The Curious Case of the Unreliable Decipherer: What LLMs struggle with while explaining Formal Proofs

Anonymous ACL submission

Abstract

Formal methods—such as symbolic proofs—offer a principled way to force Large Language Models (LLMs) to reason more reliably. However it is unclear whether LLMs can actually use these proofs to generate faithful, and yet human understandable explanations? We introduce `ProofTeller`, a new benchmark to evaluate this capability. On a new dataset with over 68,000 human-annotated tokens, we evaluate several LLMs on three tasks: identifying key proof steps, summarizing the proofs, and creating a user-targeted message. Across tasks and settings, both automated metrics and human evaluation reveal a critical reliability gap: LLMs over-emphasize steps near the final conclusion, while humans draw on evidence distributed throughout the proof. These findings expose a fundamental mismatch between the reasoning strategies of current LLMs and those of humans, underscoring the need for approaches that enable LLMs to employ formal proof methods reliably while faithfully generating a comprehensible reasoning chain.

1 Introduction

Deploying large language models (LLMs) in real-world critical domains from medical applications (Bang et al., 2025) to autonomous vehicles (Cui et al., 2024) to critical network infrastructure (Manocchio et al., 2024) warrants ensuring reliable and consistent outputs. In-context learning (Brown et al., 2020) provides LLMs with remarkable abilities to solve tasks explained in natural language, and subsequent developments like retrieval-augmented generation (RAG) (Lewis et al., 2020), have further enhanced their utility. However, LLMs outputs are not always reliable (Abbasi Yadkori et al., 2024). They can produce inconsistent answers if information is presented in slightly different ways (Sclar et al., 2024; Elazar et al., 2021) and are threfore prone to generating *plausible but unfaithful* explanations that do not reflect the true drivers of their predictions (Turpin et al., 2023).

In the past year, various "thinking" LLMs such as DeepSeek-R1 (Guo et al., 2025), Qwen3 (Yang et al., 2025), Gemini-2.5 (Comanici et al., 2025) have shown better reasoning capabilities empirically. However even these reasoning models fail to use explicit algorithms, generating inconsistent outputs (Shojaee et al., 2025). This leads to unwanted hallucinations and presents a significant barrier to LLM usage in applications where accuracy and reliability are paramount.

To mitigate these consistency issues and reduce hallucinations, one promising approach is to use formal methods to inject reliability in LLMs, such as by providing symbolic proofs as input context. Such hybrid systems can leverage the rigorous and verifiable nature of formal logic, combined with the fluency of LLMs (Kassner et al., 2021). However, as Turpin et al. (2023) highlights the nature of faithfulness in modern LLMs continues to be brittle and sensitive to the given input, and thus their reasoning can be swayed by biasing features in the input. Furthermore, symbolic proofs sometimes do not contain all the necessary information to explain a scenario perfectly (Mondorf and Plank, 2024). In such cases, LLMs has to infer the missing components by relying on knowledge gleaned during pretraining. This can lead to hallucinations if this "knowledge" is not aligned with the given input (Sun et al., 2025; Xie et al., 2024).

These observations point to a concrete question:

> *Can LLMs reliably parse a formal proof, distill the key reasoning steps, and restate them faithfully for a non-expert?*

Answering this question lets us probe—within a single, controlled setting— all failure modes identified above. If a model misunderstands or omits

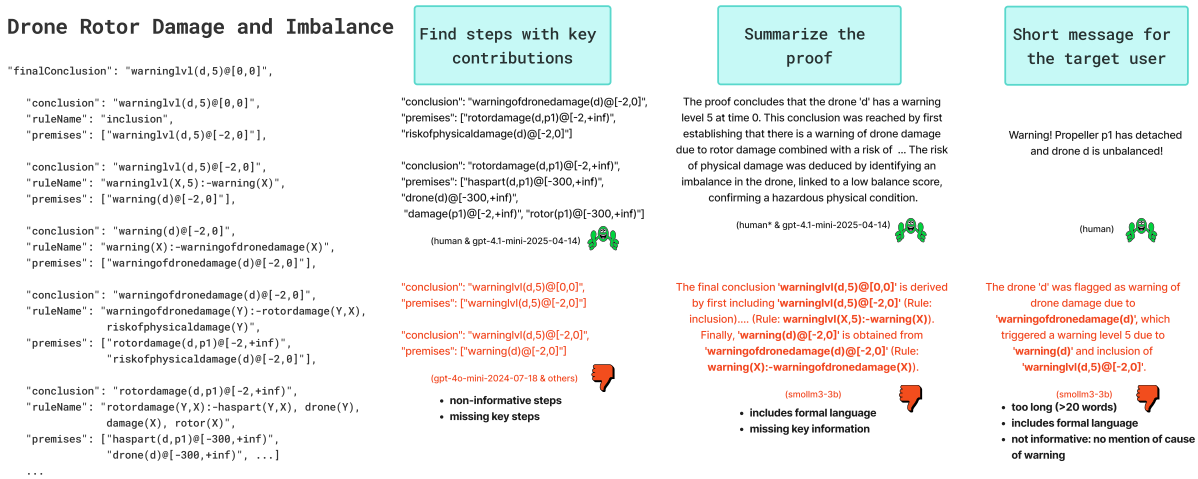

Figure 1: An example datapoint showcasing limitations of LLMs on our benchmark tasks

critical deductive steps we identify the lack of correctness, if it over or under-states what the proof entails we can identify hallucination or inconsistency and finally if it cannot translate the outcome to the user, we witness a breakdown in faithfulness.

We introduce `ProofTeller`, a benchmark expressly designed to stress-test these capabilities. Concretely, each instance in `ProofTeller` asks an LLM to

1. **Highlight** the key steps that contribute the most to the final verdict of the proof,
2. **Summarize** long symbolic proofs faithfully,
3. **Explain** the result in a *concise* message tailored to a layman end user.

We provide a rigorous evaluation of 9 LLMs, including 7 open weights and 2 proprietary LLMs. Additionally, we also conduct numerous human evaluations studies to further showcase the gaps between human and LLM abilities. With this benchmark and our evaluation framework grounded in realistic scenarios, we aim to establish a principled yardstick for measuring and ultimately improving the reliability and consistency of LLMs in real world critical domains.

2 Related Work

Symbolic Text Understanding Many recent works evaluate the symbolic reasoning of LLMs. For example, (Shalyt et al., 2025) tests LLM's abilities for mathematical problem-solving using symbolic perturbations, (Kulkarni et al., 2025) makes use of SQL for robust tabular reasoning, and (Vo et al., 2025) evaluates LLMs for an abstract causal discovery task from natural language text. While these works focus on *solving* or *discovering* symbolic relationships using LLMs, our benchmark focuses on evaluating LLM's ability to *explain* formal proofs.

Data-To-Text Generation Our work also shares similarity with data-to-text (D2T) generation since we also generate natural language text from structured data (i.e., JSON proofs). In D2T literature, ToTTo (Parikh et al., 2020) benchmark emphasizes the faithful generation of text from tables. Another such benchmark is DART (Nan et al., 2021), which focuses on generating text from complex RDF triples. Our work introduces a different structured input (i.e., logic proofs), which requires models to understand logical operators to create a summary and a user-targeted message.

Explaining Logical Proofs We find our work directly contributing to the field of explaining logical proofs. We find (Colombo et al., 2025) to be the most related to our work as it has a similar input format of formal proofs and output contains summaries. It uses LLMs to fill in templates to explain Datalog inferences in the financial domain. In contrast, our benchmark evaluates LLMs on multiple tasks including summarization.

3 Benchmark Creation

We start by generating different types of description logic (DL) proofs for all three domains. Our proof data come from three distinct domains: (1) Biology, (2) Food & Recipes, and (3) Critical Situations in Drones. For conciseness purposes, we refer to these domains as (1) Biology, (2) Recipe, and (3) Drone throughout the remainder of the text. For

this benchmark, we explore real-world scenarios and user groups that are different for each domain. For each target domain, we randomly sample 50 proofs for annotation. In this section, we describe our benchmark creation process step by step.

3.1 Generating DL proofs

Table 1 provides quantitative information about the domains and the extracted proofs. Although critical situations in drones are formally specified in DatalogMTL rather than DL, we adopt DL terminology throughout this work for consistency—for instance, referring to *rules* as *axioms*. Proof size is the number of inference steps, see "`steps`" in the proof in Figure 1. A full-sized example proof for each of these domains is provided in Appendix B.

Biology. Our first domain is based on the Cell Line Ontology[1], a community-driven ontology developed to standardize and integrate cell line information and support computer-assisted reasoning.

Recipe. We created a food and recipe ontology[2], a formal semantic model to represent and reason about culinary recipes, dietary restrictions, and allergen content. This ontology builds upon established recipe and food ontologies (Qi et al., 2018; Dooley et al., 2018), with the extracted proofs specifically related to the added recipes.

Drone. The drone ontology[2] models complex situations occurring during a drone flight. It uses metric temporal operators as well as numerical predicates. The associated proofs incorporate different urgency levels to prioritize critical situations.

3.2 Target User Group and Scenarios

Biology: a 10th-grade student learning about the characteristics of various cells.

Recipe: a 6th-grade student learning about food allergens and classification of ingredients into vegan, vegetarian and non-vegetarian.

Drone: a drone pilot monitoring an autonomous drone, who needs help in understanding current or future critical situations.

3.3 Task Annotation

To annotate these proofs for all three tasks, we recruited two graduate-level students. We divide the task-specific annotation into two phases: Pilot and Main. In the pilot phase, we gave each human annotator 10 proofs per domain and asked them to annotate the generated proofs for the following tasks: (i) identify the key contributing steps to the conclusion from the proof (maximum up to 3 steps), (ii) summarize proofs, and (iii) generate a short user-targeted message.

At the end of the pilot phase, we asked our expert annotator to review the pilot annotations from both our annotators and share detailed feedback on their annotations. We then asked our student annotators to incorporate the feedback. This feedback particularly helped align the most contributing steps between the student annotators. Finally, we divided the remaining 120 proofs equally among them for individual annotations. Throughout the annotation process, the student annotators were prohibited from using any AI tools for annotation purposes. By the end of this annotation process, we collected over 68,000 human-annotated tokens[3] across domains and tasks. We provide detailed annotation statistics in Table 2.

We provide the final version of our annotation guidelines in the Appendix G. We plan to release our full benchmark dataset and code with the camera-ready version of this work.

4 Experiments

In this section, we describe our experimental setup for benchmarking LLM performance. We provide an LLM with proofs and task-specific prompts to generate answers for each task. Given the connected nature of tasks (i.e., providing contributing steps in the context may help summarize the proof), it is important to test LLMs with and without providing the output of all the previous tasks as part of the context.

4.1 Prompting strategy

Our prompting setup contains 4 prompt variants as shown in Table 3. We write all our prompts to be clear and task-specific, following the principles of effective prompt engineering to elicit structured, reliable outputs (Levy et al., 2024). The exact prompts used for each experimental condition are detailed in Appendix C.

Our baseline prompting setup varies along two axes: the number of examples provided to the

[1] https://obofoundry.org/ontology/clo.html

[2] available in the supplementary material

[3] We use GPT-4.1-mini tokenizer to count the number of tokens throughout our work.

Domain	Ontology		Proofs	
	#Predicates (Arity)	#Axioms	#Proofs (DL)	Proof Size
Biology	43,327 (unary), 249 (binary)	72,830	2,429 ($\mathcal{EL}$)	4–100
Recipe	35,369 (unary), 195 (binary)	52,248	74 ($\mathcal{EL}$), 12 ($\mathcal{ALC}$)	4–100
Drone	201 (of maximal arity 4)	332	50 (DatalogMTL)	9–59

Table 1: Metrics of Ontologies and Logical Proofs. $\mathcal{EL}$ is a lightweight Description Logic and $\mathcal{ALC}$ is a more complex Description Logic. More details on these description logics are available in Appendix A.

Domain	Steps (in tokens)	Steps (in words)	Summary (in tokens)	Summary (in words)	Target User Msg (in tokens)	Target User Msg (in words)
Biology	13331	5530	5691	4674	900	698
Drone	14320	1315	4080	3326	878	688
Recipe	24488	10352	3858	3254	903	649
Total	52139	17197	13629	11254	2681	2035

Table 2: Token and word statistics for human-annotated proofs, summaries, and target user messages.

Strategy	Task Examples	Context Trail
Zero-shot atomic	No	No
Zero-shot chained	No	Yes
One-shot atomic	Yes	No
One-shot chained	Yes	Yes

Table 3: Baseline prompting strategies

model (i.e., zero-shot vs. one-shot) and the conversational structure of the interaction (i.e., atomic single-turn vs. chained multi-turn). This design allows us to systematically investigate the effectiveness of in-context learning and conversational history on task performance.

While in the atomic single-turn setup the order of tasks may not matter, it might matter in the multi-turn setup. We tried multiple different task orders and found that the following order performed better empirically: (1) find the most contributing steps, (2) summarize the proof and (3) provide a short message for the target user. We note that finding the most contributing steps is the first step in the chain and we also do not provide any examples for this task. For the same, the output of this task remains the same across all our prompting strategies.

In the chained prompting variants, the overall input length can become much longer as the conversational history grows. Hence, we limit our experiments to one-shot prompting for the practical given the longer input size (refer to Table 5).

4.2 LLM Coverage

We test a range of recent LLMs, including state-of-the-art proprietary models and several open-source models of varying sizes (refer to Table 4).

Model	Open	Thinking
`GPT-4.1-mini`	No	No
`GPT-4o-mini`	No	No
`SmolLM3-3B`	Yes	No[4]
`Llama-3.1-8B`	Yes	No
`Mistral-3.2-24B`	Yes	No
`Llama-3.3-70B`	Yes	No
`Qwen3-8B`	Yes	Yes
`Magistral-24B`	Yes	Yes
`Qwen3-32B`	Yes	Yes

Table 4: Information about the LLMs used

Here, LLMs such as `qwen3-8b`, `magistral-24b` and `qwen3-32b` inherently perform CoT reasoning via thinking tokens, effectively serving as implicit CoT baselines.

4.3 Implementation Details

We use `vLLM` (Kwon et al., 2023) to perform inference on all open-source models in their native precision (i.e., `fp16/bf16`). For each model, we use the recommended inference parameters (refer to the Appendix D) provided in their model card. We use 4x H100 GPUs to perform all our experiments with open-source models. For the closed-source

[4]We use the non-thinking variant here.

Domain	Minimum Tokens	Maximum Tokens	Average Tokens
Biology	296	2612	1180
Drone	466	3267	1307
Recipe	1272	4058	2021

Table 5: Number of tokens in proofs per domain

models, we use `OpenAI` hosted API services. For both open and closed-source models, we prompt LLMs to generate a JSON response and use a fixed seed value for reproducible outputs.

4.4 Metrics

Steps similarity To quantitatively evaluate the similarity in choosing the most contributing steps, we measure the average depth of the selected proof steps within the reference proof. We represent each proof as a Directed Acyclic Graph (DAG), $G = (V, E)$, where vertices $v \in V$ are the inferences and a directed edge $(u, v) \in E$ indicates that the conclusion of v is a premise for u.

The depth of a step v, denoted $d(v)$, is the length of the longest path from the proof's final conclusion (v_{root}) to v. To standardize this measure across different proofs, we compute a normalized depth, $\hat{d}(v)$, as:

$$\hat{d}(v) = \frac{d(v)}{\max_{u \in V} d(u)}$$

This ensures that $\hat{d}(v) \in [0, 1]$, where 0 corresponds to the root (final conclusion) and 1 to the deepest leaf nodes (asserted conditions). For a given explanation consisting of a set of selected steps $S = \{s_1, \ldots, s_k\}$, we calculate the mean normalized depth $\bar{d}(S)$:

$$\bar{d}(S) = \frac{1}{k} \sum_{i=1}^{k} \hat{d}(s_i)$$

By analyzing the distributions of $\bar{d}(S)$ for each LLM and human annotators, we can identify biases in their selection preferences and measure their alignment with human reasoning patterns.

Summary & Target User Message We evaluate similarity between the LLM-generated text and the human annotations using the following standard metrics, such as `BLEU` (Papineni et al., 2002), `ROUGE-L` (Lin, 2004), `chrF++` (Popović, 2017), and `BERTScore` (Zhang et al., 2020).

5 Results

Key steps similarity Figure 2 presents the distributions of the mean normalized step depth for most contributing steps identified by humans and LLMs across all domains. This result remains the same across all four baseline strategies, since it is the first step in the chain, and we also do not provide any examples for this task.

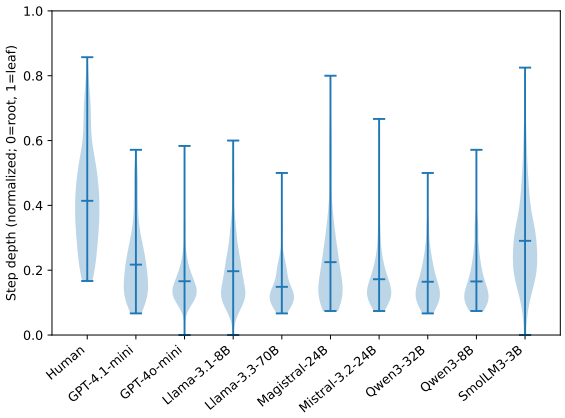

Figure 2: Most contributing steps distribution (averaged across domains)

Human-identified steps exhibit the highest mean step depth (≈ 0.4) and the widest distribution, covering nearly the entire normalized range. This indicates that humans do not adhere to a single fixed strategy. Instead, they flexibly select steps from all levels of the proof graph, from high-level conclusions to foundational premises. In contrast, most LLMs display a strong and consistent bias toward low-depth steps, indicating a preference for inferences that are structurally close to the final conclusion. We also observe that no model successfully replicates the human distribution.

For this task, `SmolLM3-3B` comes closest to the human average mean depth, outperforming even larger models such as `Magistral-24B` and `GPT-4.1-mini`, which rank second and third on this task, respectively. One major limitation we observe in `SmolLM3-3B`, `GPT-4o-mini`, and `Llama-3.1-8B` is that they sometimes pick the final conclusion step itself as the most contributing step. We then dive deeper to evaluate if these patterns are consistent across domains.

We observe clear differences in distribution patterns across domains. In the Biology domain (Figure 3), human-identified steps have a lower mean depth compared to other domains, indicating a preference for steps closer to the final conclusion, while

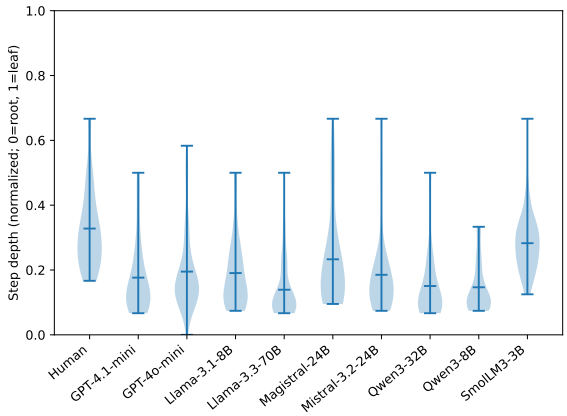

Figure 3: Most contributing steps distribution (Biology)

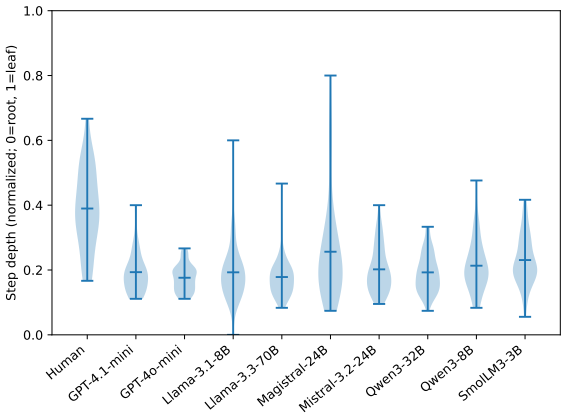

Figure 5: Most contributing steps distribution (Recipe)

still maintaining a wide distribution. Among the LLMs, `SmolLM3-3B` again exhibits a mean depth closest to the human average in this specific domain.

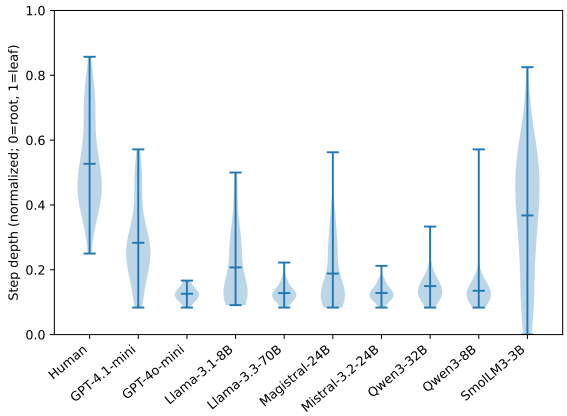

Figure 4: Most contributing steps distribution (Drone)

In the Drone domain, as shown in Figure 4, human-selected steps have the highest mean depth across all domains (≥ 0.5) and a broader distribution. On the contrary, all LLMs exhibit a strong preference for low-depth steps, creating the most significant gap between human and model distributions observed in our study. While `SmolLM3-3B` remains the best-performing model in terms of matching the mean, it also at times picks the final conclusion step as the most contributing one.

Figure 5 shows the distributions for the Recipe domain. The human distribution is similar to the cross-domain average, with a mean depth of ≈ 0.4 and noticeable variance. Most LLMs again gravitate towards steps structurally close to the conclusion. In this domain, `Magistral-24B` outperforms `SmolLM3-3B`, though all the LLMs fall short of replicating the breadth and average depth of human-selected steps.

In conclusion, the normalized step depth metric reveals a fundamental difference between human flexibility and LLM bias. While humans select key contributing steps from all levels of a reasoning chain, LLMs consistently favor low-depth steps near the final conclusion. This opens up another future research opportunity in developing LLMs that can replicate human understanding, leading to a more human-aligned identification of pivotal reasoning steps.

Summary & Target User Msg similarity We evaluate the LLMs' performance against human-written references using BLEU, ROUGE-L, chrF++, and BERTScore. The performance under the one-shot atomic strategy is nearly identical, with the chained context providing a marginal overall improvement. The results for the rest three prompting strategies are available in Appendix E.

The results for the one-shot chained strategy are presented in Figure 6. These results suggest that, for the *summary* task, `Llama-3.1-8B` and `Mistral-3.2-24B` consistently achieve the highest scores across all four metrics. However, for the more constrained *target user message* task, `GPT-4o-mini` emerges as one of the top-performing LLMs along with `Mistral-3.2-24B`. In contrast, LLMs like `GPT-4.1-mini` and `SmolLM3-3B` generally rank lower on these generation tasks compared to their performance on the step selection task.

6 Human Evaluation

While automated metrics like BLEU, ROUGE, and BERTScore offer scalable evaluation, they are lim-

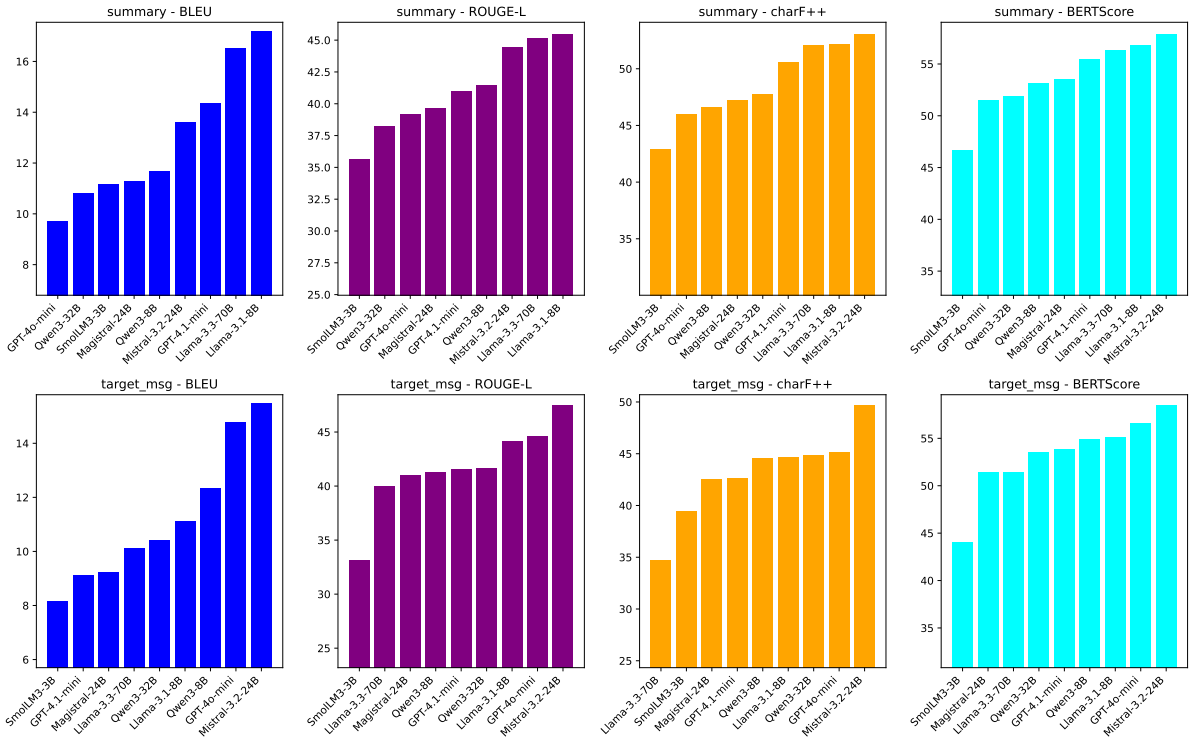

Figure 6: Automated metrics of the best performing strategy (One-shot chained)

Model Name	Summary criteria				Target Message criteria		
	Conciseness	Coverage	Faithfulness	Readability	Appropriateness	Coverage	Faithfulness
Llama-3.1-8B	4.07	3.65	3.76	4.05	3.54	3.69	3.67
Mistral-3.2-24B	4.01	4.63	3.86	3.83	3.85	4.19	4.11
GPT-4.1-mini	3.79	4.60	4.00	3.57	3.89	4.33	4.24
Llama-3.3-70B	4.14	3.84	4.00	3.84	3.62	3.46	4.44
Human	**4.55**	**4.80**	**4.80**	**4.35**	**4.59**	**4.47**	**4.96**

Table 6: Averaged human evaluation scores across all domains and annotators

ited in their ability to assess critical qualitative aspects of text generation. These metrics often fail to capture nuances of *faithfulness*, *readability*, and *appropriateness*, which are vital for user-facing summaries and messages. The results in Figure 6 show that top-performing models, such as `Llama-3.1-8B` and `Mistral-3.2-24B`, achieve very similar scores. Automated metrics alone make it difficult to determine if these small numerical differences translate into meaningful improvements in quality or to verify if the generated content is factually correct and truly useful.

To address these limitations and provide a more robust assessment, we conduct a human evaluation study. We select a diverse set of models for comparison against a Human baseline: the two top performers according to automated metrics (`Mistral-3.2-24B` and `Llama-3.1-8B`), a larger model from the same family (`Llama-3.3-70B`), and a recent proprietary model (`GPT-4.1-mini`). This selection allows us to validate whether the top automated scores correspond to actual qualitative improvements, and to explore the performance differences between various model types and sizes. To carry out the evaluation, we recruit five human annotators to rate the outputs for both tasks, using the following task-specific criteria.[5]

Summary Criteria The evaluators rate the summary based on the following aspects:

Faithfulness - assesses the degree to which the reference proof factually supports every statement in the summary.

Readability - judges the summary based on its clarity, grammar, and ease of comprehension.

Conciseness assesses if the summary only contains essential information or not.

[5]The human evaluation guide is available in Appendix G.

Coverage determines if the summary captures all key reasoning steps and the main conclusion.

Target Message Criteria The evaluators rate the target user message based on the following aspects. **Faithfulness** assesses the degree to which the reference proof factually supports the target message. **Appropriateness** judges the overall suitability of the message for its target audience, combining aspects of faithfulness, clarity, and conciseness. **Coverage** determines if the summary target message contains the main conclusion and at least one main reason.

6.1 Results

The human evaluation statistics are presented in Table 6. These indicate qualitative differences between the LLMs and Human annotators. Based on this evaluation, GPT-4.1-mini and Mistral-3.2-24B are the best-performing LLMs overall and they score higher compared to the Llama family LLMs. For the summary task, GPT-4.1-mini and Mistral-3.2-24B obtained the highest scores for the Coverage criteria, but they fall short on the Conciseness criteria. This indicate the verbose nature of these LLMs. We confirm this by plotting average summary and target message lengths for all the LLMs. The plots are available in Appendix F. In the Target Message task, GPT-4.1-mini achieves the highest scores across all three criteria: Appropriateness, Coverage, and Faithfulness.

These findings diverge from the automated evaluation results shown in Figure 6. While automated metrics ranked Llama-3.1-8B and Mistral-3.2-24B as having similar top-tier performance, human annotators rated Llama-3.1-8B lower than both Mistral-3.2-24B and GPT-4.1-mini. This suggests a limitation of automated metrics in capturing nuanced aspects of text quality and demonstrates the utility of human judgment for such assessments.

We finally assess the reliability of the human evaluation by calculating inter-annotator agreement using Fleiss' Kappa (κ). The results indicate that the five annotators reached a substantial level of agreement, which supports the reliability of our findings. For the Summary criteria, annotators achieved almost perfect agreement for Faithfulness ($\kappa = 0.813$), and substantial agreement for Readability ($\kappa = 0.677$) and Conciseness ($\kappa = 0.721$). Agreement for Coverage ($\kappa = 0.442$) was moderate, suggesting a higher degree of subjectivity for this criterion. For the Target Message criteria, all three aspects showed substantial agreement: Faithfulness ($\kappa = 0.774$), Appropriateness ($\kappa = 0.678$), and Coverage ($\kappa = 0.656$). These agreement scores demonstrate that annotators applied the evaluation criteria consistently and that the criteria themselves were well defined.

6.2 Error Analysis

In order to understand the ratings a bit better, we asked our human evaluators to provide us with the reasoning for the scores for 5 low-scoring samples per domain per LLM. We present our findings here.

Summary We observe that almost all the models use some technical terms from the proof verbatim, especially Mistral-3.2-24B and GPT-4.1-mini. This leads to lower readability scores in general. For llama-3.1-8b, we observed a consistent pattern of incorrectly using the term "equivalent" when the proof demonstrates a subclass relationship. This recurring issue indicates a potential weakness in understanding logical operators.

Target user message We observed that for the GPT-4.1-mini uses left-branching sentence structure, which increases memory load and scores lower on the appropriateness criteria. An example target message showcasing this issue is – Because saucy shepherd pie has carrots as ingredients, it needs special allergen labels about carrots. Llama-3.1-8B suffers again from its inability to understand logical operators. Interestingly, Llama-3.3-70B produces very short target messages (i.e., Cell derivation from Mus musculus) and achieves lower ratings for coverage criteria. This observation is also supported by Figure 21 in the Appendix.

7 Conclusion

We introduce `ProofTeller`, a benchmark to evaluate LLM reliability in explaining formal proofs via key step identification, summarization, and user messaging. Our experiments with nine LLMs reveal reliability gaps. We find that LLMs consistently favor low-depth steps near the conclusion, whereas humans select steps from all over the proof. This suggests LLMs lack a holistic understanding of the reasoning chain. Further, our human evaluation of summarization tasks highlights qualitative deficiencies in faithfulness and conciseness not captured by automated metrics. These findings demonstrate that even modern LLMs struggle to faithfully interpret and communicate the essence of a formal proof reliably.

Limitations

Experimental limitations While we took a systematic approach, our exploration of the vast prompt engineering space was limited. We did our initial testing with three distinct seed variations, which may not fully capture the possible variability in output. Furthermore, we restricted prompt variations for each task to five, potentially overlooking other effective phrasings or structures that could yield superior performance. Finally, we confined the initial evaluation of the system prompt's influence to three language models before finalizing the one used for this work, meaning findings may not be consistent across all possibly suited system prompts.

Scope limitation Our work is limited to three domains within DL and DatalogMTL formalisms. Moreover, to maintain the original notations, the proof syntax employs logic-specific Unicode symbols (e.g. $\sqsubseteq$, $\boxplus$) and specialized terminology (e.g. "`eliminate`" and "`Intersection Composition`") requiring LLMs to first recognize their semantic meaning before interpreting the logical implications.

Evaluation Subjectivity Evaluation of explanation quality can be inherently subjective, especially with respect to aspects such as appropriateness and faithfulness. Different annotators may interpret the relevance and accuracy of an explanation in diverse ways, leading to potential variability in the assigned scores. To mitigate this, we provided clear guidelines and examples, but acknowledge that some level of subjectivity is unavoidable in human evaluations of natural language explanations.

Ethics Statement

All annotators involved in the evaluation process are either co-authors of this paper or were fairly compensated for their time, receiving above the minimum wage of 14.5 EUR per hour. This ensures ethical standards in data annotation and helps maintain the quality and reliability of the evaluation results.

References

Yasin Abbasi Yadkori, Ilja Kuzborskij, András György, and Csaba Szepesvari. 2024. To believe or not to believe your llm: Iterative prompting for estimating epistemic uncertainty. *Advances in Neural Information Processing Systems*, 37:58077–58117.

Christian Alrabbaa, Franz Baader, Stefan Borgwardt, Patrick Koopmann, and Alisa Kovtunova. 2020. Finding small proofs for description logic entailments: Theory and practice. In *LPAR23. LPAR-23: 23rd International Conference on Logic for Programming, Artificial Intelligence and Reasoning*, volume 73 of *EPiC Series in Computing*, pages 32–67. EasyChair.

Christian Alrabbaa, Stefan Borgwardt, Tom Friese, Patrick Koopmann, Julián Méndez, and Alexej Popovic. 2022a. On the eve of true explainability for OWL ontologies: Description logic proofs with evee and evonne. In *Proceedings of the 35th International Workshop on Description Logics (DL 2022) co-located with Federated Logic Conference (FLoC 2022), Haifa, Israel, August 7th to 10th, 2022*, volume 3263 of *CEUR Workshop Proceedings*. CEUR-WS.org.

Christian Alrabbaa, Stefan Borgwardt, Patrick Koopmann, and Alisa Kovtunova. 2022b. Finding good proofs for answers to conjunctive queries mediated by lightweight ontologies. In *Proceedings of the 35th International Workshop on Description Logics (DL 2022) co-located with Federated Logic Conference (FLoC 2022), Haifa, Israel, August 7th to 10th, 2022*, volume 3263 of *CEUR Workshop Proceedings*. CEUR-WS.org.

Franz Baader, Ian Horrocks, Carsten Lutz, and Uli Sattler. 2017. *An Introduction to Description Logic*. Cambridge University Press.

Yejin Bang, Ziwei Ji, Alan Schelten, Anthony Hartshorn, Tara Fowler, Cheng Zhang, Nicola Cancedda, and Pascale Fung. 2025. Hallulens: Llm hallucination benchmark. *arXiv preprint arXiv:2504.17550*.

Stefan Borgwardt, Vera Demberg, Mayank Jobanputra, Alisa Kovtunova, and Duy Nhu. 2024. Explaining critical situations over sensor data streams using proofs and natural language. In *Proceedings of the 37th International Workshop on Description Logics (DL 2024), Bergen, Norway, June 18-21, 2024*, volume 3739 of *CEUR Workshop Proceedings*. CEUR-WS.org.

Sebastian Brandt, Elem Güzel Kalayci, Vladislav Ryzhikov, Guohui Xiao, and Michael Zakharyaschev. 2018. Querying log data with metric temporal logic. *J. Artif. Intell. Res.*, 62:829–877.

Tom Brown, Benjamin Mann, Nick Ryder, Melanie Subbiah, Jared D Kaplan, Prafulla Dhariwal, Arvind Neelakantan, Pranav Shyam, Girish Sastry, Amanda Askell, and 1 others. 2020. Language models are few-shot learners. *Advances in neural information processing systems*, 33:1877–1901.

Andrea Colombo, Teodoro Baldazzi, Luigi Bellomarini, Emanuel Sallinger, and Stefano Ceri. 2025. Template-based explainable inference over high-stakes financial knowledge graphs. In *Proceedings 28th International Conference on Extending*

Database Technology, EDBT 2025, Barcelona, Spain, March 25-28, 2025, pages 503–515. OpenProceedings.org.

Gheorghe Comanici, Eric Bieber, Mike Schaekermann, Ice Pasupat, Noveen Sachdeva, Inderjit Dhillon, Marcel Blistein, Ori Ram, Dan Zhang, Evan Rosen, and 1 others. 2025. Gemini 2.5: Pushing the frontier with advanced reasoning, multimodality, long context, and next generation agentic capabilities. *arXiv preprint arXiv:2507.06261*.

Can Cui, Yunsheng Ma, Zichong Yang, Yupeng Zhou, Peiran Liu, Juanwu Lu, Lingxi Li, Yaobin Chen, Jitesh H Panchal, Amr Abdelraouf, and 1 others. 2024. Large language models for autonomous driving (llm4ad): Concept, benchmark, experiments, and challenges. *arXiv preprint arXiv:2410.15281*.

Damion M. Dooley, Emma J. Griffiths, Gurinder S. Gosal, Pier L. Buttigieg, Robert Hoehndorf, Matthew C. Lange, Lynn M. Schriml, Fiona S. L. Brinkman, and William W. L. Hsiao. 2018. FoodOn: a harmonized food ontology to increase global food traceability, quality control and data integration. *npj Science of Food*, 2(23).

Yanai Elazar, Nora Kassner, Shauli Ravfogel, Abhilasha Ravichander, Eduard Hovy, Hinrich Schütze, and Yoav Goldberg. 2021. Measuring and improving consistency in pretrained language models. *Transactions of the Association for Computational Linguistics*, 9:1012–1031.

Daya Guo, Dejian Yang, Haowei Zhang, Junxiao Song, Ruoyu Zhang, Runxin Xu, Qihao Zhu, Shirong Ma, Peiyi Wang, Xiao Bi, and 1 others. 2025. Deepseek-r1: Incentivizing reasoning capability in llms via reinforcement learning. *arXiv preprint arXiv:2501.12948*.

Nora Kassner, Oyvind Tafjord, Hinrich Schütze, and Peter Clark. 2021. BeliefBank: Adding memory to a pre-trained language model for a systematic notion of belief. In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*, pages 8849–8861, Online and Punta Cana, Dominican Republic. Association for Computational Linguistics.

Yevgeny Kazakov and Pavel Klinov. 2014. Goal-directed tracing of inferences in EL ontologies. In *The Semantic Web - ISWC 2014 - 13th International Semantic Web Conference, Riva del Garda, Italy, October 19-23, 2014. Proceedings, Part II*, volume 8797 of *Lecture Notes in Computer Science*, pages 196–211. Springer.

Yevgeny Kazakov, Markus Krötzsch, and Frantisek Simancik. 2014. The incredible ELK – from polynomial procedures to efficient reasoning with $\mathcal{EL}$ ontologies. *J. Autom. Reasoning*, 53(1):1–61.

Patrick Koopmann. 2020. LETHE: Forgetting and uniform interpolation for expressive description logics. *KI - Künstliche Intelligenz*.

Atharv Kulkarni, Kushagra Dixit, Vivek Srikumar, Dan Roth, and Vivek Gupta. 2025. Llm-symbolic integration for robust temporal tabular reasoning. *arXiv preprint arXiv:2506.05746*.

Woosuk Kwon, Zhuohan Li, Siyuan Zhuang, Ying Sheng, Lianmin Zheng, Cody Hao Yu, Joseph Gonzalez, Hao Zhang, and Ion Stoica. 2023. Efficient memory management for large language model serving with pagedattention. In *Proceedings of the 29th symposium on operating systems principles*, pages 611–626.

Mosh Levy, Alon Jacoby, and Yoav Goldberg. 2024. Same task, more tokens: the impact of input length on the reasoning performance of large language models. In *Proceedings of the 62nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 15339–15353, Bangkok, Thailand. Association for Computational Linguistics.

Patrick Lewis, Ethan Perez, Aleksandra Piktus, Fabio Petroni, Vladimir Karpukhin, Naman Goyal, Heinrich Küttler, Mike Lewis, Wen-tau Yih, Tim Rocktäschel, and 1 others. 2020. Retrieval-augmented generation for knowledge-intensive nlp tasks. *Advances in neural information processing systems*, 33:9459–9474.

Chin-Yew Lin. 2004. ROUGE: A package for automatic evaluation of summaries. In *Text Summarization Branches Out*, pages 74–81, Barcelona, Spain. Association for Computational Linguistics.

Liam Daly Manocchio, Siamak Layeghy, Wai Weng Lo, Gayan K Kulatilleke, Mohanad Sarhan, and Marius Portmann. 2024. Flowtransformer: A transformer framework for flow-based network intrusion detection systems. *Expert Systems with Applications*, 241:122564.

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.

Linyong Nan, Dragomir Radev, Rui Zhang, Amrit Rau, Abhinand Sivaprasad, Chiachun Hsieh, Xiangru Tang, Aadit Vyas, Neha Verma, Pranav Krishna, Yangxiaokang Liu, Nadia Irwanto, Jessica Pan, Faiaz Rahman, Ahmad Zaidi, Mutethia Mutuma, Yasin Tarabar, Ankit Gupta, Tao Yu, and 5 others. 2021. DART: Open-domain structured data record to text generation. In *Proceedings of the 2021 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 432–447, Online. Association for Computational Linguistics.

Kishore Papineni, Salim Roukos, Todd Ward, and Wei-Jing Zhu. 2002. BLEU: a method for automatic evaluation of machine translation. In *Proceedings of the*

40th Annual Meeting of the Association for Computational Linguistics, pages 311–318, Philadelphia, Pennsylvania, USA. Association for Computational Linguistics.

Ankur Parikh, Xuezhi Wang, Sebastian Gehrmann, Manaal Faruqui, Bhuwan Dhingra, Diyi Yang, and Dipanjan Das. 2020. ToTTo: A controlled table-to-text generation dataset. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 1173–1186, Online. Association for Computational Linguistics.

Maja Popović. 2017. chrF++: words helping character n-grams. In *Proceedings of the Second Conference on Machine Translation*, pages 612–618, Copenhagen, Denmark. Association for Computational Linguistics.

Miao Qi, Yarden Neeman, Forest Eckhardt, and Kevin Blissett. 2018. WhatToMake: a semantic web application for recipe recommendation. *Rensselaer Polytechnic Institute*.

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

Michael Shalyt, Rotem Elimelech, and Ido Kaminer. 2025. Asymob: Algebraic symbolic mathematical operations benchmark. *arXiv preprint arXiv:2505.23851*.

Parshin Shojaee, Iman Mirzadeh, Keivan Alizadeh, Maxwell Horton, Samy Bengio, and Mehrdad Farajtabar. 2025. The illusion of thinking: Understanding the strengths and limitations of reasoning models via the lens of problem complexity. *arXiv preprint arXiv:2506.06941*.

Kaiser Sun, Fan Bai, and Mark Dredze. 2025. What is seen cannot be unseen: The disruptive effect of knowledge conflict on large language models. *arXiv preprint arXiv:2506.06485*.

Miles Turpin, Julian Michael, Ethan Perez, and Samuel Bowman. 2023. Language models don't always say what they think: Unfaithful explanations in chain-of-thought prompting. *Advances in Neural Information Processing Systems*, 36:74952–74965.

Vy Vo, Lizhen Qu, Tao Feng, Yuncheng Hua, Xiaoxi Kang, Songhai Fan, Tim Dwyer, Lay-Ki Soon, and Gholamreza Haffari. 2025. ACCESS : A benchmark for abstract causal event discovery and reasoning. In *Proceedings of the 2025 Conference of the Nations of the Americas Chapter of the Association for Computational Linguistics: Human Language Technologies (Volume 1: Long Papers)*, pages 1049–1074, Albuquerque, New Mexico. Association for Computational Linguistics.

Dingmin Wang, Pan Hu, Przemyslaw Andrzej Walega, and Bernardo Cuenca Grau. 2022. MeTeoR: Practical reasoning in datalog with metric temporal operators. In *Thirty-Sixth AAAI Conference on Artificial Intelligence, AAAI 2022*, pages 5906–5913. AAAI Press.

Jian Xie, Kai Zhang, Jiangjie Chen, Renze Lou, and Yu Su. 2024. Adaptive chameleon or stubborn sloth: Revealing the behavior of large language models in knowledge conflicts. In *The Twelfth International Conference on Learning Representations*.

An Yang, Anfeng Li, Baosong Yang, Beichen Zhang, Binyuan Hui, Bo Zheng, Bowen Yu, Chang Gao, Chengen Huang, Chenxu Lv, and 1 others. 2025. Qwen3 technical report. *arXiv preprint arXiv:2505.09388*.

Tianyi Zhang, Varsha Kishore, Felix Wu, Kilian Q Weinberger, and Yoav Artzi. 2020. Bertscore: Evaluating text generation with bert. In *International Conference on Learning Representations*.

A Preliminaries

A.1 DL Proofs

The proofs in our benchmark are based on the two description logics (DLs) $\mathcal{EL}$ and $\mathcal{ALC}$ (Baader et al., 2017) as well as DatalogMTL (Brandt et al., 2018), an extension of Datalog with metric temporal operators for querying temporal data.

The syntax of DLs is based on disjoint, countably infinite sets $\mathsf{N_C}$ and $\mathsf{N_R}$ of *concept names* A, B, ... and *role names* r, s, ..., respectively. In $\mathcal{EL}$, *concepts* are built from concept names by applying the constructors $\top$ (top), $C \sqcap D$ (conjunction), and $\exists r.C$ (existential restriction for a role name r). A *general concept inclusion (GCs)* is of the form $C \sqsubseteq D$, where C and D are $\mathcal{EL}$ concepts, and a finite set of GCIs is called a *TBox* or *ontology*. The DL $\mathcal{ALC}$ extends $\mathcal{EL}$ by the concept constructors $\bot$ (bottom), $C \sqcup D$ (disjunction), $\forall r.C$ (value restriction), and $\neg C$ (negation). For the semantics, in particular when a GCI η is *entailed* by a TBox $\mathcal{T}$ (written $\mathcal{T} \models \eta$), we refer the reader to (Baader et al., 2017).

In contrast to $\mathcal{EL}$ and $\mathcal{ALC}$, temporal reasoning in DatalogMTL also takes facts about constants into account. A (function-free first-order) *atom* has the form $P(\tau)$ with P a predicate of some arity n and τ an n-ary tuple consisting of constants and variables. A *literal* (or *metric atom*) A takes one of the following forms, where ϱ is a non-empty positive rational interval:

$$A := \top \mid \bot \mid P(\tau) \mid \Diamond_\varrho A \mid \boxplus_\varrho A \mid \boxminus_\varrho A \mid \boxplus_\varrho A \mid A \,\mathcal{S}_\varrho A \mid A \,\mathcal{U}_\varrho A$$

A *rule* with *body literals* $A_1, \ldots, A_n$, $n \geq 1$, and *head literal* B is of the form:

$$B \;:-\; A_1 \wedge \ldots \wedge A_n,$$

with B not containing the operators $\Diamond$, $\boxplus$, $\mathcal{S}$ or $\mathcal{U}$. If an atom, literal or rule contains no variable, we call it *ground*. A *fact* F is defined as an expression of the form $A@\varrho$ where A is a ground atom and ϱ a rational interval. Moreover, we call a finite set D of facts a *dataset* and a finite set Π of rules a *program*. In this context, entailments are of the form $\Pi, \mathcal{D} \models F$. However, for simplicity, we now denote entailments in DLs and DatalogMTL uniformly by $\mathcal{T} \models \eta$.

Our goal is to explain a logical consequence $\mathcal{T} \models \eta$, where $\mathcal{T}$ is either a TBox or a program together with a dataset, and η is a GCI or a fact, respectively. Following (Alrabbaa et al., 2020, 2022b), *proofs* of $\mathcal{T} \models \eta$ are finite, acyclic, directed hypergraphs, where vertices v are labeled with GCIs or facts $\ell(v)$ and hyperedges are of the form (S, d), with S a tuple of vertices and d a vertex such that $\{\ell(v) \mid v \in S\} \models \ell(d)$; the leafs of a proof must be labeled by elements of $\mathcal{T}$ and the unique sink vertex by η. In addition, an edge labeling function (see labels without boxes in Figures 8,12,10) indicates which logical rule derived a conclusion $\ell(d)$ from the premises. The *size* of a proof is the number of its vertices.

For this benchmark, $\mathcal{EL}$ proofs were generated using the reasoner ELK (Kazakov et al., 2014; Kazakov and Klinov, 2014), while for $\mathcal{ALC}$, we employed the forgetting tool LETHE (Koopmann, 2020; Alrabbaa et al., 2020). For DatalogMTL, we extended the Metric Temporal Reasoner (MeTeoR) (Wang et al., 2022) to trace the applied reasoning steps (Borgwardt et al., 2024). We also used EVEE (EVincing Expressive Entailments) (Alrabbaa et al., 2022a), a Java library that can extract size-minimal proofs from the output of a reasoner.

B Proof Examples

Biology Example In Figures 7 and 8, one can see an example of a proof that RMUG-S is an immortal human cell line cell (IhCL for short). An *immortal cell line* (ICL) is expected to be capable of an unlimited number of divisions, and is thus able to support indefinite propagation *in vitro*. RMUG-S is a human (lat. *Homo sapiens*) ovarian adenocarcinoma cell line originated from a 62 year old Japanese female.

Food & Recipes example In Figures 9 and 10 one can see an example of a vegan bread recipe.

Critical Situations in Drone example The proof in Figure 12 show a temporal proof for the drone probably having internal damage at time point 0 by detecting an internal temperature above the threshold temperature.

C Prompts

We provide all the prompts verbatim in Figures 13, 15, 14, 16.

D Inference details

For the open weights LLMs, we use the exact same inference parameters mentioned in the LLM model card on their respective HuggingFace model page.

```
finalConclusion: RMUG-S ⊑ IhCL,
inferences: [ {
        conclusion: RMUG-S ⊑ IhCL,
        ruleName: Class Hierarchy,
        premises: [ RMUG-S ⊑
            (∃dF.∃pOf.Homo sapiens ⊓
            ICL),
            (∃dF.∃pOf.Homo sapiens ⊓
                ICL) ⊑ IhCL ]
    },{
        conclusion: RMUG-S ⊑
            (∃dF.∃pOf.Homo sapiens ⊓
            ICL),
        ruleName: Intersection
            Composition,
        premises: [ RMUG-S ⊑ ICL,
            RMUG-S ⊑ ∃dF.∃pOf.Homo
                sapiens ]
    },{
        conclusion: RMUG-S ⊑ ICL,
        ruleName: Asserted Conclusion
    },{
        conclusion: RMUG-S ⊑
            ∃dF.∃pOf.Homo sapiens,
        ruleName: Asserted Conclusion
    },{
        conclusion: (∃dF.∃pOf.Homo
            sapiens ⊓ ICL) ⊑ IhCL,
        ruleName: Equivalent Classes
            Decomposition,
        premises: [ IhCL ≡
            (∃dF.∃pOf.Homo sapiens ⊓
            ICL) ]
    },{
        conclusion: IhCL ≡
            (∃dF.∃pOf.Homo sapiens ⊓
            ICL),
        ruleName: Asserted Conclusion }
            ]
```

Figure 7: Biology Proof JSON (we abbreviated 'derives from' and 'part of' as dF and pOf correspondingly)

E Additional Results on Summary & Target User Message similarity automated metrics

The results in Figure 6 and 17 show that providing the context trail leads to a slight but consistent improvement across all models and metrics for both the summary and target user message tasks. For instance, the ROUGE-L score for `Llama-3.1-8B` on the summary task improves from approximately 44.5 to 45.0 when the context trail is included. Similarly, the BERTScore for `GPT-4o-mini` on the target message task increases from around 57.0 to 57.5. While minor, this trend suggests that access to the full reasoning path, even when generating a summary of it, provides valuable context that helps the models produce outputs that are more aligned with the human-written references. This indicates that for generation tasks grounded in a logical proof, providing the complete proof structure is beneficial.

F Length analysis for Summary and Target User Message tasks

Figures 20 and 21 show this analysis.

G Annotation Guidelines

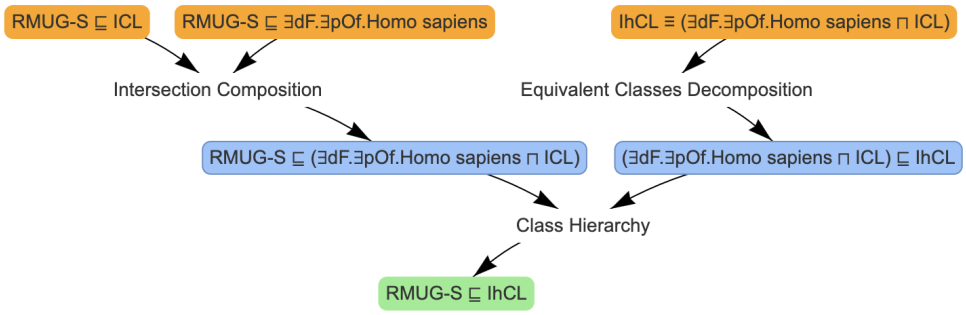

Figure 8: Biology Proof

```
finalConclusion: bread ⊑ vegan recipe,
inferences: [ {
        conclusion: bread ⊑ vegan
            recipe,
        ruleName: eliminate 'flour',
        premises: [ (∀ingr.(water ⊔
            flour) ⊓ bread) ⊑ vegan
            recipe,
            bread ⊑ ∀ingr.(water ⊔
                flour) ]
    },{
        conclusion: (∀ingr.(water ⊔
            flour) ⊓ bread) ⊑ vegan
            recipe,
        ruleName: eliminate 'vegan',
        premises: [ water ⊑ vegan,
            flour ⊑ vegan,
            (∀ingr.vegan ⊓ bread) ⊑
                vegan recipe ]
    },{
        conclusion: water ⊑ vegan,
        ruleName: asserted
    },{
        conclusion: flour ⊑ vegan,
        ruleName: asserted
    },{
        conclusion: (∀ingr.vegan ⊓
            bread) ⊑ vegan recipe,
        ruleName: eliminate 'food
            recipe',
        premises: [ bread ⊑ (food
            recipe ⊓ ∃ingr.flour ⊓
            ∃ingr.water),
            vegan recipe ≡ (∀ingr.vegan
                ⊓ food recipe) ]
    },{
        conclusion: bread ⊑ (food
            recipe ⊓ ∃ingr.flour ⊓
            ∃ingr.water),
        ruleName: asserted
    },{
        conclusion: vegan recipe ≡
            (∀ingr.vegan ⊓ food recipe),
        ruleName: asserted
    },{
        conclusion: bread ⊑
            ∀ingr.(water ⊔ flour),
        ruleName: asserted } ]
```

Figure 9: Recipe proof JSON

Table 1: Rubrics and Workflow for Evaluating Candidate Explanations

What you see	How to interpret it	Why it matters
Description Logic Proof (JSON)	Ground-truth **correct** reasoning chain – assume JSON contains the gold standard reasoning in mathematical form.	Serves as the gold standard to judge each candidate.
Candidate Summary & Target message	Model's attempt to compress the proof for an end-user.	Ratings indicate clarity and faithfulness to the proof.

Structure of the Description Logic Proof: The JSON proof structure links "premises" step by step using "ruleName" in the "inferences" field. Each step uses asserted or previously inferred "premises", applies a "ruleName", and produces a "conclusion". This builds a logical sequence from base facts to the "finalConclusion".

Field	Max Length	Purpose	Typical Content
Summary	$\approx$ 4–5 sentences	Capture candidate's full reasoning and conclusion.	Key conclusion, main supporting facts
Target Message	$\leq$ 20 words	Single-line alert user will see.	Trigger condition, consequence or instruction

Rating Rubrics (5-point scale):

1. **Summary**:
   - **Faithfulness:** Alignment with the proof.
   - **Readability:** Clarity, tone.
   - **Conciseness:** No redundancy.
   - **Coverage:** All key steps present.
2. **Target Message**:
   - **Faithfulness:** Supported by proof.
   - **Appropriateness:** Suited to end-user, no extra inference.
   - **Coverage:** Critical details for action.

Score	Faithfulness	Readability	Conciseness	Coverage
5 (Excellent)	Every statement fully justified by proof	Flawless writing	Only essential info	Every key step and conclusion covered
4 (Good)	Minor paraphrase, accurate ($\geq$95%)	Very clear, minor phrasing issue	Small redundancy	Misses one trivial step or includes one unneeded detail
3 (Fair)	Several weakly-supported statements	Understandable, awkward wording	Multiple extra phrases	Omits $\geq$2 secondary steps
2 (Poor)	Key facts misstated or unsupported	Hard to follow	Verbose or info beyond important points	Omits at least one critical step
1 (Unacceptable)	Major hallucinations/contradictions	Largely incoherent	Very lengthy or irrelevant info	Fails to cover main conclusion/reasoning

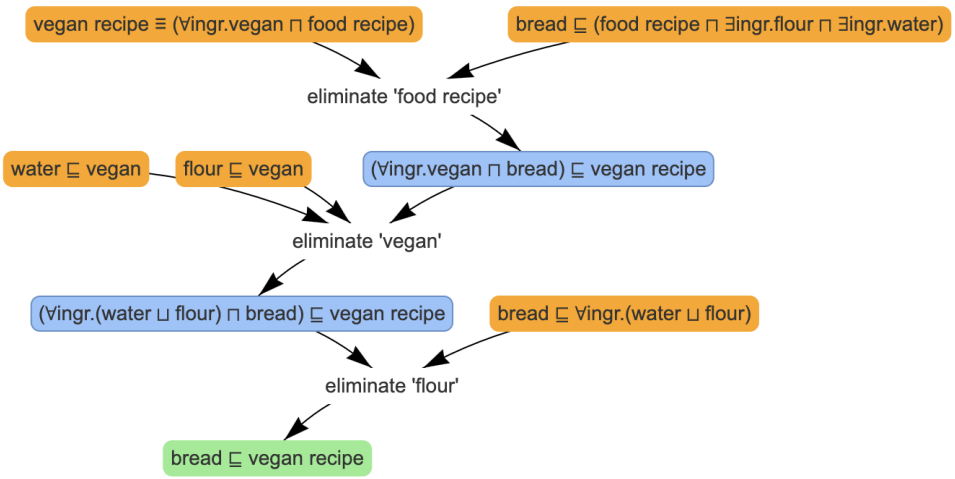

Figure 10: Simplified proof. Edges "asserted" removed, "ingr." means "has ingredient" and "vegan" — "vegan ingredient"

```
finalConclusion : warninglvl(d,4)@[0,0],
inferences : [ {
        conclusion :
            warninglvl(d,4)@[0,0],
        ruleName : inclusion,
        premises : [
            warninglvl(d,4)@[-1,9] ]
    }, {
        conclusion :
            warninglvl(d,4)@[-1,9],
        ruleName : warninglvl(X,4) :-
            risk(X),
        premises : [ risk(d)@[-1,9] ]
    }, {
        conclusion : risk(d)@[-1,9],
        ruleName : risk(X) :-
            riskofinternaldamage(X),
        premises : [
            riskofinternaldamage(d)@[-1,9]
            ]
    }, {
        conclusion :
            riskofinternaldamage(d)@[-1,9],
        ruleName : reverse ⊞,
        premises : [
            ⊞[0,10]riskofinternaldamage(d)@[
            ]
    }, {
        conclusion :
            ⊞[0,10]riskofinternaldamage(d)@[
        ruleName :
            ⊞[0,10]riskofinternaldamage(Y) :-
            internaltemperature(Y,S),drone(Y
        premises : [
            internaltemperature(d,48)@[-1,-1
            drone(d)@[-300,+∞) ]
    }, {
        conclusion :
            internaltemperature(d,48)@[-1,-1
        ruleName : Asserted
    }, {
        conclusion :
            drone(d)@[-300,+∞),
        ruleName : reverse ⊞,
        premises : [
            ⊞[0,+∞)drone(d)@[-300,-300]
            ]
    }, {
        conclusion :
            ⊞[0,+∞)drone(d)@[-300,-300],
        ruleName : ⊞[0,+∞)drone(X) :-
            drone(X),
        premises : [
            drone(d)@[-300,-300] ]
    }, {
        conclusion :
            drone(d)@[-300,-300],
        ruleName : Asserted } ]
```

Figure 11: Example proof for a critical scenario for drones

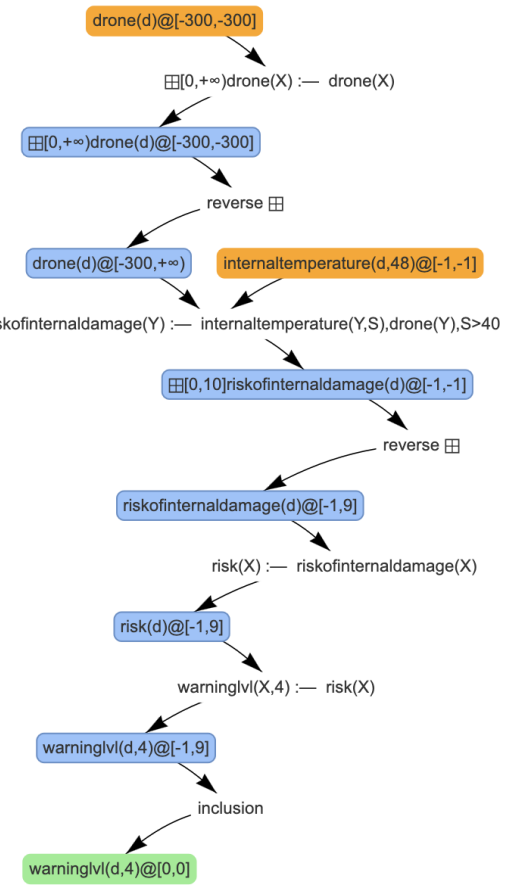

Figure 12: Proof for the drone experiencing overheating.

System Prompt

You are an expert on Description Logic and DatalogMTL proofs. You are also good at explaining complex things in an easy way.

Task 1: Identify Most Contributing Steps

`{{proof_method_description}}`

`### TASK - 1 ###`
List the steps that contribute the most in deriving the final conclusion (at most 3 steps). Put the exact steps verbatim from the proof below, including conclusion, rule name, and premises.

`{{proof}}`

Output **strictly** a JSON object matching this schema:
`{{ StepsOutput.model_json_schema() | tojson }}`

LLM response

`{{JSON_response}}`

Task 2: Summarize the Proof

`### TASK - 2 ###`
An example summary for an example proof was written by a human expert as shown below.
Example Proof: *`{{example_proof}}`*
Example Summary: *`{{example_summary}}`*

Summarize the proof shown in Task 1 (i.e., Finding the most contributing steps) similar to a human expert. The summary should contain the conclusion of the proof and how it was reached.

Output **strictly** a JSON object matching this schema:
`{{ SummaryOutput.model_json_schema() | tojson }}`

LLM response

`{{JSON_response}}`

Task 3: Generate Targeted Message

`### TASK - 3 ###`
An example message for the targeted user was written by a human expert as shown below. The human expert utilized the 'Example Proof' and the 'Example Summary' to write this message.
Example Target Message: *`{{example_target_msg}}`*

Generate a message for the targeted user similar to a human expert for the proof shown in Task 1. The message should be a maximum of 15-20 words long and it should explain to `{{user}}` what exact condition led to the final conclusion.

Output **strictly** a JSON object matching this schema:
`{{ TargetMsgOutput.model_json_schema() | tojson }}`

LLM response

`{{JSON_response}}`

Figure 13: Prompt template for One-shot chained; same color for all tasks indicates multi-turn response chain, adding the previous context for all the tasks.

System Prompt

You are an expert on Description Logic and DatalogMTL proofs. You are also good at explaining complex things in an easy way.

Task 1: Identify Most Contributing Steps

`{{proof_method_description}}`

`### TASK - 1 ###`
List the steps that contribute the most in deriving the final conclusion (at most 3 steps). Put the exact steps verbatim from the proof below, including conclusion, rule name, and premises.

`{{proof}}`

Output **strictly** a JSON object matching this schema:
`{{ StepsOutput.model_json_schema() | tojson }}`

LLM response

`{{JSON_response}}`

Task 2: Summarize the Proof

`### TASK - 2 ###`
Summarize the proof shown in Task 1 (i.e., Finding the most contributing steps) similar to a human expert. The summary should contain the conclusion of the proof and how it was reached.

Output **strictly** a JSON object matching this schema:
`{{ SummaryOutput.model_json_schema() | tojson }}`

LLM response

`{{JSON_response}}`

Task 3: Generate Targeted Message

`### TASK - 3 ###`
Generate a message for the targeted user similar to a human expert for the proof shown in Task 1. The message should be a maximum of 15-20 words long and it should explain to `{{user}}` what exact condition led to the final conclusion.

Output **strictly** a JSON object matching this schema:
`{{ TargetMsgOutput.model_json_schema() | tojson }}`

LLM response

`{{JSON_response}}`

Figure 14: Prompt template for Zero-shot chained; same color for all tasks indicates multi-turn response chain, adding the previous context for all the tasks.

System Prompt

You are an expert on Description Logic and DatalogMTL proofs. You are also good at explaining complex things in an easy way.

Task 1: Identify Most Contributing Steps

`{{proof_method_description}}`

`### TASK - FIND THE MOST CONTRIBUTING STEPS ###`
List the steps that contribute the most in deriving the final conclusion (at most 3 steps). Put the exact steps verbatim from the proof below, including conclusion, rule name, and premises.

`{{proof}}`

Output **strictly** a JSON object matching this schema:
`{{ StepsOutput.model_json_schema() | tojson }}`

LLM response

`{{JSON_response}}`

Task 2: Summarize the Proof

`### TASK - SUMMARIZE PROOF ###`
An example summary for an example proof was written by a human expert as shown below.
Example Proof: *`{{example_proof}}`*
Example Summary: *`{{example_summary}}`*

Summarize the proof shown in Task 1 (i.e., Finding the most contributing steps) similar to a human expert. The summary should contain the conclusion of the proof and how it was reached.

Output **strictly** a JSON object matching this schema:
`{{ SummaryOutput.model_json_schema() | tojson }}`

LLM response

`{{JSON_response}}`

Task 3: Generate Targeted Message

`### TASK - TARGET-USER MESSAGE ###`
An example message for the targeted user was written by a human expert as shown below. The human expert utilized the 'Example Proof' and the 'Example Summary' to write this message.
Example Target Message: *`{{example_target_msg}}`*

Generate a message for the targeted user similar to a human expert for the proof shown in Task 1. The message should be a maximum of 15-20 words long and it should explain to `{{user}}` what exact condition led to the final conclusion.

Output **strictly** a JSON object matching this schema:
`{{ TargetMsgOutput.model_json_schema() | tojson }}`

LLM response

`{{JSON_response}}`

Figure 15: Prompt template for One-shot atomic; different colors for all tasks indicate single-turn responses for each independent task.

System Prompt

You are an expert on Description Logic and DatalogMTL proofs. You are also good at explaining complex things in an easy way.

Task 1: Identify Most Contributing Steps

`### TASK - FIND THE MOST CONTRIBUTING STEPS ###`
List the steps that contribute the most in deriving the final conclusion (at most 3 steps). Put the exact steps verbatim from the proof below, including conclusion, rule name, and premises.

`{{proof}}`

Output **strictly** a JSON object matching this schema:
`{{ StepsOutput.model_json_schema() | tojson }}`

LLM response

`{{JSON_response}}`

Task 2: Summarize the Proof

`### TASK - SUMMARIZE PROOF ###`
Summarize the proof below similar to a human expert. The summary should contain the conclusion of the proof and how it was reached.

`{{proof}}`

Output **strictly** a JSON object matching this schema:
`{{ SummaryOutput.model_json_schema() | tojson }}`

LLM response

`{{JSON_response}}`

Task 3: Generate Targeted Message

`### TASK - TARGET-USER MESSAGE ###`
Generate a message for the targeted user similar to a human expert for the proof given below. The message should be a maximum of 15-20 words long and it should explain to `{{user}}` what exact condition led to the final conclusion.

`{{proof}}`

Output **strictly** a JSON object matching this schema:
`{{ TargetMsgOutput.model_json_schema() | tojson }}`

LLM response

`{{JSON_response}}`

Figure 16: Prompt template for Zero-shot atomic; different colors for all tasks indicate single-turn responses for each independent task.

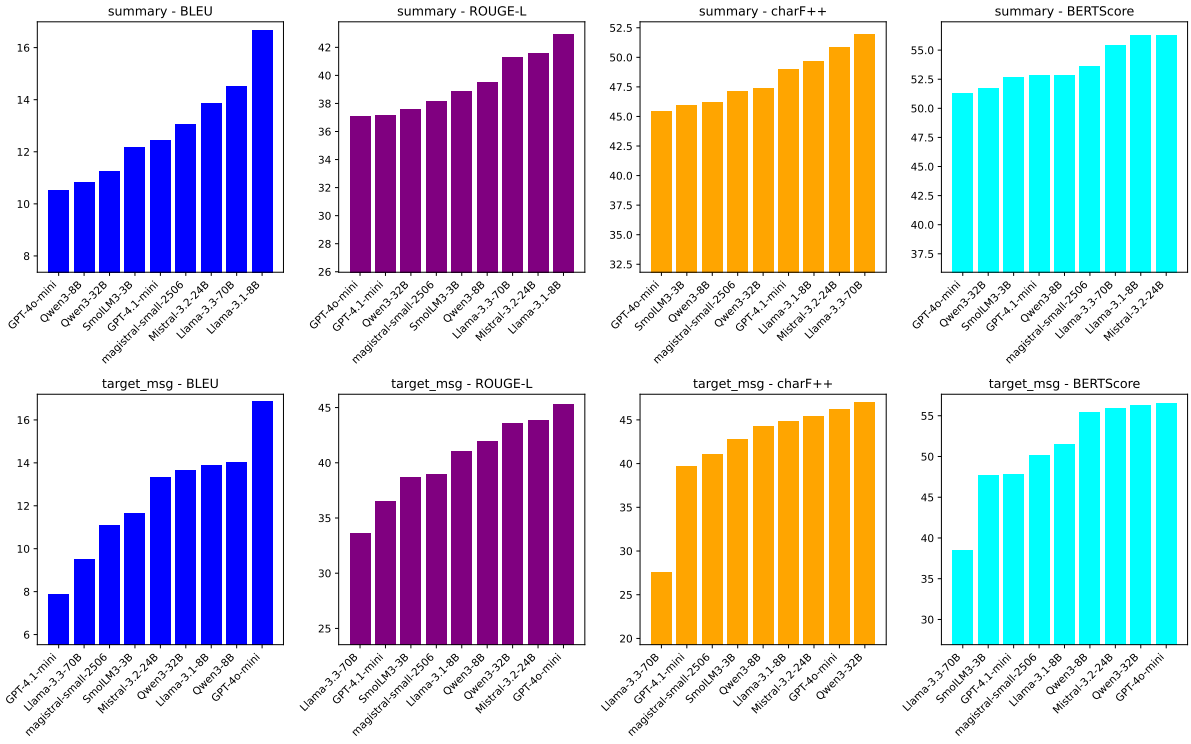

Figure 17: Automated metrics of One-shot atomic strategy

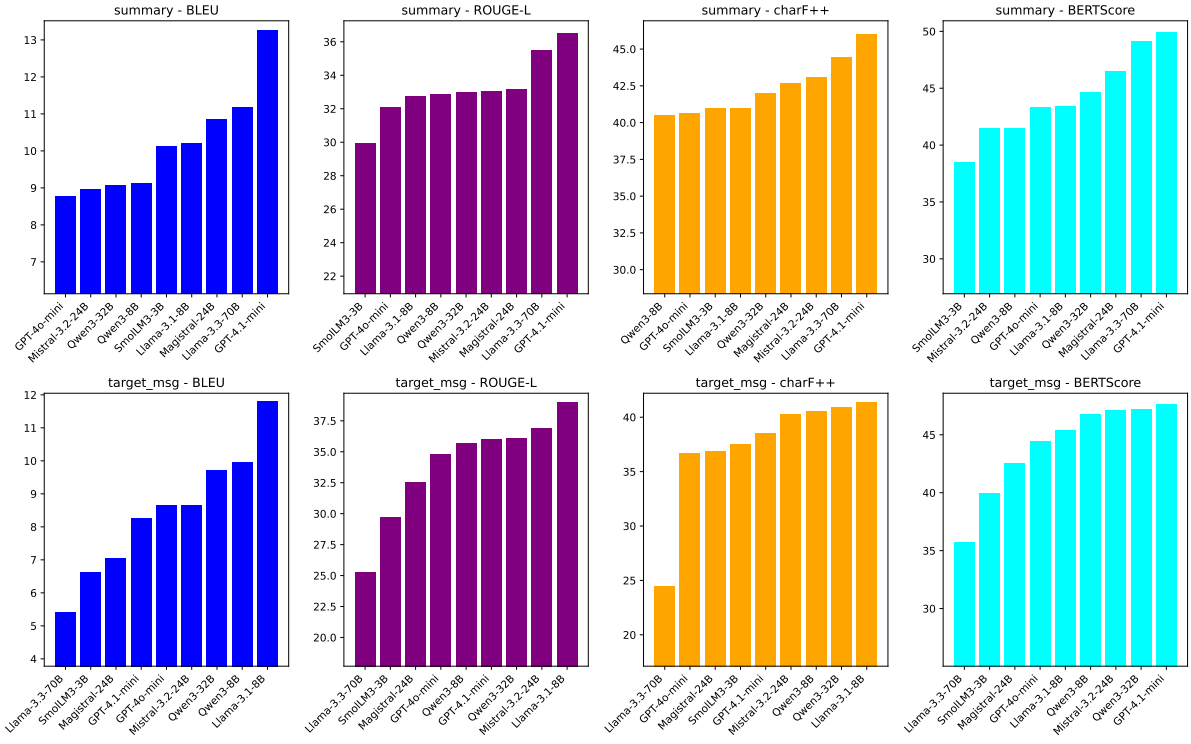

Figure 18: Automated metrics of Zero-shot chained strategy

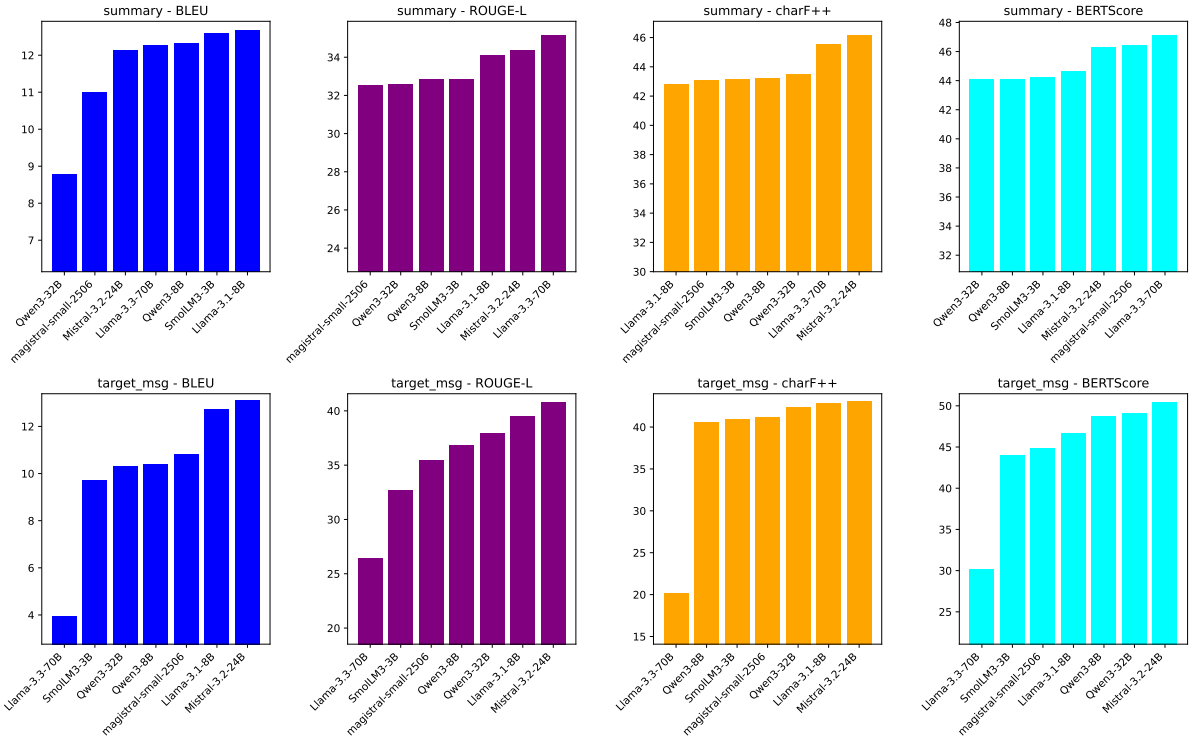

Figure 19: Automated metrics of Zero-shot atomic strategy

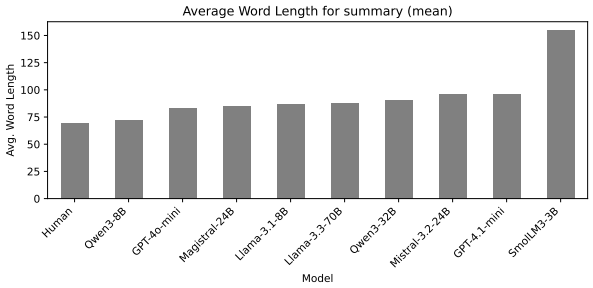

Figure 20: Average length of summary for One-shot chained strategy

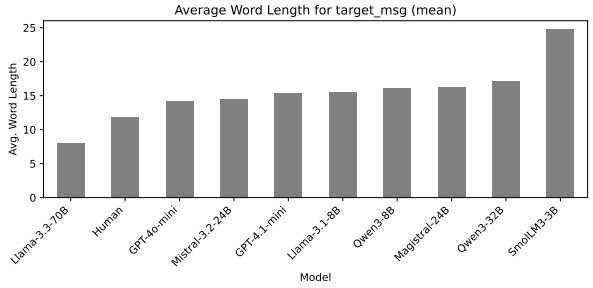

Figure 21: Average length of target user message for One-shot chained strategy