low-probability tokens in model outputs, but the disparity in token probability distributions is less pronounced in instruction-tuned MLLMs. **Masking-based methods** (Deng et al., 2024), which assess training contamination by evaluating a model's ability to predict specific missing or masked text, face challenges when images in multimodal samples provide clues, leading to overestimated contamination detection. Finally, **comparison-based methods** (Dong et al., 2024) that measure contamination by comparing model outputs with benchmark data prove to be ineffective for image caption tasks due to low output similarity. To validate these inefficiencies, we have conducted comprehensive experiments with compelling results, which are detailed in Appendix A.

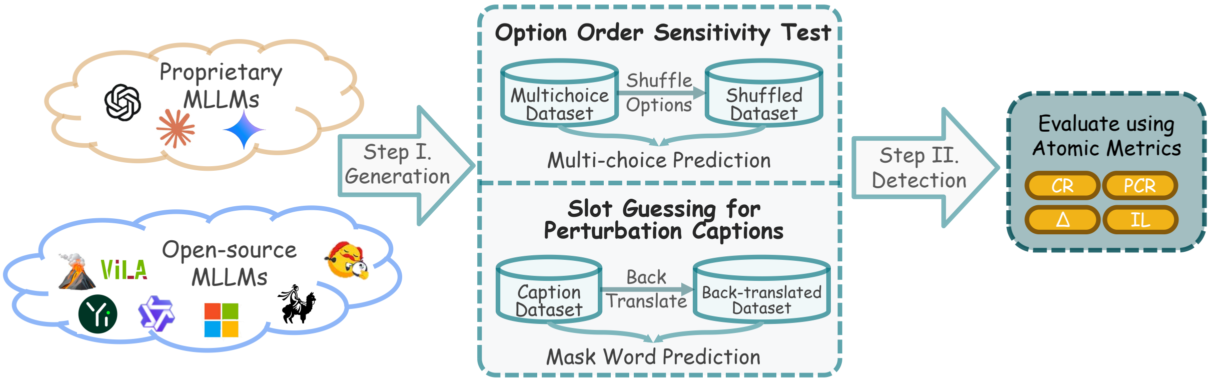

Figure 2: The overview of proposed **MM-Detect** framework.

Challenge II: Multi-stage Training in MLLMs. Another challenge in detecting contamination in MLLMs is the multi-stage nature of their training (Yin et al., 2023). Each stage may be subject to data contamination. 1) Initially, the **pretraining corpus** could contain the textual components of questions from benchmark samples. Moreover, in certain native multimodal model training (Reid et al., 2024), samples may be entirely exposed. 2) Subsequently, during **multimodal fine-tuning**, the model may utilize training samples of some benchmarks, leading to skewed performance improvements. 3) Furthermore, some models employ extensive mixed image-text data from the internet for **modality alignment training** (Lin et al., 2023; Bai et al., 2023b), potentially introducing additional contamination. Given the challenges, the development of an effective detection framework for multimodal contamination becomes an urgent need.

Based on the discussion above, we have designed a detection method specifically tailored for multimodal contamination, with a particular focus on VQA tasks. Additionally, we have developed a heuristic method to trace the introduction of contamination across different training phases.

3 MM-Detect

In this section, we introduce the multimodal contamination detection framework, **MM-Detect**. The core philosophy of MM-Detect is to detect the unusual discrepancies in model performance before and after semantic-irrelevant perturbations. As depicted in Figure 2, this framework operates in two primary steps:

- The first step is to generate perturbed datasets using two innovative methods: the *Option Order Sensitivity Test* (§3.1) and the *Slot Guessing for Perturbation Captions* (§3.2), tailored for multiple-choice and image captioning tasks, respectively.
- The second step involves the application of predefined metrics to detect contamination (§3.3), conducting thorough analyses at both the dataset and instance levels.

3.1 Option Order Sensitivity Test

This method is based on a reasonable and intuitive premise that if the model's performance is highly sensitive to the order of the options, as shown in Figure 3, it indicates potential contamination, leading the model to memorize a certain canonical order of the options.

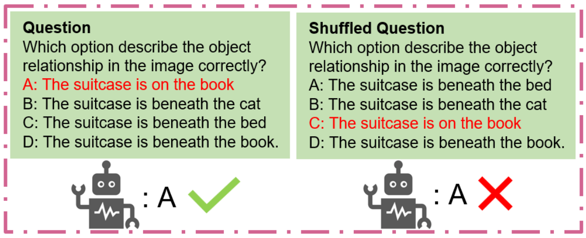

Figure 3: An example of **Option Order Sensitivity Test** applied to a contaminated model.

Method Formulation. Let D be a dataset consisting of n datapoints. Each datapoint d_i ($i \in \{1, \ldots, n\}$) comprises a question Q_i, an associated image I_i, and a set of answer choices $A_i = \{a_i^1, a_i^2, \ldots, a_i^m\}$, where m is the number of choices and the correct answer is denoted by a_i^c.

To introduce positional variation, the set A_i is randomly shuffled to obtain a new set A_i', ensuring that the index of the correct answer a_i^c in A_i' differs from its original position in A_i. The final prompts, before and after shuffling, are constructed by concatenating the image, question and choices:

$$P = \text{Concat}(I_i, Q_i, A_i),$$

$$P' = \text{Concat}(I_i, Q_i, A_i'),$$

where P and P' are the inputs to the model, and Q_i and I_i remain unchanged throughout this process.

3.2 Slot Guessing for Perturbation Caption

This method is based on the intuition that if a model can predict a missing and important part of a sentence but fails with the back-translated version (from English to Chinese, then back to English), it likely indicates that the model has encountered the original sentence during training.

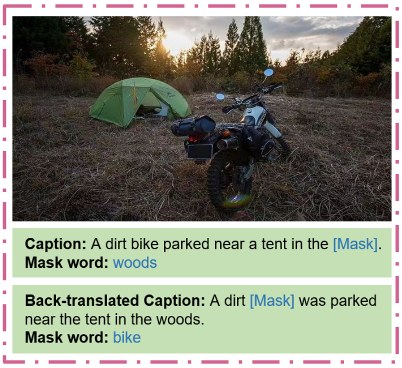

Figure 4: A simple example shows the procedure.

As shown in Figure 4, the keywords identified are "woods" and "bike". Since the image contains "woods", a correct guess by the model may stem from its multimodal capabilities rather than data contamination. However, if the model fails to predict "bike", which is also present in the image, this may indicate potential leakage of this instance.

Method Formulation. Let D be a dataset containing n datapoints. Each datapoint d_i ($i \in \{1, \ldots, n\}$) consists of an image-caption pair, where the caption S_i describes the visual features of the corresponding image I_i. We first apply a back-translation function[1] to S_i:

$$S'_i = f_{\text{back-translate}}(S_i).$$

resulting in a paraphrased version S'_i. Next, we perform keyword extraction[2] on both S_i and S'_i:

$$K_i = f_{\text{keyword}}(S_i), \quad K'_i = f_{\text{keyword}}(S'_i),$$

where K_i and K'_i denote the extracted keywords from S_i and S'_i, respectively. We then apply a masking function f_{mask} to replace the extracted keywords with a placeholder token `[MASK]`:

$$S_{i,\text{mask}} = f_{\text{mask}}(S_i, K_i), \; sS'_{i,\text{mask}} = f_{\text{mask}}(S'_i, K'_i).$$

[1]We use the Google Translate API for Python to implement back-translation.

[2]We employ the Stanford POS Tagger (Toutanvoa and Manning, 2000), targeting nouns, adjectives, and verbs, as they encapsulate the core meaning of the sentences.

The final prompt guiding the model to complete the masked-word prediction can be represented as:

$$P_i = \text{Concat}(I_i, Q_i, S_{i,\text{mask}}),$$
$$P'_i = \text{Concat}(I_i, Q_i, S'_{i,\text{mask}}).$$

3.3 Detection Metrics

Having introduced two detection methods, we now delineate the atomic metrics for the detection pipeline, which consists of two primary steps.

Step 1: Correct Rate Calculation. This step assesses the model's performance on benchmark D before and after perturbation. We denote the correct rate (CR) and perturbed correct rate (PCR) uniformly for both Option Order Sensitivity Test (using Accuracy) and Slot Guessing (using Exact Match). Here, N and N' are the counts of correct answers before and after perturbation, respectively. They are calculated as:

$$CR = \frac{N}{|D|}, \quad PCR = \frac{N'}{|D|}.$$

Step 2: Contamination Degree Analysis. This step quantifies the model's contamination degree based on the performance variation pre- and post-perturbation. Specifically, we introduce two metrics to evaluate contamination at both dataset and instance levels.

Dataset Level Metric. We evaluate the reduction in atomic metrics, denoted as Δ:

$$\Delta = PCR - CR$$

This reduction indicates the model's familiarity or memory of the original benchmark relative to the perturbed set, thereby offering insights into potential contamination at the **dataset level**. A significant negative Δ suggests potential extensive leakage in the benchmark dataset, leading to highly perturbation-sensitive model performance.

Instance Level Metric. Despite a non-significant or positive Δ, contamination may still occur at the instance level, as some instances may still have been unintentionally included during training. To identify such instances, we compute X, the count of cases where the model provided correct answers before perturbation but incorrect answers after. The **instance leakage metric** Φ is then obtained by dividing X by the dataset size:

$$\Phi = \frac{X}{|D|},$$

where a larger Φ indicates a higher likelihood of instance leakage.

Contamination Degree. We define the contamination degree solely based on Δ, capturing how much performance changes from D to D_{pert}. The threshold for each severity level (Minor, Partial, Severe) is derived from experimental findings in §4. The entire pipeline, including how these levels are assigned, is summarized in Algorithm 1.

Algorithm 1 MM-Detect Framework

Require: Benchmark dataset D, Model M
Define contamination degree $\mathcal{C}_{\text{Minor}}$, $\mathcal{C}_{\text{Partial}}$, $\mathcal{C}_{\text{Severe}}$
if D is multiple-choice **then**
  Generate perturbed set D_{pert} via §3.1
else
  Generate perturbed set D_{pert} via §3.2
end if
Compute CR, PCR, Δ, Φ using §3.3
if multiple-choice **then**

$$\mathcal{C} \leftarrow \begin{cases} \mathcal{C}_{\text{Minor}}, & \Delta \in (-1.6, -0.2] \\ \mathcal{C}_{\text{Partial}}, & \Delta \in (-2.9, -1.6] \\ \mathcal{C}_{\text{Severe}}, & \Delta \leq -2.9 \end{cases}$$

else

$$\mathcal{C} \leftarrow \begin{cases} \mathcal{C}_{\text{Minor}}, & \Delta \in (-2.4, -1.1] \\ \mathcal{C}_{\text{Partial}}, & \Delta \in (-5.0, -2.4] \\ \mathcal{C}_{\text{Severe}}, & \Delta \leq -5.0 \end{cases}$$

end if
Ensure: CR, PCR, Δ, Φ, $\mathcal{C}$

4 Intentional Contamination

In this section, we address three research questions through intentional contamination:
RQ1: Is MM-Detect an effective detector?
RQ2: How sensitive is MM-Detect?
RQ3: Does training set leakage cause bias?

To answer these questions, we adopt the LLaVA framework and train several 7B-parameter models using intentionally contaminated downstream task data during the visual instruction tuning phase. We then evaluate the degree of contamination in each model. The contamination data are identical to those described in §5.1.

4.1 MM-Detect is An Effective Detector

We reproduced the LLaVA-1.5-7B experiment to obtain a baseline model without contamination. Recognizing that contamination can occur anywhere in the training data, we inserted contaminated samples into the visual instruction tuning dataset (D_{tuning}) at three positions, early, mid, and late, creating two groups of contaminated training sets using 1340 ScienceQA test samples or 1000 NoCaps validation samples. Corresponding models, termed Early Cont., Mid Cont., and Late Cont., were then trained for comparison with the baseline.

Models	ScienceQA Test Set			NoCaps Val. Set		
	CR	PCR	Δ	CR	PCR	Δ
Baseline	61.4	61.5	0.01	33.0	32.1	-0.9
Early Cont.	71.5	68.1	**-3.4**	37.5	32.0	**-5.5**
Mid Cont.	69.4	67.3	**-2.1**	38.5	35.1	**-3.4**
Late Cont.	70.2	66.9	**-3.3**	38.7	32.6	**-6.1**

Table 1: Detection results on contamination using the ScienceQA test set and NoCaps validation set.

Table 1 shows that incorporating contaminated data during training increases both the model's performance and its sensitivity to perturbations. Compared with the baseline, ScienceQA-contaminated models exhibit average increases in CR and PCR of 9.0% and 5.9%, respectively, while NoCaps-contaminated models show increases of 5.2% and 1.1%. Moreover, all contaminated models demonstrate a marked decrease in Δ, confirming that MM-Detect effectively identifies data contamination. We define the average Δ across these models as the threshold for severe contamination ($\mathcal{C}_{\text{Severe}}$); specifically, $\Delta \leq -2.9$ indicates severe leakage in the multiple-choice dataset, and $\Delta \leq -5$ indicates severe leakage in the caption dataset.

4.2 MM-Detect is Sensitive and Fine-grained

We evaluated MM-Detect's sensitivity by varying leakage levels in the training set. Using the fully contaminated model as our baseline, we trained additional models with moderate and minimal contamination, by inserting reduced amounts (10% and 50%) of contaminated data at the late position of the training set, to assess leakage impact.

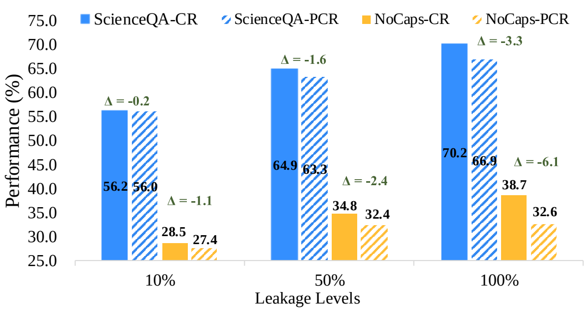

Figure 5: Performance and atomic metrics evaluated under varying leakage levels on the ScienceQA test set and NoCaps validation set.

As illustrated in Figure 5, increasing contamination from 10% to 50% to 100% results in corresponding increases in CR and PCR, alongside progressively larger Δ values. We define the Δ for 50% contamination as the threshold for partial leakage ($\mathcal{C}_{\text{Partial}}$) and that for 10% contamination as the threshold for minor leakage ($\mathcal{C}_{\text{Minor}}$), as detailed in Algorithm 1. These findings confirm that our framework can accurately differentiate between varying leakage levels in benchmark datasets.

4.3 Training Set Leakage Leads to Unfairness

We investigated whether training set leakage biases evaluations by comparing models trained with and without benchmark data contamination. For the ScienceQA experiment, we appended 2000 ScienceQA training samples to the training dataset, creating a contaminated model. For the COCO experiment, we removed the COCO-Caption2017 training data from the original training dataset, resulting in a model without leakage.

Model	Dataset	CR	PCR	Δ
Clean	ScienceQA	61.4	61.5	0.01
Leaked	ScienceQA	64.3	63.8	**-0.5**
Clean	COCO-Caption2017	32.5	31.9	**-0.6**
Leaked	COCO-Caption2017	38.1	34.9	**-3.2**

Table 2: Performance of models trained without (Clean) and with (Leaked) training set contamination.

Table 2 compares the models' performance. On the ScienceQA test set, the contaminated model outperforms the clean model by 2.9% in CR and 2.3% in PCR, with a Δ of -0.5, meeting the minor leakage threshold ($\mathcal{C}_{\text{Minor}}$). On the COCO-Caption2017 validation set, the model with COCO data shows a Δ of -3.2, exceeding the partial leakage threshold ($\mathcal{C}_{\text{Partial}}$). The results indicate that training set leakage inflates performance and that MM-Detect can effectively detect it likewise.

Takeaways

Both training and test set leakage can result in unfairness, and the degree of contamination can be detected through MM-Detect effectively.

5 Experiment

In this section, we demonstrate the practicality of our methodology in verifying the leakage of multi-modal benchmark datasets across several MLLMs.

5.1 Setup

Models. We conducted extensive evaluations on nine open-source MLLMs, including LLaVA-1.5-7B (Liu et al., 2023a), VILA1.5-3B (Lin et al., 2023), Qwen-VL-Chat (Bai et al., 2023b), fuyu-8b[3], idefics2-8b (Laurençon et al., 2024), Phi-3-vision-128k-instruct (Abdin et al., 2024), Yi-VL-6B (AI et al., 2024), InternVL2-8B (Chen et al., 2023, 2024b), DeepSeek-VL2-Tiny (Wu et al., 2024), as well as three proprietary MLLMs: GPT-4o-2024-08-06 (OpenAI, 2023), Gemini-1.5-Pro-002 (Reid et al., 2024), and Claude-3.5-Sonnet-2024-06-20[4].

Benchmark Datasets. Our analysis leverages two multi-choice datasets: ScienceQA (Lu et al., 2022) and MMStar (Chen et al., 2024a), along with three caption datasets: COCO-Caption2017 (Lin et al., 2015), NoCaps (Agrawal et al., 2019), and Vintage[5]. MMStar and Vintage, owing to their recent inception, serve to contrast leakage levels with other datasets. We randomly selected 2000 and 1340 samples from ScienceQA's training and test sets, respectively, with 1000 samples from the other datasets. Given the unavailability of public test labels for COCO-Caption2017 and NoCaps, we used their validation sets.

5.2 Main Results

Multi-choice Datasets. Table 3 yields several conclusions: (1) **Both open-source and proprietary models exhibit contamination.** For example, on the ScienceQA training set, both open-source models like LLaVA-1.5-7B and idefics2-8b and proprietary model Gemini-1.5-Pro show minor contamination degree. (2) **Proprietary models are more contaminated.** Claude-3.5-Sonnet, for instance, registers a severe Δ with higher Φ values on both ScienceQA training and test sets, indicating extensive leakage. (3) **Training set leakage is more pronounced than test set leakage.** On the ScienceQA dataset, models generally exhibit larger Δ values in the training set, for instance, Claude-3.5-Sonnet shows $\Delta = -5.3$ on training versus $\Delta = -2.4$ on the test set, while most models have near-zero Δ on the test set. (4) **Older benchmarks are more prone to leak.** The older ScienceQA test set shows more severe leakage (e.g., Claude-3.5-Sonnet's Δ of -2.4) compared to the newer MMStar validation set (e.g., fuyu-8b's Δ of -1.2).

Caption Datasets. Table 4 yields several conclusions: (1) **Both open-source and proprietary models exhibit contamination on caption datasets.** For example, in the COCO Validation Set, open-source models such as DeepSeek-VL2-Tiny and proprietary models like GPT-4o record

[3] https://www.adept.ai/blog/fuyu-8b

[4] https://www.anthropic.com/news/claude-3-5-sonnet

[5] https://huggingface.co/datasets/SilentAntagonist/vintage-artworks-60k-captioned

Model	ScienceQA Training Set				ScienceQA Test Set				MMStar Validation Set			
Metric	CR	PCR	Δ	Φ	CR	PCR	Δ	Φ	CR	PCR	Δ	Φ
Open-source MLLMs												
LLaVA-1.5-7B	59.7	58.6	-1.1	12.7	60.3	61.6	1.3	10.5	38.9	41.7	2.8	11.0
VILA1.5-3B	57.7	58.3	0.6	14.5	60.3	59.8	-0.5	14.8	38.6	37.6	-1.0	13.9
Qwen-VL-Chat	58.4	60.8	2.5	13.3	60.3	60.4	0.1	13.7	40.9	44.2	3.3	13.2
fuyu-8b	36.5	37.5	1.0	13.4	37.4	36.9	-0.5	**14.9**	28.2	27.0	**-1.2**	**17.7**
idefics2-8b	85.1	84.0	-1.2	3.7	84.0	84.3	0.3	2.8	48.2	49.3	1.1	7.9
Phi-3-vision-128k-instruct	90.5	90.4	-0.1	4.6	88.4	89.1	0.7	3.9	48.7	51.9	3.2	7.2
Yi-VL-6B	60.5	61.8	1.3	10.0	59.5	61.3	1.8	9.6	38.8	44.0	5.2	9.3
InternVL2-8B	94.1	93.9	-0.3	2.0	92.3	93.1	0.8	1.7	56.9	60.1	3.2	5.1
DeepSeek-VL2-Tiny	86.4	86.5	0.1	5.3	87.1	86.9	-0.2	5.3	51.1	52.1	1.0	10.7
Proprietary MLLMs												
GPT-4o	69.9	70.0	0.1	2.7	69.1	69.7	0.6	2.8	48.6	50.5	1.9	9.4
Gemini-1.5-Pro	68.5	67.9	-0.6	6.6	66.5	66.2	-0.3	7.1	45.7	45.5	-0.2	9.9
Claude-3.5-Sonnet	70.3	65.0	**-5.3**	**15.3**	67.3	64.9	**-2.4**	12.4	36.3	36.4	0.1	15.9

Table 3: Comparison of MLLMs on multichoice datasets. Bold values represent the most significant Δ or Φ; color codes denote contamination degree: green for $\mathcal{C}_{\text{Minor}}$, yellow for $\mathcal{C}_{\text{Partial}}$, and red for $\mathcal{C}_{\text{Severe}}$.

Model	COCO Validation Set				NoCaps Validation Set				Vintage Training Set			
Metric	CR	PCR	Δ	Φ	CR	PCR	Δ	Φ	CR	PCR	Δ	Φ
Open-source MLLMs												
LLaVA-1.5-7B	34.6	34.0	-0.6	19.0	30.9	28.5	-2.4	17.9	10.8	10.1	-0.7	9.0
VILA1.5-3B	19.1	20.5	1.4	13.0	19.1	20.5	1.4	13.0	1.5	2.2	0.7	1.5
Qwen-VL-Chat	32.2	30.3	-1.9	19.2	28.7	27.3	-1.4	16.7	15.1	15.4	0.3	12.4
fuyu-8b	9.6	10.6	1.0	7.8	10.0	9.8	-0.2	8.3	2.4	3.3	0.9	2.3
idefics2-8b	43.5	42.3	-1.2	21.2	42.6	37.5	**-5.1**	**23.3**	18.5	17.0	-1.5	14.5
Phi-3-vision-128k-instruct	38.8	39.3	0.5	19.4	36.9	33.3	-3.6	19.7	17.4	11.7	**-5.7**	14.3
Yi-VL-6B	43.9	43.3	-0.6	19.4	37.2	36.1	-1.1	17.5	3.3	4.2	0.9	2.8
InternVL2-8B	53.3	51.9	-1.4	20.4	48.0	46.2	-1.8	20.9	28.0	28.7	0.7	18.8
DeepSeek-VL2-Tiny	23.8	21.4	-2.4	13.5	19.3	18.1	-1.2	12.2	7.5	6.9	-0.6	6.3
Proprietary MLLMs												
GPT-4o	58.1	54.4	**-3.7**	**23.1**	54.2	55.1	0.9	19.4	36.3	38.4	2.1	20.1
Gemini-1.5-Pro	57.5	55.3	-2.2	21.6	51.2	52.0	0.8	18.7	46.3	41.0	-5.3	**28.3**
Claude-3.5-Sonnet	53.7	51.0	-2.7	21.8	50.8	51.5	0.7	20.0	35.2	33.0	-2.2	21.3

Table 4: Comparison of MLLMs on caption datasets.

a significant contamination degree. (2) **Leakage levels vary significantly by benchmark.** For example, on the NoCaps Validation Set, open-source models exhibit more pronounced contamination degree than proprietary models, whereas the trend reverses on the COCO Validation Set. These findings confirm that caption datasets are vulnerable to leakage, with proprietary models generally exhibiting more pronounced contamination effects.

Takeaways

Multimodal data contamination, at both dataset and instance levels, is prevalent in open-source and proprietary MLLMs across multi-choice and image caption datasets.

6 At Which Stage is Contamination Introduced?

In this section, we investigate the source of contamination in MLLMs. Although the training data for some MLLMs is openly documented, an important question remains: if contamination does not arise during the multimodal training phase, could it stem from the unimodal (pre-training) phase, as defined in §2.1? To address this possibility, we examined the underlying LLMs of the evaluated MLLMs and conducted a series of experiments (§6.1). We also explored the origins of cross-modal contamination arising during visual instruction tuning (§6.1).

6.1 A Heuristic Experiment for Unimodal Contamination Detection

We employed a heuristic approach based on the intuition that *if an LLM can correctly answer an image-required question without the image, it may indicate the leakage of that instance.*

Experiment Setup. We used MMStar as the benchmark, where **every question relies on visual input for correct answers**. The tested models include LLaMA2-7B (Touvron et al., 2023b) (used by LLaVA-1.5 and VILA), Qwen-7B (Bai et al., 2023a) (used by Qwen-VL), Mistral-7B-v0.1 (Jiang et al., 2023) (used by idefics2), Phi-3-small-128k-instruct (Abdin et al., 2024) (used by Phi-3-vision), Yi-6B (AI et al., 2024) (used by Yi-VL), and Internlm2-7B (Cai et al., 2024) (used by InternVL2). To discourage random guessing, we appended the prompt "If you do not know the answer, output I don't know" to the instructions. Accuracy, the frequency with which models correctly answer questions without image input, is reported as the primary metric. Note that we did not evaluate Fuyu-8B and proprietary models since their unimodal LLM and training data remain undisclosed.

Model	Accuracy	Φ_M
LLaMA2-7b (LLaVA-1.5 & VILA)	25.6	11.0
Qwen-7B (Qwen-VL)	13.2	13.2
Internlm2-7B (InternVL2)	11.0	5.1
Mistral-7B-v0.1 (idefics2)	10.7	7.9
Phi-3-small-128k-instruct (Phi-3-vision)	6.1	7.2
Yi-6B (Yi-VL)	3.4	9.3

Table 5: Contamination rates of LLMs used by MLLMs. Φ_M denotes the Φ of the respective MLLMs.

Main Results. Table 5 yields several conclusions: (1) **Contamination occurs in LLM.** All models exhibit varied contamination rates, indicating that their pre-training data likely included text from multimodal benchmarks. (2) **Elevated LLM contamination correlates with increased MLLM leakage.** For instance, VILA1.5-3B and Qwen-VL-Chat exhibit significant Φ values that mirror their underlying LLM contamination levels. These findings suggest that contamination in these MLLMs may originate partly from the LLMs' pre-training phase, rather than solely from multimodal training.

6.2 Tracing Origins: A Review of MLLM's Visual Instruction Tuning Data

To investigate the origins of cross-modal contamination, we scrutinize the visual instruction tuning data of MLLMs. We delve into the construction process of three benchmark datasets: ScienceQA, COCO Caption, and Nocaps, comparing them with the training data and its sources of various open-source MLLMs to analyze the degree of overlap.

As Table 6 illustrates, MLLMs marked in red and yellow typically exhibit a significant contamination degree. Yet, even MLLMs labeled in green aren't exempt from the risk of cross-modal contamination. This is because some models have been trained on large-scale interleaved image-text datasets (e.g., OBELICS (Laurenon et al., 2023)), datasets derived from online sources (e.g., Conceptual Caption (Sharma et al., 2018)), or in-house data. Furthermore, some models haven't fully disclosed their training data, which may lead to overlooked potential leaks in benchmark datasets.

Model	ScienceQA	COCO Caption	Nocaps
Phi-3-Vision	0.7	0.5	-3.6
VILA	-0.5	1.4	1.4
Idefics2	0.3	-1.2	-5.1
LLaVA-1.5	1.3	-0.6	-2.4
Yi-VL	1.8	-0.6	-1.1
DeepSeek-VL2	-0.2	-2.4	-1.2
Qwen-VL-Chat	0.1	-1.9	-1.4
InternVL2	0.8	-1.4	-1.8

Table 6: Depiction of the overlap between the training data of MLLMs and the benchmarks, as well as the contamination degree Δ of MLLMs on benchmarks. Green signifies no overlap, yellow suggests potential overlap, and Red indicates partial or entire overlap.

Takeaways

The contamination in MLLMs may not only stem from cross-modal contamination but also from unimodal contamination, both of which can significantly impact the overall performance.

7 Conclusion and Future Work

In this study, we introduce and validate a multimodal data contamination detection framework, MM-Detect, providing new perspectives for evaluating contamination in MLLMs. We discovered that popular MLLMs exhibit varying degrees of data contamination, which directly impacts their performance and generalization ability. In addition, our experiment indicates that MM-Detect is sensitive to varying degrees of contamination and can highlight significant performance improvements due to the leakage in the multimodal benchmark training set. Furthermore, we found that the contamination in MLLMs may not solely originate from the cross-modal contamination but could also stem from the unimodal contamination.

Future work will focus on two key areas:

- Firstly, standardizing the use of multimodal datasets and reporting potential contamination impacts to minimize contamination, thereby enhancing data consistency and quality.
- Secondly, creating a continuously updated benchmarking system for the ongoing evaluation of multimodal model performance.

This will support advancements and broader applications in this field.

Limitations

We acknowledge several limitations in our work. First, this work is limited to discussions around visual modalities, and does not yet cover other modalities such as audio or video. Second, we only selected widely used and representative multimodal datasets for detection, including multiple-choice datasets and caption datasets, without testing additional datasets, such as open-ended generation and cloze questions. However, we speculate that the method *Slot Guessing for Perturbation Caption* may also apply to other types of image-feature-analyzing benchmarks. Third, the effectiveness of *Option Order Sensitivity Test* can be undermined by option shuffling, which, while potentially improving model performance, is computationally expensive and may increase the training cost.

References

Marah I Abdin, Sam Ade Jacobs, Ammar Ahmad Awan, Jyoti Aneja, Ahmed Awadallah, Hany Awadalla, Nguyen Bach, Amit Bahree, Arash Bakhtiari, Harkirat S. Behl, Alon Benhaim, Misha Bilenko, Johan Bjorck, Sébastien Bubeck, Martin Cai, Caio César Teodoro Mendes, Weizhu Chen, Vishrav Chaudhary, Parul Chopra, Allie Del Giorno, Gustavo de Rosa, Matthew Dixon, Ronen Eldan, Dan Iter, Amit Garg, Abhishek Goswami, Suriya Gunasekar, Emman Haider, Junheng Hao, Russell J. Hewett, Jamie Huynh, Mojan Javaheripi, Xin Jin, Piero Kauffmann, Nikos Karampatziakis, Dongwoo Kim, Mahoud Khademi, Lev Kurilenko, James R. Lee, Yin Tat Lee, Yuanzhi Li, Chen Liang, Weishung Liu, Eric Lin, Zeqi Lin, Piyush Madan, Arindam Mitra, Hardik Modi, Anh Nguyen, Brandon Norick, Barun Patra, Daniel Perez-Becker, Thomas Portet, Reid Pryzant, Heyang Qin, Marko Radmilac, Corby Rosset, Sambudha Roy, Olatunji Ruwase, Olli Saarikivi, Amin Saied, Adil Salim, Michael Santacroce, Shital Shah, Ning Shang, Hiteshi Sharma, Xia Song, Masahiro Tanaka, Xin Wang, Rachel Ward, Guanhua Wang, Philipp Witte, Michael Wyatt, Can Xu, Jiahang Xu, Sonali Yadav, Fan Yang, Ziyi Yang, Donghan Yu, Chengruidong Zhang, Cyril Zhang, Jianwen Zhang, Li Lyna Zhang, Yi Zhang, Yue Zhang, Yunan Zhang, and Xiren Zhou. 2024. Phi-3 technical report: A highly capable language model locally on your phone. *CoRR*, abs/2404.14219.

Harsh Agrawal, Karan Desai, Yufei Wang, Xinlei Chen, Rishabh Jain, Mark Johnson, Dhruv Batra, Devi Parikh, Stefan Lee, and Peter Anderson. 2019. nocaps: novel object captioning at scale. In *2019 IEEE/CVF International Conference on Computer Vision (ICCV)*. IEEE.

01. AI, :, Alex Young, Bei Chen, Chao Li, Chengen Huang, Ge Zhang, Guanwei Zhang, Heng Li, Jiangcheng Zhu, Jianqun Chen, Jing Chang, Kaidong Yu, Peng Liu, Qiang Liu, Shawn Yue, Senbin Yang, Shiming Yang, Tao Yu, Wen Xie, Wenhao Huang, Xiaohui Hu, Xiaoyi Ren, Xinyao Niu, Pengcheng Nie, Yuchi Xu, Yudong Liu, Yue Wang, Yuxuan Cai, Zhenyu Gu, Zhiyuan Liu, and Zonghong Dai. 2024. Yi: Open foundation models by 01.ai. *Preprint*, arXiv:2403.04652.

Jinze Bai, Shuai Bai, Yunfei Chu, Zeyu Cui, Kai Dang, Xiaodong Deng, Yang Fan, Wenbin Ge, Yu Han, Fei Huang, Binyuan Hui, Luo Ji, Mei Li, Junyang Lin, Runji Lin, Dayiheng Liu, Gao Liu, Chengqiang Lu, Keming Lu, Jianxin Ma, Rui Men, Xingzhang Ren, Xuancheng Ren, Chuanqi Tan, Sinan Tan, Jianhong Tu, Peng Wang, Shijie Wang, Wei Wang, Shengguang Wu, Benfeng Xu, Jin Xu, An Yang, Hao Yang, Jian Yang, Shusheng Yang, Yang Yao, Bowen Yu, Hongyi Yuan, Zheng Yuan, Jianwei Zhang, Xingxuan Zhang, Yichang Zhang, Zhenru Zhang, Chang Zhou, Jingren Zhou, Xiaohuan Zhou, and Tianhang Zhu. 2023a. Qwen technical report. *CoRR*, abs/2309.16609.

Jinze Bai, Shuai Bai, Shusheng Yang, Shijie Wang, Sinan Tan, Peng Wang, Junyang Lin, Chang Zhou, and Jingren Zhou. 2023b. Qwen-vl: A versatile vision-language model for understanding, localization, text reading, and beyond. *Preprint*, arXiv:2308.12966.

Tom B. Brown, Benjamin Mann, Nick Ryder, Melanie Subbiah, Jared Kaplan, Prafulla Dhariwal, Arvind Neelakantan, Pranav Shyam, Girish Sastry, Amanda Askell, Sandhini Agarwal, Ariel Herbert-Voss, Gretchen Krueger, Tom Henighan, Rewon Child, Aditya Ramesh, Daniel M. Ziegler, Jeffrey Wu, Clemens Winter, Christopher Hesse, Mark Chen, Eric Sigler, Mateusz Litwin, Scott Gray, Benjamin Chess, Jack Clark, Christopher Berner, Sam McCandlish, Alec Radford, Ilya Sutskever, and Dario Amodei. 2020. Language models are few-shot learners. *CoRR*, abs/2005.14165.

Zheng Cai, Maosong Cao, Haojiong Chen, Kai Chen, Keyu Chen, Xin Chen, Xun Chen, Zehui Chen, Zhi Chen, Pei Chu, Xiaoyi Dong, Haodong Duan, Qi Fan, Zhaoye Fei, Yang Gao, Jiaye Ge, Chenya Gu, Yuzhe Gu, Tao Gui, Aijia Guo, Qipeng Guo, Conghui He, Yingfan Hu, Ting Huang, Tao Jiang, Penglong Jiao, Zhenjiang Jin, Zhikai Lei, Jiaxing Li, Jingwen Li, Linyang Li, Shuaibin Li, Wei Li, Yining Li, Hongwei Liu, Jiangning Liu, Jiawei Hong, Kaiwen Liu, Kuikun Liu, Xiaoran Liu, Chengqi Lv, Haijun Lv, Kai Lv, Li Ma, Runyuan Ma, Zerun Ma, Wenchang Ning, Linke Ouyang, Jiantao Qiu, Yuan Qu, Fukai Shang, Yunfan Shao, Demin Song, Zifan Song, Zhihao Sui, Peng Sun, Yu Sun, Huanze Tang, Bin Wang, Guoteng Wang, Jiaqi Wang, Jiayu Wang, Rui Wang, Yudong Wang, Ziyi Wang, Xingjian Wei, Qizhen Weng, Fan Wu, Yingtong Xiong, and et al. 2024. Internlm2 technical report. *CoRR*, abs/2403.17297.

Lin Chen, Jinsong Li, Xiaoyi Dong, Pan Zhang, Yuhang Zang, Zehui Chen, Haodong Duan, Jiaqi Wang, Yu Qiao, Dahua Lin, et al. 2024a. Are we on the right way for evaluating large vision-language models? *arXiv preprint arXiv:2403.20330*.

Zhe Chen, Weiyun Wang, Hao Tian, Shenglong Ye, Zhangwei Gao, Erfei Cui, Wenwen Tong, Kongzhi Hu, Jiapeng Luo, Zheng Ma, et al. 2024b. How far are we to gpt-4v? closing the gap to commercial multimodal models with open-source suites. *arXiv preprint arXiv:2404.16821*.

Zhe Chen, Jiannan Wu, Wenhai Wang, Weijie Su, Guo Chen, Sen Xing, Muyan Zhong, Qinglong Zhang, Xizhou Zhu, Lewei Lu, Bin Li, Ping Luo, Tong Lu, Yu Qiao, and Jifeng Dai. 2023. Internvl: Scaling up vision foundation models and aligning for generic visual-linguistic tasks. *arXiv preprint arXiv:2312.14238*.

Chunyuan Deng, Yilun Zhao, Xiangru Tang, Mark Gerstein, and Arman Cohan. 2024. 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), NAACL 2024, Mexico City, Mexico, June 16-21, 2024*, pages 8706–8719. Association for Computational Linguistics.

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, Bangkok, Thailand and virtual meeting, August 11-16, 2024*, pages 12039–12050. Association for Computational Linguistics.

Albert Q. Jiang, Alexandre Sablayrolles, Arthur Mensch, Chris Bamford, Devendra Singh Chaplot, Diego de las Casas, Florian Bressand, Gianna Lengyel, Guillaume Lample, Lucile Saulnier, Lélio Renard Lavaud, Marie-Anne Lachaux, Pierre Stock, Teven Le Scao, Thibaut Lavril, Thomas Wang, Timothée Lacroix, and William El Sayed. 2023. Mistral 7b. *Preprint*, arXiv:2310.06825.

Hugo Laurençon, Léo Tronchon, Matthieu Cord, and Victor Sanh. 2024. What matters when building vision-language models? *CoRR*, abs/2405.02246.

Hugo Laurenon, Lucile Saulnier, Léo Tronchon, Stas Bekman, Amanpreet Singh, Anton Lozhkov, Thomas Wang, Siddharth Karamcheti, Alexander M. Rush, Douwe Kiela, Matthieu Cord, and Victor Sanh. 2023. Obelics: An open web-scale filtered dataset of interleaved image-text documents. *Preprint*, arXiv:2306.16527.

Junnan Li, Dongxu Li, Silvio Savarese, and Steven Hoi. 2023. Blip-2: Bootstrapping language-image pre-training with frozen image encoders and large language models. In *International conference on machine learning*, pages 19730–19742. PMLR.

Ji Lin, Hongxu Yin, Wei Ping, Yao Lu, Pavlo Molchanov, Andrew Tao, Huizi Mao, Jan Kautz, Mohammad Shoeybi, and Song Han. 2023. VILA: on pre-training for visual language models. *CoRR*, abs/2312.07533.

Tsung-Yi Lin, Michael Maire, Serge Belongie, Lubomir Bourdev, Ross Girshick, James Hays, Pietro Perona, Deva Ramanan, C. Lawrence Zitnick, and Piotr Dollár. 2015. Microsoft coco: Common objects in context. *Preprint*, arXiv:1405.0312.

Haotian Liu, Chunyuan Li, Yuheng Li, and Yong Jae Lee. 2023a. Improved baselines with visual instruction tuning. *CoRR*, abs/2310.03744.

Yuan Liu, Haodong Duan, Yuanhan Zhang, Bo Li, Songyang Zhang, Wangbo Zhao, Yike Yuan, Jiaqi Wang, Conghui He, Ziwei Liu, Kai Chen, and Dahua Lin. 2023b. Mmbench: Is your multi-modal model an all-around player? *CoRR*, abs/2307.06281.

Pan Lu, Swaroop Mishra, Tony Xia, Liang Qiu, Kai-Wei Chang, Song-Chun Zhu, Oyvind Tafjord, Peter Clark, and Ashwin Kalyan. 2022. Learn to explain: Multimodal reasoning via thought chains for science question answering. In *The 36th Conference on Neural Information Processing Systems (NeurIPS)*.

OpenAI. 2023. GPT-4 technical report. *CoRR*, abs/2303.08774.

Machel Reid, Nikolay Savinov, Denis Teplyashin, Dmitry Lepikhin, Timothy Lillicrap, Jean-baptiste Alayrac, Radu Soricut, Angeliki Lazaridou, Orhan Firat, Julian Schrittwieser, et al. 2024. Gemini 1.5: Unlocking multimodal understanding across millions of tokens of context. *arXiv preprint arXiv:2403.05530*.

Piyush Sharma, Nan Ding, Sebastian Goodman, and Radu Soricut. 2018. Conceptual captions: A cleaned, hypernymed, image alt-text dataset for automatic image captioning. In *Proceedings of ACL*.

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, ICLR 2024, Vienna, Austria, May 7-11, 2024*. OpenReview.net.

Dingjie Song, Shunian Chen, Guiming Hardy Chen, Fei Yu, Xiang Wan, and Benyou Wang. 2024. Milebench: Benchmarking mllms in long context. *arXiv preprint arXiv:2404.18532*.

Kristina Toutanvoa and Christopher D. Manning. 2000. Enriching the knowledge sources used in a maximum entropy part-of-speech tagger. In *Joint SIGDAT Conference on Empirical Methods in Natural Language Processing and Very Large Corpora, EMNLP 2000, Hong Kong, October 7-8, 2000*, pages 63–70. Association for Computational Linguistics.

Hugo Touvron, Louis Martin, Kevin Stone, Peter Albert, Amjad Almahairi, Yasmine Babaei, Nikolay Bashlykov, Soumya Batra, Prajjwal Bhargava, Shruti Bhosale, Dan Bikel, Lukas Blecher, Cristian Canton-Ferrer, Moya Chen, Guillem Cucurull, David Esiobu, Jude Fernandes, Jeremy Fu, Wenyin Fu, Brian Fuller, Cynthia Gao, Vedanuj Goswami, Naman Goyal, Anthony Hartshorn, Saghar Hosseini, Rui Hou, Hakan Inan, Marcin Kardas, Viktor Kerkez, Madian Khabsa, Isabel Kloumann, Artem Korenev, Punit Singh Koura, Marie-Anne Lachaux, Thibaut Lavril, Jenya Lee, Diana Liskovich, Yinghai Lu, Yuning Mao, Xavier Martinet, Todor Mihaylov, Pushkar Mishra, Igor Molybog, Yixin Nie, Andrew Poulton, Jeremy Reizenstein, Rashi Rungta, Kalyan Saladi, Alan Schelten, Ruan Silva, Eric Michael Smith, Ranjan Subramanian, Xiaoqing Ellen Tan, Binh Tang, Ross Taylor, Adina Williams, Jian Xiang Kuan, Puxin Xu, Zheng Yan, Iliyan Zarov, Yuchen Zhang, Angela Fan, Melanie Kambadur, Sharan Narang, Aurélien Rodriguez, Robert Stojnic, Sergey Edunov, and Thomas Scialom. 2023a. Llama 2: Open foundation and fine-tuned chat models. *CoRR*, abs/2307.09288.

Hugo Touvron, Louis Martin, Kevin Stone, Peter Albert, Amjad Almahairi, Yasmine Babaei, Nikolay Bashlykov, Soumya Batra, Prajjwal Bhargava, Shruti Bhosale, Dan Bikel, Lukas Blecher, Cristian Canton-Ferrer, Moya Chen, Guillem Cucurull, David Esiobu, Jude Fernandes, Jeremy Fu, Wenyin Fu, Brian Fuller, Cynthia Gao, Vedanuj Goswami, Naman Goyal, Anthony Hartshorn, Saghar Hosseini, Rui Hou, Hakan Inan, Marcin Kardas, Viktor Kerkez, Madian Khabsa, Isabel Kloumann, Artem Korenev, Punit Singh Koura, Marie-Anne Lachaux, Thibaut Lavril, Jenya Lee, Diana Liskovich, Yinghai Lu, Yuning Mao, Xavier Martinet, Todor Mihaylov, Pushkar Mishra, Igor Molybog, Yixin Nie, Andrew Poulton, Jeremy Reizenstein, Rashi Rungta, Kalyan Saladi, Alan Schelten, Ruan Silva, Eric Michael Smith, Ranjan Subramanian, Xiaoqing Ellen Tan, Binh Tang, Ross Taylor, Adina Williams, Jian Xiang Kuan, Puxin Xu, Zheng Yan, Iliyan Zarov, Yuchen Zhang, Angela Fan, Melanie Kambadur, Sharan Narang, Aurélien Rodriguez, Robert Stojnic, Sergey Edunov, and Thomas Scialom. 2023b. Llama 2: Open foundation and fine-tuned chat models. *CoRR*, abs/2307.09288.

Zhiyu Wu, Xiaokang Chen, Zizheng Pan, Xingchao Liu, Wen Liu, Damai Dai, Huazuo Gao, Yiyang Ma, Chengyue Wu, Bingxuan Wang, Zhenda Xie, Yu Wu, Kai Hu, Jiawei Wang, Yaofeng Sun, Yukun Li, Yishi Piao, Kang Guan, Aixin Liu, Xin Xie, Yuxiang You, Kai Dong, Xingkai Yu, Haowei Zhang, Liang Zhao, Yisong Wang, and Chong Ruan. 2024. Deepseek-vl2: Mixture-of-experts vision-language models for advanced multimodal understanding. *Preprint*, arXiv:2412.10302.

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

Samuel Yeom, Irene Giacomelli, Matt Fredrikson, and Somesh Jha. 2018. Privacy risk in machine learning: Analyzing the connection to overfitting. In *31st IEEE Computer Security Foundations Symposium, CSF 2018, Oxford, United Kingdom, July 9-12, 2018*, pages 268–282. IEEE Computer Society.

Shukang Yin, Chaoyou Fu, Sirui Zhao, Ke Li, Xing Sun, Tong Xu, and Enhong Chen. 2023. A survey on multimodal large language models. *arXiv preprint arXiv:2306.13549*.

A Inefficiency of Unimodal Methods

We demonstrate the results of traditional unimodal contamination detection methods applied to MLLMs.

A.1 Logits-base

These methods determine contamination by observing the distribution of low-probability tokens in model outputs. However, MLLMs typically undergo instruction fine-tuning, which enhances their instruction-following capabilities, leading to less significant differences in token probability distributions. As shown in Table 7, LLaVA-1.5-13b exhibits extremely low perplexity on multimodal benchmark datasets.

Dataset	Perplexity	Split
ScienceQA	1.4498	Training Set
MMStar	1.4359	Validation Set
COCO-Caption2017	1.7530	Validation Set
NoCaps	1.8155	Validation Set

Table 7: Perplexity of LLaVA-1.5-13b on various multimodal benchmarks (100 samples randomly selected from each dataset).

A.2 Masking-base

These methods involve masking phrases or sentences and providing data from the benchmark to guide the model in filling in the missing parts. However, multimodal datasets often contain images that include the masked portions of sentences, effectively providing answers to the model. This results in significantly higher success rates for MLLMs in predicting missing parts compared to unimodal language models, leading to exaggerated contamination detection. As shown in Table 8, LLaVA-1.5-13b has a high probability of Exact Match for predicting the masked word.

A.3 Comparison-base

These methods identify contamination by comparing the similarity between models' outputs and

Dataset	Exact Match	ROUGE-L F1	Split
COCO-Caption2017	0.24	0.36	Validation Set
NoCaps	0.22	0.29	Validation Set

Table 8: Contamination detection of LLaVA-1.5-13b using TS-Guessing (question-based) on various multimodal benchmarks (100 samples randomly selected from each dataset).

benchmark data. However, MLLMs often undergo data augmentation, causing their outputs to diverge significantly from the labels in benchmark data, making effective contamination detection challenging. From Table 9, we can see that CDD (Contamination Detection via Output Distribution) indicates a contamination metric of 0% across all multimodal benchmark datasets.

Dataset	Contamination Metric	Split
COCO-Caption2017	0.0000%	Validation Set
NoCaps	0.0000%	Validation Set

Table 9: Contamination detection of LLaVA-1.5-13b using CDD (Contamination Detection via Output Distribution) on various multimodal benchmarks (100 samples randomly selected from each dataset).