Beyond Token Noise: Discourse-Level Rewrites Expose Brittleness in QA LLMs

Anonymous ACL submission

Abstract

Question answering (QA) systems should remain stable across meaning-preserving variations of the same problem. Earlier robustness work showed brittleness to local lexical perturbations, such as synonym replacement, word swaps, insertions, and deletions. Revisiting this robustness axis for LLMs, we find that high-performing models are now comparatively stable to such edits. However, this stability does not extend to discourse-level rewrites: answer-preserving transformations that alter a question's structure and presentation while preserving its intent and gold answer. Across reasoning-focused QA benchmarks, these rewrites induce substantially larger performance drops than local perturbations. They also change how models fail: using a compact failure taxonomy and Total Variation distance, we show that discourse-level rewrites systematically redistribute error types, revealing distinct failure regimes. Our results suggest that local lexical perturbations are no longer sufficient as a diagnostic robustness test for QA LLMs, and that discourse-level brittleness is an important axis for evaluating reasoning stability[1].

1 Introduction

Robust question answering (QA) systems should remain stable across meaning-preserving variations of the same problem. Much of the robustness literature has tested this requirement through local lexical perturbations, such as synonym replacement, token insertion, word swaps, and deletions, as popularized by EDA (Wei and Zou, 2019). Robustness and adversarial-augmentation toolkits similarly operationalize meaning preservation through token-level noise and localized edits (Wei and Zou, 2019; Morris et al., 2020; Wang et al., 2021). These perturbations exposed brittleness in earlier NLP systems, but our experiments suggest that they are becoming less diagnostic for high-performing QA LLMs: local edits typically leave both accuracy and failure profiles close to the original baseline.

We identify a different robustness axis: discourse-level brittleness. Unlike local perturbations, which test tolerance to isolated token or span changes, discourse-level rewrites test whether a model preserves the same reasoning when a problem's semantic constraints are reorganized, simplified, made more complex, embedded in a benign context, or presented in a different clause order. These rewrites alter how the question is structured and presented while preserving its intent and gold answer. They also correspond to naturally occurring reformulations in conversational QA-style data (App. M), making them a relevant robustness test rather than merely a synthetic stress test.

We evaluate this robustness axis by contrasting local lexical perturbations with discourse-level rewrites (DLRs): answer-preserving question-level transformations generated along interpretable axes while keeping the task, semantic constraints, and gold answer fixed (see App. Table 3 for examples). To reduce artifacts from automatic generation, we treat rewrites as candidates rather than ground truth, and validate semantic consistency and label invariance before evaluation; App. G reports audit statistics and filtering rates, and App. J discusses residual LLM-as-judge circularity risks. Across multiple reasoning-focused QA benchmarks and widely used LLMs, we observe a consistent pattern: relative to local lexical perturbations, discourse-level rewrites induce larger performance drops and measurable differences in the types of errors models make. Importantly, this does not imply that local edits are uninformative; rather, they test a different and less diagnostic dimension of robustness.

Beyond accuracy, we ask how models fail, not just how often. We annotate incorrect outputs with a compact taxonomy of five reasoning failures: Heuristic Short-Circuiting, Premise Rejec-

[1] https://huggingface.co/datasets/DLRewrites/unified-reasoning-augmented

tion, Hyper-Granularity Trap, Multi-step Reasoning Collapse, and Numerical Calculation Error (see Sec. 6.1 for formal definitions). We then quantify rewrite sensitivity by measuring how the failure-mode distribution shifts from the baseline using Total Variation distance. We find that discourse-level rewrites do not merely increase error rates; they systematically redistribute error types, revealing distinct failure regimes.

The contributions of this work are fourfold: **(1)** We identify discourse-level brittleness as a distinct robustness axis for QA LLMs, showing that local lexical perturbations are comparatively less diagnostic in our experiments, while discourse-level rewrites induce larger performance drops. **(2)** We introduce a controlled evaluation framework for answer-preserving discourse-level rewrites, enabling direct comparison with local perturbations under fixed semantics and validated label invariance. **(3)** We go beyond aggregate accuracy by analyzing failure modes and using Total Variation distance to quantify how rewrites redistribute model errors. **(4)** We include an initial mitigation study showing that fine-tuning on discourse-level rewrites improves LLM robustness.

2 Related Work

Prior work has studied robustness to local textual perturbations, such as synonym replacement, insertion, deletion, word swaps, and adversarial edits (Wei and Zou, 2019; Morris et al., 2020; Wang et al., 2021). These methods exposed brittleness in earlier NLP systems and remain useful baselines, but they primarily test tolerance to token or short-span noise. Our results suggest that such perturbations are less diagnostic for QA-focused LLMs: local edits typically keep accuracy and failure profiles closer to the original baseline than DLRs.

A related line of work studies prompt sensitivity, showing that LLM behavior can vary under different instructions, templates, formatting choices, or semantically similar prompt variants (Sclar et al., 2024; Mizrahi et al., 2024; Chatterjee et al., 2024; Cao et al., 2024; Zhu et al., 2024). Our setting differs in the object of variation: we keep the task instruction, answer space, and evaluation protocol fixed, and rewrite the QA instance itself.

Closest to our setting are studies of irrelevant or distracting context in QA and mathematical reasoning, including adversarial distractors and irrelevant-context benchmarks such as GSM-IC and GSM-DC (Jia and Liang, 2017; Shi et al., 2023; Yang et al., 2025). These works show that irrelevant information can degrade performance. We study a broader family of answer-preserving DLRs, where irrelevant information is just one case alongside simplification, increased linguistic complexity, question reordering, and clause movement.

Thus, while prior work establishes brittleness to local perturbations, prompt variation, and irrelevant context, we study a broader family of answer-preserving DLRs and analyze both accuracy degradation and failure-mode redistribution.

3 Discourse-Level Rewrite Families

To probe robustness beyond token-level noise, we study DLRs: semantics-preserving question rewrites that alter style, readability, benign context, or discourse structure while keeping the intent and correct answer fixed. For each original question, an LLM rewrites only the surface form while preserving all semantic conditions and required reasoning. We generate one variant per example for five DLR families: **Add Irrelevant Information**, which adds benign distracting details; **Language Harder**, which increases linguistic complexity or formality; **Language Simpler**, which uses simpler vocabulary; **Question Reordering**, which changes the order of question components; and **Clause Reversal**, which moves subordinate clauses while preserving semantic relations.

We generate DLRs using OpenAI o4-mini. To minimize semantic drift, we use: (1) prompt constraints requiring preservation of intent, constraints, and correct answer; (2) a GPT-4o-mini equivalence screening for each `(original, rewrite)` pair, with non-equivalent pairs routed to human correction; and (3) a human audit of judge reliability and generator adherence. Details and examples are provided in Appendices C and G. For implementation details and model selection please see App. F.

4 Experimental Setup

4.1 Datasets

We evaluate on five reasoning-focused QA benchmarks: ARC-C (Clark et al., 2018), BIG-MATH (Albalak et al., 2025), DROP (Dua et al., 2019), GPQA (Rein et al., 2024), and MMLU-PRO (Wang et al., 2024). Together, they span science, mathematics, discrete reasoning, expert-level QA, and multitask reasoning, with both multiple-choice and open-ended formats. Dataset statistics

are provided in App. D. All datasets are augmented with both DLRs and local techniques using the same filtering procedure described in Sec. 3.

4.2 Models

We evaluate five contemporary LLMs: **Llama-3.3-70B**, **DeepSeek-R1**, **Kimi-K2 Thinking**, **Phi-4**, and **GPT-5**. We selected them as high-performing, general-purpose systems that span key design axes: open vs. closed access, model scale, and different instruction-tuning and reasoning paradigms, including deliberation-style models. This set is intended as a representative, not exhaustive, sample of contemporary QA-capable LLMs; all models are evaluated using their default configurations.

4.3 Evaluation Protocol

We benchmark DLR against local perturbation baselines: EDA (Wei and Zou, 2019), and back-translation (Sennrich et al., 2016), which serves as a general rewrite baseline. All models use Chain-of-Thought (Wei and et al., 2022) , producing a brief reasoning trace followed by a final answer (App E); scoring uses only the final answer. We first apply Exact Match after deterministic answer extraction, and use an auxiliary LLM judge only when strict matching fails. Implementation details and validation are provided in App. I.

5 Main Results

Table 1 reports accuracy for five contemporary LLMs on five reasoning-focused QA benchmarks under three input conditions: the original questions (Baseline), local perturbations (Local), and our discourse-level, semantics-preserving rewrites. The main pattern is consistent across models and datasets: DLR cause substantially larger accuracy drops than local perturbations. A paired, one-sided permutation (sign-flip) test on per-example correctness confirms this effect is statistically significant (overall$< 10^{-4}$).

6 The Nature of Reasoning Shifts

We show that DLRs reduce accuracy, and we next study *how* models fail. We conduct an LLM-assisted error analysis (See App. H for the rigorous human in the loop validation) and find that DLR do not merely add noise; they systematically shift the distribution of reasoning failures. We focus on GPT-5, the strongest and most used model in our pool, so these shifts are relevant, and we replicate the analysis for Phi-4 as a smaller reference point to test whether the same qualitative DLR vs. local contrast holds beyond a frontier system (App. K). Unless stated otherwise, the model-specific patterns and deltas reported below refer to GPT-5.

6.1 Error Taxonomy

We label incorrect outputs using a compact, behavior-based taxonomy that identifies *where* reasoning fails. **Heuristic Short-Circuiting** denotes reliance on surface cues or familiar templates instead of the stated constraints (e.g., computing a rote mean while ignoring exclusions). **Premise Rejection** covers cases where the model dismisses or alters the given setup and solves a different problem. **Hyper-Granularity Trap** refers to over-attention to minor or irrelevant details that derail the intended solution. **Multi-step Reasoning Collapse** occurs when a multi-step solution begins correctly but breaks down mid-way due to missing steps, contradictions, or derailments. **Numerical Calculation Error** captures purely operational arithmetic or unit mistakes despite a correct plan. **Other / Misc.** includes residual errors not covered above, such as misreads after clause reordering, retrieval-like hallucinations, or answer-formatting.

6.2 Behavioral Shifts and Failure Regimes

To quantify how much an augmentation changes *how* a model fails, we represent the error profile as a probability vector over categories and compute the *Total Variation (TV) distance* to the baseline:

$$\mathrm{TV}(P_{\text{base}}, P_{\text{aug}}) = \frac{1}{2} \sum_{c \in \mathcal{C}} |P_{\text{base}}(c) - P_{\text{aug}}(c)|$$

where $\mathcal{C}$ is the set of error categories. Intuitively, TV measures the minimum probability mass that must be "shifted" to transform the baseline error distribution into the augmented one. Higher TV therefore reflects a *regime shift* in failure behavior. The full table and per-augmentation breakdowns are reported in App. K.

Local perturbations induce comparatively modest shifts in the error profile, largely reallocating mass toward *Heuristic Short-Circuiting* with smaller and less structured changes to other categories as can be seen in Table 8. In contrast, DLR yield larger and more category-concentrated shifts. For GPT-5, several discourse-level rewrites exceed 20 TV points (e.g., *Language Simpler*), and different transformation families reliably activate

Model	Model # Param.	Augmentation	ARC-C	Big-Math-RL	DROP	GPQA	MMLU-Pro
Llama-3.3-70B	70B	Baseline	97.18 ± 0.18	82.50 ± 0.30	90.12 ± 0.22	50.48 ± 0.60	68.99 ± 0.42
		Local	95.92 ± 0.20	79.11 ± 0.32	86.42 ± 0.26	47.88 ± 0.62	68.35 ± 0.42
		DLR	**92.63** ± 0.24	**76.51** ± 0.34	**82.03** ± 0.30	**43.10** ± 0.65	**62.11** ± 0.48
DeepSeek-R1	671B	Baseline	84.91 ± 0.25	68.25 ± 0.39	93.55 ± 0.18	71.38 ± 0.34	85.71 ± 0.24
		Local	83.31 ± 0.26	68.54 ± 0.39	86.41 ± 0.22	69.99 ± 0.35	84.52 ± 0.25
		DLR	**77.09** ± 0.30	**63.67** ± 0.41	**79.82** ± 0.29	**64.26** ± 0.38	**77.00** ± 0.30
Kimi-K2-Thinking	1T+	Baseline	86.09 ± 0.24	85.00 ± 0.26	93.20 ± 0.18	75.24 ± 0.32	86.46 ± 0.42
		Local	84.15 ± 0.25	81.62 ± 0.28	88.00 ± 0.21	72.25 ± 0.34	85.24 ± 0.42
		DLR	**81.83** ± 0.27	**80.72** ± 0.29	**80.74** ± 0.29	**65.57** ± 0.37	**83.91** ± 0.48
Phi-4	14B	Baseline	96.28 ± 0.60	81.15 ± 0.80	75.61 ± 0.95	54.15 ± 1.23	69.18 ± 1.00
		Local	94.84 ± 0.73	80.54 ± 0.83	73.98 ± 0.98	50.89 ± 1.23	68.84 ± 1.00
		DLR	**92.54** ± 0.80	**77.66** ± 0.90	**62.67** ± 1.20	**47.56** ± 1.16	**66.18** ± 1.03
GPT-5	Undisclosed	Baseline	97.00 ± 0.11	86.10 ± 0.23	91.94 ± 0.25	87.46 ± 0.25	86.42 ± 0.23
		Local	96.02 ± 0.13	84.20 ± 0.24	89.19 ± 0.28	84.72 ± 0.27	85.86 ± 0.23
		DLR	**92.26** ± 0.17	**76.78** ± 0.28	**82.58** ± 0.34	**79.99** ± 0.30	**82.06** ± 0.25

Table 1: Accuracy (%) under Baseline, Local, and DLR inputs; larger DLR drops indicate discourse-level brittleness. Values are reported with 95% Wilson confidence intervals computed over evaluation examples.

distinct failure regimes: structural rewrites (question reordering, clause reversal) sharply increase *Multi-step Reasoning Collapse*, while style changes and benign context injection more often amplify *Heuristic Short-Circuiting* or *Premise Rejection*.

These distribution shifts correspond to interpretable model-specific patterns (see App. K). Simplification increases short-cutting (Δ+15.9%) and also raises *Multi-step Reasoning Collapse* (Δ+5.8%), consistent with *effort miscalibration* under "easy-looking" phrasing. Structural rewrites disrupt planning, producing pronounced spikes in *Multi-step Reasoning Collapse* (Question Reorder: Δ+13.5%; Subordination Reversal: Δ+12.4%) alongside increases in *Premise Rejection* and/or *Hyper-Granularity*. Adding irrelevant but benign context increases *Heuristic Short-Circuiting* (Δ+13.2%) while reducing *Multi-step Reasoning Collapse* (Δ−11.0%), suggesting a shift from derailment to premature, pattern-driven answering. Finally, harder language primarily elevates *Premise Rejection* (Δ+4.9%) with little change in *Multi-step Reasoning Collapse* (Δ+0.3%). Overall, DLRs don't merely scale error rates; they change which failure mode dominates, indicating sensitivity to discourse-level presentation cues.

6.3 Mitigating Discourse-Level Brittleness

To test whether DLR brittleness can be reduced, we fine-tune available models on original training examples augmented with DLR variants. We compare three settings: **Base-DLR**, the unmodified model evaluated on DLR inputs; **Raw SFT**, supervised fine-tuning on original training examples only; and **DLR SFT**, supervised fine-tuning on original examples plus their DLR variants. As shown in Table 2, DLR SFT improves over Raw SFT across all model–dataset pairs. These results suggest that fine tuning on targeted DLR augmentations is a promising mitigation for discourse-level brittleness, although broader mitigation strategies remain future work. Detailed training specifications, data splits, and adversarial fine-tuning results are provided in App. P and App. Q.

Model	Data	Base-DLR	Raw SFT	DLR SFT
LLaMA-3.3-70B	Big-Math	76.51±0.34	82.30±0.28	**84.20**±0.29
	DROP	82.03±0.30	90.02±0.25	**93.12**±0.29
DeepSeek-V2-Lite	Big-Math	48.9±0.70	53.6±0.60	**55.2**±0.63
	DROP	42.0±1.00	49.1±1.22	**51.0**±1.30
Phi-4	Big-Math	77.66±0.90	82.15±0.79	**83.85**±0.78
	DROP	62.67±1.20	74.61±1.02	**76.21**±1.11
GPT-4.1-mini	Big-Math	62.2±0.60	69.0±0.55	**71.2**±0.56
	DROP	75.0±0.40	85.8±0.94	**87.2**±0.97

Table 2: Accuracy (%) under DLR inputs with fine tuning.

7 Conclusions

We show that QA LLMs are comparatively stable to local lexical perturbations but remain brittle to answer-preserving discourse-level rewrites. Across reasoning benchmarks and widely used LLMs, DLRs induce larger performance drops and systematically redistribute failure modes, especially under structural rewrites such as reordering and clause movement. These findings suggest that robust QA evaluation should include controlled discourse-level rewrites alongside canonical prompts and token-level noise. Future work should broaden rewrite families, strengthen semantic validation, and develop more robust mitigation methods for discourse-level brittleness.

8 Limitations

Scope of augmentations. Our DLR families cover common, interpretable rewrite axes (style/readability, benign context injection, and discourse structure), but they do not span all real-world variations (e.g., heavy paraphrase, cross-lingual or code-mixed rewrites, and typos). Thus, our conclusions apply to these controlled families rather than to all naturally occurring phrasing.

Model and task coverage. We study a representative set of QA-oriented LLMs and reasoning benchmarks, but results may differ in other settings (e.g., retrieval-augmented QA, tool-using agents, long-context tasks, or domain-specific QA).

Model-assisted validation. Our rewrite generation and semantic-equivalence filtering use LLM-based components to support large-scale evaluation. Although we reduce circularity through deterministic judging, manual audits, and human inspection of accepted examples, residual judge bias remains a potential risk. This approach should be used with strict human-in-the-loop validation (we discuss our process thoroughly in App. H), particularly when extending DLR to new domains, datasets, or rewrite families. App. J discusses this risk in detail.

References

Alon Albalak, Duy Phung, Nathan Lile, Rafael Rafailov, Kanishk Gandhi, Louis Castricato, Anikait Singh, Chase Blagden, Violet Xiang, Dakota Mahan, and Nick Haber. 2025. Big-math: A large-scale, high-quality math dataset for reinforcement learning in language models. *Preprint*, arXiv:2502.17387.

Bowen Cao, Deng Cai, Zhisong Zhang, Yuexian Zou, and Wai Lam. 2024. On the worst prompt performance of large language models. In *Advances in Neural Information Processing Systems*.

Anwoy Chatterjee, H S V N S Kowndinya Renduchintala, Sumit Bhatia, and Tanmoy Chakraborty. 2024. POSIX: A prompt sensitivity index for large language models. In *Findings of the Association for Computational Linguistics: EMNLP 2024*, pages 14550–14565, Miami, Florida, USA. Association for Computational Linguistics.

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:1803.05457v1*.

Dheeru Dua, Yizhong Wang, Pradeep Dasigi, Gabriel Stanovsky, Sameer Singh, and Matt Gardner. 2019. DROP: A reading comprehension benchmark requiring discrete reasoning over paragraphs. In *Proc. of NAACL*.

Robin Jia and Percy Liang. 2017. Adversarial examples for evaluating reading comprehension systems. In *Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing*, pages 2021–2031, Copenhagen, Denmark. Association for Computational Linguistics.

Moran Mizrahi, Guy Kaplan, Dan Malkin, Rotem Dror, Dafna Shahaf, and Gabriel Stanovsky. 2024. State of what art? a call for multi-prompt LLM evaluation. *Transactions of the Association for Computational Linguistics*, 12:933–949.

John Morris, Eli Lifland, Jin Yong Yoo, Jake Grigsby, Di Jin, and Yanjun Qi. 2020. TextAttack: A framework for adversarial attacks, data augmentation, and adversarial training in NLP. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing: System Demonstrations*, pages 119–126, Online. Association for Computational Linguistics.

David Rein, Betty Li Hou, Asa Cooper Stickland, Jackson Petty, Richard Yuanzhe Pang, Julien Dirani, Julian Michael, and Samuel R. Bowman. 2024. GPQA: A graduate-level google-proof q&a benchmark. In *First Conference on Language Modeling*.

Melanie Sclar, Yejin Choi, Yulia Tsvetkov, and Alane Suhr. 2024. Quantifying language models'sensitivity to spurious features in prompt design or: How i learned to start worrying about prompt formatting. In *International Conference on Learning Representations*, volume 2024, pages 25055–25083.

Rico Sennrich, Barry Haddow, and Alexandra Birch. 2016. Improving neural machine translation models with monolingual data. In *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 86–96, Berlin, Germany. Association for Computational Linguistics.

Freda Shi, Xinyun Chen, Kanishka Misra, Nathan Scales, David Dohan, Ed H. Chi, Nathanael Schärli, and Denny Zhou. 2023. Large language models can be easily distracted by irrelevant context. In *Proceedings of the 40th International Conference on Machine Learning*, volume 202 of *Proceedings of Machine Learning Research*, pages 31210–31227. PMLR.

Xiao Wang, Qin Liu, Tao Gui, Qi Zhang, Yicheng Zou, Xin Zhou, Jiacheng Ye, Yongxin Zhang, Rui Zheng, Zexiong Pang, Qinzhuo Wu, Zhengyan Li, Chong Zhang, Ruotian Ma, Zichu Fei, Ruijian Cai, Jun Zhao, Xingwu Hu, Zhiheng Yan, and 15 others. 2021. TextFlint: Unified multilingual robustness evaluation toolkit for natural language processing. In *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th*

International Joint Conference on Natural Language Processing: System Demonstrations, pages 347–355, Online. Association for Computational Linguistics.

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

Jason Wei and Wang et al. 2022. Chain-of-thought prompting elicits reasoning in large language models. In *Proceedings of the 36th International Conference on Neural Information Processing Systems*, NIPS '22, Red Hook, NY, USA. Curran Associates Inc.

Jason Wei and Kai Zou. 2019. EDA: Easy data augmentation techniques for boosting performance on text classification tasks. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 6382–6388, Hong Kong, China. Association for Computational Linguistics.

Minglai Yang, Ethan Huang, Liang Zhang, Mihai Surdeanu, William Yang Wang, and Liangming Pan. 2025. How is LLM reasoning distracted by irrelevant context? an analysis using a controlled benchmark. In *Proceedings of the 2025 Conference on Empirical Methods in Natural Language Processing*, pages 13329–13347, Suzhou, China. Association for Computational Linguistics.

Kaijie Zhu, Qinlin Zhao, Hao Chen, Jindong Wang, and Xing Xie. 2024. Promptbench: A unified library for evaluation of large language models. *Journal of Machine Learning Research*, 25(254):1–22.

A Acknowledgments

A large language model (GPT-based) was used solely to improve writing clarity. All technical content, experimental design, analysis, and conclusions are the authors' own.

B Discourse-Level Rewrite Examples

C Augmentation Errors and Manual Filtering

To validate the automated semantic equivalence screen, we conducted a manual audit of over 150 canonical–rewrite pairs that were labeled equivalent by GPT-4o-mini. The humans were instructed by the same prompt as the LLM. Each pair was independently annotated by two human annotators following a shared equivalence guideline. We report inter-annotator agreement using Cohen's κ (0.789) and raw accuracy (99.3%), indicating high reliability of the automated equivalence judgments on accepted cases.

Aug. Type	Augmented Question
Original Input	How do **plants** grow in dense forests if **sunlight** is limited?
Add Irrelevant Info (II)	How do **plants**, ***which provide oxygen to the atmosphere***, grow in dense forests if **sunlight** is limited?
Language Harder (LH)	By what ***mechanism*** does **vegetation** ***sustain development*** within canopy-covered biomes provided that **solar radiation** is ***restricted***?
Language Simpler (LS)	How do **plants** ***get big*** in thick woods when there ***isn't much*** **sun**?
Question Reordering (QR) *(Fronting Modifier)*	***In dense forests***, how do **plants** grow if **sunlight** is limited?
Clause Reversal (CR) *(Fronting the Logic)*	***If sunlight is limited***, how do **plants** grow in dense forests?

Table 3: Illustrative example generated using DLR

While the vast majority of generated DLRs preserve the original question's semantics, a small fraction of rewrites were found to violate our invariance assumptions and were therefore either rerouted for modification or excluded. These cases constitute a negligible portion of the dataset and were identified through a combination of automated screening and manual inspection.

We observe three recurring failure modes:

- **Answer Injection (66%)**: the rewriting model answers the question instead of producing a paraphrase, effectively collapsing the task into a solved statement.
- **Null Rewrite (23%)**: the output reproduces the original question verbatim or with only superficial formatting changes, failing to introduce a meaningful DLR.
- **Context Loss (11%)**: essential constraints or background information from the original question are removed or altered, rendering the rewritten question ambiguous or unanswerable.

All such cases were discarded prior to evaluation.

D Dataset

We report our results over 110K samples from the collection of domains we define in section 4.1. The

composition of this dataset is as follows: We randomly collect 5K samples from each of our dataset collection. GPQA includes only $\approx$ 1.2K samples, and for this dataset we evaluate all available instances without resampling. As a result, estimates derived from GPQA exhibit wider confidence intervals than those from larger datasets, reflecting its smaller effective sample size. Overall we have $>$ 22K samples where each of them was augmented in each of the DLR techniques, amounting to 110K samples overall.

E Answer Inference Implementation

Prompt description. *Chain-of-Thought Answering* instructs the model to explicitly reason step by step before producing a final answer. The final answer must be presented on a separate line, in this format `### {answer}`.

```
# System Prompt

- You are given a question that requires
↪  reasoning.
- Carefully reason through the problem step by
↪  step before answering.
- Do not skip intermediate reasoning steps.
- Use clear and explicit logical reasoning to
↪  reach the solution.
- After completing the reasoning, output the
↪  final answer on a new line.
- The final answer must be presented in a new
↪  line in the following format: ### {answer}
- Do not include any additional text after the
↪  final answer.

# Example

Question:
"There are 3 pufferfish in the market.
How many pufferfish are in the market?"

Reasoning:
The question states that there are 3
↪  pufferfish in the market.
No additional changes or operations are
↪  required.
Therefore, the total number of pufferfish is
↪  3.

### 3
```

F Question Augmentation process

F.1 Model Selection

We generate the DLRs using OpenAI o4-mini, chosen for its strong quality–cost tradeoff and reliable instruction following. To ensure that DLR preserve the original question's meaning and answer, we apply safeguards to minimize semantic drift: (1) explicit prompt constraints that require preserving the question's intent, constraints, and correct answer; (2) an automated semantic screen using GPT-4o-mini as an equivalence judge for each `(original, rewrite)` pair, where any pair deemed non-equivalent is routed to human annotators for correction before inclusion; and (3) a human audit that evaluates both the equivalence-judge reliability and the DLR generator's adherence to meaning preservation. Details of the augmentation and validation process, along with representative examples of observed failures, are provided in App. G and App. C respectively.

F.2 Augmentation Prompts

Shared component. All augmentation prompts in this work share the same base instruction. This component enforces semantic preservation, option consistency, and self-contained rewriting, while delegating the actual transformation to the augmentation-specific rules below.

```
# System Prompt

- You only rewrite the user's content.
- Do not answer the question or provide any
↪  solution.
- Rewrite the entire user input into a
↪  self-contained input without omitting any
↪  information.
- Preserve the original meaning and the
↪  correct answer.
- Keep all answer options (if any) present and
↪  in the same order.
- The output must be a full rewritten version
↪  of the original input and must not assume
↪  any prior context.
```

F.3 Question Reordering

Augmentation description. *Question Reordering* changes the surface order of question components while preserving semantics and the answer.

```
# Required Augmentation (extends Shared Base
↪  Prompt)

- Reorder the components of the question while
↪  maintaining its original meaning and
↪  answer.

# Example

Original:
"There are 3 pufferfish in the market.
How many pufferfish are in the market?"

Rewritten:
"How many pufferfish are in the market, given
↪  that there are 3 pufferfish available for
↪  sale?"
```

F.4 Clause Reversal

Augmentation description. *Clause Reversal* moves subordinate clauses to different positions while keeping identical semantics.

```
# Required Augmentation (extends Shared Base
↪  Prompt)

- Rewrite the question by moving subordinate
↪  clauses to different positions while
↪  preserving meaning.

# Example

Original:
"There are 3 pufferfish in the market because
↪  the vendor restocked this morning.
How many pufferfish are in the market?"

Rewritten:
"Because the vendor restocked this morning,
↪  there are 3 pufferfish in the market.
How many pufferfish are in the market?"
```

F.5 Language Simpler

Augmentation description. *Language Simpler* rewrites the input using simpler, more accessible language while preserving meaning and answer.

```
# Required Augmentation (extends Shared Base
↪  Prompt)

- Rewrite the question using simpler
↪  vocabulary and clearer phrasing while
↪  preserving meaning.

# Example

Original:
"There are 3 pufferfish in the market.
How many pufferfish are in the market?"

Rewritten:
"There are 3 pufferfish at the market.
How many pufferfish are there?"
```

F.6 Language Harder

Augmentation description. *Language Harder* rewrites the question using more complex or formal phrasing. If a modified word appears in the answer options, it must be updated consistently.

```
# Required Augmentation (extends Shared Base
↪  Prompt)

- Rewrite the question using more complex and
↪  formal language while preserving meaning.

# Example

Original:
"There are 3 pufferfish in the market.
How many pufferfish are in the market?"

Rewritten:
"Within the confines of the marketplace, there
↪  exist three individual specimens of
↪  pufferfish.
What is the total number of pufferfish present
↪  in this market environment?"
```

F.7 Add Irrelevant Information

Augmentation description. *Add Irrelevant Information* injects unrelated or esoteric details that do not affect the solution path.

```
# Required Augmentation (extends Shared Base
↪  Prompt)

- Add unrelated or descriptive details that
↪  increase length but do not affect the
↪  answer.

# Example

Original:
"There are 3 pufferfish in the market.
How many pufferfish are in the market?"

Rewritten:
"In a seaside market filled with the sound of
↪  waves and colorful coral decorations,
there are 3 pufferfish available for sale.
How many pufferfish are in the market?"
```

F.8 Backward Translation (via German)

Augmentation description. *Backward Translation* paraphrases a question by translating it into an intermediate language and then translating it back to English, while preserving the original meaning, constraints, and correct answer. In this work, we use German as the pivot language.

```
# Required Augmentation (extends Shared Base
↪  Prompt)

- Translate the input question from English
↪  into German.
- Preserve all factual content, constraints,
↪  and numerical values.
- Do not add, remove, or infer information.

# Example

Original:
"There are 3 pufferfish in the market.
How many pufferfish are in the market?"

Translated (German):
"Es gibt 3 Kugelfische auf dem Markt.
Wie viele Kugelfische sind auf dem Markt?"
```

```
# Required Augmentation (extends Shared Base
↪  Prompt)
```

```
- Translate the German question back into
↪  English.
- Preserve the original meaning and answer.
- The output should be fluent English but may
↪  differ lexically from the original.

# Example

German:
"Es gibt 3 Kugelfische auf dem Markt.
Wie viele Kugelfische sind auf dem Markt?"

Rewritten (English):
"There are 3 pufferfish at the market.
How many pufferfish are there at the market?"
```

G Augmentation Validity

To ensure the question was not augmented beyond preserving the initial question and its answer we employ an LLM (GPT-4o-mini) at temperature 0 as a step in our quality-insurance process. Below you can find the prompt we used as a system prompt, which was also used as the prompt to the human annotators.

```
# Task: Determine whether the augmentation
↪  affected the core logic or solvability of
↪  the question

You will compare a base question with its
↪  augmented version and decide whether the
↪  augmentation changes the underlying
↪  problem or its correct answer.

## Augmentation Type Applied
augmentation_tag

## Base Question
base_question

## Augmented Question
augmented_question

## Instructions
Decide if the augmentation **affected the core
↪  logic**.

Return **NOT affected** if the augmented
↪  question is still *the same problem* with
↪  the *same correct answer*, including:
- Meaning-preserving rephrasing or paraphrase
- Added background/context that does not
↪  change constraints
- Reordered information that preserves the
↪  same relationships and requirements

Return **IS affected** if the augmentation
↪  changes the problem in a way that could
↪  change the correct answer or make it not
↪  well-defined, including:
- Added/removed/modified constraints,
↪  quantities, or conditions
- Altered relationships between entities or
↪  steps required to solve
- Introduced ambiguity, contradiction, or
↪  missing information that makes the
↪  question unclear or unsolvable

**Output exactly one label:** `NOT affected`
↪  or `IS affected`.
```

Augmentation Type	Accuracy	κ
EDA Random Deletion (eda_rd)	0.94	0.884
EDA Random Insertion (eda_ri)	0.92	0.863
EDA Random Swap (eda_rs)	0.99	0.865
EDA Synonym Replacement (eda_sr)	0.99	0.844
Back Translation	0.99	0.862
Add Irrelevant Information	0.99	0.797
Question Reordering	0.99	0.797
Language Harder	0.99	0.797
Language Simpler	0.98	0.719
Clause Reversal	0.97	0.737

Table 4: Inter-annotator agreement by augmentation type. Accuracy is reported as the average agreement between annotators.

Dataset	Accuracy	κ
DROP	0.97	0.738
Big-Math	0.97	0.717
GPQA	0.98	0.745
MMLU-Pro	0.99	0.853
ARC-C	0.97	0.739

Table 5: Inter-annotator agreement by dataset. Accuracy is reported as the average agreement between annotators.

H Sequential Human Audit of LLM-as-Judge Components

Several stages of our pipeline use an LLM as a judge to support large-scale evaluation. These include semantic-equivalence validation of rewritten questions, answer resolution after deterministic extraction fails, mistake binning into failure categories, and annotation of naturally occurring DLR patterns in conversational data. Since exhaustively validating every LLM judgment is infeasible at the scale of our experiments, we estimate the reliability of each LLM-judge component using a sequential human audit.

Audit unit. For each LLM-judge component j, we define a population of judged items of size N_j. The audit unit depends on the component: an original–rewrite pair for semantic equivalence, a model response for answer resolution, an incorrect response for mistake binning, and a conversational user message for DLR-realism annotation. We sample items uniformly at random from each

component, and where relevant, stratify by dataset, augmentation family, or predicted label to avoid masking category-specific failures.

For each audited item, we define a binary agreement variable:

$$X_i^{(j)} = \begin{cases} 1 & \text{agreement,} \\ 0 & \text{disagreement.} \end{cases}$$

The estimated accuracy of the LLM judge for component j is

$$\hat{p}_j = \frac{1}{n_j} \sum_{i=1}^{n_j} X_i^{(j)},$$

where n_j is the number of manually audited items.

Stopping rule. We use a sequential Wilson confidence-interval audit. Human annotations are collected in batches, and after each batch we compute a Wilson 95% confidence interval for $\hat{p}_j$. Sampling stops once the interval half-width is at most $\epsilon = 0.03$, meaning that the estimated judge accuracy is within approximately $\pm 3\%$.

For $z = 1.96$, the standard-normal critical value for a two-sided 95% confidence interval, the Wilson interval is:

$$\mathrm{CI}_{95}(\hat{p}_j) = C_j \pm h_j,$$

where

$$C_j = \frac{\hat{p}_j + \frac{z^2}{2n_j}}{1 + \frac{z^2}{n_j}}$$

and

$$h_j = \frac{z}{1 + \frac{z^2}{n_j}} \sqrt{\frac{\hat{p}_j(1-\hat{p}_j)}{n_j} + \frac{z^2}{4n_j^2}}.$$

The stopping criterion is therefore:

$$h_j \leq 0.03.$$

Finite-population correction. When sampling without replacement from a finite population, the variance can be adjusted using the finite-population correction:

$$\mathrm{FPC}_j = \sqrt{\frac{N_j - n_j}{N_j - 1}}.$$

For our largest judged population, $N \approx 110{,}000$. Even with $n = 150$, this gives:

$$\mathrm{FPC} = \sqrt{\frac{110{,}000 - 150}{110{,}000 - 1}} \approx 0.9993 \approx 1.$$

LLM-judge component	Audit unit	Manual audit size	Criterion
Semantic-equivalence validation	Original–rewrite pair judged as preserving or changing the original problem	$n = 150$; $\hat{p} \approx 0.99$	$h \leq 0.03$
Answer resolution after extraction	Model response requiring judge-based answer resolution after exact matching fails	$n \geq 165$; $\hat{p} \approx 0.98$	$h \leq 0.03$
Mistake binning	Incorrect model response assigned to a failure category	$n \geq 215$; $\hat{p} \approx 0.95$	$h \leq 0.03$
DLR realism annotation	Conversational user message labeled for DLR-style patterns	$n \geq 388$; $\hat{p} \approx 0.90$	$h \leq 0.03$

Table 6: Human audit plan for LLM-as-judge components. Each component is audited independently using a sequential Wilson 95% confidence interval. The required audit size depends on the observed agreement rate: higher agreement requires fewer samples to achieve the same $\pm 3\%$ precision. The reported n values are the minimum sample sizes needed under the corresponding observed agreement assumptions.

Thus, for our audit sizes, the uncertainty is governed primarily by the absolute number of audited examples n_j, not by the fraction n_j/N_j. This justifies using fixed audit-size stopping criteria rather than auditing a fixed percentage of the dataset.

Example: semantic-equivalence judge. For semantic-equivalence validation, we audited $n = 150$ original–rewrite pairs. The observed agreement was:

$$\hat{p} = \frac{149}{150} = 0.993.$$

The resulting Wilson 95% confidence interval is approximately:

$$\mathrm{CI}_{95} \approx [0.963, 0.999],$$

with half-width:

$$h \approx 0.018.$$

This satisfies the stopping rule $h \leq 0.03$. Therefore, the semantic-equivalence judge has an estimated accuracy of 99.3%, with uncertainty below the target $\pm 3\%$ threshold.

Reporting convention. For each LLM-as-judge component, we report the observed agreement rate, the Wilson 95% confidence interval, and the final human audit size. Cohen's κ is reported separately when multiple human annotators are used, since it measures human–human annotation reliability rather than LLM-judge accuracy.

I Answer Check

We employ a robust method to evaluate the model's accuracy across a variety of formats:

Two-Stage Accuracy Evaluation. Our evaluation follows a two-stage protocol to ensure robustness to answer formatting while maintaining strict correctness.

We first attempt deterministic answer extraction by normalizing the model output (case-folding; stripping punctuation and whitespace) and mapping common variants (e.g., "Option C", "(c)", "answer: c") to a canonical option label. If a valid label is extracted, it is scored using Exact Match against the gold answer.

For outputs that fail strict matching—such as missing labels, multiple conflicting options, or implicitly stated choices—we apply an auxiliary LLM-based judge. The judge is given the original question, the gold answer, the list of choices (when applicable), and the model response, and determines whether the response unambiguously corresponds to a single option. If so, the inferred label is used for scoring; otherwise, the answer is marked incorrect.

We use GPT-4o-mini as the judge with deterministic decoding (temperature $= 0$). To validate this step, we conduct a human audit of over 165 randomly sampled judge-resolved cases, annotated independently by two annotators. This audit yields Cohen's $\kappa = 0.89$ and a judge accuracy of 98%, indicating high reliability of the judge-based resolution.

For reproducibility we share the system prompt used below:

```
    # Task: Answer Resolution

You are given a question, the gold answer,
↪  optional answer choices, and a model
↪  response.

Do not solve the question from scratch. Decide
↪  only whether the model response
↪  unambiguously matches the gold answer.

## Question
{question}

## Answer Choices
{answer_choices}

## Gold Answer
{gold_answer}

## Model Response
{model_response}

## Instructions

Mark correct if the response clearly gives the
↪  same answer as the gold answer, allowing
↪  harmless formatting differences such as:
- "C", "(C)", "Option C", or the text of
↪  option C;
- equivalent wording for free-form answers;
- equivalent numerical formatting, such as
↪  "1000" and "1,000".

Mark incorrect if the response gives a
↪  different answer, multiple conflicting
↪  answers, no final answer, an ambiguous
↪  answer, a refusal, or an answer to a
↪  different question.

If choices are provided, infer the selected
↪  option only when the response clearly
↪  corresponds to exactly one option.

Return ONLY valid JSON:

{
  "is_correct": true or false,
  "resolved_answer": "...",
  "match_type": "exact_option" |
  ↪  "inferred_option" | "free_form" |
  ↪  "numeric" | "incorrect" | "ambiguous"
}
```

I.1 Discarded Quantities During DLR Generation

We report the rejection rate for each DLR family during the generation and filtering process. A generated augmentation was rejected if it failed the validation criteria, e.g., if the judge determined that the rewritten instance did not preserve the original semantic conditions or gold answer.

J LLM-as-Judge Circularity Risk

Because both rewrite generation and semantic validation use LLMs, our pipeline may inherit shared model biases: a validator may accept rewrites that preserve the generator's interpretation rather than the true task semantics. We mitigate this risk by treating generated rewrites as candidates rather

DLR Family	Rejection Rate (%)
Add Irrelevant Information	5.8
Language Harder	3.4
Language Simpler	3.1
Question Reordering	2.6
Clause Reversal	2.7
Average	**3.52**

Table 7: Rejection rates during DLR generation and validation, reported overall by DLR family.

than ground truth, applying explicit equivalence prompts, routing rejected or ambiguous cases to human review, and auditing accepted pairs with independent annotators. This does not eliminate circularity entirely, so the reported results should be interpreted as robustness under validated model-assisted rewrites rather than under fully human-authored reformulations.

K Failure Pattern Analysis

We find that Phi-4 exhibits the same *qualitative trend* as GPT-5 under prompt perturbations, see table 8. DLR still induce larger behavioral shifts than local EDA-based edits—but the *structure* of the resulting failure redistribution differs. While GPT-5 is representative of frontier models in exhibiting prompt-dependent concentration into specific failure modes, Phi-4 responds to the same perturbations with a more diffuse reallocation of error mass.

As shown in Table 9, Phi-4's structured error categories (e.g., *Heuristic Short-Circuiting*, *Multiple-Step Collapse*) shift only moderately on average, whereas the dominant effect of both local and DLR is a consistent increase in the residual *Other-misc* category (up to $\Delta{+}6.5\%$ under DLR). This pattern is also evident in the raw per-augmentation analysis (Table 8), where individual perturbations induce localized fluctuations without triggering a systematic collapse into a single reasoning pathology.

Importantly, *Other-misc* does not correspond to a specific identifiable failure mechanism. In most cases, these errors reflect unconstrained outputs, including hallucinated answers or effectively random option selection, where the precise reasoning breakdown is not directly observable from the model's response. Thus, for Phi-4, DLR primarily degrade answer coherence rather than activating a distinct, rhetoric-sensitive failure mode, yielding behaviorally similar but structurally smoother degradation compared to GPT-5.

Table 8 includes the full decomposition to our failure categories of the chosen representative models: GPT-5 and Phi-4.

L LLM Error Analysis Prompt

For reproducibility purposes and future clarity we include the prompt used in error taxonomy analysis:

```
    ## Failure Category Definitions

You are analyzing failures in a model's
↪  reasoning.
For each **incorrect** response, assign
↪  **one** category based on the reasoning
↪  behavior.

### Heuristic Short-Circuiting
The model skips reasoning and jumps directly
↪  to an answer via pattern matching or a
↪  template. Reasoning is extremely short or
↪  absent.

### Premise Rejection
The model refuses the task by claiming the
↪  question is impossible, ill-defined, or
↪  lacks information, often with pedantic
↪  justification.

### Hyper-Granularity Trap
The model over-complicates the problem using
↪  dense jargon or unnecessary sub-mechanisms,
↪  obscuring or replacing the correct logic.

### Multi-step Reasoning Collapse
The model begins with a plausible reasoning
↪  plan but fails during the multi-step
↪  solution process. The failure may involve
↪  skipping an intermediate step, losing
↪  track of earlier constraints, introducing
↪  a contradiction, misapplying a condition,
↪  or following the rewritten structure in a
↪  way that derails the intended reasoning.

### Numerical Calculation Error
The model follows the correct reasoning
↪  strategy but makes an arithmetic,
↪  counting, unit-conversion, comparison, or
↪  numerical computation error. Use this
↪  category only when the plan is
↪  substantively correct and the main failure
↪  is operational rather than conceptual.

### Other (reason_name)
Free-text explanation for failures that do not
↪  fit the above categories. Explain the
↪  failure clearly.

Select the category that best explains **why**
↪  the reasoning failed, not merely that it
↪  failed.

---

# Qualitative Failure Analysis Prompt

You are performing qualitative failure
↪  analysis of an LLM.
```

Augmentation	Heuristic SC	Premise Rej.	Hyper-Gran.	M. S. Collapse	Numeric Calc.	Other-misc
Phi-4						
Baseline	37.70 (–)	2.62 (–)	6.28 (–)	27.23 (–)	6.81 (–)	19.37 (–)
Random Deletion	33.53 (−4.17)	3.07 (+0.45)	5.68 (−0.60)	26.75 (−0.48)	6.16 (−0.65)	24.81 (+5.44)
Random Insertion	34.89 (−2.81)	3.16 (+0.54)	5.28 (−1.00)	26.24 (−0.99)	8.67 (+1.86)	21.76 (+2.39)
Random Swap	33.19 (−4.51)	3.36 (+0.74)	5.22 (−1.06)	28.65 (+1.42)	4.16 (−2.65)	25.42 (+6.05)
Synonym Repl.	34.62 (−3.08)	3.07 (+0.45)	5.61 (−0.67)	26.33 (−0.90)	6.91 (+0.10)	23.46 (+4.09)
back_translation	36.62 (−1.08)	2.98 (+0.36)	6.02 (−0.26)	26.74 (−0.49)	7.21 (+0.40)	20.43 (+1.06)
Add Irrelevant Information	31.06 (−6.64)	1.04 (−1.58)	8.70 (+2.42)	24.84 (−2.39)	6.42 (−0.39)	27.95 (+8.58)
Language Harder	34.65 (−3.05)	1.98 (−0.64)	7.92 (+1.64)	24.75 (−2.48)	5.94 (−0.87)	24.76 (+5.39)
Language Simpler	32.32 (−5.38)	1.01 (−1.61)	4.04 (−2.24)	32.32 (+5.09)	4.04 (−2.77)	26.27 (+6.90)
Question Reordering	41.47 (+3.77)	4.63 (+2.01)	5.26 (−1.02)	15.38 (−11.85)	8.79 (+1.98)	24.47 (+5.10)
Clause Reversal	35.21 (−1.49)	0.48 (−2.14)	5.26 (−1.02)	24.18 (−3.05)	6.59 (−0.22)	28.28 (+8.91)
GPT-5						
Baseline	46.03 (–)	1.36 (–)	2.62 (–)	18.40 (–)	3.99 (–)	27.60 (–)
Random Deletion	52.03 (+6.0)	5.66 (+4.3)	4.32 (+1.7)	14.40 (-4.0)	2.79 (-1.2)	20.80 (-6.8)
Random Insertion	59.23 (+13.2)	0.66 (-0.7)	1.32 (-1.3)	15.84 (-2.6)	1.98 (-2.0)	20.97 (-6.7)
Random Swap	52.34 (+6.3)	4.11 (+2.8)	1.32 (-1.3)	15.21 (-3.2)	2.30 (-1.7)	24.72 (-2.9)
Synonym Repl.	55.64 (+9.6)	2.83 (+1.5)	2.40 (-0.2)	15.41 (-3.0)	3.17 (-0.8)	20.55 (-7.1)
Irrelevant Info	59.26 (+13.2)	0.00 (-1.3)	3.70 (+1.1)	7.41 (-11.0)	0.00 (-4.0)	29.63 (+2.0)
Language Harder	56.25 (+10.2)	**6.25** (+4.9)	0.00 (-2.6)	18.75 (+0.3)	0.00 (-4.0)	18.75 (-8.9)
Language Simpler	**61.90** (+15.9)	0.00 (-1.3)	4.76 (+2.1)	24.24 (+5.8)	**1.01** (-2.9)	**8.09** (-19.5)
Quest. Reorder	42.11 (-3.9)	5.26 (+3.9)	5.26 (+2.6)	**31.87** (+13.5)	3.30 (-0.7)	12.20 (-15.4)
Subord. Reversal	36.84 (-9.2)	0.00 (-1.3)	**5.27 (+2.6)**	30.77 (+12.4)	1.10 (-2.9)	26.03 (-1.6)
back_translation	47.20 (+1.17)	1.55 (+0.19)	2.50 (-0.12)	17.90 (-0.50)	4.20 (+0.21)	26.65 (-0.95)

Table 8: Error distribution per augmentation tag with Δ relative to the model-specific baseline.

Augmentation	Phi-4			GPT-5		
	Baseline	Local	DLR	Baseline	Local	DLR
Heuristic SC	37.70	34.06 (−3.64)	34.94 (−2.76)	46.05	54.81 (+8.76)	51.27 (+5.22)
Premise Rej.	2.62	3.17 (+0.55)	1.83 (−0.79)	1.32	3.31 (+1.99)	2.30 (+0.98)
Hyper-Gran.	6.28	5.45 (−0.83)	6.24 (−0.04)	2.63	2.34 (−0.29)	3.80 (+1.17)
M. S. Collapse	27.23	26.99 (−0.24)	24.29 (−2.94)	18.42	15.21 (−3.21)	22.61 (+4.19)
Numeric Calc.	6.81	6.48 (−0.33)	6.36 (−0.45)	3.95	2.56 (−1.39)	1.08 (−2.87)
Other-misc	19.37	23.86 (+4.49)	26.35 (+6.98)	27.63	21.76 (−5.87)	18.94 (−8.69)

Table 9: Aggregated error distribution for Local (EDA) vs. DLR. Values shown are *Absolute Value (Δ relative to baseline)*.

```
The error must belong to **one** of the
↪  predefined categories above.

## Augmentation Applied
augmentation_tag

## Question
question

## Expected Baseline Response
gold_answer

## Model Response
response

Return **ONLY valid JSON** with the following
↪  fields:

- `error_analysis`: A short explanation (one
↪  concise sentence) describing what went
↪  wrong and why it fits this error category.
↪  If relevant, connect the failure to the
↪  augmentation type.
- `error_category`: One of the predefined
↪  categories above (or `Other`, in which
↪  case specify the reason).

If the answer is **correct**, return empty
↪  strings for all fields and `false` for
↪  `put_in_paper`.
```

M Artificiality of Discourse-Level Rewrites

To address the concern that our DLR may not reflect how users naturally phrase questions, we performed an additional realism analysis comparing our DLR families against naturally occurring reformulations in QA-style conversational datasets, namely WildChat and lmsys-chat-1m. We annotated each sample using GPT-4.1-mini for whether it matches each DLR category. We manually validated >400 randomly sampled annotations and found the annotator to be over 90% accurate. For reproducibility considerations we include the used prompt as below:

```
    You are given a user question or
    ↪  instruction from a real conversational
    ↪  dataset.
Your task is to determine whether the message
↪  contains any of the following
↪  Discourse-Level Rewrite (DLR) patterns.
```

```
A DLR pattern refers to a discourse-level
↪  property of how the question is phrased,
↪  organized, or contextualized.
The task is multi-label: more than one
↪  category may apply to the same message.

Classify the message using the following
↪  categories:

1. Irrelevant Information:
The message contains extra details, background
↪  information, examples, constraints, or
↪  context that are not necessary for
↪  answering the main question.

2. Language Harder:
The message uses more complex, formal,
↪  technical, indirect, or syntactically
↪  dense phrasing than necessary. This
↪  includes long clauses, abstract wording,
↪  or unnecessarily difficult formulation.

3. Language Simpler:
The message uses simplified, informal,
↪  underspecified, conversational, or
↪  compressed language. This includes short
↪  phrasing, casual wording, or simplified
↪  restatement of a more complex question.

4. Question Reordering:
The message presents information in a
↪  non-standard order, such as giving
↪  constraints before the main question,
↪  placing the main request at the end, or
↪  rearranging the usual order of problem
↪  details.

5. Clause Reversal:
The message contains reversed or contrastive
↪  clause structure, such as ``if not X, then
↪  Y'', ``instead of X, do Y'', ``unless X'',
↪  ``rather than X'', or other formulations
↪  where the logical relation is expressed
↪  through reversal, negation, or contrast.

For each category, answer true if the pattern
↪  is present and false otherwise.
Be conservative: mark a category as true only
↪  when the pattern is clearly present.

Return only valid JSON in the following
↪  format:

{
  "irrelevant_information": true/false,
  "language_harder": true/false,
  "language_simpler": true/false,
  "question_reordering": true/false,
  "clause_reversal": true/false,
  "brief_rationale": "One short sentence
  ↪  explaining the decision."
}

Message:
[USER MESSAGE]
```

The resulting category frequencies seen in Table 10 show that the DLR patterns are not purely artificial, but occur frequently in real user-facing conversational data:

DLR Category	Fraction
Irrelevant Information	0.384
Language Harder	0.623
Language Simpler	0.257
Question Reordering	0.1685
Clause Reversal	0.2325

Table 10: Fraction of naturally occurring conversational samples matching each DLR category.

We further evaluate whether the augmented questions introduce an artificial fluency shift by measuring perplexity. Following common practice, we compute perplexity using the `Transformers` library with a GPT-2 model fine-tuned via SFT on a training split of the natural conversational datasets mentioned earlier. We then evaluate perplexity on both the original unaltered questions and the DLR questions.

Perplexity is defined as:

$$\mathrm{PPL}(x_{1:T}) = \exp\left(-\frac{1}{T}\sum_{t=1}^{T}\log P_\theta(x_t \mid x_{<t})\right),$$

where $x_{1:T} = (x_1, \ldots, x_T)$ is the tokenized input sequence of length T, and $P_\theta(x_t \mid x_{<t})$ is the probability assigned by the model with parameters θ to token x_t given its preceding context.

Split	PPL ↓
Unaltered questions (test set)	15.01 ± 2.20
DLR-augmented questions	16.06 ± 1.50
PPL per DLR type	
Add irrelevant information	18.5179 ± 1.005
Baseline	15.47 ± 1.72
Language harder	18.91 ± 1.53
Language simpler	13.43 ± 1.30
Question reordering	14.67 ± 1.70
Clause reversal	14.69 ± 1.60

Table 11: Perplexity comparison between original questions and DLR questions. Lower is more fluent.

As seen in Table 11 the augmented questions have perplexity comparable to the original test split. Together with the prevalence analysis on WildChat and lmsys-chat-1m, this suggests that our DLR categories capture realistic reformulation patterns observed in natural user interactions, rather than introducing purely synthetic or unnatural artifacts.

N Full Results

We share the full model performance on each independent augmentation type and each dataset for further research. This can be seen in Table 12.

O Failure Mode Examples

This appendix provides concrete examples for each failure category used in our manual error analysis. For each category, we briefly summarize the defining behavior and reference the corresponding figure containing representative failure cases.

Multi-step Reasoning Collapse. As illustrated in Fig. 1, *Multi-step Reasoning Collapse* occurs when the model initiates a plausible reasoning trajectory but degrades mid-way. Typical symptoms include missing steps, inconsistent intermediate conclusions, or matching the user's simplified undertone at the expense of correctness. These failures often arise under DLR that subtly change discourse structure, leading the model to abandon or shortcut its original reasoning path.

Hyper-Granularity Trap. Fig. 2 shows examples of the *Hyper-Granularity Trap*, where the model over-focuses on unnecessary distinctions or introduces spurious sub-structures. Instead of solving the intended problem directly, the model re-segments concepts, double-counts categories, or applies irrelevant combinatorial operations, ultimately producing incorrect answers despite otherwise valid reasoning primitives.

Premise Rejection. In *Premise Rejection* (Fig. 3), the model implicitly challenges or alters the problem setup introduced by the question. Added contextual details trigger reinterpretations that cause the model to discard the original premise, reject all answer options, or solve a different problem than the one posed.

Heuristic Short-Circuiting. Fig. 4 illustrates *Heuristic Short-Circuiting*, where the model relies on surface-level cues or prior expectations instead of the provided evidence. In these cases, discourse-level rewrites remove or obscure a critical fact, and the model fills the gap with plausible but unsupported assumptions, leading to confident yet incorrect answers.

P Mitigation Experiment Setup

To evaluate whether the observed degradation can be mitigated, we add two fine-tuning baselines. We split the 110K original questions into 70%/15%/15% train/validation/test splits. For each training question, we generate five semantically equivalent DLR variants after the split, ensuring that validation and test examples remain unchanged. In the *SFT on all DLRs* setting, each original training example is therefore augmented with five DLR variants, yielding six training instances per example. Models are fine-tuned using standard supervised fine-tuning with the causal language modeling objective. For mitigation, we use finetunable models for which supervised fine-tuning access is available. We also compare against an adversarial fine-tuning baseline that selects difficult training examples using uncertainty-based hard-example mining (See App. Q), DLR-based augmentation consistently improves performance across all model–dataset pairs and outperforms both raw SFT and adversarial fine-tuning, indicating that targeted DLR provide an effective mitigation for this failure mode.

Q Full Mitigation Results

Table 13 reports the complete mitigation results by model and dataset. The compact table in the main paper shows only part of these results across finetuning methods, while this appendix Table 13 preserves the full per-dataset comparison.

We compare five settings. *Base Model Performance* evaluates the original pretrained model on the unmodified test set. *Base Model on DLR Dataset* evaluates the same model on the discourse-level rewrite (DLR) version of the test set, measuring the robustness drop induced by answer-preserving discourse-level changes. *SFT on Raw Dataset* fine-tunes the model only on the original training examples. *SFT on all DLRs* fine-tunes the model on the original training examples together with all generated DLR variants. Since each training question is paired with five DLR variants, this setting exposes the model to both the original formulation and multiple semantically equivalent discourse-level reformulations. Finally, *Adv. Fine-tuning on Raw* is an adversarial fine-tuning baseline that selects difficult examples from the raw training data, without adding DLR variants.

R Additional Technical Details

R.1 Artifacts and Intended Use

This work uses existing QA benchmarks and creates derived discourse-level rewrite (DLR) variants.

Model	Augmentation	ARC-C	Big-Math-RL-Verified	DROP	GPQA	MMLU-Pro
DeepseekR1	back_translation	80.20	66.16	82.03	66.96	84.70
	eda_rd	87.85	69.23	86.64	68.20	84.25
	eda_ri	78.70	69.51	87.09	72.49	85.15
	eda_rs	84.82	68.61	88.00	72.49	83.80
	eda_sr	85.00	69.20	88.30	69.82	84.70
	Add Irrelevant Information	78.43	**61.56**	77.78	64.71	78.50
	Language Harder	**68.89**	61.78	**69.67**	**62.70**	**74.00**
	Language Simpler	75.93	67.54	73.67	65.90	77.50
	Question Reordering	77.59	64.55	88.00	64.53	77.00
	Clause Reversal	84.62	62.94	80.00	63.44	78.00
Aggregate rows						
	baseline	84.91	68.25	93.55	71.38	85.71
	DLR	77.09	63.67	79.82	64.26	77.00
	local	83.31	68.54	86.41	69.99	84.52
KimiK2thinking	back_translation	82.91	80.99	84.38	69.18	85.61
	eda_rd	86.50	82.99	87.14	72.90	84.98
	eda_ri	83.89	79.53	90.00	72.80	84.56
	eda_rs	84.57	82.60	88.38	73.08	87.16
	eda_sr	82.86	82.00	90.10	73.31	83.91
	Add Irrelevant Information	86.21	80.43	80.88	62.91	**82.02**
	Language Harder	82.65	82.81	77.55	**61.95**	83.21
	Language Simpler	**78.21**	**71.83**	85.00	67.79	85.53
	Question Reordering	82.10	86.73	**76.92**	68.02	84.31
	Clause Reversal	80.00	81.82	83.33	67.20	84.46
Aggregate rows						
	baseline	86.09	85.00	93.20	75.24	86.46
	DLR	81.83	80.72	80.74	65.57	83.91
	local	84.15	81.62	88.00	72.25	85.24
Llama3.370B	back_translation	94.18	77.70	84.33	45.56	68.07
	eda_rd	95.68	77.50	88.00	49.60	69.32
	eda_ri	96.49	79.23	87.14	48.70	68.15
	eda_rs	93.25	79.13	86.50	46.90	67.13
	eda_sr	100.00	82.00	86.14	48.65	69.10
	Add Irrelevant Information	96.00	77.14	78.57	43.32	62.48
	Language Harder	92.50	78.95	**75.76**	**41.62**	**60.71**
	Language Simpler	**86.11**	75.51	85.71	44.51	63.24
	Question Reordering	90.91	82.05	80.00	43.60	62.52
	Clause Reversal	97.62	**68.89**	90.11	42.47	61.61
Aggregate rows						
	baseline	97.18	82.50	90.12	50.48	68.99
	DLR	92.63	76.51	82.03	43.10	62.11
	local	95.92	79.11	86.42	47.88	68.35
Phi4	back_translation	93.71	78.93	68.32	49.35	70.07
	eda_rd	100.00	81.87	75.46	48.10	68.42
	eda_ri	96.25	82.31	74.29	50.63	68.35
	eda_rs	91.43	79.00	76.40	52.43	68.83
	eda_sr	92.80	80.60	75.46	53.94	68.52
	Add Irrelevant Information	92.50	77.42	63.55	49.20	66.62
	Language Harder	**88.75**	78.06	**60.76**	47.03	**64.91**
	Language Simpler	93.55	78.52	64.52	50.29	67.54
	Question Reordering	91.08	**76.16**	60.87	49.42	66.88
	Clause Reversal	96.84	78.15	63.64	**41.85**	64.94
Aggregate rows						
	baseline	96.28	81.15	75.61	54.15	69.18
	DLR	92.54	77.66	62.67	47.56	66.18
	local	94.84	80.54	73.98	50.89	68.84
gpt-5	back_translation	95.26	78.34	83.86	82.12	87.31
	eda_rd	95.76	84.33	88.00	83.20	85.24
	eda_ri	96.31	85.67	90.00	86.24	85.56
	eda_rs	96.67	86.26	93.08	85.34	85.87
	eda_sr	96.34	86.40	91.00	86.72	85.33
	Add Irrelevant Information	96.82	**68.00**	79.80	80.21	83.20
	Language Harder	87.95	74.70	**77.04**	**78.38**	**81.00**
	Language Simpler	96.44	75.62	82.69	81.50	82.40
	Question Reordering	**86.94**	72.02	77.14	80.81	82.10
	Clause Reversal	94.17	73.55	81.25	79.03	81.60
Aggregate rows						
	baseline	97.00	86.10	91.94	87.46	86.42
	DLR	92.26	76.78	82.58	79.99	82.06
	local	96.02	84.20	89.19	84.72	85.86

Table 12: Full model performance per augmentation type per dataset.

Model	Dataset	Base Model Performance	Base Model on DLR Dataset	SFT on Raw Dataset	SFT on all DLRs	Adv. Finetuning on Raw
DeepSeek-V2-Lite	Big-Math	52.7 ± 0.61	48.9 ± 0.70	53.6 ± 0.60	$\mathbf{55.2 \pm 0.63}$	54.4 ± 0.61
	DROP	48.1 ± 1.15	42.0 ± 1.00	49.1 ± 1.22	$\mathbf{51.0 \pm 1.30}$	50.6 ± 1.43
Phi-4	Big-Math	81.15 ± 0.80	77.66 ± 0.90	82.15 ± 0.79	$\mathbf{83.85 \pm 0.78}$	82.85 ± 0.81
	DROP	75.61 ± 0.95	62.67 ± 1.20	74.61 ± 1.02	$\mathbf{76.21 \pm 1.11}$	74.51 ± 1.20
LLaMA-3.3-70B	Big-Math	82.50 ± 0.30	76.51 ± 0.34	82.30 ± 0.28	$\mathbf{84.20 \pm 0.29}$	83.20 ± 0.31
	DROP	90.12 ± 0.22	82.03 ± 0.30	90.02 ± 0.25	$\mathbf{93.12 \pm 0.29}$	92.32 ± 0.33
GPT-4.1-mini	Big-Math	67.1 ± 0.51	62.2 ± 0.60	69.0 ± 0.55	$\mathbf{71.2 \pm 0.56}$	70.0 ± 0.58
	DROP	85.9 ± 0.87	75.0 ± 0.40	85.8 ± 0.94	$\mathbf{87.2 \pm 0.97}$	86.5 ± 1.23

Table 13: Full mitigation results across models, datasets, and fine-tuning settings. Values report accuracy with the corresponding uncertainty. The best result in each model–dataset row is shown in bold. Fine-tuning on all DLR variants yields the strongest performance in every setting, showing that DLR-based augmentation improves robustness beyond both raw supervised fine-tuning and adversarial fine-tuning on raw examples.

Figure 1: M. S. collapse

Explanation: Model matches user's undertone and replies in much simpler terms which leads to a less accurate response.

MMLU-PRO | language_simpler | Hallucinated Constraint

Original Question
What is the minimum time a stretch should be held for?
Options: (A) 2-3 min. (B) 30-50 s. (C) 0-10 s. (D) 1-2 min. (E) 10-30 s. (F) 60 s. (G) 80-90 s. (H) 50-60 s.

Baseline Model Response
... commonly provided by fitness and health experts ...
1. Stretching Goals: ...
2. Common Recommendations: ...
3. Expert Consensus: ...
4. Considering All Options: Given the options provided:
- (A) 2-3 minutes: This is longer than... which could lead to...
5. Selecting the Minimum Effective Duration: ...
Prediction: (E) *(Correct)*

Augmented Question
How long should you hold a stretch at least?
Options: (A) 2-3 min. (B) 30-50 s. (C) 0-10 s. (D) 1-2 min. (E) 10-30 s. (F) 60 s. (G) 80-90 s. (H) 50-60 s.

Augmented Model Response
Experts recommend holding a stretch for at least 30 seconds... up to 50 seconds. (C) is too short. (B) 30-50 seconds is the best choice.
Prediction: (B) *(Incorrect)*

Explanation: reordering the question made the question start with the outcome then propose the options. The model answered in the same order where it discusses air-pollution then finding from its own answer the chosen answer. Doing this it missed in reasoning over other sources that could have changed its answer (as in the baseline case).

ARC-C | question_reordering | Prediction Flip

Original Question
What is the greatest contributor to air pollution in the United States?
Options: (A) factories (B) automobiles (C) power plants (D) burning waste

Baseline Model Response
To determine the greatest contributor...
- Automobiles (B): ...transportation has been one of the largest sources...
- Power Plants (C): ...remain a key source historically.
According to the EPA... transportation is the largest contributor... **Answer: B**

Augmented Question
Of the air pollution in the United States, what is its largest source of the ones below?
Options: (A) factories (B) automobiles (C) power plants (D) burning waste

Augmented Model Response
Air pollution in the United States comes from several sources... Among these, **factories are the largest contributor**, ... **(which) include power plants** ... remain the largest contribute ...
Answer: (A)

Figure 2: Hyper-Granularity Trap

Explanation: The model overcomplicated a simple answer possibly due to match user's jargon. Correct: humans have 12 pairs of ribs (7 true + 5 false, including 2 floating) = 24 total ribs. Instead, the model incorrectly re-categorized the ribs, introduced confusing sub-counts, and double-counted pairs as "7+7+3+3" to get 20 pairs (40 ribs). Its mistake is not a lack of reasoning but an unnecessary, incorrect re-segmentation and double-counting of categories, i.e. over-complicating the problem and producing erroneous internal bookkeeping.

MMLU-PRO | language_harder | Prediction Flip

Original Question
How many ribs are there in the human body?
Options: (A) 42 (B) 24 (C) 18 (D) 36
(E) 40 (F) 28 (G) 22 (H) 30

Baseline Model Response
To determine how many ribs... - Typical Rib Pair Count: The human ribcage is normally composed of 12 pairs (one left, one right).
- Calculation: 12 pairs $\times$ 2 = 24 ribs.
Answer: (B) 24

Augmented Question
What is the total number of rib bones present within the anatomical structure of the human organism?
Options: (A) Forty-two (B) Twenty-four
(C) Eighteen (D) Thirty-six
(E) Forty (F) Twenty-eight
(G) Twenty-two (H) Thirty

Augmented Model Response
...The human rib cage typically consists of:
1. True Ribs: 7 pairs...
2. False Ribs: ... 3 pairs ...
Total: The sum of true and false ribs gives us $7 + 7 + 3 + 3 = 20$ pairs of ribs. Since each pair consists of a left and a right rib, the total number of individual rib bones is 20 pairs x 2 = 40 ribs.
Answer: (E) 40

Explanation: The model over-counted by multiplying the correct answer (540) by 6! = 720 to get 388,800. This extra factor likely came from incorrectly applying the "unique cars" constraint twice. Once at the start when setting up the problem and again at the end by permuting cars within lanes, even though their order is fixed by arrival sequence. The model added an irrelevant permutation step instead of recognizing the order was already determined.

BIG-MATH | irrelevant_info | Logic Error

Original Question
Six cars pull up to a red light, one at a time. At the light, there are three lanes... How many ways can the cars stack up so that all three lanes are occupied?
Note that... the cars are distinguishable, but pull up to the intersection in a fixed order.

Baseline Model Response
To ensure all three lanes are used, we use inclusion-exclusion on the lane choices (3^6 total):
Total sequences: $3^6 = 729$
Subtract missing 1 lane ...
Add back missing 2 lanes ...
Calculation: $729 - 192 + 3 = 540$.
Answer: 540

Augmented Question
In a bustling city intersection where the aroma of street food wafts through the air, six **distinct** cars approach a red traffic light sequentially. The intersection features three lanes...
The task is to determine the number of possible arrangements... such that each lane is occupied by at least one car... Note that each of the cars is **unique**.

Augmented Model Response
First, we calculate the number of surjective assignments of 6 cars to 3 lanes, which is 540 (using inclusion-exclusion). However, since the cars are distinct and the arrival sequence matters, we must account for the permutations of the 6 cars.

$$540 \times \mathbf{6!} = 540 \times 720 = 388,800$$

Answer: 388,800

Figure 3: Premise Rejection

Explanation: The model incorrectly focused on the added detail "works hard to chop down," treating the mechanical energy of chopping as part of the answer sequence. It failed to recognize that chopping is just the method of harvesting, not part of the photosynthesis-to-burning energy transformation being asked about. This caused it to reject all options as incorrect.

ARC-C | add_information | Refusal / Logic Error

Original Question
A person cuts down a living oak tree. The person burns the wood... to boil water. Which sequence correctly orders the energy transformations...?
Options: (A) light energy → chemical energy → thermal energy (B) thermal → chemical → light (C) chemical → mechanical → electrical (D) electrical → mechanical → chemical

Baseline Model Response
To determine the correct sequence... Photosynthesis: A living oak tree absorbs light... converted into chemical energy (glucose). (Light → Chemical) ...
Burning: Stored chemical energy is released as thermal energy. (Chemical → Thermal) The sequence is light → chemical → thermal.
Answer: (A)

Augmented Question
A person works hard to chops down a living oak tree and ... to heat water. Which sequence correctly describes the energy changes...?
Options:
(same as above)

Augmented Model Response
To solve this... Chemical Energy ... Chopping the tree uses **a lot of mechanical energy**... Thermal Energy: Burning converts chemical to thermal.
Answer: None of the options match

Figure 4: Heuristic Short-Circuiting

Explanation: ID: 2751

DROP | irrelevant_info | Hallucination / Missing Info

Original Question
The First Division... proceeded northward... A total of 10,000 Korean troops guarded the city against the **30,000 advancing Japanese**...
How many more Japanese troops were there than Korean troops?

Baseline Model Response
To determine how many more Japanese troops there were...
Japanese: 30,000 ... Korean: 10,000 ...
Calculation: $30,000 - 10,000 = 20,000$. **Answer: 20,000**

Augmented Question
In the historical context of the late 16th century... The city was defended by 10,000 Korean troops... [The text describes the First and Third Divisions having 10,000 and 20,000 respectively joining forces but **omits the specific number 30,000**].
Considering the troop numbers, how many more Japanese troops were there than Korean troops?

Augmented Model Response
To determine how many more Japanese troops... Korean Troops: 10,000 (explicit). Japanese Troops: The text references the First and Third Divisions but historically, each was around 10,000 ... Estimated total: 20,000. Calculation: $20,000 - 10,000 = 10,000$.
Answer: 10,000

Figure 5: Numerical Calculation Error

Explanation: ID: 1829

DROP | language_harder | Numerical Calculation Error

Original Question
The ages of New Haven residents are distributed as follows: **25.4%** are under 18, **16.4%** are from 18 to 24, **31.2%** are from 25 to 44, **16.7%** are from 45 to 64, and **10.2%** are 65 years of age or older.
What percentage of the population is not 65 years of age or older?

Baseline Model Response
The question asks for the percentage of residents who are not 65 or older. Since **10.2%** are 65 or older, the complement is:
$100 - 10.2 = 89.8$.
Answer: 89.8%

Augmented Question
The demographic composition of New Haven residents is partitioned as follows: **25.4%** are younger than 18, **16.4%** fall between 18 and 24, **31.2%** are between 25 and 44, **16.7%** are between 45 and 64, and **10.2%** belong to the 65-or-older age category.
What percentage of the population does not fall within the 65-or-older category?

Augmented Model Response
The question asks for the percentage of residents who are not 65 or older. I will add all age groups below 65:
$25.4 + 16.4 + 31.2 + 16.7 = 89.7$.
Therefore, the population not in the 65-or-older category is **89.7%**.
Answer: 89.7%

The source benchmarks are ARC-C, BIG-MATH, DROP, GPQA, and MMLU-Pro, which are used only for research evaluation of QA robustness and are allowed under their distribution license. The created artifact consists of DLR-augmented QA examples. Each rewrite is generated from an existing benchmark item and is intended to preserve the original task, semantic constraints, answer space, and gold answer. The artifact is intended for studying robustness to meaning-preserving reformulations, analyzing model failures, and training or evaluating QA models under controlled discourse-level variation.

R.2 Models Used

Model sizes are reported in Table 1 when publicly available. For closed-source models whose parameter counts are not disclosed, we report them as undisclosed.

Closed-model calls were run through Azure OpenAI in Azure AI Foundry using the Azure OpenAI v1 API. We used the deployed model versions available at the time of the experiments: `gpt-5 (2025-08-07)` for GPT-5 evaluation, `o4-mini (2025-04-16)` for DLR generation, and `gpt-4o-mini (2024-07-18)` for semantic-equivalence validation and answer-resolution checks.

All models were used with the default provider configuration associated with the corresponding deployed model version, unless explicitly stated otherwise in the relevant appendix. The total API budget for closed-model inference, DLR generation, and LLM-as-judge validation was approximately $1,000.

R.3 Annotation Participants

Human annotations were conducted by volunteers from varied backgrounds. Before annotation, volunteers were informed that their judgments would be used for research validation and reported only in aggregate. Participation was voluntary, and proceeding with the annotation task was treated as consent to use their annotations for this purpose. For each task, annotators were given the exact same written instructions used by the corresponding LLM judge. The tasks were text-only and required annotators to judge benchmark questions, model outputs, or rewrite pairs according to the provided prompt. No personal information was collected from annotators.