Don't Let It Hallucinate: Premise Verification via Retrieval-Augmented Logical Reasoning

Yuehan Qin
University of Southern California
yuehanqi@usc.edu

Shawn Li
University of Southern California
li.li02@usc.edu

Yi Nian
University of Southern California
yinian@usc.edu

Xinyan Velocity Yu
University of Southern California
xinyany@usc.edu

Yue Zhao*
University of Southern California
yzhao010@usc.edu

Xuezhe Ma*
University of Southern California
xuezhema@usc.edu

Abstract

Large language models (LLMs) have shown substantial capacity for generating fluent, contextually appropriate responses. However, they can produce hallucinated outputs, especially when a user query includes *false premises*—claims that contradict established facts. Such premises can mislead LLMs into offering fabricated or misleading details. Existing approaches include pretraining, fine-tuning, and inference-time techniques that often rely on access to logits or address hallucinations after they occur. These methods tend to be computationally expensive, require extensive training data, or lack proactive mechanisms to prevent hallucination before generation, limiting their efficiency in real-time applications. We propose a retrieval-based framework that identifies and addresses false premises *before* generation. Our method first transforms a user's query into a logical representation, then applies retrieval-augmented generation (RAG) to assess the validity of each premise using factual sources. Finally, we incorporate the verification results into the LLM's prompt to maintain factual consistency in the final output. Experiments show that this approach effectively reduces hallucinations, improves factual accuracy, and does not require access to model logits or large-scale fine-tuning.

1 Introduction

Large Language Models (LLMs) generate fluid, context-aware responses but can produce hallucinations when prompted with queries that include hidden factual errors [Manakul et al., 2023, Zheng et al., 2023]. These errors, known as *false premises*, are statements in a user's question that conflict with real-world facts. Even when LLMs store accurate information, they may trust the incorrect assumptions embedded in the query and generate misleading outputs [Yuan et al., 2024]. As shown in Fig. 1, these errors are common among LLM outputs [Huang et al., 2025, Snyder et al., 2024], and are particularly insidious as they can appear factually sound while being fundamentally incorrect.

Many methods attempt to address false premises after an LLM has already produced an answer [Hu et al., 2023, Shi et al., 2023, Pezeshkpour, 2023]. Although effective in some contexts, these

*Corresponding Author

39th Conference on Neural Information Processing Systems (NeurIPS 2025) Workshop: Efficient Reasoning.

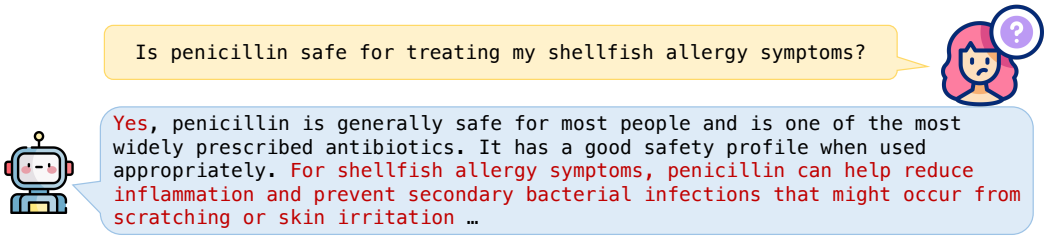

Figure 1: LLM experiences factuality hallucination when faced with a false premise question, where both entities *shellfish allergy symptom* and *penicillin* exist but are not correctly aligned. The LLM's hallucinated response could delay life-saving treatment by incorrectly recommending antibiotics for allergic reactions.

approaches can be computationally demanding and do not necessarily prevent misinformation from appearing in the first place. Additionally, questions with false premises often maintain normal semantic flow, changing only a few tokens so that they are difficult to identify using traditional out-of-distribution detection [Vu et al., 2023]. Even advanced LLMs can struggle with real-time truth evaluation, lacking the context or capacity to fully check every assumption [Hu et al., 2023].

To address this challenge, we focus on *preventing* hallucinations rather than mitigating them post hoc. In our framework, we first transform the user's query into a logical form that highlights key entities or relations. We then employ retrieval-augmented generation (RAG) to check the accuracy of these statements against a knowledge graph. If contradictions are found, the query is flagged as containing a false premise prompting the model to correct or reject the assumption before formulating a final answer. This process, shown in Fig. 2, ensures that the LLM does not rely on erroneous details during response generation. By informing the LLM about any detected false premise in advance, we reduce the likelihood of hallucinations without requiring access to model logits or large-scale fine-tuning. Our proposed method applies to knowledge graphs and datasets compatible with graph structures.

2 Related Works

False Premise. A False Premise Question (FPQ) is a question containing incorrect facts that are not necessarily explicitly stated but might be mistakenly believed by the questioner [Yu et al., 2022, Kim et al., 2021]. Recent studies [Yuan et al., 2024] have demonstrated that FPQs can induce factuality hallucination in LLMs, as they often respond directly to FPQs without verifying their validity.

Logical Forms. Symbolic solvers and logical forms are applied to logical reasoning by grounding natural language in symbolic representations. The latest trend is integrating LLMs with symbolic solvers to enhance their performance Olausson et al. [2023], Pan et al. [2023a]. Similarly, SymbCoT Xu et al. [2024] converts input text into symbolic formats such as first-order logic, generates reasoning plans through logical rule application, and verifies the reasoning process to ensure consistency.

Knowledge Graph Fact Checking and Question Answering. Knowledge graph–driven RAG recently supports structured verification via: (1) *prompt-based* methods for evidence checks and multi-hop retrieval [Pan et al., 2023b, Sun et al., 2024]; (2) *graph-based* methods framing RAG as subgraph extraction or GNN reasoning [He et al., 2024, Mavromatis and Karypis, 2024]; and (3) *training-based* methods with dual encoders for query–subgraph embedding and ranking [Zheng et al., 2024, Liu et al., 2024a], though limited by KG entity coverage and prompt-generated data.

Hallucination Mitigation. Sources of LLM hallucinations originate from different stages in the LLM life cycle [Zhang et al., 2023a], leading existing mitigation methods to target specific stages: pre-training (mitigated by emphasizing credible data [Touvron et al., 2023, Lee et al., 2023]), supervised fine-tuning (curated instruction data improves factuality [Chen et al., 2024, Cao et al., 2024]), RLHF (alignment may introduce hallucinations when prompts exceed model knowledge [Radhakrishnan et al., 2023, Wei et al., 2024]), and inference (errors snowball [Zhang et al., 2023b]), where decoding adjustments [Shi et al., 2023, Chuang et al., 2024] or uncertainty-based checks [Xu and Ma, 2025, Liu et al., 2024b, Dhuliawala et al., 2023] are applied.

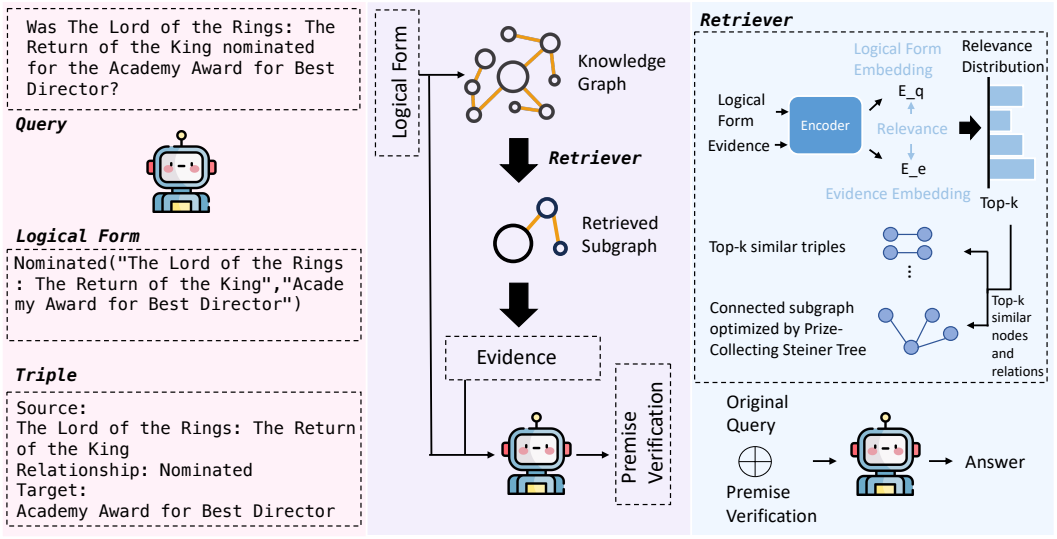

Figure 2: Overview of our approach. Left: The original query is converted into a logical form. Middle: The logical form is used to retrieve relevant elements from the knowledge graph and detect false premises. Right: Comparison of studied retrievers for aligning logical form with the knowledge graph. The LLM generates responses with reduced hallucination given prompts with premise verification.

3 Methodology

Logical Form Extraction: A logical form is a symbolic representation that captures semantic relations in a query. Given a natural language query q, we represent its logical form as $\mathcal{L}(q) = P(x_1, x_2, \ldots, x_n)$, where P is a predicate or relation and $x_1, \ldots, x_n$ are variables or constants. We use GPT-4o-mini [OpenAI, 2024] to convert q into $\mathcal{L}(q)$ and extract source, relation, and target. Details of the prompting procedure are provided in Appx. § A.3. To assess conversion quality, two annotators graded 200 samples on a three-point scale (1: no match, 2: partial, 3: match). All generated forms received a score of 3.

Retrieval: Given a natural language query q, the retrieval stage extracts the most relevant graph elements (entities, triplets, paths, or subgraphs) from a knowledge graph G: $G^* = \text{Graph-Retriever}(q, G) = \arg\max_{G \subseteq R(G)} p_\theta(G \mid q, G) = \arg\max_{G \subseteq R(G)} \text{Sim}(q, G)$, where G^* denotes the retrieved subgraph, $\text{Sim}(\cdot, \cdot)$ measures query–graph similarity, and $R(\cdot)$ restricts the candidate set for efficiency. After converting q into a logical form $\mathcal{L}(q)$, the retriever encodes $\mathcal{L}(q)$ and graph triplets, then searches G under various selection criteria: $G^* = \text{Graph-Retriever}(\mathcal{L}(q), G) = \arg\max_{G \subseteq R(G)} p_\theta(G \mid \mathcal{L}(q), G) = \arg\max_{G \subseteq R(G)} \text{Sim}(\mathcal{L}(q), G)$. We use the pre-trained encoder *all-roberta-large-v1*[2] to embed both logical forms and graph triplets, enabling similarity-based retrieval and subsequent LLM-based premise verification.

Hallucination Mitigation: For a query q, if the false premise detector $F(q) = 1$, we update the query as $q' = q + W$, where $W =$ "Note: This question contains a false premise."; otherwise, $q' = q$. Once the original query is updated, we evaluate LLM's responses and measure the effectiveness of the ensuing hallucination mitigation.

4 Experiments

Dataset KG-FPQ [Zhu et al., 2024] contains true- and false-premise questions (TPQs, FPQs) constructed from KoPL, a curated subset of Wikidata. TPQs are derived from true triplets, while FPQs are created by replacing objects in false triplets. We evaluate in the art domain with discriminative task (Yes-No questions). Dataset details are in Appx. § B.

[2]https://huggingface.co/sentence-transformers/all-roberta-large-v1

Experiment Setting Our method mitigates hallucinations in two steps: (1) detect false premises in the query; (2) feed the original query plus the detection result to the LLM. For (1), we evaluate on four types of retrievers: Direct Claim, Embedding-based retriever, Non-parametric Retriever, and LLM-based-retriver. For (2), baselines include DirecrtAsk, Prompt, Majority Vote, and Perplexity AI. See Appx. § C for details.

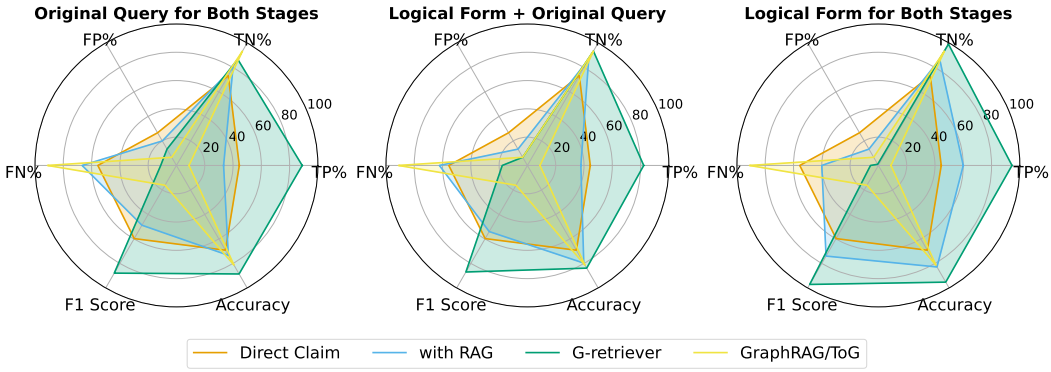

Figure 3: Comparison of performance metrics across different retrieval methods using logical forms and/or original queries.

5 Discussion

	Direct Claim	with RAG	G-retriever	GraphRAG/ToG
Original Query for Both Stages				
True Positives (TP%)	44.44	33.33	88.89	8.89
True Negatives (TN%)	73.33	80.00	86.67	93.33
F1 Score (%)	59.70	48.78	87.89	16.16
Accuracy (%)	69.20	73.33	88.57	81.27
Logical Form for Retrieval and Original Query for False Premise Detection				
True Positives (TP%)	44.44	37.78	82.22	8.89
True Negatives (TN%)	73.33	86.67	93.33	93.33
F1 Score (%)	59.70	53.97	86.97	16.16
Accuracy (%)	69.20	79.69	83.81	81.27
Logical Form for Both Stages				
True Positives (TP%)	44.44	60.00	94.44	8.89
True Negatives (TN%)	73.33	86.67	99.05	93.33
F1 Score (%)	59.70	**73.97**	**97.12**	16.16
Accuracy (%)	69.20	**82.86**	**95.24**	81.27

Table 1: Comparison of performance metrics across different retrieval methods using logical forms and/or original queries.

We show the result of the false premise detection task in Tab. 1 and the hallucination mitigation result in Tab. 2.

Using logical forms helps better identify false premises in the questions. As shown in Tab. 1, for all three retrievers, explicitly incorporating logical forms into both retrieval and false premise detection stages significantly improves the identification of false premises. Sole reliance on original queries, even though potentially yielding high accuracy, tends to neglect accurate false premise identification, underscoring the importance of utilizing structured logical forms for tasks prioritizing precise false premise detection. Using logical forms in both stages, G-retriever achieves the highest TPR (94.44%) and F1 score (97.12%), indicating strong false premise detection with balanced precision and recall. In contrast, ToG attains TNR (93.33%) but suffers from low TPR and F1 (16.16%), suggesting limited effectiveness in correctly identifying false premises. When original queries are used in either retrieval, false premise detection, or both stages, despite achieving reasonable accuracy (73.33% and 79.69%), with RAG method shows significantly lower TPR (33.33% and 37.78%) compared to the first configuration. This suggests that relying on original queries alone, or in combination with logical forms in only one stage for detection, can achieve high accuracy due to correctly identifying negatives, it is less effective at capturing false premises, which is the primary focus of our task.

Models	DirectAsk	Prompt	MajVote	Ours
GPT-4o-mini	83.8	92.4	86.7	**92.4**
GPT-3.5	93.3	93.3	92.4	**94.3**
LLama-3.1	86.7	86.7	89.5	**89.5**
Mistral-7B	87.6	86.7	87.6	**89.5**
Qwen2.5	92.4	86.7	92.4	**95.2**
Qwen1.5	89.5	90.5	90.5	**91.4**
Perplexity AI	91.4			

Table 2: Comparison of accuracy (%) of different hallucination mitigation methods.

Explicitly detecting and informing LLMs false premise mitigates hallucination, as demonstrated in Tab. 2. Our proposed method, which directly communicates the presence of false premises to the models, achieves the highest accuracy: 92.4% with GPT-4o-mini, 94.3% with GPT-3.5, 95.2% with Qwen2.5, and 91.4% with Qwen-1.5. This performance surpasses alternative approaches such as *Direct Ask*, *Prompt*, *Majority Vote*, and *Perplexity AI*. *Majority Vote* does not perform well, likely due to hallucination snowballing, where repeated querying amplifies errors rather than correcting them. Additionally, while the *Prompt* method warns the model about potential false premises, it does not specifically tell the LLM which one contains false premises, negatively impacts performance on questions with valid premises, causes unnecessary cautiousness and reduces the model's ability to provide direct and confident answers. Besides, *Perplexity AI* does not perform as well potentially because the query format does not align well with web data, leading to suboptimal retrieval of relevant information for certain types of questions. These findings emphasize the importance of tailoring hallucination mitigation strategies to both the model's reasoning process and the nature of the queries it encounters.

We provide the computational cost analysis, the evaluation of impact on multi-hop versus single-hop questions, and additional experiment results in Appx. § D.

6 Conclusion

We propose a retrieval-augmented logical reasoning framework that detects false premises to mitigate LLM hallucinations. Our method explicitly detects and signals false premises, overcoming key limitations of current approaches that rely on model parameters or post-hoc corrections. By structuring detection upfront, it improves robustness, highlighting the value of structured reasoning techniques in improving model reliability.

References

Jinze Bai, Shuai Bai, Yunfei Chu, Zeyu Cui, Kai Dang, Xiaodong Deng, Yang Fan, Wenbin Ge, Yu Han, Fei Huang, Binyuan Hui, Luo Ji, Mei Li, Junyang Lin, Runji Lin, Dayiheng Liu, Gao Liu, Chengqiang Lu, Keming Lu, Jianxin Ma, Rui Men, Xingzhang Ren, Xuancheng Ren, Chuanqi Tan, Sinan Tan, Jianhong Tu, Peng Wang, Shijie Wang, Wei Wang, Shengguang Wu, Benfeng Xu, Jin Xu, An Yang, Hao Yang, Jian Yang, Shusheng Yang, Yang Yao, Bowen Yu, Hongyi Yuan, Zheng Yuan, Jianwei Zhang, Xingxuan Zhang, Yichang Zhang, Zhenru Zhang, Chang Zhou, Jingren Zhou, Xiaohuan Zhou, and Tianhang Zhu. Qwen technical report, 2023. URL https://arxiv.org/abs/2309.16609.

Yihan Cao, Yanbin Kang, Chi Wang, and Lichao Sun. Instruction mining: Instruction data selection for tuning large language models, 2024. URL https://arxiv.org/abs/2307.06290.

Lichang Chen, Shiyang Li, Jun Yan, Hai Wang, Kalpa Gunaratna, Vikas Yadav, Zheng Tang, Vijay Srinivasan, Tianyi Zhou, Heng Huang, and Hongxia Jin. Alpagasus: Training a better alpaca with fewer data, 2024. URL https://arxiv.org/abs/2307.08701.

Yung-Sung Chuang, Yujia Xie, Hongyin Luo, Yoon Kim, James Glass, and Pengcheng He. Dola: Decoding by contrasting layers improves factuality in large language models, 2024. URL https://arxiv.org/abs/2309.03883.

Shehzaad Dhuliawala, Mojtaba Komeili, Jing Xu, Roberta Raileanu, Xian Li, Asli Celikyilmaz, and Jason Weston. Chain-of-verification reduces hallucination in large language models, 2023. URL https://arxiv.org/abs/2309.11495.

Darren Edge, Ha Trinh, Newman Cheng, Joshua Bradley, Alex Chao, Apurva Mody, Steven Truitt, Dasha Metropolitansky, Robert Osazuwa Ness, and Jonathan Larson. From local to global: A graph rag approach to query-focused summarization, 2025. URL https://arxiv.org/abs/2404.16130.

Aaron Grattafiori et al. The llama 3 herd of models, 2024. URL https://arxiv.org/abs/2407.21783.

Xiaoxin He, Yijun Tian, Yifei Sun, Nitesh V. Chawla, Thomas Laurent, Yann LeCun, Xavier Bresson, and Bryan Hooi. G-retriever: Retrieval-augmented generation for textual graph understanding and question answering, 2024. URL https://arxiv.org/abs/2402.07630.

Shengding Hu, Yifan Luo, Huadong Wang, Xingyi Cheng, Zhiyuan Liu, and Maosong Sun. Won't get fooled again: Answering questions with false premises, 2023. URL https://arxiv.org/abs/2307.02394.

Lei Huang, Weijiang Yu, Weitao Ma, Weihong Zhong, Zhangyin Feng, Haotian Wang, Qianglong Chen, Weihua Peng, Xiaocheng Feng, Bing Qin, and Ting Liu. A survey on hallucination in large language models: Principles, taxonomy, challenges, and open questions. *ACM Transactions on Information Systems*, 43(2):1–55, January 2025. ISSN 1558-2868. doi: 10.1145/3703155. URL http://dx.doi.org/10.1145/3703155.

Albert Q. Jiang, Alexandre Sablayrolles, Arthur Mensch, Chris Bamford, Devendra Singh Chaplot, Diego de las Casas, Florian Bressand, Gianna Lengyel, Guillaume Lample, Lucile Saulnier, Lélio Renard Lavaud, Marie-Anne Lachaux, Pierre Stock, Teven Le Scao, Thibaut Lavril, Thomas Wang, Timothée Lacroix, and William El Sayed. Mistral 7b, 2023. URL https://arxiv.org/abs/2310.06825.

Najoung Kim, Ellie Pavlick, Burcu Karagol Ayan, and Deepak Ramachandran. Which linguist invented the lightbulb? presupposition verification for question-answering, 2021. URL https://arxiv.org/abs/2101.00391.

Nayeon Lee, Wei Ping, Peng Xu, Mostofa Patwary, Pascale Fung, Mohammad Shoeybi, and Bryan Catanzaro. Factuality enhanced language models for open-ended text generation, 2023. URL https://arxiv.org/abs/2206.04624.

Haochen Liu, Song Wang, Yaochen Zhu, Yushun Dong, and Jundong Li. Knowledge graph-enhanced large language models via path selection, 2024a. URL https://arxiv.org/abs/2406.13862.

Ollie Liu, Deqing Fu, Dani Yogatama, and Willie Neiswanger. Dellma: Decision making under uncertainty with large language models, 2024b. URL https://arxiv.org/abs/2402.02392.

Potsawee Manakul, Adian Liusie, and Mark J. F. Gales. Selfcheckgpt: Zero-resource black-box hallucination detection for generative large language models, 2023. URL https://arxiv.org/abs/2303.08896.

Costas Mavromatis and George Karypis. Gnn-rag: Graph neural retrieval for large language model reasoning, 2024. URL https://arxiv.org/abs/2405.20139.

Theo Olausson, Alex Gu, Ben Lipkin, Cedegao Zhang, Armando Solar-Lezama, Joshua Tenenbaum, and Roger Levy. Linc: A neurosymbolic approach for logical reasoning by combining language models with first-order logic provers. In *Proceedings of the 2023 Conference on Empirical Methods in Natural Language Processing*, page 5153–5176. Association for Computational Linguistics, 2023. doi: 10.18653/v1/2023.emnlp-main.313. URL http://dx.doi.org/10.18653/v1/2023.emnlp-main.313.

OpenAI. Gpt-3.5-turbo: Large language model, 2023. URL https://platform.openai.com/docs/models/gpt-3-5. Accessed: 2024-03-18.

OpenAI. Gpt-4 technical report, 2024. URL https://arxiv.org/abs/2303.08774.

Liangming Pan, Alon Albalak, Xinyi Wang, and William Yang Wang. Logic-lm: Empowering large language models with symbolic solvers for faithful logical reasoning, 2023a. URL https://arxiv.org/abs/2305.12295.

Liangming Pan, Xinyuan Lu, Min-Yen Kan, and Preslav Nakov. Qacheck: A demonstration system for question-guided multi-hop fact-checking, 2023b. URL https://arxiv.org/abs/2310.07609.

Pouya Pezeshkpour. Measuring and modifying factual knowledge in large language models, 2023. URL https://arxiv.org/abs/2306.06264.

Qwen, :, An Yang, Baosong Yang, Beichen Zhang, Binyuan Hui, Bo Zheng, Bowen Yu, Chengyuan Li, Dayiheng Liu, Fei Huang, Haoran Wei, Huan Lin, Jian Yang, Jianhong Tu, Jianwei Zhang, Jianxin Yang, Jiaxi Yang, Jingren Zhou, Junyang Lin, Kai Dang, Keming Lu, Keqin Bao, Kexin Yang, Le Yu, Mei Li, Mingfeng Xue, Pei Zhang, Qin Zhu, Rui Men, Runji Lin, Tianhao Li, Tianyi Tang, Tingyu Xia, Xingzhang Ren, Xuancheng Ren, Yang Fan, Yang Su, Yichang Zhang, Yu Wan, Yuqiong Liu, Zeyu Cui, Zhenru Zhang, and Zihan Qiu. Qwen2.5 technical report, 2025. URL https://arxiv.org/abs/2412.15115.

Ansh Radhakrishnan, Karina Nguyen, Anna Chen, Carol Chen, Carson Denison, Danny Hernandez, Esin Durmus, Evan Hubinger, Jackson Kernion, Kamilė Lukošiūtė, Newton Cheng, Nicholas Joseph, Nicholas Schiefer, Oliver Rausch, Sam McCandlish, Sheer El Showk, Tamera Lanham, Tim Maxwell, Venkatesa Chandrasekaran, Zac Hatfield-Dodds, Jared Kaplan, Jan Brauner, Samuel R. Bowman, and Ethan Perez. Question decomposition improves the faithfulness of model-generated reasoning, 2023. URL https://arxiv.org/abs/2307.11768.

Weijia Shi, Xiaochuang Han, Mike Lewis, Yulia Tsvetkov, Luke Zettlemoyer, and Scott Wen tau Yih. Trusting your evidence: Hallucinate less with context-aware decoding, 2023. URL https://arxiv.org/abs/2305.14739.

Ben Snyder, Marius Moisescu, and Muhammad Bilal Zafar. On early detection of hallucinations in factual question answering. In *Proceedings of the 30th ACM SIGKDD Conference on Knowledge Discovery and Data Mining*, pages 2721–2732, 2024.

Jiashuo Sun, Chengjin Xu, Lumingyuan Tang, Saizhuo Wang, Chen Lin, Yeyun Gong, Lionel M. Ni, Heung-Yeung Shum, and Jian Guo. Think-on-graph: Deep and responsible reasoning of large language model on knowledge graph, 2024. URL https://arxiv.org/abs/2307.07697.

Hugo Touvron, Louis Martin, Kevin Stone, Peter Albert, Amjad Almahairi, Yasmine Babaei, Nikolay Bashlykov, Soumya Batra, Prajjwal Bhargava, Shruti Bhosale, Dan Bikel, Lukas Blecher, Cristian Canton Ferrer, Moya Chen, Guillem Cucurull, David Esiobu, Jude Fernandes, Jeremy Fu, Wenyin Fu, Brian Fuller, Cynthia Gao, Vedanuj Goswami, Naman Goyal, Anthony Hartshorn,

Saghar Hosseini, Rui Hou, Hakan Inan, Marcin Kardas, Viktor Kerkez, Madian Khabsa, Isabel Kloumann, Artem Korenev, Punit Singh Koura, Marie-Anne Lachaux, Thibaut Lavril, Jenya Lee, Diana Liskovich, Yinghai Lu, Yuning Mao, Xavier Martinet, Todor Mihaylov, Pushkar Mishra, Igor Molybog, Yixin Nie, Andrew Poulton, Jeremy Reizenstein, Rashi Rungta, Kalyan Saladi, Alan Schelten, Ruan Silva, Eric Michael Smith, Ranjan Subramanian, Xiaoqing Ellen Tan, Binh Tang, Ross Taylor, Adina Williams, Jian Xiang Kuan, Puxin Xu, Zheng Yan, Iliyan Zarov, Yuchen Zhang, Angela Fan, Melanie Kambadur, Sharan Narang, Aurelien Rodriguez, Robert Stojnic, Sergey Edunov, and Thomas Scialom. Llama 2: Open foundation and fine-tuned chat models, 2023. URL https://arxiv.org/abs/2307.09288.

Tu Vu, Mohit Iyyer, Xuezhi Wang, Noah Constant, Jerry Wei, Jason Wei, Chris Tar, Yun-Hsuan Sung, Denny Zhou, Quoc Le, and Thang Luong. Freshllms: Refreshing large language models with search engine augmentation, 2023. URL https://arxiv.org/abs/2310.03214.

Jerry Wei, Da Huang, Yifeng Lu, Denny Zhou, and Quoc V. Le. Simple synthetic data reduces sycophancy in large language models, 2024. URL https://arxiv.org/abs/2308.03958.

Jundong Xu, Hao Fei, Liangming Pan, Qian Liu, Mong-Li Lee, and Wynne Hsu. Faithful logical reasoning via symbolic chain-of-thought, 2024. URL https://arxiv.org/abs/2405.18357.

Nan Xu and Xuezhe Ma. Decoprompt : Decoding prompts reduces hallucinations when large language models meet false premises, 2025. URL https://arxiv.org/abs/2411.07457.

Xinyan Velocity Yu, Sewon Min, Luke Zettlemoyer, and Hannaneh Hajishirzi. Crepe: Open-domain question answering with false presuppositions, 2022. URL https://arxiv.org/abs/2211.17257.

Hongbang Yuan, Pengfei Cao, Zhuoran Jin, Yubo Chen, Daojian Zeng, Kang Liu, and Jun Zhao. Whispers that shake foundations: Analyzing and mitigating false premise hallucinations in large language models, 2024. URL https://arxiv.org/abs/2402.19103.

Muru Zhang, Ofir Press, William Merrill, Alisa Liu, and Noah A. Smith. How language model hallucinations can snowball, 2023a. URL https://arxiv.org/abs/2305.13534.

Yue Zhang, Yafu Li, Leyang Cui, Deng Cai, Lemao Liu, Tingchen Fu, Xinting Huang, Enbo Zhao, Yu Zhang, Yulong Chen, Longyue Wang, Anh Tuan Luu, Wei Bi, Freda Shi, and Shuming Shi. Siren's song in the ai ocean: A survey on hallucination in large language models, 2023b. URL https://arxiv.org/abs/2309.01219.

Lianmin Zheng, Wei-Lin Chiang, Ying Sheng, Siyuan Zhuang, Zhanghao Wu, Yonghao Zhuang, Zi Lin, Zhuohan Li, Dacheng Li, Eric P. Xing, Hao Zhang, Joseph E. Gonzalez, and Ion Stoica. Judging llm-as-a-judge with mt-bench and chatbot arena, 2023. URL https://arxiv.org/abs/2306.05685.

Liwen Zheng, Chaozhuo Li, Xi Zhang, Yu-Ming Shang, Feiran Huang, and Haoran Jia. Evidence retrieval is almost all you need for fact verification. In Lun-Wei Ku, Andre Martins, and Vivek Srikumar, editors, *Findings of the Association for Computational Linguistics: ACL 2024*, pages 9274–9281, Bangkok, Thailand, August 2024. Association for Computational Linguistics. doi: 10.18653/v1/2024.findings-acl.551. URL https://aclanthology.org/2024.findings-acl.551/.

Yanxu Zhu, Jinlin Xiao, Yuhang Wang, and Jitao Sang. Kg-fpq: Evaluating factuality hallucination in llms with knowledge graph-based false premise questions, 2024. URL https://arxiv.org/abs/2407.05868.

Appendix

A Methodology Details

A.1 Problem Definition

False Premise Detection: Given a user query q, the function $F(q)$ determining whether q contains a false premise can be defined as:

$$F(q) = \begin{cases} 1, & \text{if } q \text{ conflicts with retrieved evidence } R(q, G), \\ 0, & \text{otherwise.} \end{cases} \tag{1}$$

where R denotes the retrieval function that extracts relevant evidence from a knowledge graph G. The query q is evaluated against $R(q, G)$, and if contradictions are found, q is deemed to contain a false premise ($F(q) = 1$); otherwise, it is considered valid ($F(q) = 0$). In this study, the function F is achieved by RAG using a retriever that leverages logical form and a knowledge graph.

A.2 Pseudo-code Summary of the Proposed Method

We show the pseudocode summary of our approach in Algorithm 1.

A.3 Prompt Details

The following prompt is used to combine the information retrieved from the knowledge graph G (context) and the query logical form $\mathcal{L}(q)$ (query) to form the input to the LLMs discussed in the Section *False Premise Detection with Logical Form.*

```
Given the context below, does the following question contain a false
premise? Answer with 'Yes' or 'No' only. Note that the context is
provided as valid facts in a triple. Context: [context]. Query:
[query].
```

We use the following prompt for logical form conversion:

```
You are given a question. The task is to: 1) define all the predicates
used in the question. 2) parse the question into logic rules based on
the defined predicates 3) translate any logical rules implied by the
question. 4) convert the question into a logical form using predicate
logic. Provide your final answer in the following format: Logical form:
Predicate1(entity1, entity2). Keep all expressions concise and consistent.
Use standard predicate logic notation.
```

B Dataset Details

In KoPL [Zhu et al., 2024], each entity is linked to a specific concept, such as *Leonardo da Vinci* being connected to the concept of an *artist*. The knowledge graph includes 794 distinct concepts, categorized into domains based on general knowledge, enabling domain-based entity classification. For the art domain, the authors of [Zhu et al., 2024] manually selected 33 relations, ensuring that each relation is relevant to its domain and informative, avoiding ambiguity. For example, the relation *artist* is linked to the Art domain, while *family* is more ambiguous and excluded. Table 3 shows the representative concepts, relations and subjects in the art domain of KG-FPQ. The dataset comprises 4969 questions in the discriminative task for the art domain, with each true premise question modified using the following editing methods: Neighbor-Same-Concept (NSC), Neighbor-Different-Concept (NDC), Not-Neighbor-Same-Concept (NNSC), Not-Neighbor-Different-Concept (NNDC), Not-Neighbor-Same-Relation (NNSR), and Not-Neighbor-Different-Relation (NNDR).

Algorithm 1 False premise detection and hallucination mitigation

Input: User query q, Knowledge graph G
Output: Hallucination mitigated response from LLM

Convert user query q into logical representation $\mathcal{L}(q)$ ▷ (§Logical Form Extraction)
Extract logical assertions $P(x_1, x_2, \ldots, x_n)$ from $\mathcal{L}(q)$
Initialize maximum similarity score $Sim_{max} \leftarrow -\infty$ ▷ (§Retrieval)
Initialize optimal graph $G^* \leftarrow \emptyset$
Candidate set $G^* \leftarrow$ subsets of relevant subgraphs from G, i.e., $R(G)$
for triple $G' \in G$ **do**
    if retriever is embedding-based **then**
        Compute similarity via embeddings:

$$Sim \leftarrow \text{Sim}\left(\mathcal{L}(q), G'\right)$$

    else if retriever is non-parametric **then**
        Compute similarity using tree search criteria:

$$Sim \leftarrow \text{PCST}\left(\mathcal{L}(q), G'\right)$$

    else if retriever is LLM-based **then**
        Compute similarity using LLM scoring:

$$Sim \leftarrow \text{LLMScore}\left(\mathcal{L}(q), G'\right)$$

    end if
    if $Sim > Sim_{max}$ **then**
        $Sim_{max} \leftarrow Sim$
        $G^* \leftarrow G'$
    end if
end for
Define false premise indicator function: ▷ (§A.1)

$$F(q) = \begin{cases} 1, & \text{if } q \text{ conflicts with retrieved evidence } G^* = R(q, G^*) \\ 0, & \text{otherwise} \end{cases}$$

if $F(q) = 1$ **then** ▷ (§Hallucination Mitigation)
    Update query as:

$$q \leftarrow q + \text{" Note: This question contains a false premise."}$$

end if
Generate response from LLM using updated query q
return Hallucination mitigated response from LLM

Domain	Concept e.g.	Concept Qty	Subject e.g.	Subject Qty	Relation e.g.	Relation Qty
Art	film television series drama	44	Titanic Modern Family Hamlet	1754	cast member composer narrative location	33

Table 3: Representative concepts, relations, and subjects in KG-FPQ art domain.

C Experiment Details

C.1 False Premise Detection with Logical Form

In the false premise detection task, we look at different retrievers with and without the use of logical forms. Logical forms are used in 1) the retrieval stage, where the logical form $\mathcal{L}(q)$ is encoded to find the most relevant elements from knowledge G, and 2) the false premise detection stage, where the logical form is passed as input along with the retrieved evidence to LLM to determine whether the query contains false premise. The prompt detail is in Appx. § A.3. We evaluate the use of logical forms in three configurations: 1) applying logical forms in both the retrieval stage and false premise detection stage, 2) using logical forms for retrieval and employing the original query for false premise detection, and 3) utilizing the original query for both stages.

C.2 False Premise Detection Methods

We evaluate how logical form impacts retrieval for false premise detection across the following retrievers:

1) **Direct Claim**: We directly query the LLM to determine whether the given question contains a false premise. The model is prompted with: *Does the following question contain a false premise? Answer with 'Yes' or 'No' only.* 2) **Embedding-based Retriever**: ***with RAG*** selects the top-k[3] relevant triples from the knowledge graph based on the cosine similarity between the query embedding and the graph triple embedding. 3) **Non-parametric Retriever**: ***G-retriever*** [He et al., 2024] uses Prize-Collecting Steiner Tree algorithm for extracting relevant subgraph from the knowledge graph. It does not rely on a trained model with learnable parameters. 4) **LLM-based Retriever**: ***GraphRAG/ToG*** [Edge et al., 2025, Sun et al., 2024] asks the LLM to generate a score between 0 and 100, indicating how helpful the generated answer is in answering the target question. The answers are sorted in descending order of helpfulness score and used to generate the final answer returned to the user.

We use GPT-4o-mini as the LLM in the false premise detection task. These retrievers are included because they enable retrieval without task-specific fine-tuning, making them more adaptable across different domains. Unlike training-based retrievers, which require labeled data and extensive computation, non-parametric retriever uses structured knowledge, embedding-based retriever utilizes pre-trained encoders to transform queries and knowledge into a shared vector space for efficient retrieval, and LLM-based retrieval leverages pre-trained language models' generalization abilities. This setup evaluates the impact of logical forms on retrieval efficiency without the overhead of model training.

Metrics We evaluate the false premise detection task using TPR (true positive rate), TNR (true negative rate), FPR (false positive rate), FNR (false negative rate), F1 score, and accuracy of the model successfully identifying questions containing false premises or not. Here, a *positive* instance refers to a question that contains a false premise. Higher TPR indicates better detection of false premises.

C.3 Hallucination Mitigation Methods

Having used logical forms to improve query structuring and false premise detection, we wish to illustrate how our logical form-based method further reduces hallucinations. We consider the following methods as our hallucination mitigation baselines, which are all inference-time hallucination mitigation strategies that do not require access to logits or internal model weights that operate exclusively at the input level, ensuring a fair comparison:

1) **DirectAsk**: Directly query the LLMs for an answer without additional processing or external retrieval. This approach relies on the model's internal knowledge and reasoning capabilities to handle potential false premises. 2) **Prompt**: We encourage the LLM to assess potential false premises before generating a response by appending the following prompt to the original query: *This question may contain a fasle premise. [query]* 3) **Majority Vote (MajVote)**: We prompt the LLM three times with the same prompt and select the most frequent response as the final answer. This method improves reliability by reducing the impact of any single erroneous or hallucinated response. from LLM. 4) **Perplexity AI**[4]: Utilizes a search engine to retrieve and incorporate real-time information from the web, enabling it to provide answers based on the latest available web data. We use the version powered by GPT-4-Omni.

For **Direct Ask** and **Majority Vote**, we report the performances of the following LLMs: GPT-4o-mini [OpenAI, 2024], GPT-3.5-turbo [OpenAI, 2023], Llama-3.1-8B-Instruct [et al., 2024], Mistral-7B-Instruct-v0.2 [Jiang et al., 2023], Qwen2.5-7B-Instruct [Qwen et al., 2025], and Qwen-1.5-7b-chat [Bai et al., 2023].

Metrics. We evaluate question-answering accuracy on the hallucination mitigation task. Accuracy is calculated by string matching the responses of LLMs: for TPQs, answering "Yes" is considered correct; for FPQs, answering "No" is considered correct.

[3]This work focuses on top-1 selection.

[4]https://www.perplexity.ai

D Additional Analysis

D.1 Performance on Single- vs. Multi-hop Questions

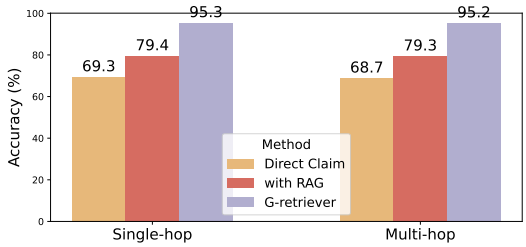

Figure 4: GPT-4o-mini and G-retriver: False premise detection accuracy across single-hop and multi-hop queries. Using logical form-based RAG mainly helps detect false premises in multi-hop questions.

Our approach mostly improves false premise detection performance on multi-hop questions, according to Fig. 4. The incorporation of logical form-based RAG leads to notable performance gains compared to direct claim evaluation. Specifically, while single-hop questions see moderate improvement, multi-hop questions benefit more, with false premise detection performance increasing from 68.7% in the direct claim setting to 79.3% with RAG and further to 95.2% when using the G-retriever. These results suggest that leveraging retrieval mechanisms enhances reasoning over multiple pieces of evidence, reinforcing the importance of retrieval-augmented methods for complex question-answering tasks. We present a case study to illustrate how our method improves performance on multi-hop questions in Appx. § D.2.

D.2 Case Study

We perform a case study demonstrating how our method enhances multi-hop question performance. We use the *with RAG* method example here, where the edited object is not directly connected to the subject within the KoPL knowledge graph.

Question: `Is "The Dark Knight" the recipient of the 16th Screen Actors Guild Awards?`

Answer: `Yes, "The Dark Knight" won the award for Outstanding Performance by a Stunt Ensemble in a Motion Picture at the 16th Screen Actors Guild Awards.`

Logical Form: `is a recipient of('The Dark Knight', 16th Screen Actors Guild Awards)`

Retrieved Graph Triple: `['The Dark Knight', 'award received', '81st Academy Awards']`

After Detecting and Informing LLM of the Presence of a False Premise:

Corrected Answer:

`No, "The Dark Knight" was not the recipient of the 16th Screen Actors Guild Awards. That year's SAG Award for Outstanding Performance by a Cast in a Motion Picture went to Inglourious Basterds, not The Dark Knight.`

D.3 Computational Cost Analysis

Tab. 4 briefly shows the comparison between our work and previous post-hoc hallucination mitigation method. Our method incurs lower training cost, requires less number of tokens and less training time, and is model agnostic as well as black-box compatible.

In Tab. 5, we compare our method with the post-hoc Contrastive Decoding [Shi et al., 2023] approach in terms of computational efficiency and model compatibility (accuracy result based on Llama-3.1-8B). Our method reduces running time, uses fewer tokens by leveraging logical forms, and supports

Method	Training cost	Number of tokens	Training time	Model agnostic	Black-box Compatible
Post-hoc method	Depends on fine-tuning	Train: original query + answer Inference: original query	Depends on fine-tuning	No	No
Ours	Zero	Original query + logical form	Zero	Yes	Yes

Table 4: Comparison of training and compatibility between post-hoc method and our method.

Method	Accuracy	Number of tokens	Running time*	Model agnostic	Black-box Compatible
Contrastive Decoding	84.8	Original Query + Reasoning Step (Length $\gg$ Logical Form)	Context Retrieval Time + 10.6s	Agnostic to White Box Models	No
Our Method	89.5	Original Query + Logical Form	Context Retrieval Time + 0.6s	Yes	Yes

Table 5: Comparison of performance and efficiency between contrastive decoding and our method. *Average running time of each query on NVIDIA RTX A6000 GPU using Llama-3.1-8B Instruct model. Both methods require context retrieval.

both model-agnostic and black-box settings. In contrast, post-hoc methods rely on fine-tuning and lack general applicability across different model architectures. We also include performance comparison of Contrastive Decoding with other LLMs in Supplementary Material §4.

D.4 Comparison with Post-hoc Hallucination Mitigation Method

Tab. 6 presents a performance comparison between Contrastive Decoding [Shi et al., 2023], a post-hoc hallucination mitigation method, and other LLMs (Mistral-7B, Qwen1.5, Qwen2.5-7B-Instruct, Llama-3.1-8B-Instruct). Our method achieves improved performance over Contrastive Decoding on all models except Mistral-7B.

	Mistral-7B	Qwen1.5	Qwen2.5-7B-Instruct	Llama-3.1-8B-Instruct
Contrastive Decoding	89.5	76.2	85.7	84.8
Ours	87.6	89.5	92.4	86.7

Table 6: Comparison between contrastive decoding and our method across different LLMs. Note: GPT-3.5 and GPT-4o-mini are not included as logits are not available for contrastive decoding approach.

D.5 Additional Result on False Premise Detection

We additionally evaluate GPT-3.5-turbo and G-retriever on the false premise detection task using our method. The results are presented below (Tab. 7 and Fig. 5). Notably, when original queries are used in either retrieval, false premise detection, or both stages, despite achieving high accuracy (91.11%), G-retriever shows a markedly lower TPR (37.78%) compared to the first configuration. This suggests that relying on original queries alone, or in combination with logical forms in only one stage for detection, can achieve high accuracy due to correctly identifying negatives, it is less effective at capturing false premises, which is the primary focus of our task.

Metric	G-retriever
Original Query for Both Stages	
True Positives (TP%)	37.78
True Negatives (TN%)	100.00
False Positives (FP%)	0.00
False Negatives (FN%)	62.22
F1 Score (%)	54.84
Accuracy (%)	91.11
Logical Form + Original Query	
True Positives (TP%)	37.78
True Negatives (TN%)	100.00
False Positives (FP%)	0.00
False Negatives (FN%)	62.22
F1 Score (%)	54.84
Accuracy (%)	91.11
Logical Form for Both Stages	
True Positives (TP%)	75.56
True Negatives (TN%)	80.00
False Positives (FP%)	20.00
False Negatives (FN%)	24.44
F1 Score (%)	84.47
Accuracy (%)	79.37

Table 7: False Premise Detection Performance using GPT-3.5-turbo and G-retriever.

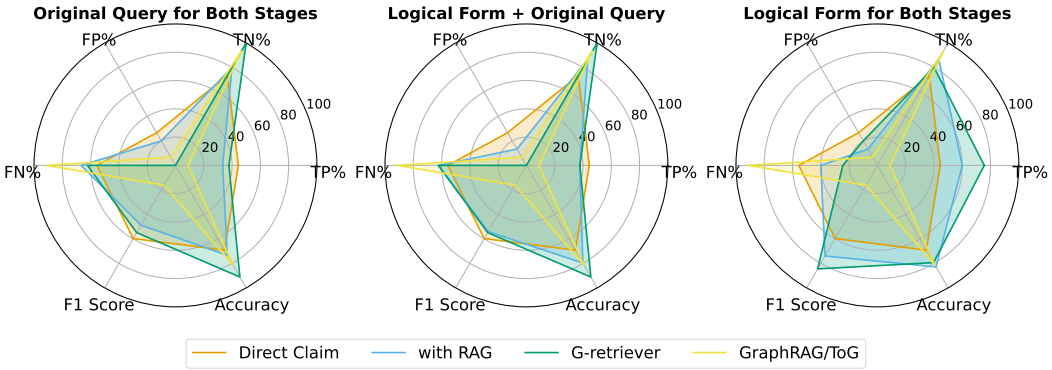

Figure 5: Additional comparison of performance metrics across different retrieval methods using logical forms and/or original queries.