ReviseQA: A Benchmark for Belief Revision in Multi-Turn Logical Reasoning

Chadi Helwe [1] **Sultan AlRashed** [1] **Francesco Orabona** [1]

Abstract

Large Language Models (LLMs) are increasingly used in conversational applications that require interactive reasoning, such as tutoring systems and legal assistants. While these models perform well on static QA tasks, it remains unclear whether they can consistently revise their beliefs and perform logical reasoning when prior information is retracted or new information is introduced over multiple conversational turns. To address this, we introduce *ReviseQA*, a benchmark for belief revision in multi-turn logical reasoning. Each turn gradually modifies the previous context by removing or adding facts and rules, requiring the model to reassess its conclusion. This dynamic setting reflects real-world reasoning, where agents must update their conclusions based on evolving information. Our experiments show that current LLMs often fail to maintain logical consistency when updating beliefs, highlighting *ReviseQA* as a necessary benchmark toward evaluating and improving multi-turn reasoning in LLMs.

1. Introduction

Large Language Models (LLMs) excel in different Natural Language Processing (NLP) applications, including Question-Answering (QA), Summarization, and Translation. Their success in these applications has led to widespread adoption across several fields, such as education, law, and healthcare. These models are often used in an interactive conversational format, where users ask questions and receive responses from the model in a back-and-forth manner. Examples of such conversational models include ChatGPT and Claude. Despite their performances and growing popularity, it remains important to understand their logical reasoning capabilities during multi-turn conversations.

[1]King Abdullah University of Science and Technology (KAUST) Thuwal, 23955-6900, Kingdom of Saudi Arabia {firstname.lastname}@kaust.edu.sa
. Correspondence to: Chadi Helwe <chadi.helwe@kaust.edu.sa>.

ICML 2025 Workshop on Assessing World Models.

Logical reasoning is the process of applying inference rules to a set of premises, consisting of facts and logical rules, to derive a conclusion that is logically entailed by them (Helwe et al., 2022), such that the reasoning is valid (e.g., from $p \rightarrow q$ and p, we can conclude q by *Modus Ponens*). Different benchmarks have been created to evaluate the logical reasoning capabilities of LLMs in a static manner, such as through QA formats, including LogiQA (Liu et al., 2021; 2023) and ReClor (Yu et al., 2020). However, there is a lack of datasets designed to assess the logical reasoning abilities of LLMs in dynamic settings. In these cases, LLMs must adapt to new information or discard outdated information in multi-turn conversations to provide accurate answers, requiring models to revise their beliefs accordingly. Without this capability, they risk providing outdated or logically invalid conclusions during conversations, which limits their reliability in real-world applications.

To address this gap, we introduce *ReviseQA*, a benchmark designed to evaluate the ability of LLMs to revise their beliefs and reason logically at each step of a multi-turn conversation. *ReviseQA* is synthetically constructed in four stages: we first use the *ProverGen* framework (Qi et al., 2025) to generate an initial QA example. This includes a context of logical facts and rules (in both First-Order Logic (FOL) and natural language), a conclusion that serves as a question, and the corresponding answer. Next, to simulate conversational turns, we prompt an LLM to symbolically manipulate the previous context by adding or removing facts or rules. These manipulations are intended to either preserve or alter the answer to the original question. Each revised context is then validated using the FOL prover *Prover9* (McCune, 2005) to ensure logical consistency. If the modification is invalid, feedback is provided to the LLM to prompt a corrected revision. Following this, the symbolic changes made at each step are translated into natural language. Finally, we use LLM-as-a-judge (Gu et al., 2024) to validate these translations for data quality. Our reproducible pipeline creates logic-based conversational examples to test models' ability to reason logically with updated context.

Using *ReviseQA*, we evaluated several LLMs and found that they struggle to revise beliefs and reason logically as conversational turns increase.

Reproducibility. All our code is available.[1]

2. Related Work

Logical Reasoning Benchmarks. Several benchmarks have been developed to assess the logical reasoning abilities of language models. Several of these logical reasoning datasets have been collected from national exams or standardized tests, such as the LSAT and GMAT. Examples of such datasets include LogiQA (Liu et al., 2021; 2023), RECLor (Yu et al., 2020), and AR-LSAT (Zhong et al., 2021). In contrast, other datasets, like FOLIO (Han et al., 2024) and BIG-Bench (Srivastava et al., 2023), have been created by humans rather than sourced from exams.

An alternative method for developing logical reasoning datasets is through synthetic generation. In these cases, data is initially created symbolically in propositional logic or FOL and then translated into natural language. Many of these synthetically generated datasets use rule-based methods for the translation steps, such as ProofWriter (Tafjord et al., 2021), RuleTaker (Clark et al., 2021), LogicNLI (Tian et al., 2021), and PrOntoQA (Saparov & He, 2023). However, LogicBench (Parmar et al., 2024) and ProverGen (Qi et al., 2025) use LLMs to translate the FOL symbols into natural language, resulting in datasets that are more natural and diversified.

These datasets evaluate logical reasoning statically, failing to reflect how LLMs revise their conclusions when presented with new information.

Belief Revision. Belief revision refers to the process of changing beliefs when new information is received. While much research emphasizes updating models' beliefs through adjusting their parameters (De Cao et al., 2021; Dai et al., 2022; Hase et al., 2023), there has been limited attention to evaluating belief revision in the context of multi-turn reasoning conversations. One work, Belief-R (Wilie et al., 2024), assesses how LLMs revise their beliefs. In this framework, the model starts with two initial premises. In the next turn, an additional premise is introduced, prompting the LLM to revise its beliefs. The model can either maintain its prior beliefs or update them. If it chooses to update, it should alter its conclusion accordingly. However, Belief-R does not evaluate belief revision within a multi-turn reasoning conversation. Instead, it assesses it in a single turn. Additionally, while Belief-R uses LLMs to generate the premises, the conclusions are labeled by human annotators.

Our benchmark, *ReviseQA*, is based on the *ProverGen* framework, which creates the initial examples. We then use an LLM and a prover to generate multi-turn reasoning conversations. The prover ensures that our edits are logically valid. Importantly, our dataset does not rely on human annotations. To prevent data contamination, which occurs when evaluation examples overlap with a model's pretraining data, inflating performance and misrepresenting reasoning (Cheng et al., 2025), similar to the *ProverGen* framework, our method generates new datasets using both an LLM and a prover, ensuring they remain uncontaminated.

[1] https://github.com/ChadiHelwe/reviseqa

3. *ReviseQA* Construction

Our pipeline for constructing *ReviseQA* consists of four stages. First, we generate the initial QA example, which includes a context of facts and rules, a conclusion (which also serves as the question), and an answer that indicates whether the conclusion is `TRUE`, `FALSE`, or `UNCERTAIN`. Second, we verify the logical validity of these edits using an FOL prover. Third, we translate these edits into natural language. Finally, we conduct a data quality control check using LLM-as-a-judge.

3.1. Initial QA Example Generation

We build upon the approach introduced in the *ProverGen* framework (Qi et al., 2025), which generates logically grounded QA examples by first creating an FOL example and then translating it into natural language. A QA example from *ProverGen* includes the following: a context (comprising facts and rules), a conclusion (which also serves as the question), an answer, and the reasoning path if the conclusion is provable. Their framework is designed to generate a single static QA example. In our approach, we adopt a similar initial generation process to construct the starting QA pair of a conversation, but extend it by using its context as the foundation for multi-turn belief revision.

3.2. Symbolic Manipulation

After generating the initial QA example, we construct multi-turn QA examples by incrementally modifying the previous context at each turn. To do this, we use *DeepSeek Prover V2* (Ren et al., 2025), an LLM designed for formal theorem proving. At every step, the model is prompted to manipulate the preceding FOL context, either by preserving the answer (e.g., keeping `TRUE`) or flipping it (e.g., from `TRUE` to `FALSE`). In cases where the initial answer is `UNCERTAIN`, meaning that both the conclusion and its negation are not provable, we prompt the model to edit the FOL context to prove the conclusion while ensuring that its negation cannot be proven. The allowed edits are: remove facts, remove rules, add facts, and add rules.

After each edit, we use the *DeepSeek Prover V2* model to generate a *Prover9* syntax representation of the edited context and the conclusion. We then verify whether the

Table 1: Example of initial context, conclusion, answer, and a "flip" edit.

Context	Lukas is vibrant. Lukas is not outgoing. Lukas values life. Lukas accepts his flaws. Lukas is passionate. Lukas embraces himself. If Lukas is vibrant, then he is either outgoing or authentic, but not both. If Lukas loves himself, then he may not necessarily value life, and if Lukas values life, then he may not necessarily love himself. Anyone who loves themselves or accepts their flaws has inner strength. If Lukas is authentic, then he is either passionate or confident, but not both. If someone embraces themselves, then they have inner strength and are beautiful. Lukas is either confident or deserves respect, but not necessarily both.
Conclusion	Lukas deserves respect and is beautiful.
Answer	`TRUE`
Edit #1	**Removed fact:** Lukas is passionate. **Added fact:** Lukas is not passionate.

conclusion is provable and if its negation is unprovable. If the conclusion is not provable, we prompt the model to perform *forward reasoning*. On the other hand, if the negation of the conclusion is provable, we ask the model to conduct *backward reasoning*.

Forward Reasoning Prompt. In situations where the prover is unable to find a proof for the conclusion, we prompt the model by indicating that it has made a mistake according to the prover. We then request the model to engage in forward reasoning to identify why these edits do not lead to the conclusion. Next, we request that the model propose a set of edits that would enable it to derive the conclusion but not its negation.

Backward Reasoning Prompt. In cases where the prover can find a proof for the negation of the conclusion, we indicate to the model that it is able to prove this negation, which results in an invalid answer. Along with this, we provide the proof trace generated by *Prover9*. We ask the model to examine the proof trace and engage in backward reasoning from the negation of the conclusion to identify which facts or rules contributed to deriving that negation. We then ask the model to propose a set of edits that would allow it to derive the conclusion instead of its negation.

After generating the proposed edits by one of these prompts, we re-ask *DeepSeek Prover V2* to convert them into *Prover9* syntax for rechecking. Then, we also verify that the negation of the conclusion is not derivable. This process is repeated for i retries until the prover validates the edits. To create n edit steps, each step is based on the edited context of the previous one.

The intuition behind using two different prompts is that *forward reasoning* lets the model explore which conclusions were led by the edits and what it needs to do to arrive at the correct conclusion, while *backward reasoning* helps the model diagnose why the negation was reached and how to undo that reasoning path.

3.3. Natural Language Translation and Quality Control

After successfully generating n edit steps for each example, we use *Claude Sonnet* to translate the FOL edits into their equivalent natural language. To ensure the quality of the benchmark, we use three LLMs acting as judges to evaluate the quality of the translation. If any one of the models identifies an issue with any of the edits, the entire example is discarded; otherwise, it is retained. Table 1 shows an example of one edit that flips the answer from `TRUE` to `FALSE`.

In Appendix A, we report the different prompts used in our generation pipeline.

4. Experiments

To build our dataset, we first generated initial QA examples requiring 6 to 9 reasoning steps to reach the correct answer, corresponding to the highest difficulty setting within the *ProverGen* framework. Although we focused on the most challenging cases, our framework is generalizable and can be applied to any difficulty level. For each example, we then generated 7 sequential edit turns. After data verification, our dataset comprises 210 examples, each with 7 edit turns. It is important to note that our benchmark can be easily extended to include more examples using our generation pipeline.

We evaluated different LLMs on our dataset using two prompting strategies: standard prompting and Chain-Of-Thought (COT) prompting. In COT, the model is provided with reasoning steps that lead to the correct answer of the initial QA example before the edit turns. In contrast, the standard approach does not provide these reasoning steps. For each prompting strategy, we explored different multi-turn interaction settings. One method updates the context implicitly by embedding the edits within it without explicitly highlighting them. The other method makes the edits explicit by directly stating the changes to the model. Additionally, we simulated conversations to assess the model's behavior under two scenarios: one in which we inform the model whether its previous answer at step $t-1$ was correct

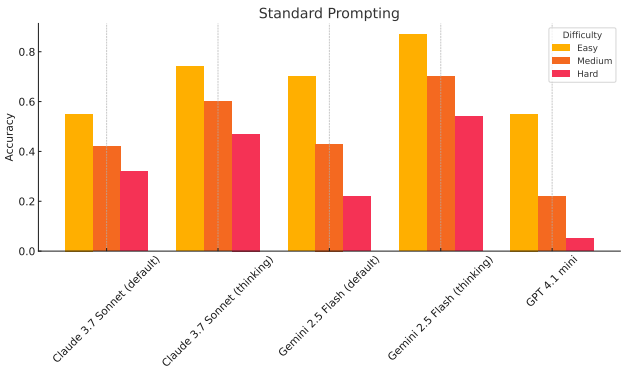

Figure 1: Standard Prompting

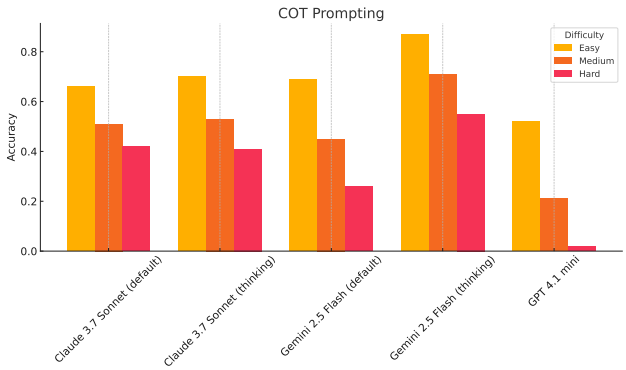

Figure 2: COT Prompting

Figure 3: Accuracy results for different prompting strategies on the explicit edits task without feedback.

or incorrect, and another in which the model continues without receiving any feedback. In the following, we report only the results with explicit edits and without feedback, while the complete results in the other settings can be found in Appendix D. Moreover, all the prompts that we used are in Appendix B.

4.1. Results

Our benchmark consists of three levels of difficulty, which correspond to the number of edits completed correctly. The easy level indicates that the first 2 edit steps are correct. The medium level means that the first 4 edit steps are correct. The hard level signifies that all 7 edit steps are correct.

Figure 3 illustrates the results of different models using two different prompt strategies, namely standard and COT (i.e., the reasoning path used in the initial QA example), on the task of explicit edit examples without providing feedback to the models at each turn. All models used a temperature of 0.7. We notice that the performance of all models decreases significantly as the number of editing steps increases. This decline suggests that models struggle to revise their beliefs and maintain consistent logical reasoning. However, models that engage in *thinking* before providing an answer perform better than those that respond immediately. This is evident in the performance of *Gemini 2.5 Flash Thinking*, which achieves the highest scores across the levels, and *Claude Sonnet 3.7 Thinking*. These results indicate that *thinking* is necessary for models to revise their beliefs. Interestingly, we find that providing a COT does not enhance the performance of models that already engage in internal reasoning. This may be due to these models being more effective when reasoning independently, as presenting initial reasoning steps could limit their flexibility or lead to inconsistencies in their thought processes. Additionally, as it can be seen in the additional results in Appendix D, the feedback at each edit turn does not improve performance. These results also include evaluations of other models.

5. Limitations

The two main limitations of our generation pipelines are their high costs and the time required to produce examples. These challenges arise from the numerous iterations needed to create an edit step of the context that is provable by the prover. However, our approach ensures that the examples generated are logically valid. A further limitation is the lack of human experts in logic involved in the verification of the dataset. While our approach is fully automated, allowing us to generate a large number of examples, which would be unfeasible with human experts involved, yet we still validate the logic using *Prover9*, and we verify the translations to natural language using an approach that uses LLM-as-a-judge.

6. Conclusion

This paper introduces *ReviseQA*, a QA dataset designed to evaluate LLMs for belief revision in multi-turn logical reasoning. Our generation pipeline starts with the *ProverGen* framework to create the initial examples, followed by LLMs for context edits, and a formal prover to maintain logical validity. Then, we evaluated the performance of LLMs on our dataset and found that their performance declines as the number of editing turns increases.

While our evaluations primarily focus on maintaining beliefs within the context of the models, future work will explore the effectiveness of Retrieval-Augmented Generation by storing and retrieving facts and rules from a vector database to support multi-turn reasoning.

References

Cheng, Y., Chang, Y., and Wu, Y. A survey on data contamination for large language models. *arXiv preprint arXiv:2502.14425*, 2025.

Clark, P., Tafjord, O., and Richardson, K. Transformers as soft reasoners over language. In *Proceedings of the Twenty-Ninth International Conference on International Joint Conferences on Artificial Intelligence*, pp. 3882–3890, 2021.

Dai, D., Dong, L., Hao, Y., Sui, Z., Chang, B., and Wei, F. Knowledge neurons in pretrained transformers. In *Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pp. 8493–8502, 2022.

De Cao, N., Aziz, W., and Titov, I. Editing factual knowledge in language models. In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*, pp. 6491–6506, 2021.

Gu, J., Jiang, X., Shi, Z., Tan, H., Zhai, X., Xu, C., Li, W., Shen, Y., Ma, S., Liu, H., Wang, S., Zhang, K., Wang, Y., Gao, W., Ni, L., and Guo, J. A survey on LLM-as-a-judge. *arXiv preprint arXiv:2411.15594*, 2024.

Han, S., Schoelkopf, H., Zhao, Y., Qi, Z., Riddell, M., Zhou, W., Coady, J., Peng, D., Qiao, Y., Benson, L., Sun, L., Wardle-Solano, A., Szabó, H., Zubova, E., Burtell, M., Fan, J., Liu, Y., Wong, B., Sailor, M., Ni, A., Nan, L., Kasai, J., Yu, T., Zhang, R., Fabbri, A., Kryscinski, W. M., Yavuz, S., Liu, Y., Lin, X. V., Joty, S., Zhou, Y., Xiong, C., Ying, R., Cohan, A., and Radev, D. FOLIO: Natural language reasoning with first-order logic. In Al-Onaizan, Y., Bansal, M., and Chen, Y.-N. (eds.), *Proceedings of the 2024 Conference on Empirical Methods in Natural Language Processing*, pp. 22017–22031, Miami, Florida, USA, November 2024. Association for Computational Linguistics.

Hase, P., Diab, M., Celikyilmaz, A., Li, X., Kozareva, Z., Stoyanov, V., Bansal, M., and Iyer, S. Methods for measuring, updating, and visualizing factual beliefs in language models. In *Proceedings of the 17th Conference of the European Chapter of the Association for Computational Linguistics*, pp. 2714–2731, 2023.

Helwe, C., Coumes, S., Clavel, C., and Suchanek, F. TINA: Textual inference with negation augmentation. In Goldberg, Y., Kozareva, Z., and Zhang, Y. (eds.), *Findings of the Association for Computational Linguistics: EMNLP 2022*, pp. 4086–4099, Abu Dhabi, United Arab Emirates, December 2022. Association for Computational Linguistics. doi: 10.18653/v1/2022.findings-emnlp.301. URL https://aclanthology.org/2022.findings-emnlp.301/.

Liu, H., Liu, J., Cui, L., Teng, Z., Duan, N., Zhou, M., and Zhang, Y. LogiQA 2.0—an improved dataset for logical reasoning in natural language understanding. *IEEE/ACM Transactions on Audio, Speech, and Language Processing*, 31:2947–2962, 2023.

Liu, J., Cui, L., Liu, H., Huang, D., Wang, Y., and Zhang, Y. LogiQA: a challenge dataset for machine reading comprehension with logical reasoning. In *Proceedings of the Twenty-Ninth International Joint Conference on Artificial Intelligence*, IJCAI'20, 2021.

McCune, W. Release of Prover9. In *Mile high conference on quasigroups, loops and nonassociative systems, Denver, Colorado*, 2005.

Miller, E. Adding error bars to evals: A statistical approach to language model evaluations. *arXiv preprint arXiv:2411.00640*, 2024.

Parmar, M., Patel, N., Varshney, N., Nakamura, M., Luo, M., Mashetty, S., Mitra, A., and Baral, C. LogicBench: Towards systematic evaluation of logical reasoning ability of large language models. In *Proceedings of the 62nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pp. 13679–13707, 2024.

Qi, C., Ma, R., Li, B., Du, H., Hui, B., Wu, J., Laili, Y., and He, C. Large language models meet symbolic provers for logical reasoning evaluation. *arXiv preprint arXiv:2502.06563*, 2025.

Ren, Z. Z., Shao, Z., Song, J., Xin, H., Wang, H., Zhao, W., Zhang, L., Fu, Z., Zhu, Q., Yang, D., Wu, Z., Gou, Z., Ma, S., Tang, H., Liu, Y., Gao, W., Guo, D., and Ruan, C. DeepSeek-Prover-V2: Advancing formal mathematical reasoning via reinforcement learning for subgoal decomposition. *arXiv preprint arXiv:2504.21801*, 2025.

Saparov, A. and He, H. Language models are greedy reasoners: A systematic formal analysis of chain-of-thought. In *The Eleventh International Conference on Learning Representations*, 2023.

Srivastava, A., Rastogi, A., Rao, A., Shoeb, A. A. M., Abid, A., Fisch, A., Brown, A. R., Santoro, A., Gupta, A., Garriga-Alonso, A., Kluska, A., Lewkowycz, A., Agarwal, A., Power, A., Ray, A., Warstadt, A., Kocurek, A. W., Safaya, A., Tazarv, A., Xiang, A., Parrish, A., Nie, A., Hussain, A., Askell, A., Dsouza, A., Slone, A., Rahane, A., Iyer, A. S., Andreassen, A. J., Madotto, A., Santilli, A., Stuhlmüller, A., Dai, A. M., La, A., Lampinen, A. K., Zou, A., Jiang, A., Chen, A., Vuong, A., Gupta, A., Gottardi, A., Norelli, A., Venkatesh, A., Gholamidavoodi, A., Tabassum, A., Menezes, A., Kirubarajan, A., Mullokandov, A., Sabharwal, A., Herrick, A., Efrat, A., Erdem,

A., Karakaş, A., Roberts, B. R., Loe, B. S., Zoph, B., Bojanowski, B., Özyurt, B., Hedayatnia, B., Neyshabur, B., Inden, B., Stein, B., Ekmekci, B., Lin, B. Y., Howald, B., Orinion, B., Diao, C., Dour, C., Stinson, C., Argueta, C., Ferri, C., Singh, C., Rathkopf, C., Meng, C., Baral, C., Wu, C., Callison-Burch, C., Waites, C., Voigt, C., Manning, C. D., Potts, C., Ramirez, C., Rivera, C. E., Siro, C., Raffel, C., Ashcraft, C., Garbacea, C., Sileo, D., Garrette, D., Hendrycks, D., Kilman, D., Roth, D., Freeman, C. D., Khashabi, D., Levy, D., González, D. M., Perszyk, D., Hernandez, D., Chen, D., Ippolito, D., Gilboa, D., Dohan, D., Drakard, D., Jurgens, D., Datta, D., Ganguli, D., Emelin, D., Kleyko, D., Yuret, D., Chen, D., Tam, D., Hupkes, D., Misra, D., Buzan, D., Mollo, D. C., Yang, D., Lee, D.-H., Schrader, D., Shutova, E., Cubuk, E. D., Segal, E., Hagerman, E., Barnes, E., Donoway, E., Pavlick, E., Rodolà, E., Lam, E., Chu, E., Tang, E., Erdem, E., Chang, E., Chi, E. A., Dyer, E., Jerzak, E., Kim, E., Manyasi, E. E., Zheltonozhskii, E., Xia, F., Siar, F., Martínez-Plumed, F., Happé, F., Chollet, F., Rong, F., Mishra, G., Winata, G. I., de Melo, G., Kruszewski, G., Parascandolo, G., Mariani, G., Wang, G. X., Jaimovitch-Lopez, G., Betz, G., Gur-Ari, G., Galijasevic, H., Kim, H., Rashkin, H., Hajishirzi, H., Mehta, H., Bogar, H., Shevlin, H. F. A., Schuetze, H., Yakura, H., Zhang, H., Wong, H. M., Ng, I., Noble, I., Jumelet, J., Geissinger, J., Kernion, J., Hilton, J., Lee, J., Fisac, J. F., Simon, J. B., Koppel, J., Zheng, J., Zou, J., Kocon, J., Thompson, J., Wingfield, J., Kaplan, J., Radom, J., Sohl-Dickstein, J., Phang, J., Wei, J., Yosinski, J., Novikova, J., Bosscher, J., Marsh, J., Kim, J., Taal, J., Engel, J., Alabi, J., Xu, J., Song, J., Tang, J., Waweru, J., Burden, J., Miller, J., Balis, J. U., Batchelder, J., Berant, J., Frohberg, J., Rozen, J., Hernandez-Orallo, J., Boudeman, J., Guerr, J., Jones, J., Tenenbaum, J. B., Rule, J. S., Chua, J., Kanclerz, K., Livescu, K., Krauth, K., Gopalakrishnan, K., Ignatyeva, K., Markert, K., Dhole, K., Gimpel, K., Omondi, K., Mathewson, K. W., Chiafullo, K., Shkaruta, K., Shridhar, K., McDonell, K., Richardson, K., Reynolds, L., Gao, L., Zhang, L., Dugan, L., Qin, L., Contreras-Ochando, L., Morency, L.-P., Moschella, L., Lam, L., Noble, L., Schmidt, L., He, L., Oliveros-Colón, L., Metz, L., Senel, L. K., Bosma, M., Sap, M., Hoeve, M. T., Farooqi, M., Faruqui, M., Mazeika, M., Baturan, M., Marelli, M., Maru, M., Ramirez-Quintana, M. J., Tolkiehn, M., Giulianelli, M., Lewis, M., Potthast, M., Leavitt, M. L., Hagen, M., Schubert, M., Baitemirova, M. O., Arnaud, M., McElrath, M., Yee, M. A., Cohen, M., Gu, M., Ivanitskiy, M., Starritt, M., Strube, M., Swkedrowski, M., Bevilacqua, M., Yasunaga, M., Kale, M., Cain, M., Xu, M., Suzgun, M., Walker, M., Tiwari, M., Bansal, M., Aminnaseri, M., Geva, M., Gheini, M., T, M. V., Peng, N., Chi, N. A., Lee, N., Krakover, N. G.-A., Cameron, N., Roberts, N., Doiron, N., Martinez, N., Nangia, N., Deckers, N., Muennighoff, N., Keskar, N. S., Iyer, N. S., Constant, N., Fiedel, N., Wen, N., Zhang, O., Agha, O., Elbaghdadi, O., Levy, O., Evans, O., Casares, P. A. M., Doshi, P., Fung, P., Liang, P. P., Vicol, P., Alipoormolabashi, P., Liao, P., Liang, P., Chang, P. W., Eckersley, P., Htut, P. M., Hwang, P., Miłkowski, P., Patil, P., Pezeshkpour, P., Oli, P., Mei, Q., Lyu, Q., Chen, Q., Banjade, R., Rudolph, R. E., Gabriel, R., Habacker, R., Risco, R., Millière, R., Garg, R., Barnes, R., Saurous, R. A., Arakawa, R., Raymaekers, R., Frank, R., Sikand, R., Novak, R., Sitelew, R., Bras, R. L., Liu, R., Jacobs, R., Zhang, R., Salakhutdinov, R., Chi, R. A., Lee, S. R., Stovall, R., Teehan, R., Yang, R., Singh, S., Mohammad, S. M., Anand, S., Dillavou, S., Shleifer, S., Wiseman, S., Gruetter, S., Bowman, S. R., Schoenholz, S. S., Han, S., Kwatra, S., Rous, S. A., Ghazarian, S., Ghosh, S., Casey, S., Bischoff, S., Gehrmann, S., Schuster, S., Sadeghi, S., Hamdan, S., Zhou, S., Srivastava, S., Shi, S., Singh, S., Asaadi, S., Gu, S. S., Pachchigar, S., Toshniwal, S., Upadhyay, S., Debnath, S. S., Shakeri, S., Thormeyer, S., Melzi, S., Reddy, S., Makini, S. P., Lee, S.-H., Torene, S., Hatwar, S., Dehaene, S., Divic, S., Ermon, S., Biderman, S., Lin, S., Prasad, S., Piantadosi, S., Shieber, S., Misherghi, S., Kiritchenko, S., Mishra, S., Linzen, T., Schuster, T., Li, T., Yu, T., Ali, T., Hashimoto, T., Wu, T.-L., Desbordes, T., Rothschild, T., Phan, T., Wang, T., Nkinyili, T., Schick, T., Kornev, T., Tunduny, T., Gerstenberg, T., Chang, T., Neeraj, T., Khot, T., Shultz, T., Shaham, U., Misra, V., Demberg, V., Nyamai, V., Raunak, V., Ramasesh, V. V., vinay uday prabhu, Padmakumar, V., Srikumar, V., Fedus, W., Saunders, W., Zhang, W., Vossen, W., Ren, X., Tong, X., Zhao, X., Wu, X., Shen, X., Yaghoobzadeh, Y., Lakretz, Y., Song, Y., Bahri, Y., Choi, Y., Yang, Y., Hao, S., Chen, Y., Belinkov, Y., Hou, Y., Hou, Y., Bai, Y., Seid, Z., Zhao, Z., Wang, Z., Wang, Z. J., Wang, Z., and Wu, Z. Beyond the imitation game: Quantifying and extrapolating the capabilities of language models. *Transactions on Machine Learning Research*, 2023. Featured Certification.

Tafjord, O., Dalvi, B., and Clark, P. ProofWriter: Generating implications, proofs, and abductive statements over natural language. In Zong, C., Xia, F., Li, W., and Navigli, R. (eds.), *Findings of the Association for Computational Linguistics: ACL-IJCNLP 2021*, pp. 3621–3634, Online, August 2021. Association for Computational Linguistics.

Tian, J., Li, Y., Chen, W., Xiao, L., He, H., and Jin, Y. Diagnosing the first-order logical reasoning ability through LogicNLI. In Moens, M.-F., Huang, X., Specia, L., and Yih, S. W.-t. (eds.), *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*, pp. 3738–3747, Online and Punta Cana, Dominican Republic, November 2021. Association for Computational Linguistics.

Voracek, V. Treatment of statistical estimation problems in randomized smoothing for adversarial robustness. *Advances in Neural Information Processing Systems*, 37: 133464–133486, 2024.

Wilie, B., Cahyawijaya, S., Ishii, E., He, J., and Fung, P. Belief revision: The adaptability of large language models reasoning. In *Proceedings of the 2024 Conference on Empirical Methods in Natural Language Processing*, pp. 10480–10496, 2024.

Xiao, X., Su, Y., Zhang, S., Chen, Z., Chen, Y., and Liu, T. Confidence in large language model evaluation: A bayesian approach to limited-sample challenges. *arXiv preprint arXiv:2504.21303*, 2025.

Yu, W., Jiang, Z., Dong, Y., and Feng, J. ReClor: A reading comprehension dataset requiring logical reasoning. In *International Conference on Learning Representations (ICLR)*, April 2020.

Zhong, W., Wang, S., Tang, D., Xu, Z., Guo, D., Wang, J., Yin, J., Zhou, M., and Duan, N. AR-LSAT: Investigating analytical reasoning of text. *arXiv preprint arXiv:2104.06598*, 2021.

Appendix

A. Generation Prompts

In our generation pipeline, we extensively use LLMs for symbolic manipulations and translations. Below, you will find the different prompts used to build *ReviseQA*.

Invariant Prompt: This prompt is used to edit the context without changing the conclusion.

Invariant Prompt

You are an expert in formal logic and Prover9. You will receive:

1. A list of assumptions (facts and rules).
2. An initial goal G, which is currently provable.
3. The negation of the goal ¬G, which is currently not provable.

Your objective is to edit the assumptions in a minimal way that preserves the current reasoning outcome:
– G must remain provable.
– ¬G must remain unprovable.

Your task is to produce a **non-simple minimal edit set** that:

• **Adds** logically redundant facts or rules that are already entailed by the original assumptions.
• **Rewrites** existing rules into logically equivalent forms (e.g., implication distribution, de Morgan's laws, etc.) without changing the semantics.
• **You are encouraged to use non-obvious but logically derivable rules** — statements that don't change the deductive power but offer alternative expressions.
• **Does not** remove any assumption, reorder rules, or rename variables.
• **Does not** introduce any new path that could derive ¬G or block the proof of G.

Ensure that after editing:
1. Forward reasoning still derives G.
2. Backward reasoning from ¬G still fails (¬G remains not derivable).

Step-by-step approach:
1. **Analyze** the assumptions and confirm that G is provable and ¬G is not.
2. **Apply** logically neutral and semantically safe edits as described above.
3. **Validate**:
- Use forward reasoning to prove G.
- Use backward reasoning from ¬G to confirm it cannot be derived.

##
Initial Assumptions:
{{assumptions}}

Initial Goal G:
{{initial_goal}}
##
Goal ¬G:
{{new_goal}}

Flip Prompt: This prompt is used to edit the context and flip the conclusion.

Flip Prompt

You are an expert in formal logic and Prover9. You will receive:

1. A list of assumptions (facts and rules).
2. An initial goal G, which is provable.
3. The negation of the goal ¬G, which is currently not provable.

Your objective is to **edit the assumptions** so that ¬G becomes provable (but G is not).

Your task is to produce a **non-simple minimal edit set** that:

• **Removes** only those facts or rules that directly enable a proof of G.
• **Negates** existing facts or rules as needed - never add ¬G as a standalone fact, though you may add a rule implying ¬G.
• **Adds** only the smallest number of new facts or rules (all derived from original assumptions) to **indirectly** derive ¬G.
• **You are encouraged to use non-obvious but logically derivable rules** — rules not explicitly present but inferable from the existing assumptions — to construct a valid proof of ¬G.
• **Ensures** that after editing:
1. Use forward reasoning to derive ¬G.
2. Use backward reasoning from G fails (G is no longer derivable).

Step-by-step approach:

1. **Analyze** the assumptions and confirm that G is provable and ¬G is not.
2. **Break** G's derivation by removing or negating the minimal set of those assumptions.
3. **Introduce** new facts or rules—based on original assumptions—to build a proof of ¬G.
4. **Validate**:
- Use forward reasoning to prove ¬G.
- Use backward reasoning from G to confirm it cannot be derived.

##
Initial Assumptions:
{{assumptions}}

Initial Goal G:
{{initial_goal}}
##
Goal ¬G:
{{new_goal}}

Uncertain to True Prompt: This prompt is used to edit the context when the conclusion is `UNCERTAIN` and wants to make it `TRUE`.

Uncertain to True Prompt

You are an expert in formal logic and Prover9. You will receive:

1. A list of assumptions (facts and rules).
2. A target goal G, which is currently not provable.
3. The negation of the goal ¬G, which is currently not provable.

Your objective is **to edit the assumptions** so that G becomes provable, but the negation ¬G remains unprovable.

Your task is to produce a **non-simple minimal edit set** that:

• **Adds** only the minimal number of new facts or rules (based strictly on the original assumptions) to derive G.
• **Does not enable** the derivation of ¬G in any form.
• **Never adds** G as a standalone fact, but you may add a rule that implies G.
• **You are encouraged to use non-obvious but logically derivable rules** — that is, rules that are logically entailed by the assumptions but not explicitly stated.
• **Ensures** that after editing:
1. Use forward reasoning to derive G.
2. Use backward reasoning from ¬G fails (¬G remains not derivable).

Step-by-step approach:
• **Analyze** the assumptions and verify that G is not currently provable.
• **Introduce** minimal additions (based on existing assumptions) to make G provable.
• **Avoid** any additions or changes that could imply or derive ¬G.
• **Validate**:
1. Use forward reasoning to prove G.
2. Use backward reasoning from ¬G to confirm it is not provable.

##
Initial Assumptions:
{{assumptions}}

Initial Goal G:
{{initial_goal}}
##
Goal ¬G:
{{new_goal}}

Forward Reasoning Prompt: This prompt is used when the conclusion of the edited context cannot be proven using *Prover9*.

Forward Reasoning Prompt

Prover9 was not able to prove **{{new_goal}}**, which is the intended goal.

Perform forward reasoning using the current assumptions to determine why **{{new_goal}}** is not derivable.

Identify the missing, conflicting, or overly permissive facts and rules that prevent the proof.

Then, propose **non-simple minimal edits** (i.e., avoid directly inserting the goal) to the assumptions that would allow **{{new_goal}}** to be correctly derived.

After applying the edits, re-run forward reasoning to confirm that **{{new_goal}}** is now provable and that **{{initial_goal}}** is no longer derivable.

Backward Reasoning Prompt: This prompt is used when the negation of the conclusion of the edited context can be proven using *Prover9*.

Backward Reasoning Prompt

Prover9 Proof:
{{proof}}

Prover9 was able to prove **{{initial_goal}}**, but this result is incorrect — it should not be provable.

Your task is to:

1. Carefully examine the Prover9 proof to identify the sequence of steps that led to the derivation of **{{initial_goal}}**.

2. Use **backward reasoning**, starting from **{{initial_goal}}**, to trace which facts and rules were used at each step and which assumptions directly or indirectly contributed to the conclusion.

3. Based on your analysis, propose a set of **non-simple minimal edits** to the assumptions that would break the proof chain — i.e., prevent **{{initial_goal}}** from being provable — **without simply deleting the goal or inserting its negation as a fact**.
- You may negate or adjust specific rules.
- You may remove enabling assumptions.
- You may introduce new rules that imply **{{new_goal}}**, but not **{{initial_goal}}**.

4. Finally, perform **forward reasoning** with the updated assumptions to confirm that:
- **{{initial_goal}}** is **no longer provable**.
- **{{new_goal}}** **is now derivable**.

Think step-by-step, and make sure your proposed changes are both logically valid and minimal.

Prover9 Syntax Translation Prompt: This prompt is used to translate the FOL of the edited context into *Prover9* syntax.

Prover9 Syntax Translation Prompt

Once you have applied the **non-simple minimal edits**, generate a valid Prover9 input along with its equivalent FOL syntax using the updated assumptions.
Do not remove or add any rules or facts beyond those specified in the non-simple minimal edits.

Your output must include:
- A complete ‘formulas(assumptions).‘ block with the updated facts and rules.
- A ‘formulas(goals).‘ block with the new goal: **{{new_goal}}**.
- Ensure that the original goal **{{initial_goal}}** is not included anywhere in the assumptions or goals.

Please provide the output in a valid JSON format, structured as follows:

```
json{
  "prover9_input": {
    "formulas(assumptions)": ["¬p_1(Novah)", "all x (p_3(x) → p_2(x))", ...],
    "formulas(goals)": ["¬p_2(Novah)", ...]
  },
  "fol_input": {
    "formulas(assumptions)": ["¬p_1(Novah)", "∀x (p_3(x) → p_2(x))", ...],
    "formulas(goals)": ["¬p_2(Novah)", ...]
  }
}
```

For the ‘fol_input‘, use the following standard **FOL symbols**:
- Universal quantifier: ‘∀‘
- Existential quantifier: ‘∃‘
- Negation: ‘¬‘
- Conjunction: ‘∧‘
- Disjunction: ‘∨‘
- Implication: ‘→‘
- Exclusive disjunction: ‘⊕‘
- Biconditional: ‘↔‘

If an expression uses XOR (⊕) in the FOL block (e.g., ‘p(x) ⊕ q(x)‘), translate it in the Prover9 block as ‘-(p(x) ¡-¿ q(x))‘.
Always enclose the entire negated biconditional in parentheses when it appears inside larger expressions, like conjunctions or implications.
For example:

- FOL: ‘(p(x) ⊕ q(x)) ∧(x)‘
- Prover9: ‘(-(p(x) < − > q(x))) & r(x)‘

Do not use Prover9 syntax (e.g., ‘all‘, ‘− >‘) in the ‘fol_input‘. Think step by step and make sure both inputs are consistent.

Natural Language Translation Prompt: This prompt is used to translate the edited FOL context into natural language.

Natural Language Translation Prompt

You are a Formal Logic, FOL-to-Natural-Language Translation Expert and JSON Schema Specialist.
When you respond, follow these instructions precisely:

Given:
1. Subject name (string)
2. Subject category (string)
3. Context with its original FOL formulas
4. Conclusion sentence with its FOL formula
5. A sequence of edits, each defined by:
- edit_number (integer)
- edited_context_fol: full list of FOL formulas after the edit (array of strings)
- edits_made: four lists (removed_facts, removed_rules, added_facts, added_rules), each containing FOL strings

Your job:
- Translate **every** FOL formula in edited_context_fol into a corresponding natural-language (NL) sentence, preserving the original meaning.
- Translate **every** FOL string in each of the four edits_made lists into a natural-language (NL) description.

Output:
A single JSON object matching this schema exactly (no additional keys or text):

```
json{
  "original_context": [string, ...],
  "original_context_fol": [string, ...],
  "conclusion": string,
  "conclusion_fol": string,
  "edits": [
    {
      "edit_number": integer,
      "edited_context_fol": [string, ...],
      "edited_natural_language_context": [string, ...],
      "edits_made": {
        "removed_facts": ["fol": string, "nl": string, ...],
        "removed_rules": ["fol": string, "nl": string, ...],
        "added_facts": ["fol": string, "nl": string, ...],
        "added_rules": ["fol": string, "nl": string, ...]
      }
    },
    ...
  ]
}
```

Once you have generated every ' "fol": ..., "nl": ... ' pair, perform a review pass:
1. Confirm each 'nl' sentence accurately and completely expresses the original 'fol'.
2. Ensure no formula is omitted or combined with another.
3. If you spot any mismatch or omission, correct the 'nl' so it matches the 'fol' exactly.
The output must be a valid JSON matching the schema—no extra keys and no prose outside the JSON.

Verification Prompt: This prompt is used to verify the generated examples. We used three LLMs as judges: *Gemini 2.5 Pro*, *GPT 4.1*, and *GPT o4 mini high*.

Verification Prompt

SYSTEM: You are an assistant that reasons before only answering True or False, identifying the mistake correctly so we can reflect on it later. Verify if the FOL statement matches the natural language.

USER: For the following pair, does the FOL statement correctly correspond to the natural language statement? Reason first, point out the mistake afterwards (can be none), and finally answer True or False. Do not be overly strict, have some kind of understanding. Your point is to look out for incorrect mappings, only. Think carefully.

FOL: {{**fol_formula**}}
NL: {{**nl_translation**}}

B. Evaluation Prompts

We evaluated different LLMs on our dataset. Below, you will find the different prompts used for evaluations.

Here, by "implicit" we denote the scenario in which we update the context implicitly, by embedding the edits within it without explicitly highlighting them. Instead, in "explicit" the edits are directly stated to the model.

COT Implicit Prompt: This prompt is used to evaluate LLMs on implicit edits using COT.

COT Implicit Prompt

Context: {{**context**}}

Question: {{**question_conclusion**}}

Options:
A) True
B) False
C) Uncertain

The correct option is:
"reasoning": {{**initial_reasoning**}},
"answer": {{**initial_answer**}}

Context: {{**edited_context**}}

Question: Does the context entail the conclusion: {{**conclusion**}}

Options:
A) True
B) False
C) Uncertain

Standard Implicit Prompt: This prompt is used to evaluate LLMs on implicit edits without using COT.

Standard Implicit Prompt

Context: {{**context**}}

Question: {{**question_conclusion**}}

Options:
A) True
B) False
C) Uncertain

The correct option is:
"reasoning": {{**initial_reasoning**}},
"answer": {{**initial_answer**}}

Context: {{**edited_context**}}

Question: Does the context entail the conclusion: {{**conclusion**}}

Options:
A) True
B) False
C) Uncertain

COT Explicit Prompt: This prompt is used to evaluate LLMs on explicit edits using COT.

COT Explicit Prompt

Context: {{**context**}}

Question: {{**question_conclusion**}}

Options:
A) True
B) False
C) Uncertain

The correct option is: "reasoning": {{**initial_reasoning**}},
"answer": {{**initial_answer**}}

Context:
removed facts: {{**removed_facts**}}
removed rules: {{**removed_rules**}}
added facts: {{**added_facts**}}
added rules: {{**added_rules**}}

Question: Does the context entail the conclusion: {{**conclusion**}}

Options:
A) True
B) False
C) Uncertain

Standard Explicit Prompt: This prompt is used to evaluate LLMs on explicit edits without using COT.

Standard Explicit Prompt

Context: {{**context**}}

Question: {{**question_conclusion**}}

Options:
A) True
B) False
C) Uncertain

The correct option is: "answer": {{**initial_answer**}}

Context:
removed facts: {{**removed_facts**}}
removed rules: {{**removed_rules**}}
added facts: {{**added_facts**}}
added rules: {{**removed_rules**}}

Question: Does the context entail the conclusion: {{**conclusion**}}

Options:
A) True
B) False
C) Uncertain

C. An Example of a Standard Explicit Prompt

Context: Lukas is vibrant. Lukas is not outgoing. Lukas values life. Lukas accepts his flaws. Lukas is passionate. Lukas embraces himself. If Lukas is vibrant, then he is either outgoing or authentic, but not both. If Lukas loves himself, then he may not necessarily value life, and if Lukas values life, then he may not necessarily love himself. Anyone who loves themselves or accepts their flaws has inner strength. If Lukas is authentic, then he is either passionate or confident, but not both. If someone embraces themselves, then they have inner strength and are beautiful. Lukas is either confident or deserves respect, but not necessarily both.

Question: Does the context entail the conclusion: Lukas deserves respect and is beautiful.

Options:
A) True
B) False
C) Uncertain

The correct option is:
"answer": True

Context:
removed facts: Lukas is passionate
removed rules: None
added facts: Lukas is not passionate
added rules: None

Question: Does the context entail the conclusion: Lukas deserves respect and is beautiful.

Options:
A) True
B) False
C) Uncertain

D. Additional Results

In this section, we report the precise numerical results in all our settings. For the calculation of the confidence intervals, it is common in the literature to use *approximately* valid confidence intervals, e.g., using a Gaussian approximation (Miller, 2024) or a Bayesian approach (Xiao et al., 2025). However, such approaches are valid only in the large-sample regime, that is, when confidence intervals are less needed. Instead, Voracek (2024) showed that it is possible to calculate *exact and valid* confidence intervals for binomial random variables, using a randomized procedure. So, we use this algorithm to calculate *valid* confidence intervals with an *exact* coverage of 95%, reporting lower and upper confidence bounds in parentheses.

D.1. Implicit Results with No Feedback

Model	Variant	Easy		Medium		Hard	
		COT	Standard	COT	Standard	COT	Standard
Claude 3.7 Sonnet	default	0.72 (0.66, 0.78)	0.68 (0.62, 0.74)	0.6 (0.53, 0.66)	0.54 (0.47, 0.60)	0.45 (0.38, 0.52)	0.44 (0.38, 0.51)
	thinking	0.80 (0.74, 0.85)	0.84 (0.78, 0.88)	0.68 (0.63, 0.74)	0.80 (0.74, 0.85)	0.62 (0.55, 0.68)	0.72 (0.66, 0.78)
Gemini 2.5 Flash	default	0.86 (0.81, 0.90)	0.91 (0.86, 0.94)	0.79 (0.73, 0.84)	0.87 (0.82, 0.91)	0.72 (0.66, 0.77)	0.79 (0.73, 0.84)
	thinking	0.92 (0.88, 0.95)	0.93 (0.89, 0.95)	0.86 (0.81, 0.90)	0.87 (0.82, 0.91)	0.78 (0.72, 0.83)	0.80 (0.74, 0.84)
GPT 4.1 mini	-	0.78 (0.72, 0.83)	0.74 (0.68, 0.80)	0.68 (0.62, 0.74)	0.62 (0.55, 0.68)	0.59 (0.52, 0.65)	0.49 (0.42, 0.56)
DeepSeek V3 Chat	-	0.75 (0.68, 0.80)	0.70 (0.63, 0.76)	0.62 (0.55, 0.68)	0.60 (0.53, 0.67)	0.47 (0.40, 0.54)	0.42 (0.36, 0.49)
Qwen 32B	-	0.83 (0.78, 0.88)	0.70 (0.63, 0.76)	0.66 (0.60, 0.72)	0.45 (0.38, 0.51)	0.41 (0.35, 0.48)	0.19 (0.14, 0.25)

D.2. Explicit Results with No Feedback

Model	Variant	Easy		Medium		Hard	
		COT	Standard	COT	Standard	COT	Standard
Claude 3.7 Sonnet	default	0.66 (0.59, 0.73)	0.55 (0.49, 0.62)	0.51 (0.44, 0.58)	0.42 (0.36, 0.49)	0.42 (0.35, 0.49)	0.32 (0.26, 0.39)
	thinking	0.70 (0.63, 0.76)	0.74 (0.68, 0.79)	0.53 (0.46, 0.60)	0.60 (0.54, 0.67)	0.41 (0.34, 0.48)	0.47 (0.41, 0.54)
Gemini 2.5 Flash	default	0.69 (0.63, 0.75)	0.70 (0.63, 0.76)	0.45 (0.38, 0.51)	0.43 (0.37, 0.50)	0.26 (0.20, 0.33)	0.22 (0.17, 0.28)
	thinking	0.87 (0.82, 0.91)	0.87 (0.82, 0.91)	0.71 (0.65, 0.77)	0.70 (0.64, 0.76)	0.55 (0.48, 0.61)	0.54 (0.47, 0.60)
GPT 4.1 mini	-	0.52 (0.46, 0.59)	0.55 (0.48, 0.62)	0.21 (0.16, 0.27)	0.22 (0.17, 0.28)	0.02 (0.01, 0.06)	0.05 (0.03, 0.09)
DeepSeek V3 Chat	-	0.64 (0.58, 0.71)	0.54 (0.47, 0.61)	0.32 (0.26, 0.39)	0.27 (0.21, 0.33)	0.14 (0.10, 0.19)	0.06 (0.03, 0.10)
Qwen 32B	-	0.78 (0.72, 0.83)	0.68 (0.61, 0.74)	0.50 (0.44, 0.57)	0.37 (0.30, 0.44)	0.25 (0.20, 0.31)	0.18 (0.13, 0.24)

D.3. Implicit Results with Feedback

Model	Variant	Easy		Medium		Hard	
		COT	Standard	COT	Standard	COT	Standard
Claude 3.7 Sonnet	default	0.72 (0.66, 0.78)	0.66 (0.60, 0.72)	0.60 (0.54, 0.67)	0.54 (0.48, 0.61)	0.48 (0.41, 0.55)	0.45 (0.38, 0.52)
	thinking	0.80 (0.75, 0.85)	0.77 (0.71, 0.82)	0.72 (0.65, 0.78)	0.70 (0.63, 0.76)	0.63 (0.56, 0.69)	0.60 (0.54, 0.67)
Gemini 2.5 Flash	default	0.87 (0.82, 0.91)	0.88 (0.83, 0.92)	0.82 (0.76, 0.87)	0.82 (0.77, 0.87)	0.75 (0.68, 0.80)	0.74 (0.68, 0.80)
	thinking	0.92 (0.88, 0.95)	0.91 (0.87, 0.95)	0.87 (0.82, 0.90)	0.86 (0.81, 0.90)	0.79 (0.73, 0.84)	0.79 (0.73, 0.84)
GPT 4.1 mini	-	0.76 (0.70, 0.81)	0.68 (0.61, 0.74)	0.65 (0.58, 0.71)	0.53 (0.47, 0.60)	0.54 (0.48, 0.61)	0.43 (0.35, 0.50)
DeepSeek V3 Chat	-	0.73 (0.66, 0.78)	0.70 (0.63, 0.76)	0.59 (0.52, 0.65)	0.56 (0.50, 0.63)	0.45 (0.38, 0.52)	0.41 (0.35, 0.48)
Qwen 32B	-	0.83 (0.78, 0.88)	0.81 (0.76, 0.86)	0.67 (0.60, 0.73)	0.66 (0.59, 0.72)	0.47 (0.41, 0.54)	0.49 (0.42, 0.55)

D.4. Explicit Results with Feedback

Model	Variant	Easy		Medium		Hard	
		COT	Standard	COT	Standard	COT	Standard
Claude 3.7 Sonnet	default	0.64 (0.58, 0.70)	0.57 (0.50, 0.63)	0.48 (0.41, 0.54)	0.40 (0.34, 0.47)	0.35 (0.29, 0.42)	0.30 (0.23, 0.36)
	thinking	0.71 (0.65, 0.77)	0.67 (0.61, 0.73)	0.57 (0.50, 0.63)	0.53 (0.47, 0.60)	0.44 (0.37, 0.51)	0.40 (0.33, 0.46)
Gemini 2.5 Flash	default	0.70 (0.63, 0.75)	0.70 (0.63, 0.75)	0.45 (0.38, 0.52)	0.45 (0.38, 0.52)	0.23 (0.18, 0.30)	0.26 (0.20, 0.32)
	thinking	0.86 (0.80. 0.90)	0.86 (0.80, 0.90)	0.68 (0.62, 0.74)	0.68 (0.62, 0.74)	0.48 (0.41, 0.54)	0.50 (0.43, 0.57)
GPT 4.1 mini	-	0.55 (0.48, 0.61)	0.52 (0.46, 0.59)	0.24 (0.19, 0.30)	0.21 (0.16, 0.27)	0.06 (0.03, 0.10)	0.03 (0.01, 0.07)
DeepSeek V3 Chat	-	0.64 (0.57, 0.70)	0.54 (0.47, 0.61)	0.32 (0.26, 0.39)	0.27 (0.21, 0.33)	0.12 (0.08, 0.18)	0.06 (0.03, 0.10)
Qwen 32B	-	0.78 (0.72, 0.83)	0.68 (0.61, 0.74)	0.50 (0.43, 0.56)	0.41 (0.34, 0.48)	0.21 (0.16, 0.27)	0.15 (0.11, 0.21)

E. Models and Cost

For our paper, we used several open-source and closed-source LLM models using the OPENROUTER API. The total cost of generating our dataset and evaluating different models was approximately $3,500.