Discover Data

Research

You believe your LLM is not delusional? Think again! a study of LLM hallucination on foundation models under perturbation

Anirban Saha[1] · Binay Gupta[1] · Anirban Chatterjee[1] · Kunal Banerjee[1]

Received: 2 January 2025 / Accepted: 2 May 2025
Published online: 29 May 2025

Abstract
Large Language Model (LLM) has recently become almost a household term because of its wide range of applications and immense popularity. However, hallucination in LLMs is a critical issue as it affects the quality of an LLM's response, reduces user trust and leads to the spread of misinformation. Detecting hallucination in the presence of the context or a golden response is relatively easier but it becomes considerably more challenging when both of these are absent, which is typically the case post deployment of an LLM. In this study, we present a framework that relies on query perturbation and consistency score calculation between the responses generated against the original query and the perturbed query to identify the potential hallucination scenarios. This framework has no dependency on the availability of the context or the ground truth. In this study, we focus on the popular foundation models because majority of the LLM applications leverage these specific models since training an LLM from scratch or even finetuning LLMs may require a lot of capital investment. Moreover, we specifically investigate LLM hallucinations under different levels of perturbation: character-level, word-level and sentence-level — robustness towards these perturbations indicates that an LLM has a good understanding of a concept, and thus is less susceptible to hallucinations – this, in turn, should help in the LLM's user adoption. Our study shows that GPT-4 hallucinates the least when faced with perturbations; in contrast, other LLMs start hallucinating even with minor typos.

Keywords Large language model (LLM) · LLM hallucination · Perturbation · OpenAI GPT · Google Gemini

1 Introduction

Large Language Model (LLM) is currently one of the most discussed topics in technology. Although the benefits of LLMs are undeniable, there are few concerns regarding these models which hinder their wider adoption – hallucination [1] likely being the primary one. It may be important to note that in a recent work [2], the authors claim that hallucinations in LLMs is inevitable. Specifically, the authors formalize hallucinations as inconsistencies between a computable LLM and a computable ground truth function, then based on the results from information theory, they argue that it is impossible for an LLM to learn all possible computable functions – thus, inescapably leading to hallucinations. Moreover, the real world being much more complex than the formal world, hallucinations are much more likely to manifest in LLMs in practice.

✉ Kunal Banerjee, kunal.banerjee1@walmart.com; Anirban Saha, anirban.saha@walmart.com; Binay Gupta, binay.gupta@walmart.com; Anirban Chatterjee, anirban.chatterjee@walmart.com | [1]Walmart Global Tech, Bangalore 560103, Karnataka, India.

Therefore, it is important to monitor an LLM post deployment to ensure its correct usage in production. However, unlike the training stage, where ground truths or full contexts may be available, a practical strategy to check for hallucinations post deployment is to invoke the LLM multiple times with the same (or semantically similar) questions and then check for its consistency across the responses – this method was proposed in SelfCheckGPT [3]. Another line of work [4, 5] prescribes to add different levels of perturbations (word, character or sentence-level) to the questions and then check for consistency among the generated responses; a high consistency score even with perturbations indicates that the model has truly understood the concept and therefore, it is less likely to generate a hallucinated response. Note that these papers [4, 5] applied perturbations in a non-LLM context.

Thus, the contributions of this work are as follows:

1. We propose for the first time to use perturbation based methodology to better evaluate robustness to hallucinations for LLMs.
2. We carry out extensive experiments with eight popular foundation models to underline the efficacy of the proposed strategy; specifically, we show that the perturbation based method subsumes the one proposed by SelfCheckGPT [3], and one's choice for an LLM based on its robustness to hallucination may alter by following our strategy compared to SelfCheckGPT.
3. We make our code publicly available at https://github.com/anirbans403/robustness_framework. — all our codes are modularized, thereby allowing plug-n-play with different models, perturbation strategies and evaluation metrics easily.

2 Related work

Hallucination detection in LLMs In spite of the challenges involved in detecting hallucinations, significant progress has been made in the case of Retrieval-Augmented Generation (RAG) [6] based systems for LLMs [7, 8]. In fact, the RAGAs library [9], that is primarily developed to evaluate RAG pipelines, provides metrics such as, *context precision*, *context recall* and *answer correctness* which may be re-purposed to measure LLM hallucinations. None of these methods, however, work for monitoring an LLM post its deployment especially for a non-RAG use-case (i.e., when the context is not present) because ground truths are hard to come by. Therefore, for such cases, the approach of SelfCheckGPT [3] is applicable whereby, the same (or semantically similar) query is asked multiple times and if the LLM has understood the concept that the query pertains to, then all its responses will be consistent with the original response. There are various models that may be used for checking consistency (hallucination) between a pair of strings such as, DeBERTa-v3 [10] based hallucination detector [11] and DistilRoBERTa [12] based consistency checker [13]; however, based on our experience, we found that the hallucination evaluation model [14] by Vectara [15] performed the best since it had maximum matches with the human evaluations.

Perturbation of textual data The faith in the robustness of a model is bolstered if it continues to perform equally well (i.e., with minor degradation) when its inputs are perturbed, for example, in [4], the authors apply character and word-level perturbations to the inputs of various language models and test their performance on tasks such as, text classification, named entity recognition and question answering; in another work [5], the authors apply word, character and sentence-level perturbations to check the robustness of sentence encoders. In a more recent work [16], the authors explore character, word and sentence-level perturbations to check a LLM's robustness to adversarial attacks [17]. To motivate why perturbation-based hallucination detection may be preferred, we use the example shown in Table 1 that shows two original queries (having perturbation-level "N/A") involving the monthly growth of a baby boy and the annual growth of Mount Everest. The responses recorded in this table are obtained from Gemini–1.5-Flash [18] – note that we have made minor cosmetic changes to the responses for better illustration and brevity. As can be seen from Table 1, Gemini–1.5-Flash gives consistent answers even with higher-level perturbations for the baby boy related queries compared to Mount Everest, and thus it may be concluded that the concept of growth for a baby boy is better understood by this LLM than that of a mountain.

Table 1 An example using Gemini-1.5-Flash to illustrate perturbation-based hallucination detection

Perturbation-level	Query	Response
N/A	What is the monthly growth in height of a 2 weeks old baby boy?	A 2-week-old baby boy typically grows about 1/2 inch to 1 inch per month.
Character	What is the monhtly growth in height of a 2 weeks old bayby boy?	A two-week-old baby boy typically grows about 1 inch per month.
Word	What is the monthly expansion in height of a 2 weeks old baby boy?	A 2-week-old baby boy typically grows about 1 inch per month.
Sentence	What is the rate of height increase per month for a baby boy who is two weeks old?	It's impossible to determine a specific rate of height increase per month for a two-week-old baby boy without more information but typically it's 1/2 to 1 inch.
N/A	How much does Mount Everest grow higher every year?	Mount Everest grows about 4 to 6 millimeters higher each year.
Character	How much does Mount Everest grw highr every year?	Mount Everest grows about 4 millimeters higher each year.
Word	How much does Mount Everest elevate higher every year?	Mount Everest does not elevate higher every year.
Sentence	What is the annual increase in height of Mount Everest?	Mount Everest is not growing in height annually.

3 Methodology

Algorithm 1 Measure average consistency score for LLMs

```
Require: A list of LLMs: $\{M_1, M_2, \ldots, M_K\}$, a list of queries: $\{Q_1, Q_2, \ldots, Q_L\}$, a list of
    perturbations $\{P_1, P_2, \ldots, P_N\}$, and a consistency measuring function $\mathcal{F}$.
Ensure: Average hallucination scores $\{S_1, S_2, \ldots, S_K\}$ corresponding to each LLM
    $\{M_1, M_2, \ldots, M_K\}$.
                              ▷ Create the set of perturbed queries from the original queries
Initialize a set of queries $\mathcal{Q} \leftarrow \{Q_1., Q_2, \ldots, Q_L\}$
for each query $Q_l, 1 \leq i \leq L$ do
    for each perturbation $P_n, 1 \leq j \leq N$ do
        Generate a perturbed query $Q_{l,n}$ by applying $P_n$ to $Q_l$
        $\mathcal{Q} \leftarrow \mathcal{Q} \cup Q_{l,n}$
    end for
end for
                              ▷ Generate the set of LLM-specific responses from the queries
Initialize a set of responses $\mathcal{R} \leftarrow \Phi$
for each LLM $M_k, 1 \leq K$ do
    for each query $Q$ in $\mathcal{Q}$ do
        Run $M_k$ with $Q$ and store the response $R$ in $\mathcal{R}$
    end for
end for
                              ▷ Compute the average consistency score for each LLM
for each LLM $M_k, 1 \leq k \leq K$ do
    Initialize $S_k \leftarrow 0$
    for each query $Q_l, 1 \leq l \leq L$ do
        $s_k^l \leftarrow 0$
        for each perturbation $P_n, 1 \leq n \leq N$ do
            $s_k^l \leftarrow s_k^l + \mathcal{F}(R_l^k, R_{l,n}^k)$
        end for
        $S_k \leftarrow S_k + s_k^l$
        $s_k^l \leftarrow s_k^l \div N$
    end for
    $S_k \leftarrow S_k \div (L \times N)$
end for
return $S_1, S_2, \ldots, S_K$
```

Our method is described concisely in Algorithm 1. Specifically, we explore six proprietary LLMs – the former three are from OpenAI while the latter three are from Google, and two open-source models – one from Meta and the other from Google.

- GPT-3.5-Turbo: This model has 175 billion parameters that is the same as GPT-3, and is obtained by finetuning the GPT-3 model.
- GPT-4: Some sources say that this model has 1.7 trillion parameters although OpenAI has not officially revealed its size [19]. In contrast to its predecessor, this model supports multi-modality and has enhanced safety and alignment features.
- GPT-4-Turbo: This model is trained on data till December 2023 while GPT-4 is trained on data till September 2021 [20]. Our primary motivation for including both GPT-4 and GPT-4-Turbo is to check whether training on more data has any significant effect on LLM hallucinations.[1]

[1] Note that in practicality, the choice between GPT-4 and GPT-4-Turbo may depend on their costs, the supported context window length, etc. – these differences, albeit important, do not play any role in this study.

- Gemini-1.0-Pro: We could not find any documentation that explicitly mentions the parameter size of any of the Google's LLMs mentioned in this paper. This model [21], at the time of its creation, was the most capable and most generic generative AI model that Google had to offer.
- Gemini-1.5-Pro: This model [18] improves upon its predecessor by introducing a new mixture-of-experts architecture with support for context window up to a million tokens.
- Gemini-1.5-Flash: This model is a smaller version of Gemini-1.5-Pro that performs slightly worse but still better than Gemini-1.0-Pro [18] on NLP tasks.
- Llama3–8B: This an open-source model from Meta having 8 billion parameters. This improves upon its predecessor [22] by reducing false refusal rates and increasing diversity in model responses among others.
- Gemma2–9B: This is an open-source model from Google that is developed using the same research and technology as that of Germini. It has 9 billion parameters and has been built *responsibly* to ensure safe usage. A primary motivation for choosing this model is to compare the performances of a proprietary and an open-source LLM which have been developed using the same methodology.

Next we need a set of queries (called *original queries*) which will be used to check whether any of these LLMs are prone to hallucination with respect to the concepts that these queries pertain to. In case of real-world LLM monitoring, these queries can be sampled from the prompts which are actually fed to the LLMs by its users. We perturb these queries to get an extended list of queries. For perturbation, we use the approach mentioned in [16] as described below briefly – note that each level of perturbation has three different strategies to inject perturbation.

- Character-level perturbation: (i) introduce typos to at most two words, (ii) change at most two letters in the sentence, (iii) add at most two extraneous characters to the end of the sentence.
- Word-level perturbation: (i) replace at most two words with their synonyms, (ii) delete at most two words that do not alter the sentence's meaning, (iii) add at most two semantically neutral words to the sentence.
- Sentence-level perturbation: (i) add a randomly generated irrelevant handle at the end of the sentence, e.g., @abak, (ii) paraphrase the sentence, (iii) change the syntactic structure of the sentence.

It is important to note that replacing words by their synonyms, paraphrasing the sentence or changing the syntactic structure (e.g., active to passive voice) do not alter the semantic meaning of the original query, and thus the perturbation approach of [16] subsumes the traditional approach that checks hallucination by modifying the original queries while preserving their semantic meanings.

As the next step, we fire each LLM with both the original queries and their perturbed versions. Finally, we check the pair-wise consistency between the responses generated for the original query and each of its perturbed variants. One may be interested in knowing whether an LLM M_k has understood the concept that query Q_l is related to, in such a case she may look into the average consistency score computed in step 25 of Algorithm 1. If the user is interested in knowing the overall average consistency score for an LLM for all the original queries, then they may refer to the score computed in step 27 of Algorithm 1. *Note that consistency and hallucination are inversely related, i.e., higher consistency implies lesser hallucination and vice versa.* It may be further noteworthy that our chosen model for computing consistency [14] produces a real-valued score between 0 and 1 whereas, other models [11] may produce a binary score of either 0 or 1 to indicate whether there is hallucination or no hallucination – the method described in Algorithm 1 is generic enough to handle such cases as well.

4 Experimental results

4.1 Dataset description

We have used the LongBench multitask dataset [23] for our experiments. LongBench is the first and widely adopted benchmark dataset for evaluating the long-context understanding capabilities of LLMs. This dataset includes various datasets for English and Chinese language related tasks. For our experiments, we selected datasets exclusively from the English language related tasks. Among the chosen datasets, four datasets (TriviaQA, WikiQA, HotpotQA, and Qasper) correspond to regular question-answer tasks, one (SAMSum) corresponds to a summarization task, and one (TREC) involves text classification.

4.2 Experiment 1

Average consistency scores of LLMs for character, word and sentence-level perturbations

We have measured the average consistency score of the six LLMs explored in this paper against perturbations at the character, word, and sentence-levels aggregated across different datasets. From Fig. 1, we can see that, as expected, the performance of the models degrade as one moves from the character-level to the sentence-level perturbations except for GPT-4-Turbo and Gemini–1.5-Flash which see a small bump in performance for word-level perturbation with respect to character-level. Moreover, we observe that GPT-4 has the highest average consistency score compared to other LLMs, and surprisingly, GPT-4-Turbo has the least average consistency score. In case of Google's LLMs, both the 1.5 models outperform that of 1.0 as expected; however, astonishingly, Gemini–1.5-Flash does better than the bigger Gemini–1.5-Pro model. The two open-source models achieve much lower average consistency scores compared to the proprietary ones with Gemma2–9B doing slightly better than Llama3–8B.

4.3 Experiment 2

GPT-4's consistency score at sentence-level perturbation vs other models' consistency scores at character-level perturbation

We now zoom into the average consistency score of GPT-4 for sentence-level perturbation and the scores for the rest of the models for character-level perturbation as shown in Fig. 2. Our scrutiny reveals that GPT-4's performance is superior in spite of this skewed comparison; in other words, GPT-4 faced with sentence-level perturbation fares better than other models even when the latter faced only minor typos. This finding reaffirms that GPT-4 is the most consistent model and is least prone to generating hallucinations.

4.4 Experiment 3

SelfCheckGPT vs our perturbation-based framework: Comparing average consistency scores when different LLMs are asked only semantically similar questions vs when asked questions with all types of perturbations

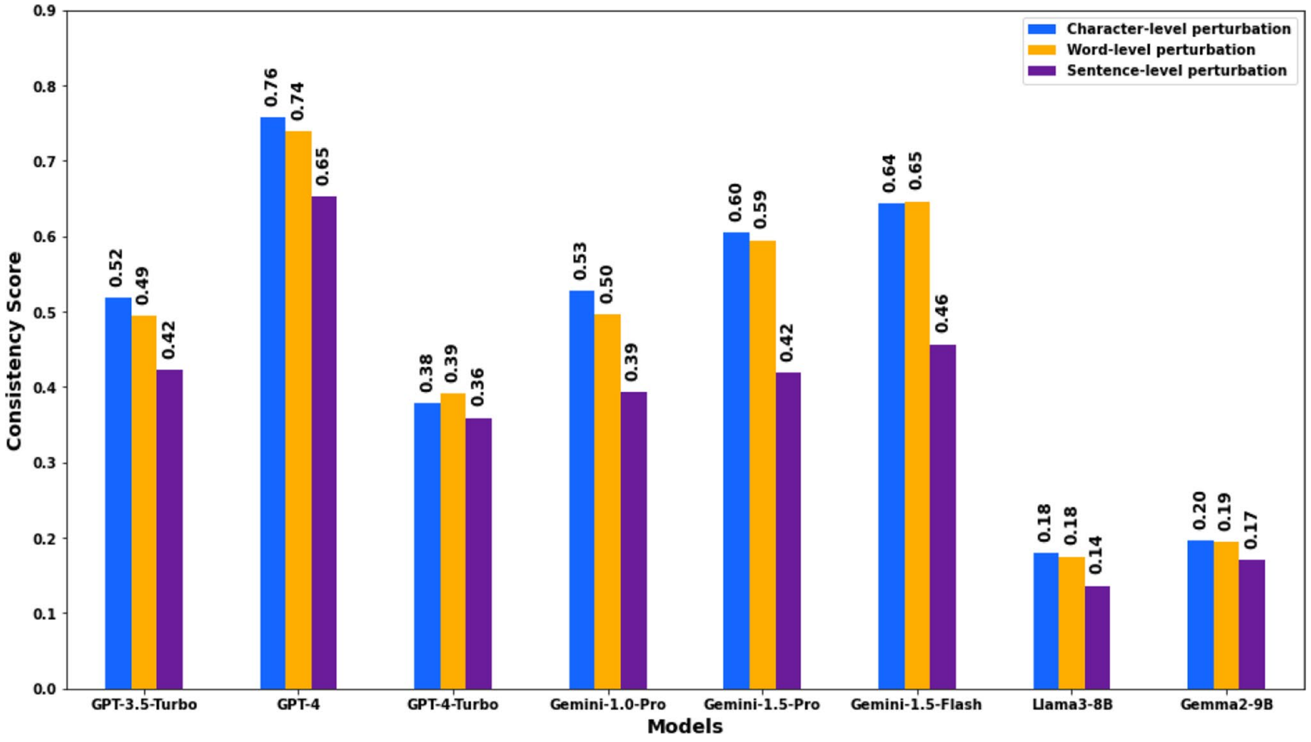

Fig. 1 Average consistency scores (after aggregating across datasets) of LLMs for character, word and sentence-level perturbations

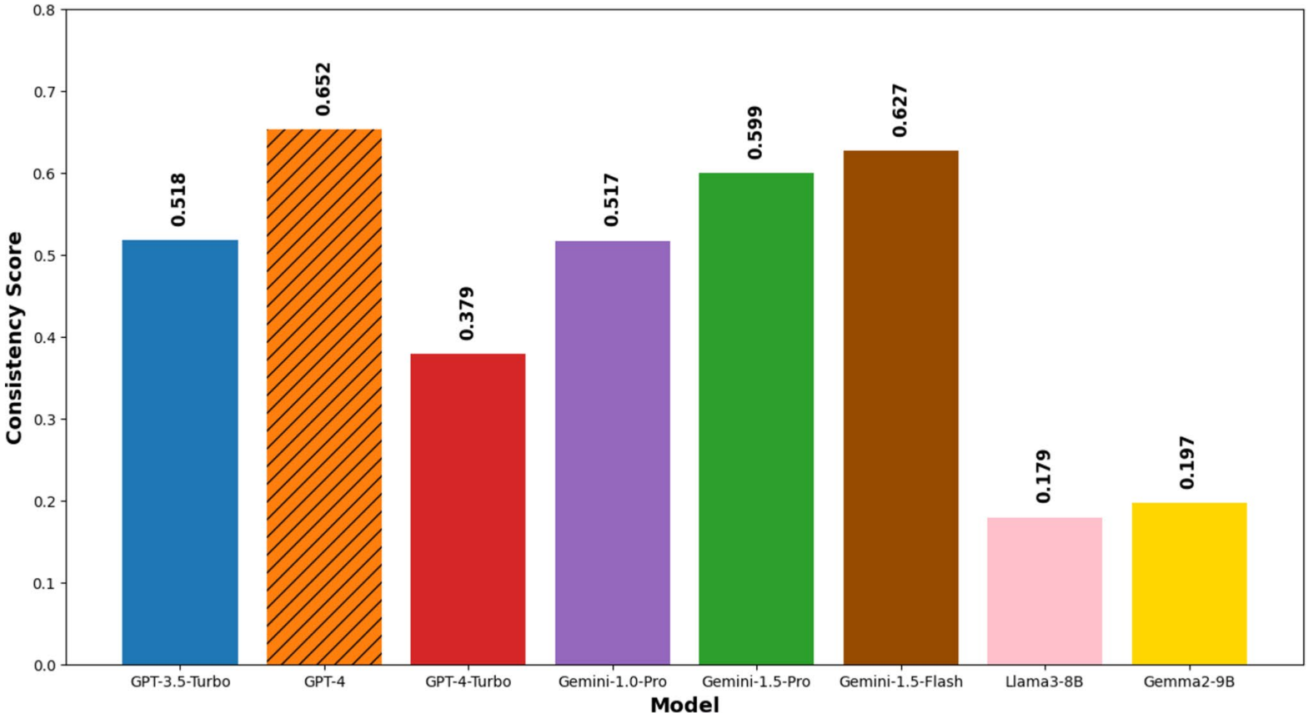

Fig. 2 GPT-4's average consistency score at sentence-level perturbation vs other models' average consistency score at character-level perturbation

In this experiment, we did a comparative analysis between two comparison frameworks. We compared the results of SelfCheckGPT framework, where the model was asked only semantically similar questions, against our proposed framework where the model was asked questions with all three types of perturbations. As demonstrated in Fig. 3, the relative ordering of the LLMs are different for the two frameworks. GPT-4 holds the top-most rank in both whereas, Gemini–1.0-Pro and GPT-4-Turbo, among the proprietary LLMs, hold the bottom-most two ranks in both. However, with

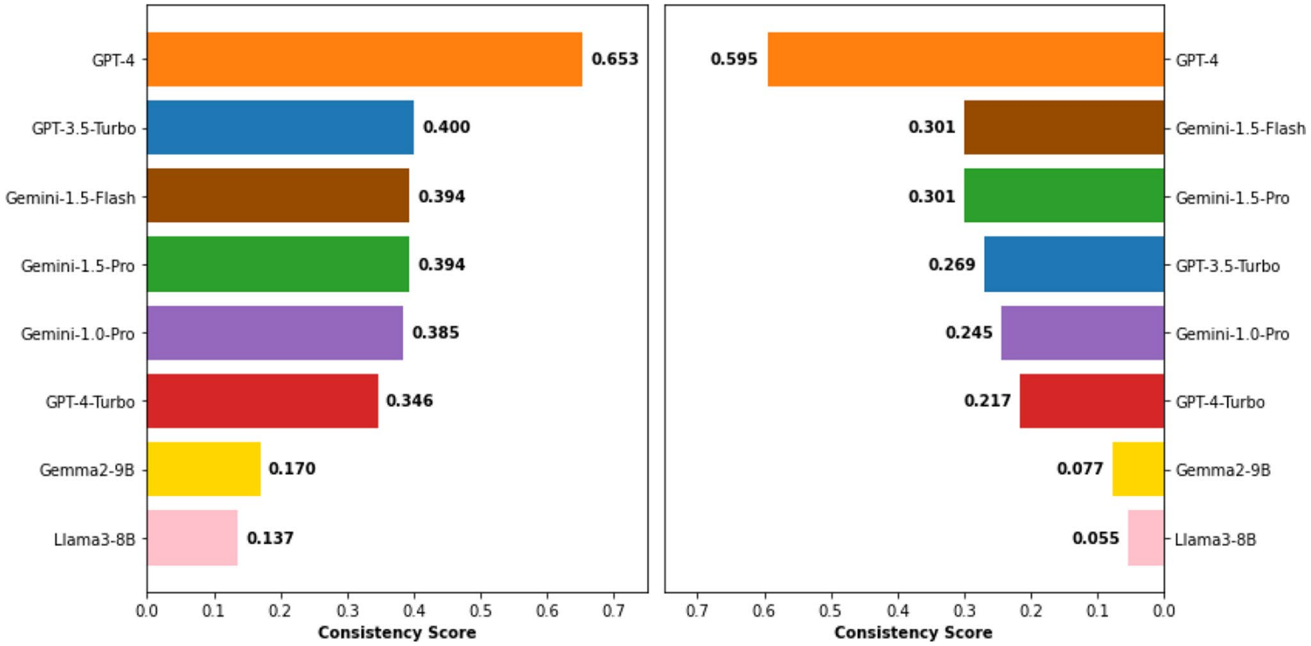

Fig. 3 LLMs ranked by average consistency scores (across all datasets) for (left) semantically similar questions, (right) questions with character, word and sentence-level perturbations

SelfCheckGPT, GPT–3.5-Turbo was ranked second but it slips to the fourth place in our proposed perturbation-based framework. As previously mentioned in Sect. 3, the perturbation-based framework subsumes the framework that only asks semantically similar questions, and therefore we conclude that Gemini–1.5 models should be preferred over GPT–3.5-Turbo if one is aiming for a model that is less susceptible to hallucination – note that our recommendation lies in contrast to what SelfCheckGPT would have recommended. The two open-source models rank the lowest in both the lists with Gemma2–9B outperforming Llama3–8B in both the scenarios. Further note that although Gemini and Gemma2 are developed using the same methodology, there is a substantial gap in their performances.

5 Conclusion

Hallucinations in LLMs may hinder their adoption for critical applications. Consequently, it may be important for a client to choose an LLM that is least likely to hallucinate in production. Typically, following the concept of SelfCheckGPT [3], one asks semantically similar queries (to the original query) to evaluate an LLM's hallucination susceptability. However, there is a line of work [4, 5, 16] that advocates to introduce perturbations to the LLM's inputs (at character, word and sentence-level), and then check for hallucination — if the LLM continues to generate consistent answers compared to the original query, then this should prove the LLM's robustness to hallucination even further. Additionally, a more robust LLM should ideally be less affected by adversarial attacks [16]. In this work, we present a framework that applies perturbations to LLM's queries, generates their corresponding responses and then checks for hallucination; our code is modularized to accommodate different perturbation strategies, LLM models and hallucination detection models/metrics. We evaluate this framework using the perturbations mentioned in PromptAttack [16] and Vectara's hallucination detection model [14] on eight prominent foundation models. Our evaluation shows that GPT-4 is most robust to hallucinations.

Acknowledgements This is an independent research done by the authors and not endorsed by Walmart Global Tech in any manner.

Author contributions A.S. was the primary contributor for the code; he also prepared all the figures. B.G. also contributed to the code and helped identify the illustrative example. A.C. first proposed the idea and pointed out relevant literature, which form the foundation of the current work. K.B. supervised the entire work; he wrote the paper and helped devise the experiments.

Funding Not applicable

Data availability The datasets generated and/or analyzed during the current study are available in the GitHub repository – https://github.com/anirbans403/robustness_framework

Declarations

Ethics approval and consent to participate Not applicable

Consent for publication Not applicable

Competing interests Not applicable

References

1. Huang L, et al. A survey on hallucination in large language models: principles, taxonomy, challenges, and open questions. CoRR arXiv:abs/2311.05232 2023.
2. Xu Z, Jain S, Kankanhalli MS. Hallucination is inevitable: an innate limitation of large language models. CoRR arXiv:abs/2401.11817 2024.
3. Manakul P, Liusie A, Gales MJF. Selfcheckgpt: Zero-resource black-box hallucination detection for generative large language models 2023.
4. Moradi M, Samwald M. Evaluating the robustness of neural language models to input perturbations 2021.

5. Chavan T, et al. SenTest: evaluating robustness of sentence encoders. CoRR arXiv:abs/2311.17722 2023.
6. Lewis PSH, et al. Retrieval-augmented generation for knowledge-intensive NLP tasks 2020.
7. Li J, Yuan Y, Zhang Z. Enhancing LLM factual accuracy with RAG to counter hallucinations: A case study on domain-specific queries in private knowledge-bases. CoRR arXiv:abs/2403.10446 2024.
8. Friel R, Sanyal A. Chainpoll: A high efficacy method for LLM hallucination detection. CoRR arXiv:abs/2310.18344 2023.
9. Es S, James J, Espinosa Anke L. Schockaert, S. RAGAs: automated evaluation of retrieval augmented generation 2024.
10. He P, Gao J, Chen W. DeBERTaV3: improving DeBERTa using ELECTRA-style pre-training with gradient-disentangled embedding sharing 2023.
11. Chowdary V. DeBERTa-v3-base fine-tuned for hallucination detection. https://huggingface.co/Varun-Chowdary/hallucination_detect 2024. Online; accessed 25-Jul-2024.
12. Sanh V, Debut L, Chaumond J. Wolf, T. DistilBERT, a distilled version of BERT: smaller, faster, cheaper and lighter. CoRR arXiv:abs/1910.01108 2019.
13. cross encoder. Cross-encoder for natural language inference. https://huggingface.co/cross-encoder/nli-distilroberta-base 2021. ; Accessed 25 Jul 2024.
14. Vectara. Hallucination evaluation model. https://huggingface.co/vectara/hallucination_evaluation_model 2024. Accessed 25 Jul 2024.
15. Vectara. The vectara platform. https://docs.vectara.com/docs/ 2024. Accessed 25 Jul 2024.
16. Xu X, et al. An LLM can fool itself: a prompt-based adversarial attack. CoRR arXiv:abs/2310.13345 2023.
17. Wang B, et al. Adversarial GLUE: a multi-task benchmark for robustness evaluation of language models 2021.
18. Reid M, et al. Gemini 1.5: unlocking multimodal understanding across millions of tokens of context. CoRR arXiv:abs/2403.05530 2024.
19. Heaven WD. Gpt-4 is bigger and better than chatgpt-but openai won't say why. https://www.technologyreview.com/2023/03/14/1069823/gpt-4-is-bigger-and-better-chatgpt-openai/ 2023. Online; accessed 25-Jul-2024.
20. OpenAI Platform. Models. https://platform.openai.com/docs/models/gpt-4-turbo-and-gpt-4 2024. Accessed 30 Jul 2024.
21. Anil R, et al. Gemini: a family of highly capable multimodal models. CoRR arXiv:abs/2312.11805 2023.
22. Touvron H, et al. Llama 2: open foundation and fine-tuned chat models. CoRR arXiv:abs/2307.09288 2023.
23. Bai Y, et al. Longbench: a bilingual, multitask benchmark for long context understanding. CoRR arXiv:abs/2308.14508 2023.