JOURNAL OF
EMERGING INVESTIGATORS

Article

An explainable AI framework for studying query complexity and hallucinations in LLM scheduling tasks

Baelyn Thantron[1], Gopal Gupta[2]
[1]Cabrillo Point Academy, Poway, CA
[2]Department of Computer Science, University of Texas at Dallas, Richardson, Texas

SUMMARY

Large language models (LLMs) are AI systems that generate text and can assist with tasks such as answering questions and scheduling appointments. However, they are susceptible to "hallucinations," where the model provides inaccurate information or misinterprets tool outputs. We hypothesized that increasing query difficulty (i.e., multi-day or ambiguous time window queries) would facilitate higher occurrences of tool-use hallucinations by LLMs in appointment scheduling tasks because models would misinterpret structured calendar data. To test our hypothesis, we built a framework combining GPT-4o with tool-use (LangChain), rule-based validation (Prolog), and explainability methods (LIME and attention visualizations) to test generated queries categorized as simple, complex, or edge case. For simple queries, there was a baseline hallucination rate of 10.2%, 14.8% for complex, and 18.5% for edge cases, with explainability analyses showing that increased error rates resulted from misplaced model attention to more difficult phrases. This framework not only helps diagnose query types that cause LLMs to hallucinate but also shows how we can mitigate these errors to make models more dependable in real-world settings.

INTRODUCTION

Large language models (LLMs) are increasingly used in everyday applications, such as customer support chatbots and automated task assistants. However, it is difficult to rely on these models because they are frequently affected by hallucinations, which are responses that sound believable but are incorrect or purely inventive (1). Hallucination rates can be as high as 19 – 31%, depending on the retrieval setting (3). This issue may arise from the inherent design of LLMs, which are trained to prioritize producing a plausible answer instead of revealing their uncertainty (2). Tool use allows an LLM to call external software, such as a calendar service, to retrieve information or perform an action. An application programming interface (API) allows software to request structured information from external systems, such as fetching calendar availability. One especially problematic type of error is a tool-use hallucination, in which an LLM misrepresents outputs retrieved from external sources such as APIs or databases (3). These hallucinations often make systems unreliable and can also prove dangerous with AI's ability to access and modify sensitive data. Previous research has found that even advanced systems, such as GPT-4, struggle with this problem, often producing incorrect API calls due to their inability to generate accurate input arguments (3).

The most common reasons for hallucinations include vague or unclear instructions given to the model, incorrectly structured data and errors in interpreting that data, or the model placing excessive emphasis on what it learned during training (ignoring the new information it is given) and indicating a "reliance on fixed and parametric knowledge" (4). These issues are exacerbated by complex user queries that involve multiple constraints or vague phrasing. In this study, we examine how the complexity of a query can influence tool hallucinations in LLMs. We will do this by using a case study in appointment setting, an involved task requiring the LLM to consider user preferences and calendar availability to suggest slots to book a call. This task can help understand the nature behind why the models hallucinate because of its inherent difficulty and the overwhelming number of hallucination cases our authors have observed in past projects with appointment setting agents.

We hypothesized that increasing query complexity, by adding time constraints, such as multi-day or ambiguous time ranges, would lead to higher rates of tool-use hallucinations in large language models during appointment scheduling tasks due to misinterpretations of structured availability data. To test our hypothesis, we developed an explainable-AI (XAI) framework to both measure and explain hallucinations across query types **(Figure 1)**.

We selected GPT-4o for this study because it supported the LangChain-based tool-calling workflow needed to retrieve calendar availability while responding to scheduling requests. Prolog validated the model's responses and provided corrective explanations (6), while LIME and attention visualizations were used to examine query features associated with errors (4, 5).

The results confirmed our hypothesis, indicating that hallucination rates increased with query complexity, with time-based misinterpretation as the primary cause. Additionally, in some cases, the feedback from these explanations reduced hallucination frequency when fed back to the LLM, which reveals the relevance of these findings for future work. This framework allows us to automate the detection, explanation, and mitigation of hallucinations **(Figure 1)** to create more reliable systems in high stakes, real world applications.

RESULTS

To investigate whether query complexity was associated with scheduling errors, we tested 9,999 total queries evenly distributed amongst three categories: simple (single unambiguous requests), complex (multi-date or time range requests) and edge case (ambiguous or confusing phrasing). For each query, we recorded whether the model's response matched the calendar availability and calculated

JOURNAL OF
EMERGING INVESTIGATORS
https://doi.org/10.59720/25-313

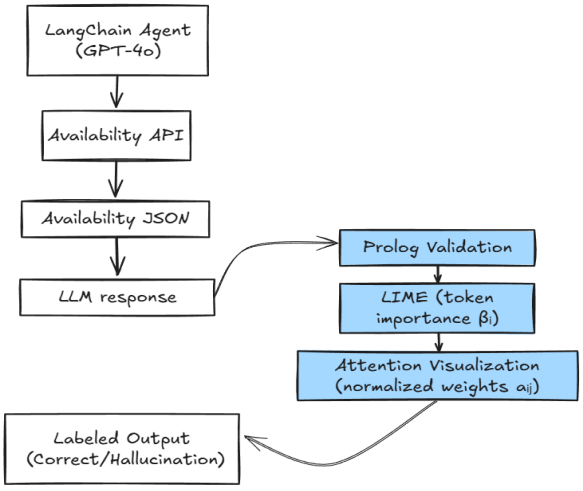


Figure 1. Experimental pipeline for LangChain–GPT-4o agent. Diagram of the experimental setup. Blue boxes represent the validation and explainability steps, while white boxes represent the agent workflow and final labeled output. The LangChain agent queries the availability API, which returns calendar availability in JavaScript Object Notation (JSON) format. The large language model (LLM) then generates a response, which is evaluated through Prolog validation (to label the output as valid or hallucinated). Local Interpretable Model-agnostic Explanations (LIME) token importance analysis (β_i) and attention visualization (normalized weights a_{ij}) are used to attribute which parts of the query are responsible for the model response. This figure details how each step of the framework comes together, specifically highlighting the explainability portion used to diagnose hallucinations.

the hallucination rate for each category **(Figure 2)**. Simple queries showed an average hallucination rate of 10.2 ± 0.6%, complex 14.8 ± 0.7%, and edge cases 18.5 ± 0.8% (two-sided two-proportion z-tests, simple vs. complex, $p = 1.5 \times 10^{-8}$; complex vs. edge case, $p = 6.05 \times 10^{-5}$; simple vs. edge case, $p = 7.56 \times 10^{-22}$; **Figure 2**). The corresponding 95% confidence intervals were 9.0 – 11.4% for simple queries, 13.3 – 16.4% for complex queries, and 17.0 – 20.2% for edge cases. These errors largely involved misinterpretations of the availability output. We used LIME and attention visualizations to examine query features associated with errors in complex and edge-case queries **(Figures 3, 4)**. In a Thursday/Friday query where Thursday was available but Friday was not, LIME showed that the model over emphasized "Friday" ($\beta_j = 0.38$) and under emphasized "Thursday" ($\beta_j = 0.12$), causing it to overlook the available Thursday slot **(Figure 3)**. Attention weights provided a complementary view of how tokens were weighted during model processing **(Figure 4)**. In the complex query, the attention was split across explicit day/time constraints. However, in the edge case query, attention was more diffuse across tokens, which is consistent with the model's uncertainty when interpreting vague language, such as "morning."

Next, we performed an exploratory re-feeding experiment using 500 previously evaluated queries. Prolog corrections were given back to the model, and responses were regenerated. Across these paired trials, the hallucination rate decreased from 16.8% (95% CI: 12.1 – 19.2) to 4.6% (95% CI: 1.7 – 5.3) after incorporating the Prolog explanations (**Figure 5**; $p < 0.0001$). This represents a 12.2-percentage-point reduction in the hallucination rate.

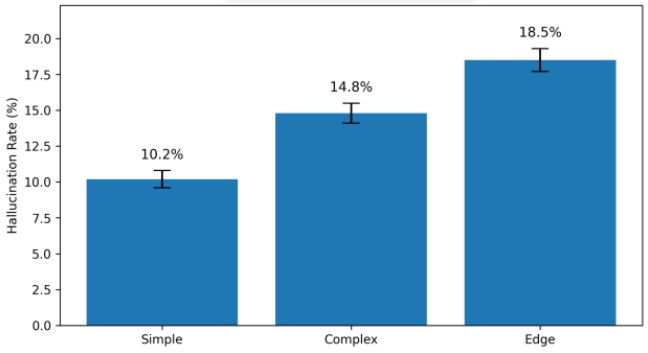


Figure 2. Hallucination rate (%) across query complexity. Bar chart comparing hallucination rates for simple, complex, and edge-case scheduling queries (n = 3,333 per category; 9,999 total). Error bars represent ±1 standard error. The two-sided two-proportion z-tests found differences between simple and complex queries ($p = 1.5 \times 10^{-8}$), complex and edge-case queries ($p = 6.05 \times 10^{-5}$), and simple and edge-case queries ($p = 7.56 \times 10^{-22}$); all three comparisons met the Bonferroni-adjusted significance threshold of $\alpha = 0.0167$.

DISCUSSION

In this study, we tested 9,999 appointment-scheduling queries and used Prolog to validate the model's responses. Our hypothesis was supported: increasing query complexity resulted in higher rates of tool-use hallucinations in

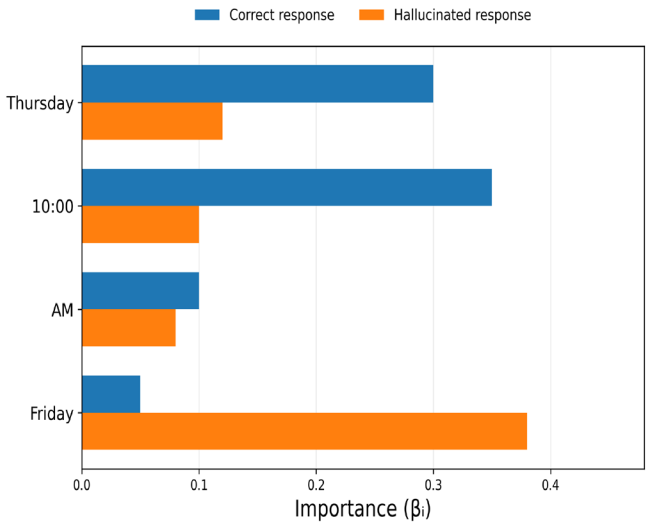


Figure 3. LIME token importance (β_i) for correct vs. hallucinated responses. Grouped horizontal bar chart comparing Local Interpretable Model-agnostic Explanations (LIME) local feature importance weights for a representative query, "Can we meet Thursday or Friday at 10:00 AM?" The 10 AM slot was available on Thursday but unavailable on Friday. The correct response selected Thursday, while the hallucinated response incorrectly concluded that neither slot was available. The hallucinated response showed greater importance for "Friday" and lower importance for "Thursday" than the correct response.

https://doi.org/10.59720/25-313

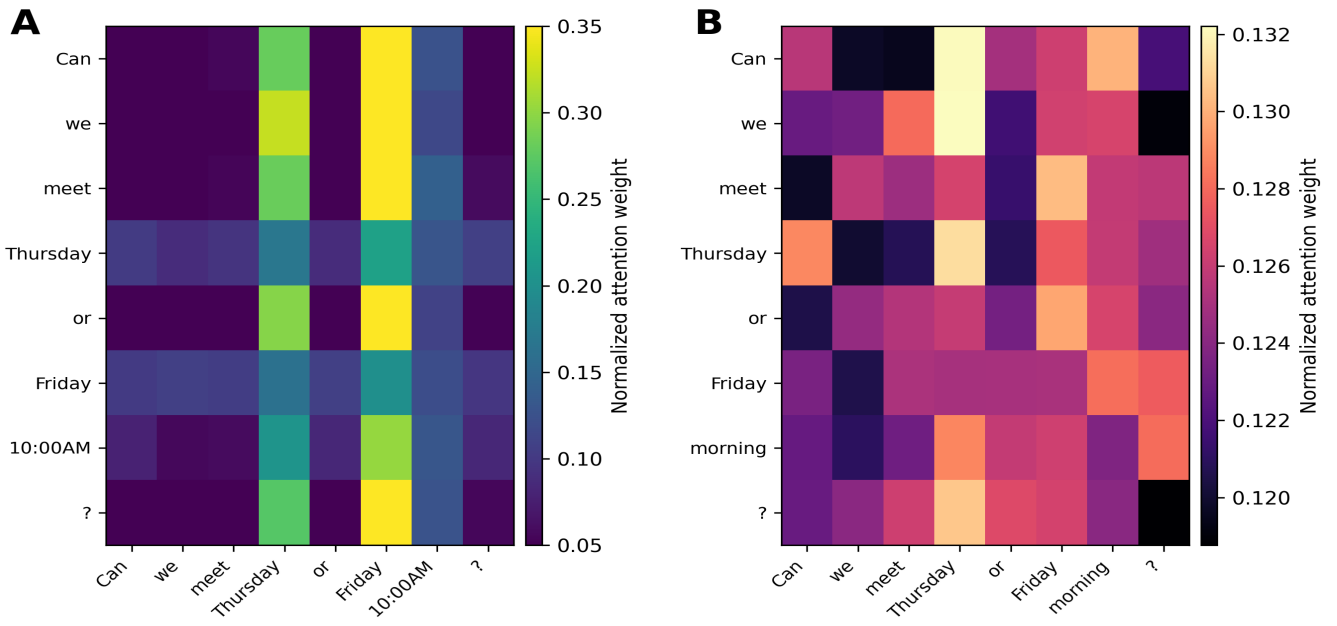


Figure 4. Attention visualization of multi-day vs. edge case queries. Heatmaps showing token-to-token attention weights for the example prompts in each panel. Weights were normalized across all displayed tokens for each query token, so that each row sums to 1. Tokens appear on both axes because each cell represents the attention weight between a pair of tokens. Darker colors indicate lower weights, while lighter colors indicate higher weights, as shown by each panel's color bar. Because the panels use different numerical ranges and color scales, colors should be interpreted within each panel rather than compared directly across panels. (A) Complex query: "Can we meet Thursday or Friday at 10:00 AM?" (B) Edge case query: "Can we meet Thursday or Friday morning?"

appointment scheduling LLMs, specifically due to the misinterpretation of availability. As the LLM was forced to handle more complicated user requests, the hallucination rate increased substantially, from 10.2% in simple queries to 18.5% in edge cases **(Figure 2)**. These rates are consistent with prior reports of tool-use hallucination rates (3). Furthermore, LIME and attention visualizations confirmed that, amongst the errors, misplaced focus was primarily on incorrect time tokens **(Figures 3, 4)**.

These findings revealed an important connection between how LLMs interpret both the user's request and the provided availability data. Across all queries, the availability output from the tool remained consistent. Simple queries, where the model might simply have to match one time slot against a structured list of available times, were performed with a high level of accuracy. In these cases, model weights were almost entirely devoted to the single constraint given by the user. However, once time ranges and/or multiple days were introduced, the same availability output was ignored. The model would incorrectly inform users that an available slot was booked or that a booked slot was available. For complex cases (time range/multiple days), LIME and attention visualizations showed split weighting among various phrases from the query. This outcome suggests that the model struggles to balance its "attention" across multiple constraints simultaneously, resulting in the misinterpretation of constant, unmistakable data. The error rate was even higher in edge cases with ambiguous phrasing, such as "morning" or "evening". This was also seen in cases where only a single day was mentioned. In these cases, attention visualizations showed scattered, low-intensity focus across multiple unrelated parts of the prompt, a reflection of the model's uncertainty when encountering vague phrasing.

It is important to note that results are based on a specific use case of LLMs, an appointment scheduling agent tasked with giving users information on call availability. However, the context of appointment scheduling exhibits tool-use scenarios that are found in broader domains, such as querying databases through APIs (as our model did to retrieve calendar availability), or any other use cases where models get information from tools, make decisions, and respond to queries. Therefore, our generalizations should be treated as a hypothesis for future work rather than a direct conclusion of this scheduling-based study.

The major significance of our findings is that by understanding how hallucinations arise, we gain insights about how to prevent them. We saw two distinct causes behind tool use hallucinations: overload and uncertainty. In the complex queries involving multiple days or time ranges, the model exhibited split attention across the various constraints. It struggled to determine which constraints were relevant when several valid options existed. As a result, the model would often overlook some potential options and return false information about availability. Although it is tempting to conclude here and advocate for the need for simpler, more direct user queries, in real-world deployment, complexity and ambiguity are not exceptions, but rather inevitable. AI technology continues to be adopted in many industries, and its use cases are relied upon more heavily for increasingly consequential and complex tasks (7). Outside of appointment setting, AI agents are used for retrieval-augmented question answering, customer support, writing to databases, screening

https://doi.org/10.59720/25-313

for eligibility, etc. However, even within scheduling, one can imagine the catastrophe of missed healthcare visits, legal consultations, or internal financial processes due to incorrect time-based authorization. In practice, users phrase queries flexibly (like "anytime in the morning") and real scheduling requests inherently involve multiple constraints across time. Thus, models must adapt to interpret uncertainty and reconcile a myriad of constraints to improve the reliability of AI systems in real life. Our findings provide the foundation for research around improving models' prioritization of multi-constraints by studying how attention can be reweighted or guided when multiple tokens compete for relevance. For instance, future experiments could test whether structured prompting or attention regularization (like penalizing overly distributed weights) reduces errors in complex tasks. Similarly, the observed pattern of diffuse attention in edge cases suggests developing context-resolution modules that prompt the model to request clarification when uncertainty passes a certain threshold.

Our findings also present a complementary path of improving correction after an error occurs. We observed that feeding Prolog's corrected explanations back to the model made it less likely for hallucinations to repeat: across 500 paired trials, hallucinations decreased from 16.8% to 4.6% after refeeding **(Figure 5)**. This improvement provides a basis that future studies can expand upon: the use of explainable AI tools to form self-correcting systems where errors are caught, fed back to the model, and responses are regenerated. Compared to other approaches, such as regex format validation (which mainly verifies surface-level structure and syntax) and observability tools like Langsmith (which detect and quantify failures during runs), Prolog is an explicit way to check against structured availability, provide justification for errors, and suggest other candidate slots. An approach like this combines automated reasoning, which is traceable and useful for complex logic, with generative AI, which is useful for direct user interaction, to build more trustworthy machine learning systems.

There are various ways this study could have been strengthened to ensure that these observed results were consistent. First, having repeated trials would help address one potential limitation of the data collected, which is the role of luck and timing. Unfortunately, testing for hallucinations in LLMs is not always reliable; assigning a cause to a hallucination is difficult when one query (of the same difficulty type) can produce both correct and incorrect responses depending on chance (1, 2). Consequently, repeating trials and averaging data across all sets would result in higher confidence in observed hallucination rates. More limitations include the small number of query types used and their generated, controlled nature. Our standardized definitions of ambiguous time phrases may also influence the results; for instance, future studies could test whether defining "morning" as starting at 8 AM instead of 9 AM changes the hallucination rates. We used the introduction of multi-day and time ranges for the complex cases, and ambiguous time-of-day phrases for the edge cases to show an increasing level of difficulty for queries, which was helpful in observing hallucination rates differing across well-defined categories. However, different input types, such as different date formats, typos, and informal language, are more common in everyday situations, and the inclusion of these queries would help validate our generalization of "query complexity" in practice. Furthermore, including natural, messy queries may help diagnose and pinpoint further words or phrases that are likely to cause hallucinations.

Significantly, this study provided quantifiable evidence that tool-use hallucinations are not random, but predictable and traceable. The use of LIME and attention visualizations showed split attention in complex queries and diffuse attention in edge case queries. These findings are relevant for end users and suggest the use of more explicit, direct queries. Future research could explore preventing hallucinations during response generation and correcting errors after they occur. Although this study focused on appointment scheduling, its findings suggest a path toward AI systems that verify information from external tools and correct errors before passing inaccurate responses to users, greatly improving their reliability in real-world applications.

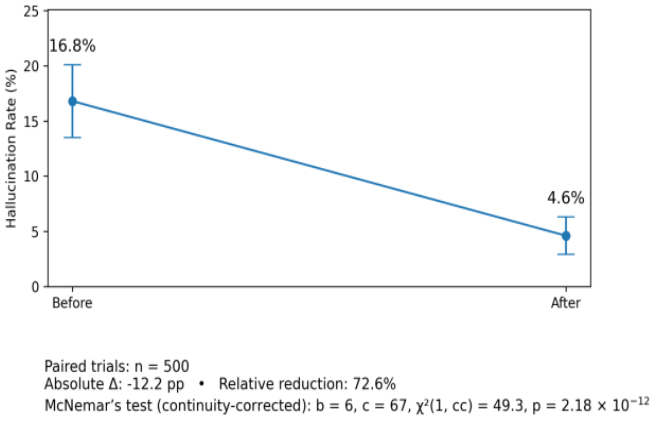


Figure 5. Effect of Prolog feedback on hallucination reduction. Paired slopegraph showing hallucination rate before and after introducing Prolog feedback for the same randomly sampled trials (n = 500). Error bars denote 95% confidence intervals for the hallucination rate. The intervention reduced hallucinations by 72.6% (absolute Δ = −12.2 pp). McNemar's test (b = 6, c = 67; $p < 0.0001$) confirmed the change was statistically significant.

MATERIALS AND METHODS

Overview of experimental framework.

A LangChain agent with GPT-4o, developed by OpenAI, was used to read and process user queries (8). It was given access to tools that fetch availability to book a call and prompted to return whether there were matching slots for the user's call preferences. After the response was generated, Prolog was used for checking the validity of the LLM response (i.e., was the time slot truly available?). Prolog's rule-based nature allowed us to encode the returned tool availability into a set of rules that could be parsed through and compared against the model's reply. Notably, Prolog facts were generated dynamically from the availability retrieved by the model and checked using fixed validation rules, ensuring that both used the same availability data (see **Appendix**). If the Prolog code found a discrepancy, it would mark the instance of the hallucination and LIME would determine what aspects of the user's query the model was placing the most emphasis on, thus isolating the cause of the hallucination. Finally,

https://doi.org/10.59720/25-313

attention visualizations would show where the model's focus had been misplaced, like weighing an irrelevant date/time range more heavily than the correct one.

System components and software

All experiments were performed on a cloud-based virtual machine with four virtual CPUs and eight GB of RAM. A LangChain agent (Python v3.14.0) incorporating GPT-4o (temperature = 0.2) was used to generate responses to user queries. Each query prompted a tool call to an availability API that returned structured data in JSON format, which contained day-time key-value pairs representing open appointment slots. The same JSON object was converted into Prolog facts and rules for logical validation using SWI-Prolog (v9.2.9). Interpretability was achieved using the LIME (Python lime library) package and attention visualizations derived from the Hugging Face transformers library. Combining tools was handled with Flask and Python requests, and all code and configuration files are available in the Appendix and GitHub repository.

Data generation and query categories

A dataset of 9,999 total queries was used, evenly divided into simple, complex, and edge-case categories, based on the number of time constraints and whether there was any ambiguity. Simple queries contained exactly one explicit day and time and contained no vague temporal language (e.g., "Is Thursday at 10:15 AM available?"). Complex queries contained two or more potential days, a multi-hour time range, and/or an "OR" across days or times (e.g., "Can we meet Thursday or Friday between 10 and 12?"), but the time-related terms themselves always remained explicitly numeric. Edge case queries contained at least one ambiguous phrasing such as "morning," "evening," or "around 10 or 11" (e.g. "Can we meet Thursday morning?"), regardless of how many days or times were included. The same availability JSON remained constant for all trials to isolate the effect of query phrasing. Queries were randomly ordered before testing to avoid bias.

Validation with Prolog

Each response from the LLM was verified using Prolog logic rules that compared the model's answer to the ground-truth availability data. JSON data were translated into Prolog facts of the form available_slot(Date, Day, Label, StartMin, EndMin), representing open time intervals. Logical rules defined that a time T was valid if it occurred within the specified start-end range on the corresponding day. For ambiguous terms such as "morning," mappings were standardized (e.g., morning = 09:00–12:00). Any contradiction between the model's response and Prolog's validation output was marked as a hallucination.

For example, a model could receive structured availability data, such as: {"Thursday": ["9:00 AM–12:00 PM"], "Friday": ["1:00 PM–4:00 PM"]} **(Table 1)**. A simple query such as "Can we meet Friday at 1 PM?" would typically be answered with sufficient reliability: "Yes, Friday at 1 PM is available." An edge-case query like "Can we meet Thursday or Friday morning around 10 or 11?" could elicit the response: "There is no availability on Friday morning, but Thursday morning is booked, so there are no open times." In this case, the model overlooks the available Thursday 10–11 AM block.

Category	Example Query	Ambiguity Source
Simple	"Can we meet Friday at 1 PM?"	None; single time, single day
Complex	"Can we meet Thursday or Friday between 10:00 and 11:00 AM?"	Multi-day, overlapping time window (only Thursday fits given {Thu 9–12, Fri 1–4})
Edge	"Anytime in the morning this week."	Implicit time window ("morning" = 9:00–12:00)

Table 1. Query categories and example prompts Example scheduling queries used to evaluate hallucination behavior. "Ambiguity Source" refers to the element of the prompt introducing uncertainty. These examples are illustrative and do not encompass all possible sources of ambiguity or complexity.

Explainability analysis with LIME and attention visualizations

LIME approximated the local behavior of the model by fitting a simple linear model to the perturbed input samples of each query, identifying which words in a query most influenced the model's output and interpretation of scheduling availability (4). For each explained query, we created 200 perturbed versions by removing or hiding individual tokens, evaluated the LLM on these variants, and then fit the linear surrogate model to approximate the relationship. For an input x and an LLM output f(x), LIME represented the relationship as:

$$f(x) \approx \Sigma\, \beta_i\, x_i, \text{Importance}(x_i) = \beta_i \qquad \text{.....(Equation 1)}$$

Here, β_i represents the importance assigned to each token x_i in the user query. For instance, in the query "Is John available at 10:15 AM?", LIME might assign a higher weight ($\beta_i = 0.35$) to the token "10:15 AM." In cases of hallucinations, misweighted tokens (e.g., overemphasis on "Friday" instead of "Thursday") were reflected in elevated β_i scores, allowing identification of the cause of the error. To keep explanations readable, we picked only the top 5 most influential words (highest β_i) per query, a rule used across all query categories. This choice of threshold was made to exclude redundant, low-influence words with near zero contributions to the underlying LIME scoring.

Attention visualizations were derived from GPT-4o's transformer architecture to compute attention weights aij between tokens (5). Each token representation is projected into a query vector (Q_i) and a key vector (K_j) by said learned attention layer weights, coming from the transformer attention computation, not LIME. The normalized attention weight aij represents how strongly token i attends to token j:

$$a_{ij} = (\exp(Q_i \cdot K_j)) / (\Sigma_k \exp(Q_i \cdot K_k)) \qquad \text{.....(Equation 2)}$$

where Q_i and K_j denote the query and key vectors for

https://doi.org/10.59720/25-313

tokens i and j respectively. High attention weights (a_{ij}) indicate strong influence between tokens. While LIME estimated token importance for the model's output, the attention visualizations showed how strongly the model distributed focus across tokens in the prompt. These maps were used to identify where focus was misplaced, such as assigning high attention to "Friday" when the correct context was "Thursday."

Statistical analysis

For each category, the hallucination rate was calculated as the proportion of invalid responses according to Prolog validation, with 95% confidence intervals via nonparametric bootstrap (1000 resamples). Where required, the standard error (SE) for a proportion was estimated as $\sqrt{((p(1-p))/n)}$. Because the validity of LLM replies can vary even for identical queries, we treated every response as an independent binary trial (hallucination positive or negative). Although we did not repeatedly sample identical queries, we estimated category level hallucination rates by averaging across 3,333 queries per category, which reduces the influence of random variation from any single response. Differences in hallucination rates between categories were analyzed using two-proportion z-tests. For the three pairwise comparisons (simple vs. complex, complex vs. edge case, and simple vs. edge case), we used a Bonferroni-adjusted significance threshold of $\alpha = 0.05/3 = 0.0167$. All three comparisons were statistically significant.

Observation of refeeding effects

Post experiment, a sample of 500 trials from the original experiment were chosen using uniform random sampling (Python "random" module, seed = 2025). For each selected trial, the Prolog-corrected explanation was appended to the original prompt, and the model was prompted to regenerate. Differences in hallucination rates before and after Prolog feedback were evaluated using McNemar's test with continuity correction on paired before-after outcomes (discordant counts b = 6, c = 67), yielding $\chi 2$ (1) = 49.3, $p = 2.18 \times 10^{-12}$. Although not part of the primary experimental design, we logged regenerated responses to analyze later and determine whether corrective feedback could help reduce hallucinations.

ACKNOWLEDGMENTS

We would like to express gratitude to the University of Texas at Dallas's applied logic, programming languages, and systems laboratory (ALPS) summer workshop that initiated this project.

Received: November 19, 2025
Accepted: July 07, 2026
Published: November09, 2026

REFERENCES

1. Borji, Ali. "A Categorical Archive of ChatGPT Failures." *arXiv*, 2023, arxiv.org/abs/2302.03494.
2. Wang, Xuezhi, et al. "Self-Consistency Improves Chain-of-Thought Reasoning in Language Models." *International Conference on Learning Representations (ICLR)*, 2023, arxiv.org/abs/2203.11171.
3. Patil, Shishir G., et al. "Gorilla: Large Language Model Connected with Massive APIs." *arXiv*, 2023, arxiv.org/abs/2305.15334.
4. Cambria, Erik, et al. "XAI Meets LLMs: A Survey of the Relation Between Explainable AI and Large Language Models." *arXiv*, 2024, arxiv.org/abs/2407.15248.
5. Devlin, Jacob, et al. "BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding." *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies (NAACL-HLT)*, 2019, pp. 4171–4186, arxiv.org/abs/1810.04805.
6. Gupta, Gopal. "Automating Commonsense Reasoning with ASP and s(CASP)." *Technical Report, The University of Texas at Dallas*, 2022.
7. Sarkar, Surjodeep, et al. "A Review of the Explainability and Safety of Conversational Agents for Mental Health to Identify Avenues for Improvement." *Frontiers in Artificial Intelligence*, vol. 6, 12 Oct. 2023, https://doi.org/10.3389/frai.2023.1229805.
8. OpenAI. "GPT-4o System Card." *OpenAI*, 8 Aug. 2024, https://openai.com/index/gpt-4o-system-card. Accessed 24 Sept. 2026.

Copyright: © 2026 Baelyn Thantron and Gopal Gupta. All JEI articles are distributed under the Creative Commons Attribution Noncommercial No Derivatives 4.0 International License. This means that you are free to share, copy, redistribute, remix, transform, or build upon the material for any purpose, provided that you credit the original author and source, include a link to the license, indicate any changes that were made, and make no representation that JEI or the original author(s) endorse you or your use of the work. The full details of the license are available at https://creativecommons.org/licenses/by-nc-nd/4.0/deed.en.

APPENDIX

Supporting programming scripts, including the Prolog availability-validation rules, are available on GitHub at https://github.com/cyborgal/xai-llm-hallucination-framework for replication and detailed review.