Reverse Question Answering: Can an LLM Write a Question so Hard (or Bad) that it Can't Answer?

Nishant Balepur[1] **Feng Gu**[1] **Abhilasha Ravichander**[2] **Shi Feng**[3]
Jordan Boyd-Graber[1] **Rachel Rudinger**[1]
[1]University of Maryland [2]University of Washington
[3]George Washington University
{nbalepur, rudinger}@umd.edu jbg@.umiacs.umd.edu

Abstract

Question answering (QA)—giving correct answers to questions—is a popular task, but we test ***reverse* question answering (RQA)**: for an input answer, give a question with that answer. Past work tests QA and RQA separately, but we test them jointly, comparing their difficulty, aiding benchmark design, and checking reasoning consistency. We run 16 LLMs on QA and RQA with trivia questions/answers, revealing: 1) Versus QA, LLMs are much less accurate in RQA for numerical answers, but slightly more accurate in RQA for textual answers; 2) LLMs often answer their own invalid questions from RQA accurately in QA, so RQA errors are not from knowledge gaps alone; 3) RQA errors correlate with question difficulty and inversely correlate with answer frequencies in the Dolma corpus; and 4) LLMs struggle to provide valid multi-hop questions. By finding question and answer types that lead to RQA errors, we suggest improvements for LLM reasoning.[1]

1 Reversing the Question Answering Task

Question answering (QA) is a long-standing task in NLP (Green Jr et al., 1961). For an input question q, QA deduces the correct answer a (Reiter, 1989). More recently, large language models (LLMs) do the reverse—given an answer a, generate a valid question q to which a is the answer—which we call ***reverse* question answering** (RQA).[2] RQA thus can be a part of downstream tasks like exam question generation (Biancini et al., 2024) or search query reformulation (Dang and Croft, 2010).

QA and RQA are often tested separately, but we test them jointly, offering two key benefits. First, it gives insights into open questions on LLM abilities, as some show LLMs excel in generation over comprehension (West et al., 2023, RQA), while others claim verification is easier (Kadavath et al., 2022, QA). Uncovering which task is harder can guide benchmark design (Chen et al., 2024) and inform data collection practices in writing question-answer pairs (§3.1; e.g., if RQA is easy, get answers manually and then generate synthetic questions).

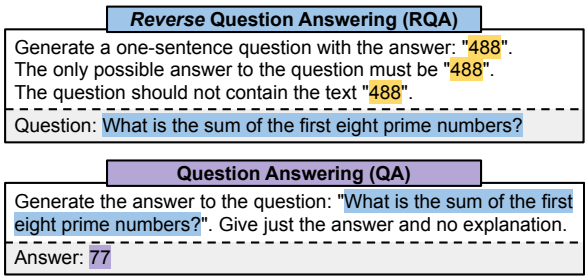

Figure 1: RQA/QA consistency check using GPT-4. The LLM fails to give a valid question with answer 488 (top), but correctly gives the answer 77 for its own question (bottom).

Second, chaining RQA and QA forms a consistency check for LLM reasoning (Liu et al., 2024a). RQA—inferring just one of many valid questions—is *abductive* (Abe, 1998), while QA—inferring an answer from question premises—is *deductive* (Reiter, 1989). Thus, by seeing if $\text{QA}(\text{RQA}(a)) \approx a$, i.e., checking if an LLM can answer its own question from RQA (Fig 1), we can assess LLMs' logical robustness in abduction and deduction (§3.2). This analysis can also help determine if LLMs can reliably self-verify (Pan et al., 2024) in downstream RQA tasks like writing exams (Wang et al., 2018).

To reap these benefits, we test if 16 LLMs can produce 1) questions correctly answered by input entities (RQA); and 2) accurate answers for input questions (QA). We collect 3443 trivia question/answer pairs (Rodriguez et al., 2019), grouped by answer as either numerical or textual entities, forming inputs to evaluate RQA and QA in varied domains.

In numerical domains, LLMs are much less accurate in RQA than QA, especially integers (Fig 1); the accuracy difference when LLMs do these tasks **exceed 0.80** for Command-R and LLaMA-3 (§3.1).

[1]Code and data available at https://github.com/nbalepur/Reverse-QA

[2]This definition differs from question generation (Zhang et al., 2021), which grounds the answer to an input context.

Answer Type	Description	Example Question	Example Answer	Count
(1) Number	Integers in $[100, 1000)$	What is 26 times 4?	104	900
(2) Number+Text	Integers with a text entity	When did Pope Hormisdas die?	523 AD	743
(3) Easy Fact	Well-known factual entity	Who is the artist that painted Starry Night?	Vincent van Gogh	900
(4) Hard Fact	Obscure factual entity	What is the final painting by Paolo Uccello?	The Hunt in the Forest	900

Table 1: Description of our collected dataset for Question Answering and Reverse Questioning Answering tasks.

Interestingly, in textual domains, the trend reverses, so LLMs are not consistently better generators or validators (Li et al., 2024a). We then design a consistency check (§3.2) to see if LLMs can answer their own RQA questions; numerical RQA failures are not solely due to knowledge gaps, as LLMs often **answer their own invalid questions correctly** in QA (33% of cases for Claude-Opus). We then study questions from RQA (§3.3, §3.4) and find errors occur when LLMs give overly-complex, multi-step questions, giving insights into strategies—like complexity bias mitigation in preference data and calibrating models using difficulty scores—to improve LLM RQA reliability. Our contributions are:

1. We use Reverse Question Answering (RQA) to test if LLMs can provide accurate questions for input answers using abductive reasoning.

2. We reveal many LLMs have a surprising weakness in RQA on numerical entities, struggling on input answers with lower pretraining token counts and when creating multi-hop questions.

3. We design a consistency check between RQA and QA, showing LLMs answer their own invalid questions from RQA correctly via QA.

2 Experimental Setup

We evaluate LLM abilities in question answering (**QA**) and *reverse* question answering (**RQA**):

- **QA**$(q) \rightarrow \hat{a}$: Given a question q with a single answer a, the LLM produces an answer $\hat{a}$ for q. QA succeeds if a matches $\hat{a}$ semantically. For example, given the input "What is the name of the polygon with three sides?" for q, an LLM using QA should give an $\hat{a}$ that matches "triangle" for a. This typical QA setup tests *deduction*, since the model must reason to the correct answer of a based on the premises in q.

- **RQA**$(a) \rightarrow \hat{q}$: Given an input answer a, the LLM must produce a question $\hat{q}$. RQA succeeds if the correct answer to $\hat{q}$ is a (verified via oracle, §2.3). For example, given the input "triangle" for a, an LLM using RQA could succeed with $\hat{q}$ as "In eight-ball pool, what shape is used to rack the balls?". RQA tests *abduction*, as the model must reason toward one of the many valid questions with the answer a.

This section describes the datasets (§2.1), models (§2.2), and metrics (§2.3) used for RQA and QA.

2.1 Dataset Collection

We study question/answer pairs (q, a) in four domains for QA and RQA inputs, based on a's answer type (Table 1). We group them as numerical (Number, Number+Text) or textual (Easy Fact, Hard Fact), providing varied domains for testing.

When a is a Number, q is a random, one-step math operation (what is 118+211?). Other types are from QANTA (Rodriguez et al., 2019), an expert-curated dataset of multi-sentence trivia QA pairs. For Easy and Hard Facts, a is the answer to sampled QANTA questions, with the last sentence[3] as q. We use middle school questions for Easy Facts and college questions for Hard Facts. We obtain Number+Text answers a in QANTA by finding numbers in full questions via regex and q from the sentence a appears in. One PhD student checks all QA pairs to ensure they are accurate (details in Appendix A.1).

2.2 Models

We evaluate 16 LLMs: GPT (Achiam et al., 2023, 3.5, 4, 4o), Command R (Cohere, 2023, Command-R, Command-R+), Claude (Anthropic, 2023, Sonnet, Haiku, Opus), LLaMA-3 Instruct (Dubey et al., 2024, 8B, 70B), Yi-1.5 Chat (Young et al., 2024, 6B, 9B, 34B), and Mistral Instruct (Jiang et al., 2024, 7B, 8x7B, 8x22B). All LLMs use temperature 0. We list all parameters in Appendix A.3.

The QA and RQA prompts are zero-shot, since few-shot exemplars test inductive reasoning, not deduction/abduction (Liu et al., 2024a) in QA/RQA.

[3] QANTA questions are paragraph-long and describe a single answer. Sentences in paragraphs are ordered in decreasing difficulty, so we use the last one, forming the easiest question.

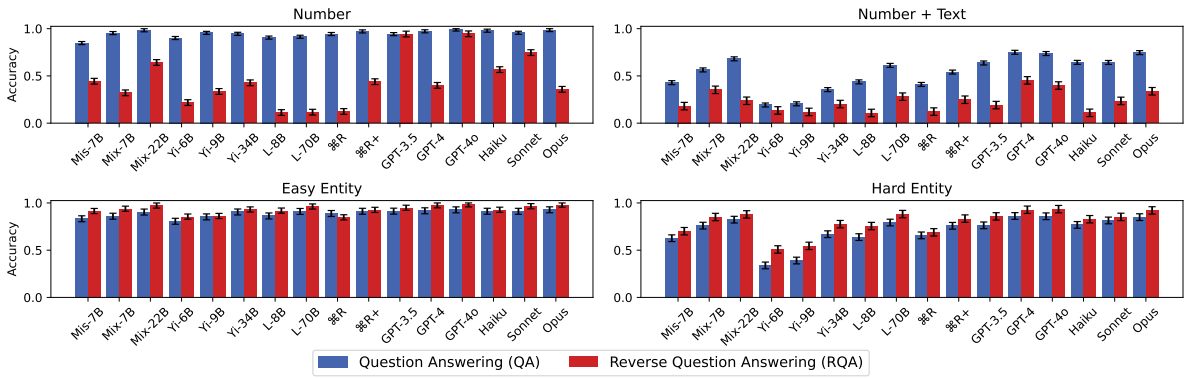

Figure 2: LLM RQA (**blue**) and QA (**red**) accuracy with 95% CIs for metric error rate. LLMs are much weaker in *abductive* RQA in numerical settings (Number/Number+Text), but in text settings (Easy/Hard Entity), *deductive* QA is slightly weaker.

Exemplars also do not improve LLM RQA accuracy (Appendix A.6). Prompts follow the same template as Figure 1 with format rules to parse outputs (Liu et al., 2024c). Two NLP graduate students write the prompts, with all design steps in Appendix A.2.

2.3 Evaluation Metrics

To compute QA accuracy, two graduate students annotate if 1280 LLM QA answers $\hat{a}$ for a question q match its true answer a (20 per answer type/model).[4] We test seven metrics (Li et al., 2024b) that evaluate if $\hat{a}$ and a are equivalent. We select DSPy-optimized (Khattab et al., 2024) GPT-4o for easy/hard entities and a rule-based method for numerical entities, since these methods had the highest agreement with humans (94% on average).

For RQA accuracy, students annotate if the answer to 1280 questions $\hat{q}$ from RQA is a (20 per answer type/model), following rules from Li et al. (2024b). We use DSPy-optimized GPT-4o as an oracle ($\text{VERIFY}^*(\hat{q}, a)$) to assess if a answers $\hat{q}$, which has high (90%) human agreement. Metric agreement is high but imperfect, so we also show QA/RQA accuracy using our 1280 annotations in Figure 6, which has the same trend as our metrics.

3 Evaluation of QA and RQA

Having designed our tasks (§2), this section tests LLMs abilities in QA and RQA. LLMs struggle in RQA on numerical entities (§3.1) but surprisingly can often detect their own errors (§3.2). We study the types of entities that lead to RQA errors (§3.3) and qualitatively analyze differences between accurate and inaccurate questions from RQA (§3.4).

[4]The 1280 total annotations are derived from 16 LLMs, 4 splits, and 20 annotations on each LLM/split combination.

3.1 LLMs Struggle with Numerical RQA

We first see if RQA (**red**, no stripe) or QA (**blue**, striped) is consistently harder for LLMs (Figure 2). In numerical domains (Number, Number+Text), LLMs are much more accurate in QA versus RQA, revealing a clear abduction weakness. Interestingly, in text domains (Easy, Hard), the trend reverses—RQA slightly beats QA 31/32 times. Thus, LLMs cannot be categorized as always stronger in generation or validation (West et al., 2023; Li et al., 2024a): their abilities are domain-specific. If users (e.g. teachers) want to write question-answer pairs with LLMs, we advise manually writing questions for numerical pairs and answers for text pairs, and using LLMs to generate the counterparts, given their strengths in numerical QA and textual RQA.

The Numbers domain has the largest QA/RQA accuracy gaps, over 0.8 for LLaMA and Command-R. Some view LLMs as strong math reasoners, but they excel just in deductive QA tasks, as QA is the main testbed for math abilities (Ahn et al., 2024). In contrast, abduction in textual domains appears in instruction-tuning datasets with queries like "Tell me about Germany". Thus, researchers should design more *abductive* math benchmarks, like RQA, to holistically evaluate LLM math capabilities.

3.2 QA Can Self-Verify Numerical RQA

We chain RQA and QA for consistency, i.e., see if $\text{QA}(\text{RQA}(a)) \approx a$ (Figure 1). If the check fails, the RQA question $\hat{q}$ is invalid, the LLM fails to answer its own valid $\hat{q}$, or both failures occur. We discuss how we disentangle these cases below.

LLMs give: 1) a question $\hat{q}$ with answer a; and 2) an answer $\hat{a}$ to their own $\hat{q}$ without using a. We find three yes/no judgments via our metrics (§2.3): a) does a answer $\hat{q}$ (RQA succeeds); b) does $\hat{a}$ answer

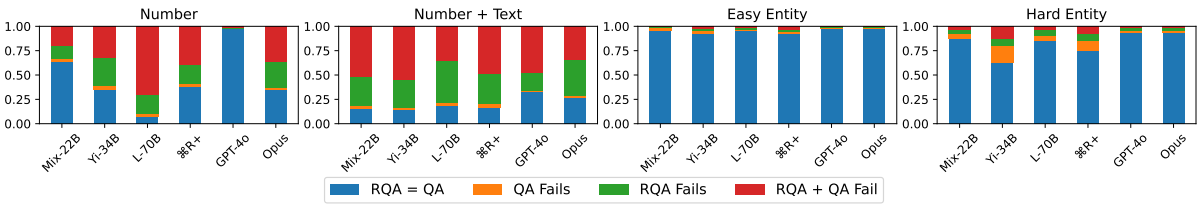

Figure 3: Logical consistency of RQA and QA. For Number and Number+Text entities, most LLMs lack consistency (except GPT-4o), with RQA as the main failure point. Otherwise, LLMs are fairly consistent, with QA as the failure point for Hard Factual Entities. We display the strongest LLM from each model family for brevity, with all results shown in Appendix A.7.

$\hat{q}$ (QA succeeds); and c) are a and $\hat{a}$ equivalent? Answers $\mathcal{A} = (\text{a}, \text{b}, \text{c})$ to these judgments form a truth table to diagnose LLM inconsistencies, which in 91% of cases, fall into the four cases of $\mathcal{A}$ below:

1. $(\texttt{y}, \texttt{y}, \texttt{y})$: RQA = QA (consistent).
2. $(\texttt{n}, \texttt{y}, \texttt{n})$: Just RQA fails.
3. $(\texttt{y}, \texttt{n}, \texttt{n})$: Just QA fails.
4. $(\texttt{n}, \texttt{n}, \texttt{n})$: RQA *and* QA fail.

Other rare cases of $\mathcal{A}$ are metric prediction errors or errors in $\hat{q}$ (e.g. ambiguity), which we omit for this analysis. Appendix A.7 shows all cases of $\mathcal{A}$.

LLMs are fairly consistent in textual domains, but often fail the check in numerical domains, except GPT-4o (Figure 3, left). Thus, our LLMs are logically inconsistent in numerical abduction and deduction. In such cases, QA rarely fails alone: either both RQA and QA fail, where the LLM gives an invalid question that it cannot answer, or just RQA fails, where the LLM detects its error. The latter is akin to hallucination snowballing (Zhang et al., 2024)—inaccurate questions are not just due to knowledge gaps, as LLM can answer their invalid question accurately (e.g. 33% of cases for Opus).

For instance, given the answer "127 countries", Opus incorrectly produces the question "How many countries are members of the United Nations that do not have veto power in the UN Security Council?". However, when Opus answers its own question, it knows there are 193 countries in the UN and five of them have veto power,[5] returning the correct answer of "188". Thus, self-verification (Weng et al., 2023) could be a useful way to verify the correctness of responses in numerical RQA tasks.

3.3 Number+Text RQA Errs on Rare Entities

To find when LLMs fail in numerical RQA (§3.1), we test two indicators of RQA error. We first see

[5]At the time of writing this paper.

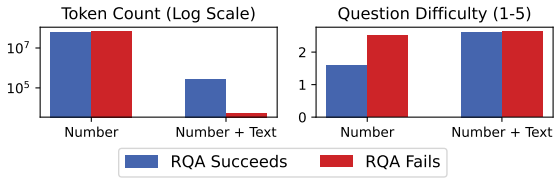

Figure 4: Answer answer token count in Dolma and question difficulty of when RQA succeeds/fails, averaged over LLMs.

how often a appears in the Dolma pretraining corpus (Soldaini et al., 2024) via infini-gram (Liu et al., 2024b), a proxy for the size of all valid questions an LLM must abductively reason over in RQA. Next, as §3.2 hints LLMs may give overly-hard questions (RQA+QA fail), we use the Prometheus LLM (Kim et al., 2024) to get a 1-5 difficulty score for $\hat{q}$. We average metrics pivoted by RQA success/failure on the subset containing human annotations (§2.3).

Number+Text a have lower Dolma token counts when RQA fails (Fig 4), so LLMs struggle to recall long-tail numerical facts (Kandpal et al., 2023). In Numbers, RQA $\hat{q}$ are harder when RQA fails. Thus, calibrating LLMs with desired difficulty (Srivastava and Goodman, 2021) could help designers avoid errors from overly-hard questions in RQA on numbers. Also, difficulty and token count are similar in RQA success/failure for Numbers+Text and Numbers, respectively, so RQA errors depend on answer type, like in QA (Vakulenko et al., 2020).

3.4 LLMs Fail to Write Multi-Step Questions

For qualitative insights into question types $\hat{q}$ from RQA, we analyze 30 $\hat{q}$ when RQA fails/succeeds in strong LLMs with low RQA accuracy (§3.1): L-70B, GPT-4, and Opus. For brevity, we just study the Numbers split, as its similar answers yield $\hat{q}$ with similar patterns, and group $\hat{q}$ as: 1) **Single-Step**: has one math operation; 2) **Multi-Step**: has 2+ math operations; 3) **Fact-Based**: tests factual knowledge; and 4) **Metric Error**: metric misclassification. In Appendix A.8, we analyze more questions $\hat{q}$ in aspects like question novelty, answerability, similarity across models, and memorization.

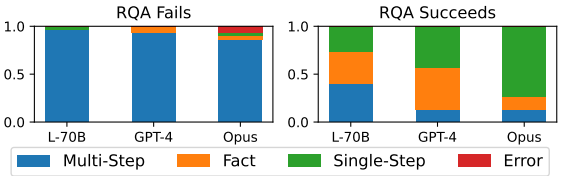

Figure 5: Analysis of Number RQA errors. RQA often fails when the LLM tries to give a complex, multi-step question.

When RQA fails, $\hat{q}$ is often multi-step (Fig 5)—combining math and facts (*how many legs are on a human, cat, & spider?*) or adding primes (Fig 1). In contrast, valid $\hat{q}$ are often single-step (*what is* 19^2*?*) or factual (McCarthy, 1959) (*how many days is a leap year?* for *366*). We believe the errors in multi-step RQA are from preference tuning; users favor a complex output even if it is wrong (Wen et al., 2025). Thus, curbing complexity bias in alignment, or multi-hop QA decoding methods (Zhao et al., 2021), may improve LLMs in multi-step RQA.

4 Related Work

LLM Reasoning: Several works have explored LLM reasoning to improve accuracy (Qiao et al., 2023) or explainability (Si et al., 2024). More recently, works explore if LLMs can execute diverse reasoning strategies, including inductive (Bowen et al., 2024; Yang et al., 2024), deductive (Sanyal et al., 2022; Mondorf and Plank, 2024), and abductive (Zhao et al., 2023; Balepur et al., 2024b) reasoning. However, we are the first to pinpoint abduction abilities via RQA, which differs from traditional question generation setups as we do not have access to an input context (Zhang et al., 2021).

LLM Consistency: LLMs must be consistent to reliably help users (Visani et al., 2022), but LLMs are inconsistent under perturbations like prompt format (Sclar et al., 2024a), entity reversal (Berglund et al., 2024), negation (Ravichander et al., 2022; Balepur et al., 2024a), and ordering (Zheng et al., 2024). Recent work finds inconsistencies in LLM generation and verification in math, QA, style transfer, and coding (Li et al., 2024a; Gu et al., 2024), which we reproduce via an RQA/QA consistency check. Deb et al. (2023) and Yu et al. (2024) similarly compare LLMs in forwards (QA) and backwards (filling question blanks for an answer) reasoning in math. While Deb et al. (2023) claim backwards reasoning is abductive, we argue it is deductive as there is just one answer; we more aptly test abduction/deduction consistency via RQA/QA.

5 Conclusion

We test LLM RQA and QA abilities. LLMs have notably low accuracy in numerical RQA which is not just due to knowledge gaps, as models can often answer their own invalid questions correctly. These weaknesses can be excised in future benchmarks to more holistically evaluate LLM numerical abductive reasoning and math capabilities. To reduce inaccuracies in numerical RQA, often from generating overly-complex questions, we suggest calibrating models using difficulty scores, collecting user preferences that control for complexity bias, and adapting prior multi-hop QA methods—key steps for reliable LLM reasoning in downstream tasks.

6 Limitations

LLMs are sensitive to prompt formats (Sclar et al., 2024b), so varying prompts could impact LLM accuracy in RQA and QA. To ensure our prompts are reliable, we followed best practices (Schulhoff et al., 2024) and kept refining prompts as LLM errors surfaced; the full prompt engineering process is documented in Appendix A.2. Our final prompts will be released and are considered very reasonable implementations of RQA and QA. Further, in Appendix A.6, we test if common prompt engineering strategies (few-shot exemplars, chain-of-thought) can alleviate the low numerical RQA accuracy of GPT-4 but find minimal benefits, suggesting that accuracy gaps between QA and RQA cannot be attributed to prompt formatting alone.

7 Ethical Considerations

RQA uses abduction, a core reasoning strategy that aims to arrive at a plausible explanation given a set of facts. However, our current findings suggest that LLM abductive reasoning in numerical settings is highly unreliable. We advise practitioners to take caution when using LLMs to reason via numerical abduction in downstream tasks, including designing math exam questions, explaining financial forecasts, proposing economic policies, or diagnosing medical patients from numerical data.

Acknowledgements

We would like to thank the CLIP lab at University of Maryland and our external collaborators for their feedback, including Neha Srikanth, Haozhe An, Yu Hou, Abhilasha Sancheti, Connor Baumler, Seraphina Goldfarb-Tarrant, Aidan Peppin, Jack

Wang, and Vishakh Padmakumar. This material is based upon work supported by the National Science Foundation under Grant No. IIS-2403436 (Boyd-Graber), IIS-2339746 (Rudinger), DMS-2134012 (Ravichander), and DGE-2236417 (Balepur). Any opinions, findings, and conclusions or recommendations expressed in this material are those of the author(s) and do not necessarily reflect the views of the National Science Foundation. Cloud computing resources were made possible by a gift from Adobe Research. Access to Command-R was made possible with a Cohere for AI Research Grant.

References

Akinori Abe. 1998. Applications of abduction. In *Proc. of ECAI98 Workshop on Abduction and Induction in AI*, pages 12–19. Citeseer.

Josh Achiam, Steven Adler, Sandhini Agarwal, Lama Ahmad, Ilge Akkaya, Florencia Leoni Aleman, Diogo Almeida, Janko Altenschmidt, Sam Altman, Shyamal Anadkat, et al. 2023. GPT-4 technical report. *arXiv preprint arXiv:2303.08774*.

Janice Ahn, Rishu Verma, Renze Lou, Di Liu, Rui Zhang, and Wenpeng Yin. 2024. Large language models for mathematical reasoning: Progresses and challenges. In *Proceedings of the 18th Conference of the European Chapter of the Association for Computational Linguistics: Student Research Workshop*, pages 225–237, St. Julian's, Malta. Association for Computational Linguistics.

Anthropic. 2023. Meet claude. https://www.anthropic.com/product. Accessed: 2024-09-10.

Nishant Balepur, Shramay Palta, and Rachel Rudinger. 2024a. It's not easy being wrong: Large language models struggle with process of elimination reasoning. In *Findings of the Association for Computational Linguistics ACL 2024*, pages 10143–10166, Bangkok, Thailand and virtual meeting. Association for Computational Linguistics.

Nishant Balepur, Abhilasha Ravichander, and Rachel Rudinger. 2024b. Artifacts or abduction: How do LLMs answer multiple-choice questions without the question? In *Proceedings of the 62nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 10308–10330, Bangkok, Thailand. Association for Computational Linguistics.

Lukas Berglund, Meg Tong, Maximilian Kaufmann, Mikita Balesni, Asa Cooper Stickland, Tomasz Korbak, and Owain Evans. 2024. The reversal curse: LLMs trained on "a is b" fail to learn "b is a". In *The Twelfth International Conference on Learning Representations*.

Giorgio Biancini, Alessio Ferrato, and Carla Limongelli. 2024. Multiple-choice question generation using large language models: Methodology and educator insights. In *Adjunct Proceedings of the 32nd ACM Conference on User Modeling, Adaptation and Personalization*, pages 584–590.

Chen Bowen, Rune Sætre, and Yusuke Miyao. 2024. A comprehensive evaluation of inductive reasoning capabilities and problem solving in large language models. In *Findings of the Association for Computational Linguistics: EACL 2024*, pages 323–339, St. Julian's, Malta. Association for Computational Linguistics.

Yulong Chen, Yang Liu, Jianhao Yan, Xuefeng Bai, Ming Zhong, Yinghao Yang, Ziyi Yang, Chenguang Zhu, and Yue Zhang. 2024. See what LLMs cannot answer: A self-challenge framework for uncovering LLM weaknesses. In *First Conference on Language Modeling*.

Cohere. 2023. Cohere command. https://cohere.com/command. Accessed: 2024-09-10.

Van Dang and Bruce W. Croft. 2010. Query reformulation using anchor text. In *Proceedings of the Third ACM International Conference on Web Search and Data Mining*, WSDM '10, page 41–50, New York, NY, USA. Association for Computing Machinery.

Aniruddha Deb, Neeva Oza, Sarthak Singla, Dinesh Khandelwal, Dinesh Garg, and Parag Singla. 2023. Fill in the blank: Exploring and enhancing LLM capabilities for backward reasoning in math word problems. *arXiv preprint arXiv:2310.01991*.

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

Shangbin Feng, Weijia Shi, Yike Wang, Wenxuan Ding, Vidhisha Balachandran, and Yulia Tsvetkov. 2024. Don't hallucinate, abstain: Identifying LLM knowledge gaps via multi-LLM collaboration. In *Proceedings of the 62nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 14664–14690, Bangkok, Thailand. Association for Computational Linguistics.

Bert F Green Jr, Alice K Wolf, Carol Chomsky, and Kenneth Laughery. 1961. Baseball: an automatic question-answerer. In *Papers presented at the May 9-11, 1961, western joint IRE-AIEE-ACM computer conference*, pages 219–224.

Alex Gu, Baptiste Roziere, Hugh James Leather, Armando Solar-Lezama, Gabriel Synnaeve, and Sida Wang. 2024. CRUXEval: A benchmark for code reasoning, understanding and execution. In *Proceedings of the 41st International Conference on Machine Learning*, volume 235 of *Proceedings of Machine Learning Research*, pages 16568–16621. PMLR.

Shotaro Ishihara. 2023. Training data extraction from pre-trained language models: A survey. In *Proceedings of the 3rd Workshop on Trustworthy Natural Language Processing (TrustNLP 2023)*, pages 260–275, Toronto, Canada. Association for Computational Linguistics.

Albert Q Jiang, Alexandre Sablayrolles, Antoine Roux, Arthur Mensch, Blanche Savary, Chris Bamford, Devendra Singh Chaplot, Diego de las Casas, Emma Bou Hanna, Florian Bressand, et al. 2024. Mixtral of experts. *arXiv preprint arXiv:2401.04088*.

Saurav Kadavath, Tom Conerly, Amanda Askell, T. J. Henighan, Dawn Drain, Ethan Perez, Nicholas Schiefer, Zachary Dodds, Nova DasSarma, Eli Tran-Johnson, Scott Johnston, Sheer El-Showk, Andy Jones, Nelson Elhage, Tristan Hume, Anna Chen, Yuntao Bai, Sam Bowman, Stanislav Fort, Deep Ganguli, Danny Hernandez, Josh Jacobson, John Kernion, Shauna Kravec, Liane Lovitt, Kamal Ndousse, Catherine Olsson, Sam Ringer, Dario Amodei, Tom B. Brown, Jack Clark, Nicholas Joseph, Benjamin Mann, Sam McCandlish, Christopher Olah, and Jared Kaplan. 2022. Language models (mostly) know what they know. *ArXiv*, abs/2207.05221.

Nikhil Kandpal, Haikang Deng, Adam Roberts, Eric Wallace, and Colin Raffel. 2023. Large language models struggle to learn long-tail knowledge. In *International Conference on Machine Learning*, pages 15696–15707. PMLR.

Omar Khattab, Arnav Singhvi, Paridhi Maheshwari, Zhiyuan Zhang, Keshav Santhanam, Sri Vardhamanan A, Saiful Haq, Ashutosh Sharma, Thomas T. Joshi, Hanna Moazam, Heather Miller, Matei Zaharia, and Christopher Potts. 2024. DSPy: Compiling declarative language model calls into state-of-the-art pipelines. In *The Twelfth International Conference on Learning Representations*.

Gangwoo Kim, Sungdong Kim, Byeongguk Jeon, Joonsuk Park, and Jaewoo Kang. 2023a. Tree of clarifications: Answering ambiguous questions with retrieval-augmented large language models. In *Proceedings of the 2023 Conference on Empirical Methods in Natural Language Processing*, pages 996–1009, Singapore. Association for Computational Linguistics.

Najoung Kim, Phu Mon Htut, Samuel R. Bowman, and Jackson Petty. 2023b. $(QA)^2$: Question answering with questionable assumptions. In *Proceedings of the 61st Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 8466–8487, Toronto, Canada. Association for Computational Linguistics.

Seungone Kim, Jamin Shin, Yejin Cho, Joel Jang, Shayne Longpre, Hwaran Lee, Sangdoo Yun, Seongjin Shin, Sungdong Kim, James Thorne, and Minjoon Seo. 2024. Prometheus: Inducing fine-grained evaluation capability in language models. In *The Twelfth International Conference on Learning Representations*.

Xiang Lisa Li, Vaishnavi Shrivastava, Siyan Li, Tatsunori Hashimoto, and Percy Liang. 2024a. Benchmarking and improving generator-validator consistency of language models. In *The Twelfth International Conference on Learning Representations*.

Zongxia Li, Ishani Mondal, Yijun Liang, Huy Nghiem, and Jordan Lee Boyd-Graber. 2024b. Pedants: Cheap but effective and interpretable answer equivalence.

Emmy Liu, Graham Neubig, and Jacob Andreas. 2024a. An incomplete loop: Instruction inference, instruction following, and in-context learning in language models. In *First Conference on Language Modeling*.

Jiacheng Liu, Sewon Min, Luke Zettlemoyer, Yejin Choi, and Hannaneh Hajishirzi. 2024b. Infini-gram: Scaling unbounded n-gram language models to a trillion tokens. In *First Conference on Language Modeling*.

Michael Xieyang Liu, Frederick Liu, Alexander J Fiannaca, Terry Koo, Lucas Dixon, Michael Terry, and Carrie J Cai. 2024c. "We need structured output": Towards user-centered constraints on large language model output. In *Extended Abstracts of the CHI Conference on Human Factors in Computing Systems*, pages 1–9.

John McCarthy. 1959. Programs with common sense.

William Merrill, Noah A. Smith, and Yanai Elazar. 2024. Evaluating n-gram novelty of language models using rusty-DAWG. In *Proceedings of the 2024 Conference on Empirical Methods in Natural Language Processing*, pages 14459–14473, Miami, Florida, USA. Association for Computational Linguistics.

Sewon Min, Julian Michael, Hannaneh Hajishirzi, and Luke Zettlemoyer. 2020. AmbigQA: Answering ambiguous open-domain questions. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 5783–5797, Online. Association for Computational Linguistics.

Seyed Iman Mirzadeh, Keivan Alizadeh, Hooman Shahrokhi, Oncel Tuzel, Samy Bengio, and Mehrdad Farajtabar. 2025. GSM-symbolic: Understanding the limitations of mathematical reasoning in large language models. In *The Thirteenth International Conference on Learning Representations*.

Philipp Mondorf and Barbara Plank. 2024. Comparing inferential strategies of humans and large language models in deductive reasoning. In *Proceedings of the 62nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 9370–9402, Bangkok, Thailand. Association for Computational Linguistics.

Liangming Pan, Michael Saxon, Wenda Xu, Deepak Nathani, Xinyi Wang, and William Yang Wang. 2024. Automatically correcting large language models: Surveying the landscape of diverse automated correction strategies. *Transactions of the Association for Computational Linguistics*, 12:484–506.

Shuofei Qiao, Yixin Ou, Ningyu Zhang, Xiang Chen, Yunzhi Yao, Shumin Deng, Chuanqi Tan, Fei Huang, and Huajun Chen. 2023. Reasoning with language model prompting: A survey. In *Proceedings of the 61st Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 5368–5393, Toronto, Canada. Association for Computational Linguistics.

Abhilasha Ravichander, Matt Gardner, and Ana Marasovic. 2022. CONDAQA: A contrastive reading comprehension dataset for reasoning about negation. In *Proceedings of the 2022 Conference on Empirical Methods in Natural Language Processing*, pages 8729–8755, Abu Dhabi, United Arab Emirates. Association for Computational Linguistics.

Raymond Reiter. 1989. Deductive question-answering on relational data bases. In *Readings in Artificial Intelligence and Databases*, pages 431–443. Elsevier.

Pedro Rodriguez, Shi Feng, Mohit Iyyer, He He, and Jordan L. Boyd-Graber. 2019. Quizbowl: The case for incremental question answering. *ArXiv*, abs/1904.04792.

Anna Rogers, Matt Gardner, and Isabelle Augenstein. 2023. Qa dataset explosion: A taxonomy of NLP resources for question answering and reading comprehension. *ACM Computing Surveys*, 55(10):1–45.

Soumya Sanyal, Harman Singh, and Xiang Ren. 2022. FaiRR: Faithful and robust deductive reasoning over natural language. In *Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 1075–1093, Dublin, Ireland. Association for Computational Linguistics.

Sander Schulhoff, Michael Ilie, Nishant Balepur, Konstantine Kahadze, Amanda Liu, Chenglei Si, Yinheng Li, Aayush Gupta, HyoJung Han, Sevien Schulhoff, et al. 2024. The prompt report: A systematic survey of prompting techniques. *arXiv preprint arXiv:2406.06608*.

Melanie Sclar, Yejin Choi, Yulia Tsvetkov, and Alane Suhr. 2024a. 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*.

Melanie Sclar, Yejin Choi, Yulia Tsvetkov, and Alane Suhr. 2024b. 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*.

Chenglei Si, Navita Goyal, Tongshuang Wu, Chen Zhao, Shi Feng, Hal Daumé Iii, and Jordan Boyd-Graber. 2024. Large language models help humans verify truthfulness – except when they are convincingly wrong. In *Proceedings of the 2024 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies (Volume 1: Long Papers)*, pages 1459–1474, Mexico City, Mexico. Association for Computational Linguistics.

Luca Soldaini, Rodney Kinney, Akshita Bhagia, Dustin Schwenk, David Atkinson, Russell Authur, Ben Bogin, Khyathi Chandu, Jennifer Dumas, Yanai Elazar, Valentin Hofmann, Ananya Jha, Sachin Kumar, Li Lucy, Xinxi Lyu, Nathan Lambert, Ian Magnusson, Jacob Morrison, Niklas Muennighoff, Aakanksha Naik, Crystal Nam, Matthew Peters, Abhilasha Ravichander, Kyle Richardson, Zejiang Shen, Emma Strubell, Nishant Subramani, Oyvind Tafjord, Evan Walsh, Luke Zettlemoyer, Noah Smith, Hannaneh Hajishirzi, Iz Beltagy, Dirk Groeneveld, Jesse Dodge, and Kyle Lo. 2024. Dolma: an open corpus of three trillion tokens for language model pretraining research. In *Proceedings of the 62nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 15725–15788, Bangkok, Thailand. Association for Computational Linguistics.

Megha Srivastava and Noah Goodman. 2021. Question generation for adaptive education. In *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing (Volume 2: Short Papers)*, pages 692–701, Online. Association for Computational Linguistics.

Svitlana Vakulenko, Shayne Longpre, Zhucheng Tu, and Raviteja Anantha. 2020. A wrong answer or a wrong question? an intricate relationship between question reformulation and answer selection in conversational question answering. In *Proceedings of the 5th International Workshop on Search-Oriented Conversational AI (SCAI)*, pages 7–16, Online. Association for Computational Linguistics.

Pat Verga, Sebastian Hofstatter, Sophia Althammer, Yixuan Su, Aleksandra Piktus, Arkady Arkhangorodsky, Minjie Xu, Naomi White, and Patrick Lewis. 2024. Replacing judges with juries: Evaluating llm generations with a panel of diverse models. *ArXiv*, abs/2404.18796.

Giorgio Visani, Enrico Bagli, Federico Chesani, Alessandro Poluzzi, and Davide Capuzzo. 2022. Statistical stability indices for lime: Obtaining reliable explanations for machine learning models. *Journal of the Operational Research Society*, 73(1):91–101.

Zichao Wang, Andrew S Lan, Weili Nie, Andrew E Waters, Phillip J Grimaldi, and Richard G Baraniuk. 2018. QG-net: a data-driven question generation model for educational content. In *Proceedings of the fifth annual ACM conference on learning at scale*, pages 1–10.

Jiaxin Wen, Ruiqi Zhong, Akbir Khan, Ethan Perez, Jacob Steinhardt, Minlie Huang, Samuel R. Bowman, He He, and Shi Feng. 2025. Language models learn to mislead humans via RLHF. In *The Thirteenth International Conference on Learning Representations*.

Yixuan Weng, Minjun Zhu, Fei Xia, Bin Li, Shizhu He, Shengping Liu, Bin Sun, Kang Liu, and Jun Zhao. 2023. Large language models are better reasoners with self-verification. In *Findings of the Association for Computational Linguistics: EMNLP 2023*, pages 2550–2575, Singapore. Association for Computational Linguistics.

Peter West, Ximing Lu, Nouha Dziri, Faeze Brahman, Linjie Li, Jena D Hwang, Liwei Jiang, Jillian Fisher, Abhilasha Ravichander, Khyathi Chandu, et al. 2023. The generative AI paradox:"What it can create, it may not understand". In *The Twelfth International Conference on Learning Representations*.

Zonglin Yang, Li Dong, Xinya Du, Hao Cheng, Erik Cambria, Xiaodong Liu, Jianfeng Gao, and Furu Wei. 2024. Language models as inductive reasoners. In *Proceedings of the 18th Conference of the European Chapter of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 209–225, St. Julian's, Malta. Association for Computational Linguistics.

Alex Young, Bei Chen, Chao Li, Chengen Huang, Ge Zhang, Guanwei Zhang, Heng Li, Jiangcheng Zhu, Jianqun Chen, Jing Chang, et al. 2024. Yi: Open foundation models by 01. AI. *arXiv preprint arXiv:2403.04652*.

Longhui Yu, Weisen Jiang, Han Shi, Jincheng YU, Zhengying Liu, Yu Zhang, James Kwok, Zhenguo Li, Adrian Weller, and Weiyang Liu. 2024. Metamath: Bootstrap your own mathematical questions for large language models. In *The Twelfth International Conference on Learning Representations*.

Xinyan Yu, Sewon Min, Luke Zettlemoyer, and Hannaneh Hajishirzi. 2023. CREPE: Open-domain question answering with false presuppositions. In *Proceedings of the 61st Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 10457–10480, Toronto, Canada. Association for Computational Linguistics.

Muru Zhang, Ofir Press, William Merrill, Alisa Liu, and Noah A. Smith. 2024. How language model hallucinations can snowball. In *Forty-first International Conference on Machine Learning*.

Ruqing Zhang, Jiafeng Guo, Lu Chen, Yixing Fan, and Xueqi Cheng. 2021. A review on question generation from natural language text. *ACM Transactions on Information Systems (TOIS)*, 40(1):1–43.

Chen Zhao, Chenyan Xiong, Jordan Boyd-Graber, and Hal Daumé III. 2021. Multi-step reasoning over unstructured text with beam dense retrieval. In *Proceedings of the 2021 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 4635–4641, Online. Association for Computational Linguistics.

Wenting Zhao, Justin Chiu, Claire Cardie, and Alexander Rush. 2023. Abductive commonsense reasoning exploiting mutually exclusive explanations. In *Proceedings of the 61st Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 14883–14896, Toronto, Canada. Association for Computational Linguistics.

Chujie Zheng, Hao Zhou, Fandong Meng, Jie Zhou, and Minlie Huang. 2024. Large language models are not robust multiple choice selectors. In *The Twelfth International Conference on Learning Representations*.

A Appendix

A.1 Dataset Details

We show details for our dataset in Table 2. Our entities are derived from Quizbowl questions (Rodriguez et al., 2019) from the QB Reader API,[6] which is free to use and publicly available online. We verify that all questions are answerable by the given answer via Google search. If any question was found to be unanswerable, we manually edited the question such that it was answerable. Thus, all of our collected data is within their license and terms of use, and our use of these questions are within their intended use. Since expert trivia writers curated these questions for academic competitions, we did not need to check that our data has PII. All questions and answers are in English.

A.2 Prompting Details

Below, we document our prompt engineering process for the QA and RQA prompts shown in Figure 1. To assess each prompt version, we ran inference on a small subset of examples with the Yi and LLaMA LLMs and manually assessed the quality of questions/answers to identify prevalent issues that could be avoided through prompt engineering. In all adjacent prompt boxes below, **blue text** corresponds to us adding instructions to the previous version of the prompt, and ~~red text~~ corresponds to us removing instructions from the previous version.

Our initial RQA prompt is in Prompt A.1. With this prompt, our LLMs generated verbose answers, so we added the instruction that all questions must be "one-sentence" (Prompt A.2). Next, we observed that it was difficult to reliably parse the question from the model's generated output, so we added formatting constraints (Prompt A.3). At this point, when we looked at the model's generated questions more closely, we saw that models could cheat—adding the answer in the question itself (e.g. giving the question "How many of the 150 people attended the conference" for the answer "150 people"). Thus, we added an instruction to forbid this behavior (Prompt A.4). Finally, as we noticed many of the questions were inaccurate, we wanted to study if abstention could alleviate these issues, so we added an instruction (Prompt A.5) allowing the model to respond with "IDK" §2.2. We added abstention to test LLM calibration (Feng et al., 2024), but abstention rates are only 3% in QA and <1% in RQA, so we do not study it in this work. We keep abstention to avoid re-running all LLMs and omit rare cases of abstention. Our final RQA prompt is in Prompt A.6.

We then designed our QA prompt by mimicking the format of the final RQA prompt, shown in Prompt A.7. We initially wrote the constraint that the answer must be "short" and "just a few words," but we felt these instructions were ambiguous, and the easy and hard entities split of our dataset had answers that were longer than just a few words; as a result, we removed these instructions, and used "the" instead of "a" to make it clear that there is only one valid answer (Prompt A.8). After removing these instructions, we noticed that models would often generate very long explanations before or after answering the question. To avoid this, we added an instruction stating that we were just looking for the answer and no explanation (Prompt A.8). Our final QA prompt is in Prompt A.10.

A.3 Model Details

The LLMs used in this work are from the following endpoints:

- LLaMA-8B: `Meta-Llama-3-8B-Instruct`
- LLaMA-70B: `Meta-Llama-3-70B-Instruct`
- Mistral-7B: `Mistral-7B-Instruct-v0.3`
- Mixtral-8x7B: `Mixtral-8x7B-Instruct-v0.1`
- Mixtral-8x22B: `Mixtral-8x22B-Instruct-v0.1`
- Yi-6B: `Yi-1.5-6B-Chat`
- Yi-9B: `Yi-1.5-9B-Chat`
- Yi-34B: `Yi-1.5-34B-Chat`
- Command-R: `command-r`
- Command-R+: `command-r-plus`
- GPT-3.5: `gpt-3.5-turbo-0125`
- GPT-4: `gpt-4-turbo-2024-04-09`
- GPT-4o: `gpt-4o-2024-05-13`
- Haiku: `claude-3-haiku-20240307`
- Sonnet: `claude-3-sonnet-20240229`
- Opus: `claude-3-opus-20240229`

LLaMA, Mistral, and Yi models are accessed via huggingface, and all other models are accessed through their respective API endpoints. We allocated 8 NVIDIA:A6000s for Mixtral-8x22B, 8 NVIDIA:A5000s for Mixtral-8x7B, Yi-34B, and LLaMA-70B, 2 NVIDIA:A6000s for Yi-9B and LLaMA-8B, and 1 NVIDIA:A6000 for all other

[6] https://www.qbreader.org/api-docs/

non-API models (which were run on CPU only). Each model was allocated 24 hours to run both QA and RQA on our dataset.

LLMs generate with 0 temperature, a minimum token length of 5, and a maximum token length of 5. All other unspecified parameters are set to their respective default values.

A.4 Metric Details

To design a metric for QA accuracy, we consider seven answer equivalence metrics, which check if a candidate answer a_{cand} is semantically equivalent to a ground-truth answer a_{true}: 1) DSPy-optimized GPT-4o; 2) A rule-based method designed specifically for each dataset; 3) Exact match; 4) Token F1 score; 5) Token Recall Score; 6) Token Precision Score; and 7) PEDANTS (Li et al., 2024b), a classifier designed for answer equivalence. The DSPy method in (1) uses a maximum of 10 bootstrapped demos, a maximum of 10 labeled demos, and 20 candidate programs; it uses 64 examples for training (seeding the prompts) and 64 examples for validation. We decide the optimal decision thresholds for (4), (5), and (6) using the 64 validation examples. We present the agreement with human annotations of each metric in Table 3, which is how we picked the metric to use for each dataset split. In all, our QA accuracy metric has 94% raw agreement with humans on 1152 held-out examples.

Since there are no automated metrics to check whether a question q can correctly be answered by an entity a, we design our own metric for RQA accuracy. Given the strength of the DSPy GPT-4o approach in QA accuracy, we similarly design a DSPy-optimized GPT-4o classifier that determines if q is correctly answered by a, using the same hyperparameters for QA accuracy. Overall, this RQA accuracy metric has 90% raw agreement with humans on 1152 held-out examples. We also considered Jury approaches (Verga et al., 2024), which ensemble multiple LLMs instead of relying just on a single LLM. However, using majority vote with three/five LLMs boosted our metric's accuracy by less than 2%, which we did not feel justified the much larger computational expenses.

All metrics are reported for a single run, and we provide confidence intervals in Figure 2 corresponding to the error rates in our metrics.

A.5 Abduction/Deduction Human Accuracy

In Figure 6, we show a version of Figure 2 using our human annotations on a subset of data versus the automated metrics on the entire splits. Our trend holds on the human-annotated subset; LLMs are still much weaker in numerical RQA versus QA, but their QA capabilities slightly beat RQA in the text-based settings.

A.6 RQA with Prompting Engineering

To explore if RQA weaknesses can be alleviated with prompt engineering efforts (Schulhoff et al., 2024), we test three prompting strategies: 1) Zero-Shot Chain-of-Thought Prompting (asking the LLM to "Think step by step" before answering); 2) Self-Verification (asking the LLM to "Check if the question is accurate after generating a question"); and 3) Five-Shot Prompting (including five exemplars showing the model how to generate a question for an answer). To write exemplars for (3), we pick question/answer pairs when RQA succeeds in the zero-shot setting to make the priors in the exemplars most similar to the model's original generations. The prompts for (1), (2), and (3) are in Prompts A.11, A.12, and A.13, respectively.

We experiment with GPT-4 on Numbers and Numbers+Text, as the model showed a surprising RQA weakness in these settings. GPT-4 is also considered to respond well to prompt engineering efforts, making it a suitable candidate for our prompting strategies. Overall, none of these prompting strategies can close the accuracy gap between RQA and QA (Figure 7). Chain-of-thought prompting increases GPT-4's RQA accuracy by ~ 0.15, but it is still significantly lower than QA, which does not use chain-of-thought. This shows that the accuracy gap between QA and RQA may be an inherent reasoning flaw of current LLMs that cannot be fully mitigated via prompt engineering.

A.7 Full Consistency Analysis

In this section, we describe the consistency analysis for all values of our truth table $\mathcal{A}$, introduced in §3.2. Apart from the four categories described before, the truth table outcome can also be "Ambiguous Question" if $\mathcal{A} = (\mathtt{y}, \mathtt{y}, \mathtt{n})$, as both steps succeeded but converged to different answers (meaning the question had more than one possible correct answer). Another option is for the mistakes to cancel out, which is a rare scenario $\mathcal{A} = (\mathtt{n}, \mathtt{n}, \mathtt{y})$ where the model generated an inaccurate question and answered its own question incorrectly, but managed to arrive at the original entity a. The final category is a Metric Prediction Error, a scenario that only occurs if either just QA or RQA was

predicted to fail, but a and $\hat{a}$ were predicted to be matching ($\mathcal{A} = (\mathtt{n}, \mathtt{y}, \mathtt{y})$ or $\mathcal{A} = (\mathtt{y}, \mathtt{n}, \mathtt{y})$). These scenarios are summarized in Table 4.

Figure 8 reports the full consistency analysis for all 16 of our LLMs and all truth table scenarios. The four categories reported in Figure 3 encompass most of the truth table. Further, even for smaller LLMs, our claims hold; LLMs can often detect their own question inaccuracies from RQA through QA.

A.8 Further Analysis of RQA Questions

Due to page limit constraints of a short paper, we were unable to show the entire qualitative analysis we conducted on questions generated in RQA. Below, we give more qualitative results on the answerability of questions from RQA (Appendix A.8.1), a cross-model comparison of question duplicates in RQA (Appendix A.8.2), the ability of LLMs to match the ground-truth question during RQA (Appendix A.8.3), and a brief investigation into memorization in the RQA task (Appendix A.8.4).

A.8.1 Are RQA questions unanswerable?

We now seek to understand the types of RQA questions generated in the Number+Text setting, complementing our analysis in §3.4. The Number+Text questions have higher variance and cannot be as neatly categorized as in §3.4 (e.g. single-step computation). So instead, we study the *answerability* of 30 generated questions from each LLM, i.e., if the question is clear but leads to an incorrect answer, or if the question has an issue that makes it difficult to answer. We adopt five categories of unanswerable questions from Rogers et al. (2023):

1) **Invalid Premise:** the question contains a false assumption, so it is impossible to answer. For example, Opus generates the question *How old was the world's oldest tortoise, Jonathan, when he passed away in 2022?*, but this Tortoise is still alive.

2) **No Consensus on the Answer:** the question does not have a single, agreed-upon answer . For example, LLaMA generates the question *What is the unique property of the Lie algebra E8 that makes it particularly interesting in theoretical physics?*, but Lie algebra has many distinct, interesting properties that would answer the question.

3) **Information not yet Discovered:** the answer to this question is not yet known. For example, GPT-4 generates the question *How long, in terms of word count, is the sentence that holds the record for being the longest in the English language without using any punctuation?*, but it is not yet known what could theoretically be the longest sentence.

4) **Missing Information:** the question does not have enough information, or it is too vague. For example, GPT-4 generates the question *How many individuals attended the annual community festival last year according to the final headcount?*, which cannot be answered without knowing more details.

5) **Answerable:** The question has one right answer.

As expected, when RQA succeeds, questions are mostly answerable (Figure 9). However, a non-trivial proportion of generated questions when RQA fails are unanswerable, reaching nearly 60% for GPT-4. The most common types of unanswerable questions are those that are missing information, meaning that they are too vague or ambiguous, or those that have false premises or assumptions. While several works explore methods to *answer* ambiguous questions (Min et al., 2020; Kim et al., 2023a) or questions with false presuppositions (Yu et al., 2023; Kim et al., 2023b), our analysis reveals a need to *avoid generating* ambiguous or faulty-presupposition questions in RQA.

In Tables 5 and 6, we provide examples of question/error types in our qualitative analysis on the Number and Number+Text split, respectively.

A.8.2 Do LLMs give the same RQA questions?

While most of our analysis treated LLMs independently, we now study whether LLMs generate the same exact questions (i.e. duplicates) in RQA. Figure 11 shows that LLMs more frequently generate duplicated questions across entities versus matching questions from other models. For example, LLaMA-3 70B generates 379 duplicate questions in the Numbers setting, even when the input answer is altered. This aligns with very recent work suggesting that LLMs may often conduct pattern-matching rather than engaging in true, generalizable reasoning (Mirzadeh et al., 2025).

Interestingly, models in the same family are more likely to generate duplicated questions. For example, GPT-3.5, GPT-4, and GPT-4o generate the same questions in RQA more often than when compared to other LLM families. Thus, we speculate that these model families likely share similar pre-training and alignment data, which is optimized on through different training recipes.

A.8.3 Does RQA match the gold question?

We now explore whether the questions generated for an answer in RQA match the gold question we collected for that answer. When determining if

the two questions are semantically equivalent, we follow the protocol of Balepur et al. (2024b) and analyze whether the two questions test the exact same knowledge. Figure 10 shows that the LLMs can often match the true question when RQA succeeds in Number+Text settings, reaching as high as 40% of cases for GPT-4; the questions never matched for Number. One explanation for the high match rate is dataset contamination (Ishihara, 2023), but it is also possible that the most likely question the LLM abductively reason towards is the ground-truth question. For example, for the answer "120 counties," the only salient fact linked to the entity is that Kentucky has 120 counties (McCarthy, 1959); this led GPT-4's question and the ground-truth question to both ask about Kentucky.

A.8.4 Are any RQA questions memorized?

Since the duplicates in Appendix A.8.3 suggest that LLMs may just be retrieving similar questions from pretraining rather than reasoning towards new questions in RQA, we now investigate the *novelty* of the RQA questions (Merrill et al., 2024), i.e., whether they are exactly copied from pretraining. We do not know which corpora all of our LLMs are trained on, so we use the Dolma (Soldaini et al., 2024) corpus as a proxy for pretraining data. For each generated RQA question $\hat{q}$, we compute how frequently the exact question $\hat{q}$ appears in Dolma via infini-gram (Liu et al., 2024b).

Table 7 reveals in total, 2.87% of RQA questions are exactly found in Dolma. For comparison, 1.25% of our ground-truth questions exist in Dolma. While we did not explicitly prompt the model to give a new question that it has not seen in pretraining, practitioners may need to design specialized techniques if they desire novel RQA questions.

When comparing exact question match frequency by model, weaker/smaller LLMs tend to copy more from pretraining data, suggesting that smaller LLMs are more prone to RQA memorization. Further, the Hard Fact setting is much less prone to question copying in RQA, likely because the RQA input answers have very low pretraining token count (§3.3), which further supports that LLMs may struggle to retrieve exact pretraining knowledge for long-tail facts (Kandpal et al., 2023).

We present examples of RQA questions that appear the most in Dolma in Table 8. The tendency to generate inaccurate or ambiguous questions may be influenced by pretraining, as many of these questions appear directly in Dolma.

Prompt A.1: Reverse Question Answering Prompt V1 (RQA)

```
Generate a question with the answer: "a".
```

Prompt A.2: Reverse Question Answering Prompt V2 (RQA)

```
Generate a one-sentence question with the answer: "a".
```

Prompt A.3: Reverse Question Answering Prompt V3 (RQA)

```
Generate a one-sentence question with the answer: "a". Please format your output
as "Question: [insert generated question]"
```

Prompt A.4: Reverse Question Answering Prompt V4 (RQA)

```
Generate a one-sentence question with the answer: "a". The question should not
contain the text "a". Please format your output as "Question: [insert generated
question]"
```

Prompt A.5: Reverse Question Answering Prompt V5 (RQA)

```
Generate a one-sentence question with the answer: "a". The only possible answer
to the question must be "a". The question should not contain the text "a".
Please format your output as "Question: [insert generated question]". If no
possible question exists say "IDK".
```

Prompt A.6: Final Reverse Question Answering Prompt (RQA)

```
Generate a one-sentence question with the answer: "a". The only possible answer
to the question must be "a". The question should not contain the text "a".
Please format your output as "Question: [insert generated question]". If no
possible question exists say "IDK".
```

Prompt A.7: Question Answering Prompt V1 (QA)

Generate a short answer to the question: "q". The answer should just be a few words long. Please format your output as "Answer: [insert generated answer]". If no possible answer exists say "IDK".

Prompt A.8: Question Answering Prompt V2 (QA)

Generate ~~a short~~ **the** answer to the question: "q". ~~The answer should just be a few words long.~~ Please format your output as "Answer: [insert generated answer]". If no possible answer exists say "IDK".

Prompt A.9: Question Answering Prompt V3 (QA)

Generate the answer to the question: "q". **Give just the answer and no explanation.** Please format your output as "Answer: [insert generated answer]". If no possible answer exists say "IDK".

Prompt A.10: Final Question Answering Prompt (QA)

Generate the answer to the question: "q". Give just the answer and no explanation. Please format your output as "Answer: [insert generated answer]". If no possible answer exists say "IDK".

Prompt A.11: RQA with Chain-of-Thought

```
Generate a one-sentence question with the answer: "a". The only possible answer
to the question must be "a". The question should not contain the text "a". Think
step by step and reason before generating the question. After reasoning, please
format your final output as "Question: [insert generated question]".
```

Prompt A.12: RQA with Self-Verification

```
Generate a one-sentence question with the answer: "a". The only possible answer
to the question must be "a". The question should not contain the text "a".
Please format your output as "Question: [insert generated question]". After
generating a question, answer your own question to verify that the answer is "a",
formatted as "Answer: [insert answer to generated question]".
```

Prompt A.13: RQA with Five Exemplars

```
Generate a one-sentence question with the answer: "a". The only possible answer
to the question must be "a". The question should not contain the text "a".
Please format your output as "Question: [insert generated question]".

Answer: 328
Question: What is the sum of the first 15 prime numbers?

Answer: 710 survivors
Question: How many people survived the sinking of the RMS Titanic in 1912?

Answer: 648
Question: What is the product of 12 and 54?

Answer: 286 ayats
Question: How many verses are there in the longest chapter of the Quran, Surah
Al-Baqarah?

Answer: 311
Question: What is the sum of the first three prime numbers greater than 100?

Answer: a
Question:
```

	Number	Number+Text	Easy Entity	Hard Entity
Count	900	743	900	900
Average Answer Length (Tokens)	1.00	2.49	2.77	5.18
Average Question Length (Tokens)	8.75	21.9	18.9	22.9

Table 2: Dataset details of each split (Number, Number+Text, Easy Entity, Hard Entity), including the number of data instances, average length of answers (in tokens), and average length of questions (in tokens). Tokens are computed using `tiktoken`.

Metric	Number	Number + Text	Easy Entity	Hard Entity
DSPy (GPT-4o)	0.972	0.924	**0.917**	**0.897**
Rule-Based	**0.979**	**0.965**	0.817	0.790
Exact Match	**0.979**	0.819	0.752	0.537
Token F1	0.969	0.771	0.845	0.829
Token Recall	0.969	0.760	0.848	0.826
Token Precision	0.969	0.760	0.848	0.826
PEDANTS	0.972	0.760	0.872	0.786

Table 3: Raw agreement with human annotators (i.e. accuracy) of seven tested answer equivalence metrics. The best metric for each dataset split is in **bold**.

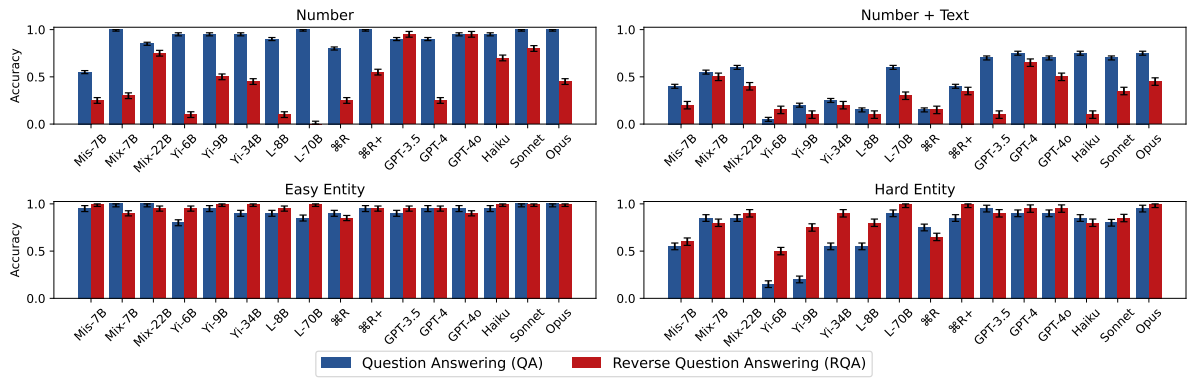

Figure 6: LLM deduction (**blue**) and abduction (**red**) accuracy based on human annotations on a subset of data (20 labels per model/dataset). The plot shows a similar trend as the automated metrics (LLMs are weaker in abduction in numerical settings, but stronger in abduction in non-numerical settings), confirming the validity of our metrics.

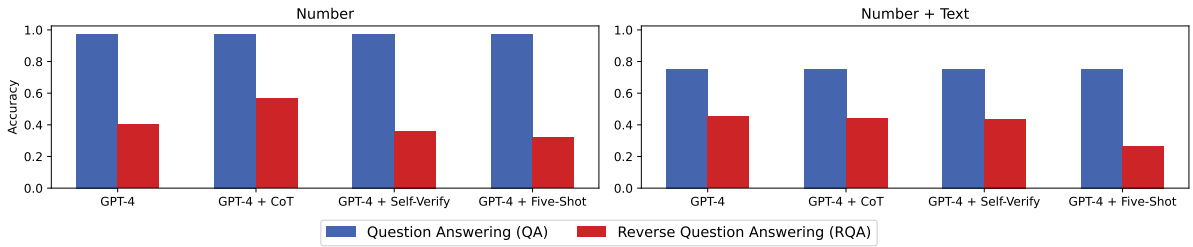

Figure 7: LLM deduction (**blue**) and abduction (**red**) accuracy with GPT-4 on numerical entities. For QA, we present the zero-shot prompt used in §3.1. For RQA, we test adding chain-of-thought instructions (GPT-4 + CoT), asking the LLM to verify its question post-generation (GPT-4 + Self-Verification), and including five exemplars (GPT-4 + 5-Shot). None of these strategies allow the model to fully match the QA accuracy.

Is a_{true} the answer to q_{bwd}?	Is a_{bwd} the answer to q_{bwd}?	Is a_{true} equal to a_{bwd}?	**Outcome**
Yes	Yes	Yes	RQA = QA
Yes	Yes	No	Ambiguous Question
Yes	No	Yes	QA Fails
Yes	No	No	Metric Error (Impossible)
No	Yes	Yes	RQA Fails
No	Yes	No	Metric Error (Impossible)
No	No	Yes	RQA + QA Fail
No	No	No	Mistakes Cancel (lucky!)

Table 4: All truth table outcomes for the consistency analysis in §3.2.

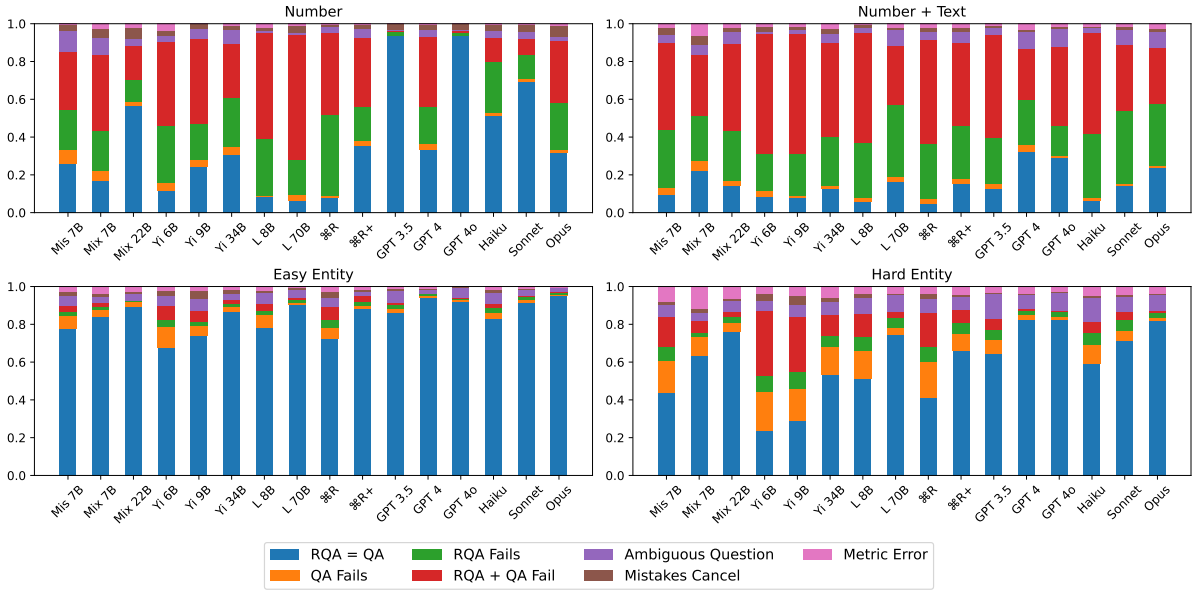

Figure 8: QA and RQA logical consistency across all models. The consistency trends are also prevalent for smaller/less capable LLMs; RQA and QA consistency is higher for easy/hard entities, but LLMs can often detect their own RQA inaccuracies in numerical settings.

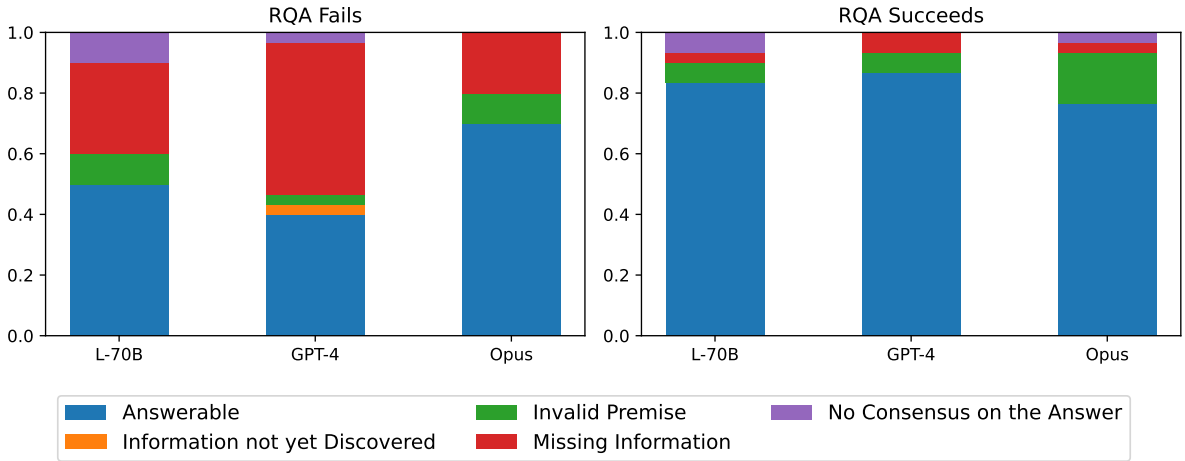

Figure 9: Error analysis of questions from RQA on Number+Text. When RQA fails, questions are often unanswerable (30-60%), and frequently include false premises or omit key information that is needed to answer the question.

Question	Answer	Model	Valid?	Question Type
What is the sum of the numbers on a standard roulette wheel?	369	L-70B	No	Multi-Step
What is the sum of the first 37 natural numbers?	749	GPT-4	No	Multi-Step
What is the sum of the first 18 positive odd integers?	855	Opus	No	Multi-Step
What is the result of multiplying 25 by 25?	625	L-70B	Yes	Single-Step
What is the smallest prime number greater than 357?	359	GPT-4	Yes	Single-Step
What is the product of 30 and 23?	690	Opus	Yes	Single-Step
What is the emergency telephone number in the United States and many other countries?	911	L-70B	Yes	Fact-based
What is the atomic number of the element with the highest atomic number ... as of 2023?	223	GPT-4	No	Fact-based
What is the number of characters allowed in a single tweet on Twitter?	280	Opus	Yes	Fact-based

Table 5: Examples of RQA question types and errors on the Number split.

Question	Answer	Model	Error Type
How many British soldiers were killed or wounded during the Battle of Thermopylae in 480 BCE?	266 men	L-70B	Invalid Premise
What is the numerical designation..., if we humorously assume there were 111 before it?	112 Ark	GPT-4	Invalid Premise
According to a 2011 census, how many officially recognized ethnic groups are there in India?	634 distinct peoples	Opus	Invalid Premise
In what year did the Vietnamese king Le Hoan defeat the Song Dynasty army at the Battle of Bach Dang?	988 AD	L-70B	No Consensus
How long did the construction of the Great Wall of China continue...?	264 years	GPT-4	No Consensus
What is the wavelength of yellow light in the visible spectrum?	587 nanometers	L-70B	Missing Info
How many individuals attended the annual community festival last year according to the final headcount?	178 people	GPT-4	Missing Info
How old was the world's oldest tortoise, Jonathan, when he passed away in 2022?	179 years of age	Opus	Missing Info

Table 6: Examples of RQA question types and errors on the Number+Text split.

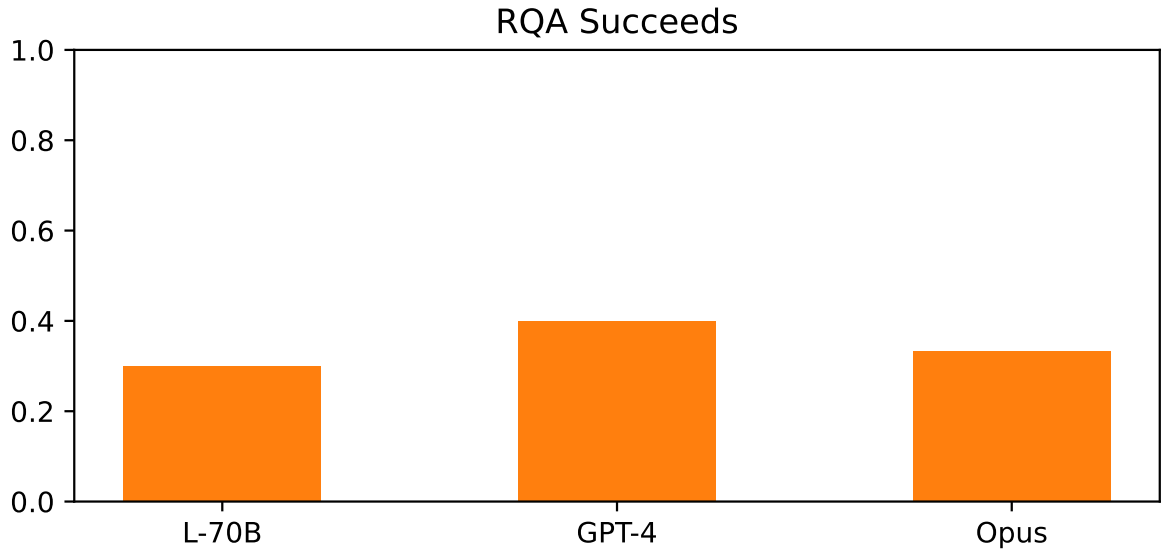

Figure 10: Proportion of RQA questions on Numbers+Text that semantically match the ground-truth question when RQA succeeds. LLaMA-3 70B, GPT-4, and Opus can all match the ground-truth question over 25% of the time.

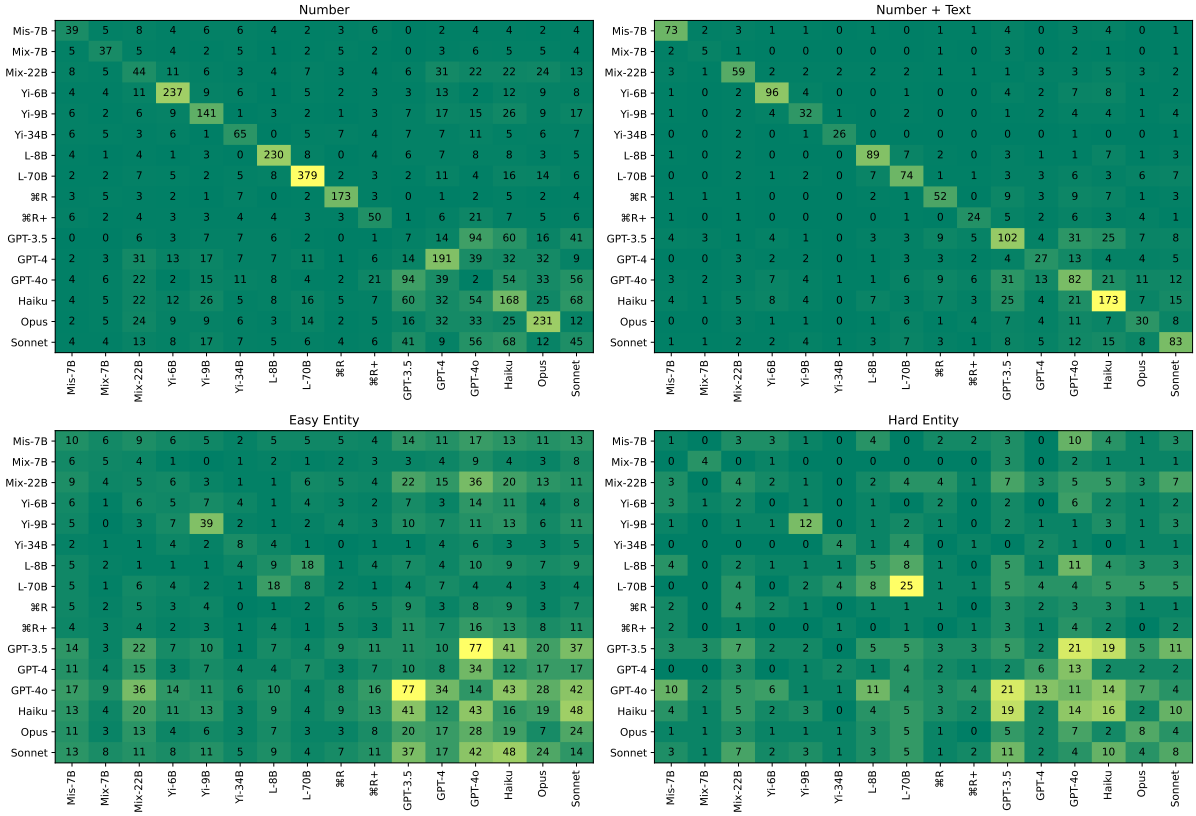

Figure 11: Cross-model frequency of questions from RQA that are exact duplicates. LLMs often generate the same question in RQA even though the input answer changes, reaching as high as 379 for LaMA-3 70B on Numbers.

Model	Easy Fact	Hard Fact	Number	Number+Text	**Model Sum**
Mis-7b	25	6	6	7	44
Mix-7B	7	0	10	1	18
Mix-22B	41	3	17	12	73
Yi-6B	18	0	98	40	156
Yi-9B	17	2	8	1	28
Yi-34B	7	0	18	0	25
L-8B	12	2	21	8	43
L-70B	4	1	19	4	28
Command-R	48	3	61	47	159
Command-R+	28	3	17	20	68
GPT-3.5	111	19	16	105	251
GPT-4	29	0	32	3	64
GPT-4o	96	10	27	39	172
Haiku	88	12	67	95	262
Sonnet	51	5	18	17	91
Opus	41	0	48	12	101
Dataset Sum	623	66	483	411	**1583**

Table 7: Number of generated RQA questions that are exact matches to a question in the Dolma pretraining corpus. On average, models are prone to copying questions from pretraining $\sim 3\%$ of the time. Smaller/weaker LLMs are more susceptible to copying questions from pretraining in RQA. Further, easy facts and numerical answers are more likely to lead to copied questions in RQA versus our hard facts.

Question	Answer	Model(s)	Split	Valid	Count
What is the answer to this question?	Lucy poems	Haiku	Hard Fact	No	21313
Who lives in a pineapple under the sea?	Spongebob Squarepants	GPT-3.5, GPT-4o	Easy Fact	Yes	1452
Where does the story take place?	In the Penal Colony	GPT-3.5	Hard Fact	No	1395
How many countries are there in the world?	195 nations	GPT-3.5	Num+Text	Yes	380
What is the capital of France?	Paris, France	Command-R+	Easy Fact	Yes	338
Who was the first president of the United States?	George Washington	Haiku, Sonnet	Easy Fact	Yes	281
How many days are there in a week?	357	Yi-6B	Number	No	194
What is the capital of the United States?	Washington, D.C.	Command-R, GPT-3.5	Easy Fact	Yes	192
How many days are there in a year?	365	Command-R	Easy Fact	Yes	166
How many days are there in a year?	800	Haiku	Number	No	166

Table 8: Questions generated from RQA that are most frequently found in the Dolma corpus. The LLM's tendency to generate inaccurate questions (e.g. *How many days are there in a year?* for *800*) or ambiguous questions (*What is the answer to this question?*) could be influenced by how often these questions appear in pretraining.