

LIBRARY HALLUCINATIONS IN LLM-GENERATED CODE: A RISK ANALYSIS GROUNDED IN DEVELOPER QUERIES

Lukas Twist Mark Harman Helen Yannakoudakis Jie M. Zhang



MOTIVATION

- LLMs continue to hallucinate, even when generating code.
- Code hallucinations *create reliability and security risks*.
- Whilst well studied, there is no understanding of how realistic developer queries can impact the hallucination rates.
- We test how prompt variations, such as user-inspired descriptions and mistakes, trigger library hallucinations.



METHODOLOGY

- 7 *diverse LLMs* evaluated, inc. GPT, Claude, DeepSeek.
- LLMs solve Python *BigCodeBench* tasks with...
 - *Developer-style descriptions* from StackExchange.
 - Generated *library misspellings and fabrications*.
- Then imports and members are extracted from generated code and validated against PyPI and documentation.



WHEN LLMs GENERATE LIBRARY-ORIENTED CODE...

THEY OFTEN IGNORE DESCRIPTIVE ADJECTIVES.



Requests for “fast”, “simple”, “modern”, or “lightweight” libraries rarely changed library choices or triggered hallucinations.

THEY HALLUCINATE UNDER RECENCY PRESSURE.



Year-based prompts caused sharp increases in hallucinated libraries, reaching up to 85% of tasks when asking for a library “from 2025”.

THEY COMPLY WITH SMALL LIBRARY-NAME ERRORS.

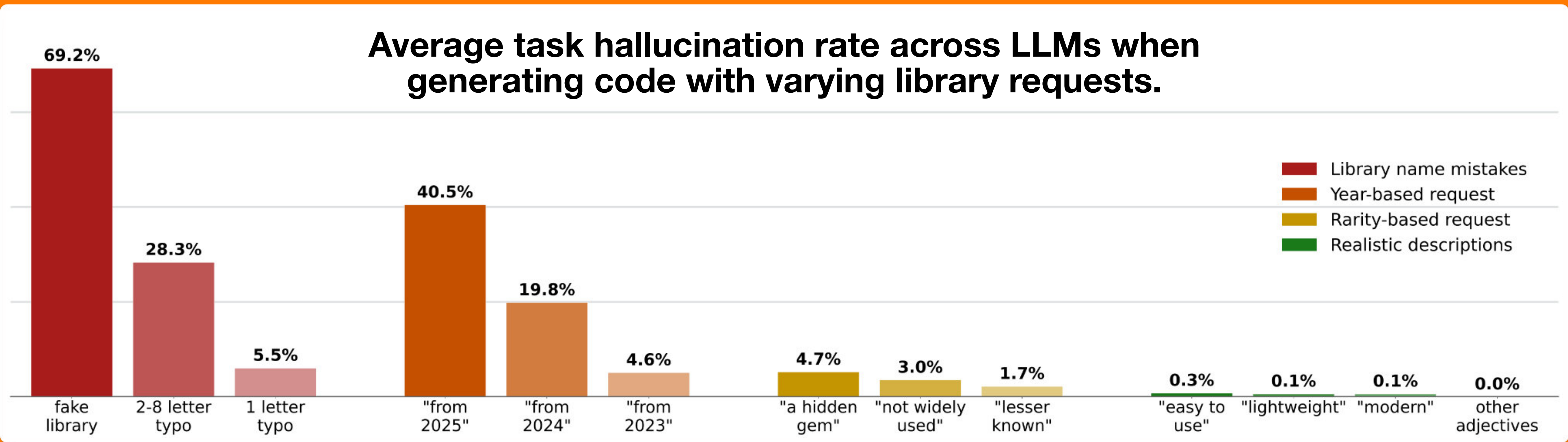


Even one-character typos were sometimes treated as valid libraries, causing hallucinations in up to 26% of tasks.

THEY CONSISTENTLY USE FABRICATED LIBRARIES.



Fabricated library names were used in up to 99% of tasks, showing strong compliance with invalid user references.



ADDITIONAL RESULTS

- Rarity-seeking prompts increase hallucinations.
- Similar risks appear across *JavaScript* and *Rust*.
- Prompt engineering helps inconsistently.
- Tool access can reduce hallucinations, but models can still misuse or underuse it.



DISCUSSION

- Library hallucinations create practical supply chain risks through *slopsquatting* and *typosquatting*.
- Solutions need to be at the model interface level.
- Temporal requests require *cut-off-aware behaviour*.
- Incorrect libraries require *anti-sycophancy behaviour*.

