You're reading LLM leaderboards wrong: Disentangling models from pipelines in engineering benchmarks

Anonymous Authors[1]

Abstract

LLM leaderboard scores are widely treated as measures of model capability. We argue they are not — they are joint outcomes of the model *and* the evaluation pipeline. We reproduce four benchmarks (MMLU, ScienceQA, SceMQA, MatSciBench) and show two concrete ways pipelines distort scores: prompt design shifts accuracy by 5–9 percentage points and produces opposite effects depending on task type, and removing tool access from MatSciBench drops o4-mini from 74% to 38%. Engineering benchmarks are especially affected because they combine tool-dependent computation with multimodal inputs, making the pipeline contribution uniquely large compared to general NLP tasks. We call for benchmark papers to, at minimum, provide full pipeline specifications and key ablations for reproducibility, and ideally report score ranges across reasonable pipeline variations rather than single point estimates.

1. Introduction

Leaderboards reduce model evaluation to a single number and invite direct comparison. Implicit in that comparison is the assumption that benchmark scores are intrinsic properties of the models being ranked. Our central claim is that they are not: they are outcomes of specific evaluation design choices, and the gap between these two interpretations is large enough to reverse practical conclusions. This echoes a longer-standing critique that static leaderboards can create misleading impressions of progress when evaluation protocols are not continuously stress-tested (Kiela et al., 2021), and that model rankings themselves can flip under alternative but reasonable prompting schemes (Sclar et al., 2024).

[1]Anonymous Institution, Anonymous City, Anonymous Region, Anonymous Country. Correspondence to: Anonymous Author <anon.email@domain.com>.

Preliminary work. Under review by the International Conference on Machine Learning (ICML). Do not distribute.

This distinction matters particularly for engineering benchmarks. Engineering tasks often involve quantitative computation, where tool access such as calculators or code execution provides a large performance boost. Tool-augmented language models have long been shown to substantially alter measured reasoning performance (Schick et al., 2023). Engineering tasks also frequently include multimodal inputs such as diagrams and schematics, where the image processing pipeline affects results. Multimodal benchmarks are especially sensitive to preprocessing and visual prompting choices (Chen et al., 2024). In these settings, the gap between "model score" and "system score" can be enormous.

Prior work has noted prompt sensitivity in general benchmarks (Wang et al., 2024; Mizrahi et al., 2024) and called for standardised evaluation (Liang et al., 2023). Even the placement of instructions relative to inputs has been shown to materially change model behavior (Liu et al., 2024), and seemingly minor choices such as decoding settings, answer parsing, and seed selection shift scores in quantifiable ways (Polo et al., 2024). The apparent emergent abilities of LLMs have been shown to depend heavily on evaluation choices rather than genuine capability shifts (Schaeffer et al., 2023). Benchmark contamination — where training data overlaps with test sets — is an additional confounder that further decouples scores from true model capability (Golchin & Surdeanu, 2024); even without direct contamination, benchmark saturation can compress performance differences so that small pipeline changes appear larger than genuine capability gaps (Akhtar et al., 2026). Recent work has also critiqued agentic benchmarks for conflating model and scaffold contributions (Kapoor et al., 2025). Prior work, however, has largely examined these factors in isolation. Our contribution is to make the combined problem concrete with quantitative evidence from reproduction experiments, and to show it is systematically worse for engineering-focused benchmarks, where prompt, tool, and modality effects interact and the evaluation pipeline itself becomes a dominant determinant of leaderboard position. We use MMLU and SceMQA as contrast cases to establish the general phenomenon across task types, then show the effect is substantially amplified in MatSciBench, a domain-specific engineering benchmark where the pipeline contribution alone exceeds the difference between competing models on the leaderboard.

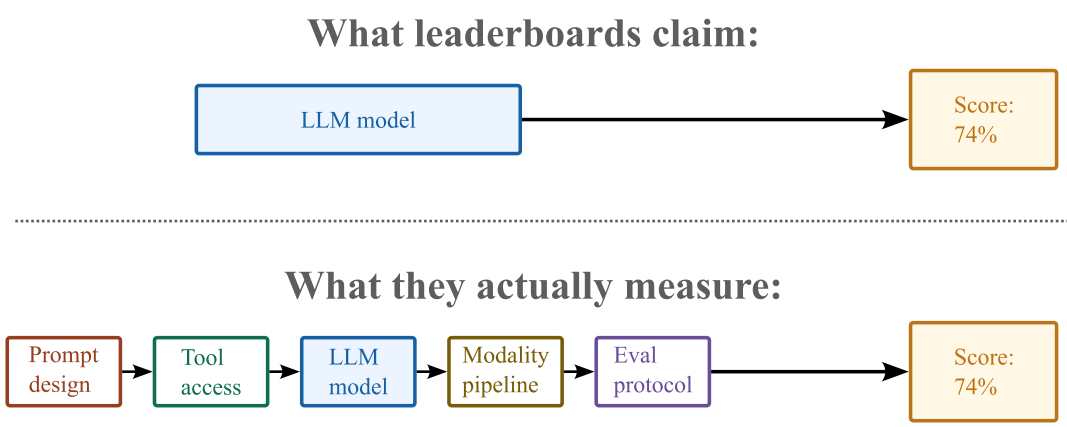

Figure 1. Leaderboards present a single score as a direct measure of the LLM model. In reality, the score is produced by a pipeline of interconnected stages — prompt design, tool access, the model itself, modality processing, and evaluation protocol. All five contribute to the final number, but only the model gets labelled on the leaderboard. Changing any stage changes the score; our results show the effect can exceed 36 percentage points.

Figure 1 illustrates the core issue. A leaderboard score looks like a function of the model. It is actually a function of the model, the prompt, the tools, the modality pipeline, and the scoring protocol — connected stages of an evaluation pipeline, each of which shapes the final number. Changing any of these changes the score, sometimes by more than switching the model.

2. Experimental Setup

We selected four benchmarks representing a deliberate progression from general to engineering-specific evaluation. MMLU (Hendrycks et al., 2021) covers 57 academic subjects and is the most widely cited general reasoning benchmark — most readers will have encountered it, but few will have reproduced it independently. ScienceQA (Lu et al., 2022) and SceMQA (Liang et al., 2024) introduce multimodal reasoning at K-12 and college-entrance level respectively, bridging general and domain-specific evaluation. MatSciBench (Zhang et al., 2025) represents the engineering endpoint: 1,340 university-level materials science problems requiring quantitative reasoning, with an agentic evaluation pipeline that makes the model-versus-system distinction especially sharp.

All experiments used the OpenRouter API at temperature 0 for determinism. Models were matched to each benchmark: GPT-3.5-turbo for MMLU (widely reproduced baseline), GPT-4o for ScienceQA and SceMQA (multimodal), and o4-mini for MatSciBench (the original paper's primary reasoning model).

For prompt sensitivity we tested four variants on MMLU and SceMQA, holding everything else fixed: *Direct*, *Chain-of-Thought*, *Minimal*, and *Constrained* (see Appendix A for full prompts). The MMLU paper specifies a prompt template and provides five example questions per subject for few-shot evaluation, but does not define which subset of these examples to use for $k < 5$ shots. Researchers must therefore make an undocumented choice — selecting two examples for 2-shot evaluation means choosing two out of five, leaving results that cannot be exactly reproduced without knowing the original selection. This is the kind of under specification that creates reproducibility gaps even in well-

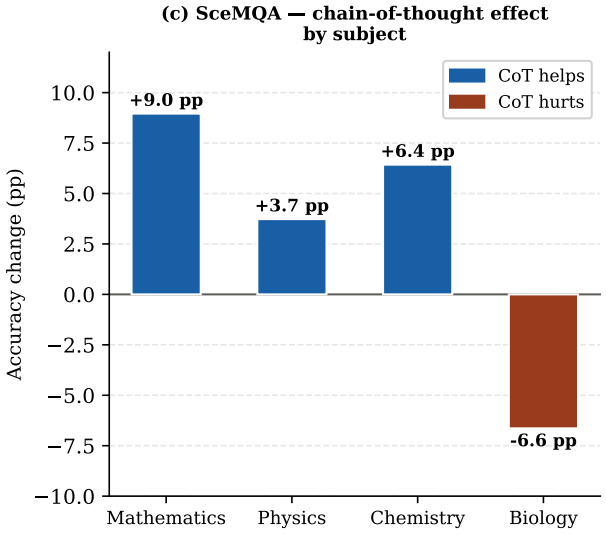

Figure 2. Per-subject change in SceMQA accuracy when switching from the original prompt to chain-of-thought (CoT) with GPT-4o. Computation-heavy subjects (Mathematics, Physics, Chemistry) benefit as CoT scaffolds multi-step calculation, while Biology — conceptual and recall-based — suffers, mirroring the MMLU result. **Whether CoT helps or hurts depends on task structure, not on the model.**

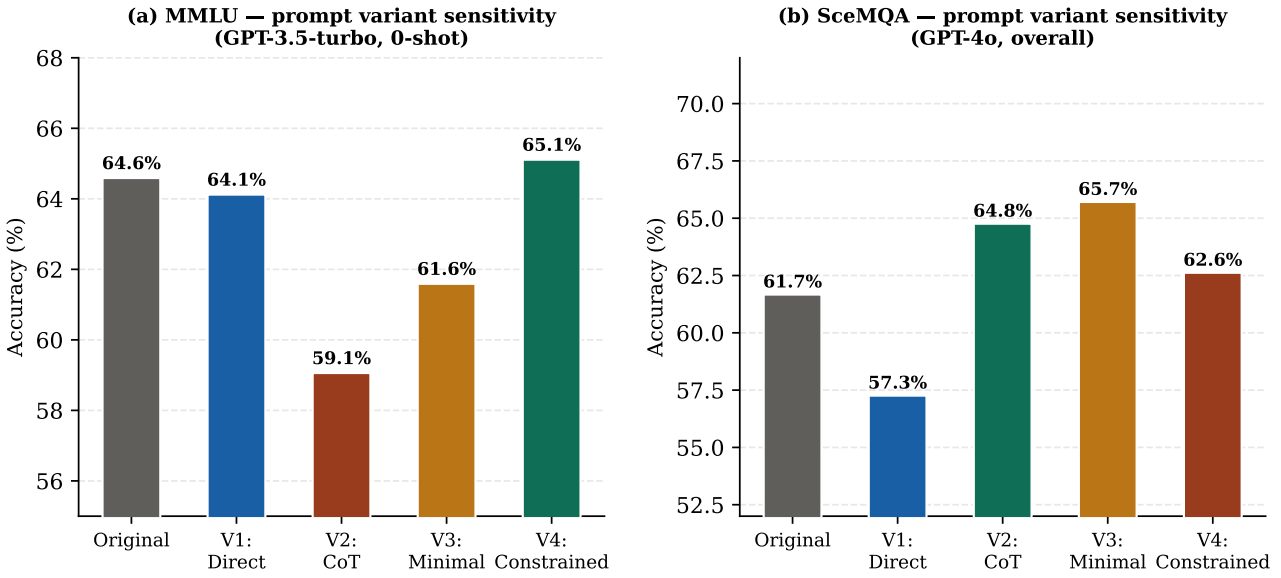

Figure 3. Accuracy across four prompt variants on **(a)** MMLU (GPT-3.5-turbo, 0-shot, 40% sample) and **(b)** SceMQA (GPT-4o, full 840-question multiple-choice set). The *original* variant reproduces each benchmark's published prompt; V1–V4 vary instruction wording and format constraints with model, dataset, and temperature held fixed (see Appendix A). Since neither benchmark mandates a zero-shot prompt, all are valid choices. The spread reaches 5.5 pp (MMLU) and 8.5 pp (SceMQA) — enough to affect rankings between models. **The prompt alone determines a substantial fraction of the reported score.**

documented benchmarks. For MatSciBench we compare two evaluation modes: our *direct reasoning* reproduction, where the model solves problems in natural language only, against the *tool-augmented* results from the original paper, where the model writes Python code that executes locally and feeds results back for synthesis. While the paper describes this strategy and provides a GitHub repository, the implementation is sufficiently complex that replicating it from scratch is not straightforward — the pipeline requires setting up a local sandboxed Python execution environment integrated into the API evaluation loop, handling execution errors and timeouts, and feeding code outputs back to the model in a structured format. This represents a practical reproducibility barrier even when documentation exists.

Accuracy is computed differently depending on the benchmark and task type. For MMLU and SceMQA, which are purely multiple-choice, accuracy is the fraction of questions where the model's selected option matches the ground truth. For MatSciBench, which contains predominantly numerical free-response answers, we follow the original paper's hybrid evaluation protocol: rule-based exact matching with a 5% numerical tolerance for approximate answers, supplemented by Gemini-2.0-Flash as judge for formula-type responses where symbolic equivalence cannot be determined by string comparison alone. The 5% tolerance accounts for rounding differences and unit conversion approximations that would cause a strict string-match evaluator to mark equivalent answers as incorrect.

3. Results

3.1. Prompt design matters — and the direction depends on the task

Figure 3 shows accuracy for MMLU and SceMQA across four prompt variants. On MMLU, chain-of-thought (CoT) prompting is the worst configuration, reducing accuracy by 5.5 pp versus the best direct-answer variant. MMLU questions are mostly recall-based, and reasoning chains sometimes lead the model to a wrong conclusion that direct memory retrieval would have avoided. Crucially, the MMLU paper does not mandate a prompt template for zero-shot evaluation, so different researchers can legitimately report different numbers for the same model and dataset simply by rephrasing the instruction.

On SceMQA the same CoT instruction produces the opposite result: overall accuracy improves by 3.1 pp over the original prompt, and the spread across all four variants reaches 8.5 pp — comparable to the difference between model generations. One might suspect CoT only hurts MMLU because GPT-3.5-turbo is older and gets distracted by its own reasoning, but our SceMQA results with the frontier GPT-4o rule this out: on the conceptual, recall-heavy Biology subset, CoT still penalises GPT-4o by 6.6 pp. The degradation is therefore tied to a fundamental mismatch between prompt structure (step-by-step reasoning) and task requirement (direct fact retrieval), across model generations.

Figure 2 shows *why* CoT has opposite effects: it depends

on task structure. Computation-heavy subjects in SceMQA (mathematics, chemistry, physics) benefit from step-by-step reasoning, gaining 4–9 pp. Biology, which requires conceptual recall rather than calculation (Wang et al., 2025) loses 6.6 pp — exactly mirroring the MMLU result. This means there is no universally optimal prompt: the best choice depends on what the benchmark is actually testing.

This task-structure dependence matters directly for engineering tasks, which are computation-dominant — applying equations, propagating units, solving numerical problems — making CoT the natural default. But as SceMQA shows, the effect varies by subject even within one benchmark. An engineering benchmark that does not specify and justify its prompt choice is reporting an unanchored number.

3.2. Tool access can drastically change benchmark outcomes

Figure 4 shows our most striking result. On MatSciBench, o4-mini scores 74.3% under the tool-augmented evaluation used in the original paper. Under direct reasoning only — the same model, same questions, same judge — it scores 38.3%. Because the tool-use pipeline is prohibitively complex to reproduce from the original repository, we compare our rigorous direct-reasoning baseline against the paper's reported system score; while minor prompt or temperature differences could account for a small fraction of the 36 pp gap, its sheer magnitude points overwhelmingly to the execution pipeline as the cause.

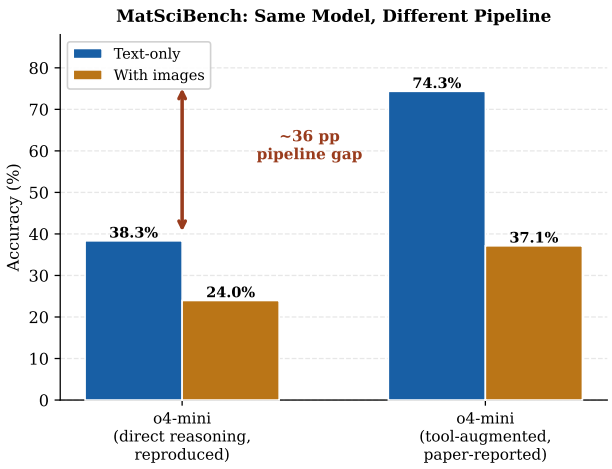

Figure 4. MatSciBench accuracy for o4-mini under two pipelines: direct reasoning (our reproduction) versus tool-augmented evaluation with Python execution (paper-reported), split by text-only and with-images questions. The ∼36 pp gap on text-only questions comes from removing code execution — model, dataset, and judge are identical — while consistently lower image accuracy shows modality handling as a further pipeline factor. **More than half of the reported MatSciBench score reflects the evaluation pipeline, not the model.**

Both text-only and with-images conditions are shown because MatSciBench includes 315 image-based questions (phase diagrams, crystal structures, schematics). The consistently lower accuracy on image questions in both conditions confirms that multimodal processing is an additional pipeline factor — a concern especially relevant for engineering benchmarks, where diagrams are common.

4. Discussion and Conclusions

Our results show that evaluation pipelines contribute substantially to benchmark scores: progress on LLM leaderboards may reflect progress in the pipelines around the model as much as in the model itself. This is not a minor technical detail — it changes what leaderboard numbers mean. A limitation of our study is that we vary models across benchmarks (GPT-3.5, GPT-4o, o4-mini) to show the breadth of pipeline sensitivity; future work should map these ablations across a unified set of frontier models to quantify how scale interacts with prompt brittleness and tool dependence. We acknowledge that tool use and code-writing are themselves genuine model capabilities, but current leaderboards bundle them with knowledge, reasoning, and pipeline engineering into a single number — these should instead be reported as separate, clearly labelled axes.

For **benchmark designers**, we suggest three concrete changes: (1) always report a reasoning-only baseline alongside any tool-augmented result so that the pipeline contribution is explicit; (2) report prompt sensitivity across at least two or three template variants — the gaps are large enough to matter and cost little to measure; and (3) specify the complete inference setup, including tool access, few-shot selection procedure, and exact prompts, to ensure full reproducibility.

Practitioners should be reminded that a score like 74% on MatSciBench means little without understanding the setup behind it — removing tool access alone can drop that number to 38%. Reproducing leaderboard results depends as much on reproducing the inference pipeline as on having access to the model itself.

For the **community**, we propose distinguishing *model leaderboards* (tool-free, standardised prompts, measuring intrinsic model capability) from *system leaderboards* (full pipeline, measuring what a well-engineered system can achieve). Current leaderboards conflate the two, misleading both audiences.

Ultimately, engineering benchmarks make this distinction unavoidable: tool access rewards computation and diagrams require image processing, so the pipeline contribution is largest exactly where reliable evaluation matters most. Until benchmark papers report both, their scores should be read accordingly.

Impact Statement

This paper presents work whose goal is to advance the field of Machine Learning. There are many potential societal consequences of our work, none which we feel must be specifically highlighted here.

References

Akhtar, M., Reuel, A., Soni, P., Ahuja, S., Ammanamanchi, P. S., Rawal, R., Zouhar, V., Yadav, S., Whitehouse, C., Ki, D., et al. When ai benchmarks plateau: A systematic study of benchmark saturation. *arXiv preprint arXiv:2602.16763*, 2026.

Chen, L., Li, J., Dong, X., Zhang, P., Zang, Y., Chen, Z., Duan, H., Wang, J., Qiao, Y., Lin, D., et al. Are we on the right way for evaluating large vision-language models? *Advances in Neural Information Processing Systems*, 37: 27056–27087, 2024.

Golchin, S. and Surdeanu, M. Time travel in llms: Tracing data contamination in large language models, 2024. URL https://arxiv.org/abs/2308.08493.

Hendrycks, D., Burns, C., Basart, S., Zou, A., Mazeika, M., Song, D., and Steinhardt, J. Measuring massive multitask language understanding, 2021. URL https://arxiv.org/abs/2009.03300.

Kapoor, S., Stroebl, B., Siegel, Z. S., Nadgir, N., and Narayanan, A. Ai agents that matter. *Transactions on Machine Learning Research*, 2025, 2025.

Kiela, D., Bartolo, M., Nie, Y., Kaushik, D., Geiger, A., Wu, Z., Vidgen, B., Prasad, G., Singh, A., Ringshia, P., et al. Dynabench: Rethinking benchmarking in nlp. In *Proceedings of the 2021 conference of the North American chapter of the Association for Computational Linguistics: human language technologies*, pp. 4110–4124, 2021.

Liang, P., Bommasani, R., Lee, T., Tsipras, D., Soylu, D., Yasunaga, M., Zhang, Y., Narayanan, D., Wu, Y., Kumar, A., Newman, B., Yuan, B., Yan, B., Zhang, C., Cosgrove, C., Manning, C. D., Re, C., Acosta-Navas, D., Hudson, D. A., Zelikman, E., Durmus, E., Ladhak, F., Rong, F., Ren, H., Yao, H., WANG, J., Santhanam, K., Orr, L., Zheng, L., Yuksekgonul, M., Suzgun, M., Kim, N., Guha, N., Chatterji, N. S., Khattab, O., Henderson, P., Huang, Q., Chi, R. A., Xie, S. M., Santurkar, S., Ganguli, S., Hashimoto, T., Icard, T., Zhang, T., Chaudhary, V., Wang, W., Li, X., Mai, Y., Zhang, Y., and Koreeda, Y. Holistic evaluation of language models. *Transactions on Machine Learning Research*, 2023. ISSN 2835-8856.

Liang, Z., Guo, K., Liu, G., Guo, T., Zhou, Y., Yang, T., Jiao, J., Pi, R., Zhang, J., and Zhang, X. Scemqa: A scientific college entrance level multimodal question answering benchmark, 2024. URL https://arxiv.org/abs/2402.05138.

Liu, Y., Zeng, X., Shao, C., Meng, F., and Zhou, J. Instruction position matters in sequence generation with large language models. In *Findings of the Association for Computational Linguistics: ACL 2024*, pp. 11652–11663, 2024.

Lu, P., Mishra, S., Xia, T., Qiu, L., Chang, K.-W., Zhu, S.-C., Tafjord, O., Clark, P., and Kalyan, A. Learn to explain: Multimodal reasoning via thought chains for science question answering, 2022. URL https://arxiv.org/abs/2209.09513.

Mizrahi, M., Kaplan, G., Malkin, D., Dror, R., Shahaf, D., and Stanovsky, G. State of what art? a call for multi-prompt llm evaluation, 2024. URL https://arxiv.org/abs/2401.00595.

Polo, F. M., Weber, L., Choshen, L., Sun, Y., Xu, G., and Yurochkin, M. tinybenchmarks: evaluating llms with fewer examples. In *Forty-first International Conference on Machine Learning*, 2024.

Schaeffer, R., Miranda, B., and Koyejo, S. Are emergent abilities of large language models a mirage?, 2023. URL https://arxiv.org/abs/2304.15004.

Schick, T., Dwivedi-Yu, J., Dessì, R., Raileanu, R., Lomeli, M., Hambro, E., Zettlemoyer, L., Cancedda, N., and Scialom, T. Toolformer: Language models can teach themselves to use tools. *Advances in neural information processing systems*, 36:68539–68551, 2023.

Sclar, M., Choi, Y., Tsvetkov, Y., and Suhr, A. Quantifying language models' sensitivity to spurious features in prompt design or: How i learned to start worrying about prompt formatting. In *The Twelfth International Conference on Learning Representations*, 2024.

Wang, X., Antoniades, A., Elazar, Y., Amayuelas, A., Albalak, A., Zhang, K., and Wang, W. Y. Generalization vs memorization: Tracing language models' capabilities back to pretraining data. In *The Thirteenth International Conference on Learning Representations*, 2025.

Wang, Y., Ma, X., Zhang, G., Ni, Y., Chandra, A., Guo, S., Ren, W., Arulraj, A., He, X., Jiang, Z., et al. Mmlu-pro: A more robust and challenging multi-task language understanding benchmark. *Advances in Neural Information Processing Systems*, 37:95266–95290, 2024.

Zhang, J., Gan, J., Wang, X., Jia, Z., Gu, C., Chen, J., Zhu, Y., Ma, M. D., Zhou, D., Li, L., and Wang, W. Matscibench: Benchmarking the reasoning ability of large language models in materials science, 2025. URL https://arxiv.org/abs/2510.12171.

A. Appendix: Prompt Variants and Reproduction Details

V1 Direct: `Return ONLY the letter: A, B, C, or D.`

V2 Chain-of-Thought: `Think through this step-by-step, then provide your final answer as a single letter (A, B, C, or D).`

V3 Minimal: `Answer: [A/B/C/D]`

V4 Constrained: `IMPORTANT: Your response must end with ONLY the letter of your answer. No other text after the letter.`

Answer extraction used a priority-based regex: for CoT responses it searches for explicit answer-statement patterns (e.g., "the answer is B") before falling back to the last standalone letter, avoiding first-match errors on mid-sentence letters.

For MatSciBench, the judge model was Gemini-2.0-Flash with a 5% numerical tolerance, matching the original paper's protocol. Missing image files in the public release were skipped and documented as a dataset integrity issue unrelated to the pipeline comparison.