

Hallucination Detection with Small Language Models

Ming Cheung

dBeta Labs, The Lane Crawford Joyce Group

Hong Kong, China

mingcheung@lcjgroup.com

Abstract—Since the introduction of ChatGPT, large language models (LLMs) have demonstrated significant utility in various tasks, such as answering questions through retrieval-augmented generation. Context can be retrieved using a vectorized database, serving as a foundation for LLMs to generate responses. However, hallucinations in responses can undermine the reliability of LLMs in practical applications, and they are not easily detectable in the absence of ground truth, particularly in question-and-answer scenarios. This paper proposes a framework that integrates multiple small language models to verify responses generated by LLMs using the retrieved context from a vectorized database. By breaking down the responses into individual sentences and utilizing the probability of generating "Yes" tokens from the outputs of multiple models for a given set of questions, responses, and relevant context, hallucinations can be detected. The proposed framework is validated through experiments with real datasets comprising over 100 sets of questions, answers, and contexts, including responses with fully and partially correct sentences. The results demonstrate a 10% improvement in F_1 scores for detecting correct responses compared to hallucinations, indicating that multiple small language models can be effectively employed for answer verification, providing a scalable and efficient solution for both academic and practical applications.

Index Terms—Large Language Models, Hallucination.

I. INTRODUCTION

The development of large language models (LLMs) has revolutionized various fields, offering remarkable capabilities in natural language understanding and generation. However, despite their impressive performance, LLMs are prone to generating inaccurate information, commonly referred to as "hallucinations." Table I presents examples of common types of hallucinations found in responses to prompts: Logical, Prompt, and Factual contradictions. The Logical contradiction illustrates a scenario where the city of Madison is misrepresented regarding its population, claiming that a town with a population of 500K is not a small town. The Prompt contradiction showcases a misleading description of a healthy breakfast that contradicts common dietary guidelines. Lastly, the Factual contradiction highlights an inaccurate ingredient in a traditional Margherita pizza, which incorrectly lists chocolate as a key component. These inaccuracies can undermine the reliability of LLMs, particularly in high-stakes applications where precision is essential.

Hallucinations in LLMs arise from various factors, including source-reference divergence in the training data, which may result from heuristic data collection methods or the

TABLE I
CONTRADICTION TYPES AND THEIR EXAMPLES

Hallucination Types	Prompt	Generated Response
Logical	Can you introduce Madison?	The city of Madison has over 500K residents. It is known for its small-town charm and quiet atmosphere.
Prompt	Describe a healthy breakfast that includes fruits and whole grains.	A bowl of sugary cereal with milk and a side of bacon is a great choice for breakfast.
Factual	What are the main ingredients in a traditional Margherita pizza?	A traditional Margherita pizza is made with tomatoes, mozzarella cheese, fresh basil, and a secret ingredient: chocolate.

inherent complexities of certain natural language generation tasks [1].

Detecting hallucinations in LLM-generated responses is both critical and challenging. Traditional LLM performance measurements, such as ROUGE [2], are not suitable for real-time evaluation, as the necessary reference answers are not readily available for detection. An alternative approach leverages the fact that language models are trained to answer yes or no questions [3]. This method prompts an LLM to answer "yes" or "no" in response to a given set of questions and answers to detect hallucinations in generated responses. It utilizes the knowledge of the LLM, though this knowledge is not always accessible. The capabilities of these language models can be enhanced with retrieval-augmented generation (RAG) [4] using vectorized database [5]. Even if the information is not in the language model, it can utilize RAG context to verify responses from other LLMs. One can call an LLM multiple times, similar to an API, to obtain probability estimates, but this requires more time. Deploying the model locally enables the computation of "yes" and "no" probabilities without multiple calls, allowing for direct extraction of probabilities. However, due to the models being closed-source, such as ChatGPT, or their large size requiring significant computational resources to host, this is not always feasible. In contrast, smaller language models (SLMs), defined as having 100M to 5B parameters [6], present a promising alternative for

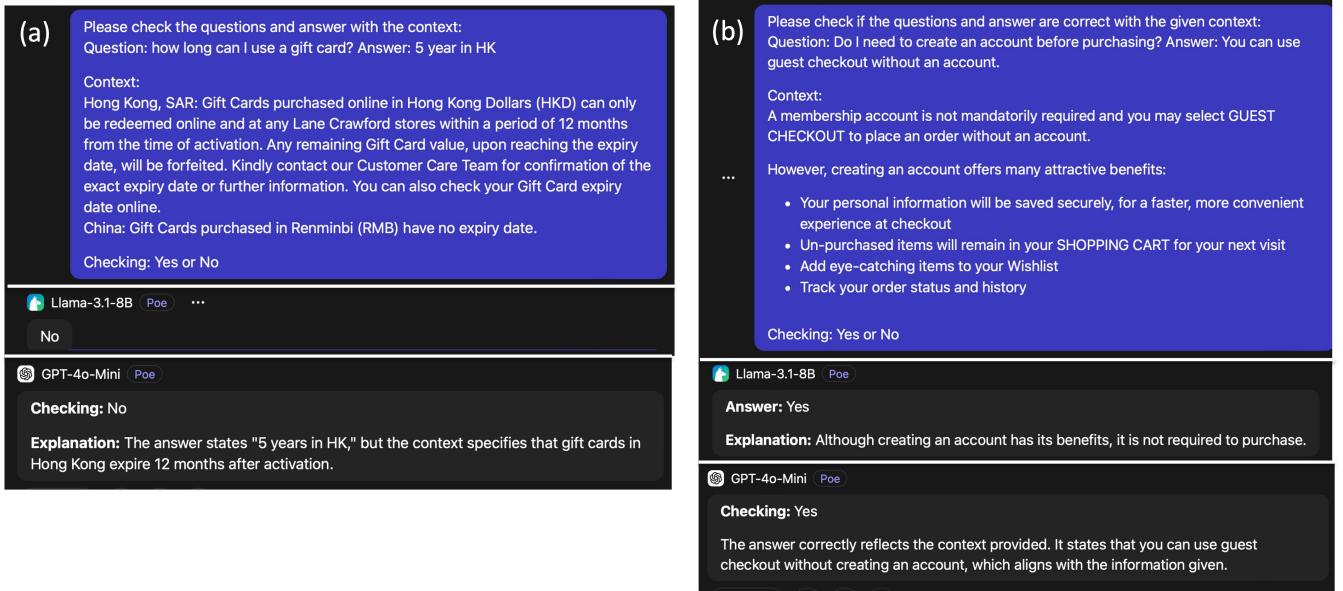


Fig. 1. Examples of multiple choices for checking responses.

answer verification. While these models may not match the performance of their larger counterparts in generating free-text responses, they can be more efficient and cost-effective for specific tasks. Figure 1 illustrates examples of checking with questions, answers, and contexts. It has been observed that even a relatively small model can still produce accurate results.

This paper proposes a framework for detecting hallucinations by exploring the potential of integrating SLMs for contextual answer checking, aiming to improve verification accuracy while mitigating the resource demands associated with larger models. The main contributions of this paper are:

- A framework for computing hallucination scores using multiple SLMs;
- An algorithm for the framework to compute scores from segmented responses using different SLMs;
- A dataset prepared to verify the proposed framework, demonstrating an improvement of 10% over the baseline.

The paper is organized as follows: Section II discusses related works, and Section III introduces how questions can be handled with retrieval-augmented generation in LLMs. Section IV discusses the proposed framework, while Section V covers the experimental settings and results. Finally, Section VI concludes the paper.

II. RELATED WORKS

The emergence of large language models (LLMs) has significantly transformed multiple domains, providing extraordinary capabilities in natural language understanding and generation. LLMs demonstrate substantial proficiency across a variety of NLP tasks, including language translation [7], sentiment analysis, named entity recognition [8], [9], and SQL generation, among others [10]. Historical models, such as recurrent neural networks (RNNs) [11] and long short-term memory

(LSTM) models [12], laid the foundational understanding of sequential data, particularly in text processing. However, the advancement of the transformer architecture [13] marked a pivotal turning point, enabling models to effectively capture long-range dependencies within textual data. This architecture employs self-attention mechanisms [14], allowing the model to concentrate on disparate segments of the input text, thereby generating coherent and contextually appropriate responses. Consequently, numerous LLMs have been developed, including GPT-3 [15], Google Bard [16], and NeevaAI [17]. These models, refined through human feedback [18], encompass billions of parameters and are capable of engaging in conversational interactions [19].

LLMs have drawn attention for their capabilities; however, they are known to produce hallucinations in their outputs, and there is no clear definition of hallucination [20]. Detecting hallucinations is not analogous to conventional LLM measurements, such as the ROUGE metric [21] and BLEU score [22], which measure response quality by comparing the output with a ground truth. The difficulty in detecting hallucinations arises from the absence of ground truth during response generation. Another approach involves building a classifier to evaluate the response [23], [24]. The hidden states in the models are also useful for constructing classifiers [25]. Reinforcement learning can be applied to detect hallucinations in logic [26] and to correct them [27].

Another solution is to prompt LLMs to answer yes or no questions [3] to detect hallucinations in a response. The research in [28] employs entropy and sentence splitting to generate multiple outputs. Additionally, the knowledge required for verification may not exist within the LLM, necessitating retrieval-augmented generation (RAG) [4] to enhance its capacity by providing knowledge that does not exist in the LLM. Related context can be extracted from vectorised

database [5]. Based on the context, score-based hallucination tests utilize distribution tests approximated solely by the answer [29]. However, this approach necessitates multiple rounds of generation and precise probability computations, making it challenging to apply to large LLMs that are either not open (e.g., ChatGPT) or excessively large. The work in [3] investigates the probability of correctness, $P(\text{True})$, by estimating the likelihood of a "yes" response, leveraging the strengths of LLMs in predicting answers for multiple-choice questions [30].

This paper proposes a framework to utilize multiple small language models (SLMs) for detecting hallucinations with $P(\text{True})$.

III. QUESTIONS AND ANSWERING USING RAG

This section discusses how responses can be generated using RAG, as shown in Fig. 2 (a). The first part discusses some common Large Language Models, followed by RAG.

A. Large Language Model (LLM)

The Large Language Model (LLM) serves as the primary engine for processing user queries. The LLM leverages vast datasets to understand context and provide accurate, human-like answers, making it a crucial component of the interaction. The prompt may include the role, the questions, as well as the required context to answer the questions. Even with a prompt containing suitable context, there is no guarantee that the responses will be free of hallucinations.

Some common LLMs include ChatGPT 3.5¹, which is a conversational AI model developed by OpenAI, utilizing the GPT (Generative Pre-trained Transformer) architecture [31]. It generates human-like responses for various tasks, trained on 570 gigabytes of text from sources like Wikipedia and Twitter, with 175 billion parameters. Another popular LLM is Meta's Llama-2-70b², which employs the GPT-3.5 architecture and is designed for high-quality response generation across applications like content creation and natural language understanding. It features 70 billion parameters and is trained on 2 trillion tokens from publicly available sources [32]. APIs are available for different applications, allowing users easier access to these models for generating responses to their questions.

B. Retrieval Augmented Generation

There are many different LLMs available, and due to variations in model structures, training data, and decoding strategies, they can generate different responses for the same prompt and question. Additionally, users may want to answer questions based on specific documents, such as inquiries from company employees where the documents are staff handbooks. Although one can fine-tune a model to respond accordingly [33], this requires substantial resources and expertise to train and host the model.

One solution is to provide related context for a question, known as Retrieval Augmented Generation [34]. As the models are trained to follow human instructions [18], LLMs can generate responses based solely on the provided context. The goal of the next section is to discuss how the response can be verified for hallucinations.

IV. PROPOSED FLOW

The flow diagram, as shown in Fig. 2, outlines the proposed framework, from the LLM generating responses to returning a hallucination score for that response. This response is first processed by a splitter, which divides the information into two separate streams: one confirming the accuracy of the details and the other assessing the context. The system subsequently checks if all conditions are met, determining whether the working hours align with the specified criteria. Hence, the hallucination score, s_i^m , can be computed as:

$$s_i^{(m)} = P(h_i = 1 | q_i, r_i, c_i, \text{prompt}) \quad (1)$$

where s_i^m is the score for response i , while m is the m -th SLM. The value h_i indicates hallucination, and prompt is the prompt for the check. The prompt asks the SLM to generate a "yes" or "no" answer, as shown in Fig. 1. The parameters q_i , r_i , and c_i represent the question, response, and context, respectively.

By considering the first generated token, Eq. 1 becomes:

$$s_i^{(m)} = P(\text{token}_1 = \text{yes} | q_i, r_i, c_i) \quad (2)$$

where token_1 is the first token generated by the SLM, and s_i^m is the probability that the first token is yes. Note that prompt is omitted as all SLMs use the same prompt. This section introduces each part of the flow one by one.

A. Splitter

The Splitter acts as an intermediary that takes the LLM's response and divides it into sentences for further processing. It separates critical details, such as the working hours and contextual information shown in Fig. 2, to ensure that each piece of information can be individually evaluated. For a response, r_i , it will be split into several sub-responses, denoted as $r_{i,j}$. Without this step, evaluating the whole sentence with both correct and incorrect information would confuse the checker. One method to split a response is using SpaCy³, a powerful NLP library in Python, which effectively segments text into sentences by identifying the relationships between words and punctuation marks. The response, r_i , can be computed by splitting r_i into $r_{i,j}$.

B. Small Language Models (SLMs)

The Small Language Model (SLM) is responsible for evaluating the segmented response. It checks for accuracy, such as the working hours and operational days, against the context. The question, context, and answer are given in the prompt, and

¹<https://openai.com/>

²<https://huggingface.co/meta-llama/Llama-2-70b-chat-hf>

³spacy.io

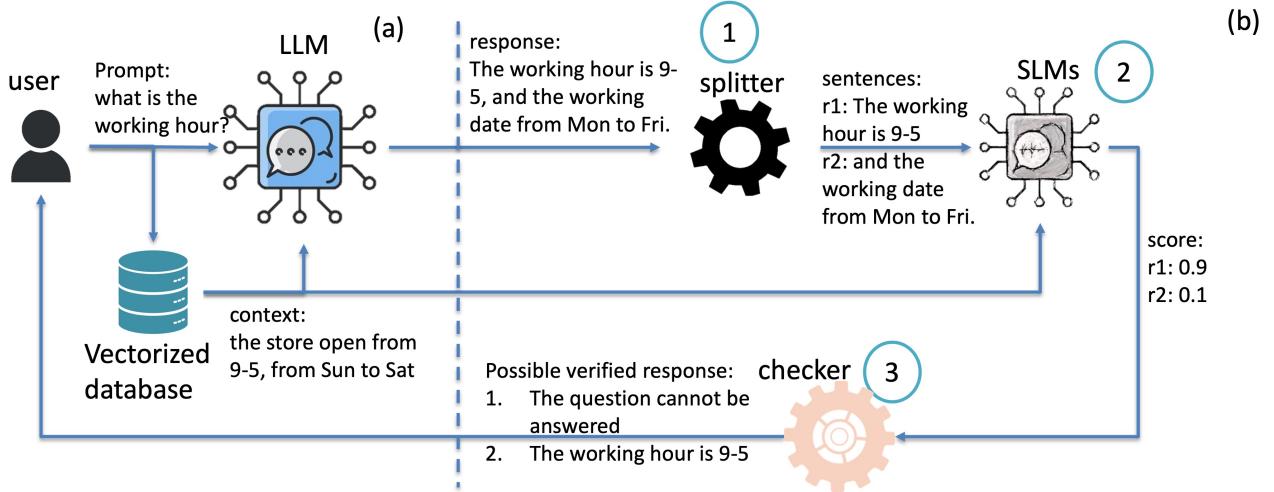


Fig. 2. Flow of using the proposed framework: (a) LLMs generate responses; (b) the proposed framework.

the SLM is asked to generate a response starting with YES or NO [3]. The score for a split sentence, $s_{i,j}^{(m)}$, is computed as:

$$s_{i,j}^{(m)} = P(\text{token}_1 = \text{yes} | q_i, c_i, r_{i,j}) \quad \text{where} \quad r_{i,j} \subset S(r_i) \quad (3)$$

The checking is conducted on each split sentence. There may be more than one SLM in the validation, and the score from each SLM is sent to the checker. SLMs, such as Qwen2 [35] and MiniCPM [36], demonstrate good performance [6] across many tasks, including math and problem-solving, making them suitable candidates for the framework.

C. Checker

The checker is the final decision-making component that combines the scores from the split sentences, processed by multiple SLMs, into a single score. By confirming or rejecting the details based on the scores, the answers can be evaluated for accuracy, enhancing the reliability of the overall system. Different SLMs have different scales, meaning they possess varying means and variances for the same set of data. Consequently, the values of the responses from different SLMs are normalized as:

$$\tilde{s}_{i,j}^{(m)} = \frac{s_{i,j}^{(m)} - \mu_m}{\sigma_m} \quad (4)$$

where μ_j and σ_j are the means and variances of the SLM j . These can be computed based on previous responses.

The score, $s_{i,j}$, for a segmented response j , based on multiple SLMs, can be calculated as:

$$s_{i,j} = \frac{1}{M} \sum_{j=1}^M \tilde{s}_{i,j}^{(m)} \quad (5)$$

where M is the total number of SLMs. The final score, s_i , of response i , can be computed using the harmonic mean:

$$s_i = \frac{|S(r_i)|}{\sum_{j=1}^{|S(r_i)|} \frac{1}{s_{i,j}}}, \quad s_{i,j} > 0 \quad (6)$$

To avoid issues with non-positive values, any values less than or equal to zero are adjusted. More discussions regarding the choice of means can be found in later sections.

V. EXPERIMENTS AND MEASUREMENT

This study conducts experiments and measurements on a real dataset sourced from Lane Crawford. The dataset comprises questions related to Human Resources policies, with corresponding contexts extracted from the employee handbook.

A. Dataset

The dataset is generated from the handbook of Lane Crawford. The questions are formulated based on the context of various topics from the handbook, ranging from Employment (such as probation, salary, leave, and benefits) to Policy (such as uniform and emails), as well as other matters (such as handling media requests and bringing personal devices to work). The following is an example of context and the question regarding working hours, derived from the context:

- Context: The store operates from 9 AM to 5 PM, from Sunday to Saturday. There should be at least three shopkeepers to run a shop.
- Question: What are the working hours?

Based on the context and questions, three responses are generated for each pair of context and question. Note that the context may contain more information than is necessary to formulate the question. The responses belong to three different categories: one labeled as correct, one as partially correct, and one as incorrect. The labeled responses are as follows:

- Correct: The working hours are 9 AM to 5 PM, and the store is open from Sunday to Saturday.
- Partial: The working hours are 9 AM to 5 PM, and the store is open from Monday to Friday.
- Wrong: The working hours are 9 AM to 9 PM, and you do not need to work on weekends.

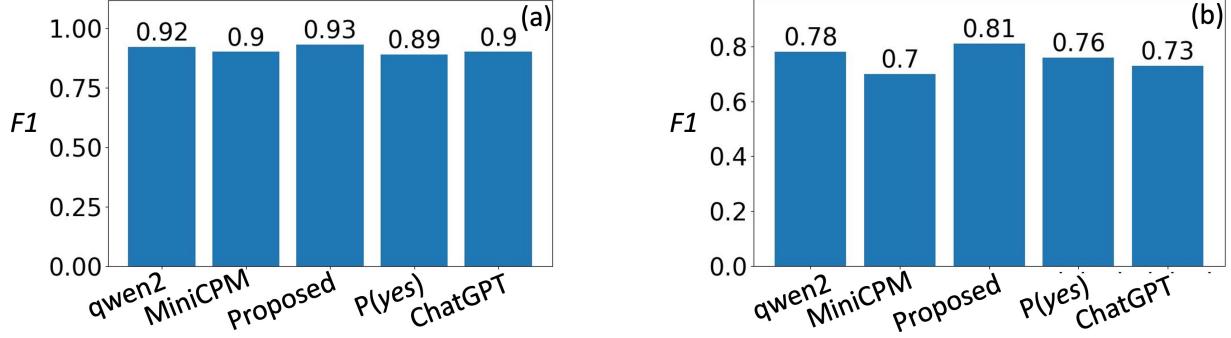


Fig. 3. Experiment results on the best F_1 for different approaches in detecting correct responses from: (a) wrong, (b) partial.

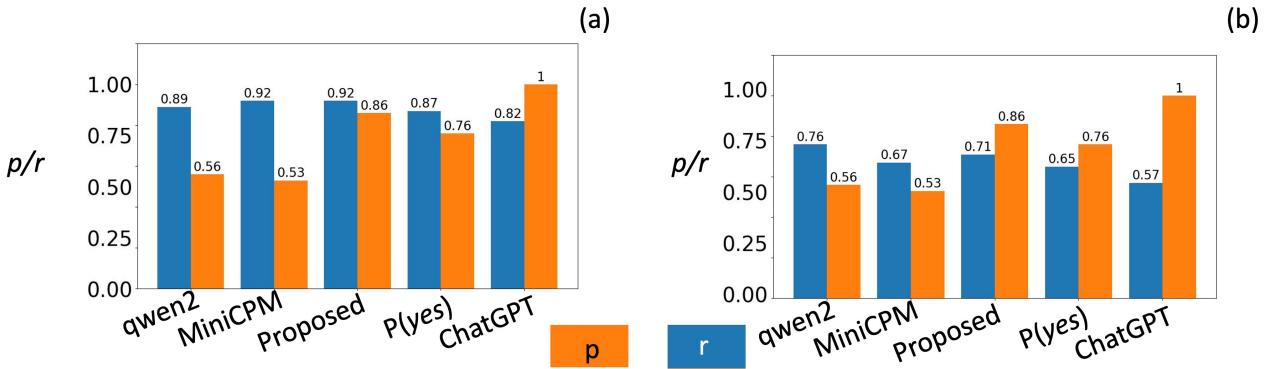


Fig. 4. Experiment results on the best p and r for detecting correct responses from: (a) wrong, (b) partial.

The incorrect parts are italicized. The reason the response is labeled as partial is that while the working hours are accurate, the days are not. Each question in the dataset is accompanied by a context that contains the essential information needed to formulate an answer and includes three responses labeled as “correct,” “partial,” and “incorrect.” It is important to note that the labels are not applied at the sentence level, particularly in the case of partial responses. The primary aim of these experiments is to assess whether the SLM can effectively identify incorrect or partial responses in comparison to the correct ones. Using the same contexts and questions to form multiple answers ensures that the models are not biased toward certain contexts.

B. Small Language Models

Two models are used as the SLM in the proposed framework: Qwen2 [35] and MiniCPM [36]. This section introduces them one by one.

1) *Qwen2*: Qwen2 [35] is an advanced Language Model (LLM) designed to enhance natural language understanding and generation⁴. Building on the capabilities of its predecessors, Qwen2 features improved contextual awareness, enabling it to generate more coherent and contextually relevant

responses. It utilizes a vast dataset for training, allowing it to perform a variety of tasks, from conversational agents to content creation⁵.

2) *MiniCPM*: MiniCPM [36] is a series of edge-optimized large language models designed for efficient deployment on devices with limited computational resources⁶. The flagship model, MiniCPM-2B, features 2.4 billion parameters and excels in tasks such as language understanding and coding, often outperforming larger models like Llama2-13B and MPT-30B. MiniCPM also supports extensive customization, allowing users to fine-tune outputs for various NLP tasks, thus broadening its applicability in real-time and on-device scenarios.

C. Approaches and Baselines

This section introduces the baselines and the approaches for the frameworks.

- Proposed: Using Qwen2 and MiniCPM as the SLMs in the proposed framework;
- ChatGPT: Prompt ChatGPT to answer “Yes” or “No” ($P(\text{True})$) [3];

⁵available: huggingface.co/Qwen/Qwen2-1.5B-Instruct

⁶<https://huggingface.co/openbmb/MiniCPM-2B-sft-bf16>

⁴<https://huggingface.co/Qwen/Qwen2.5-1.5B-Instruct>

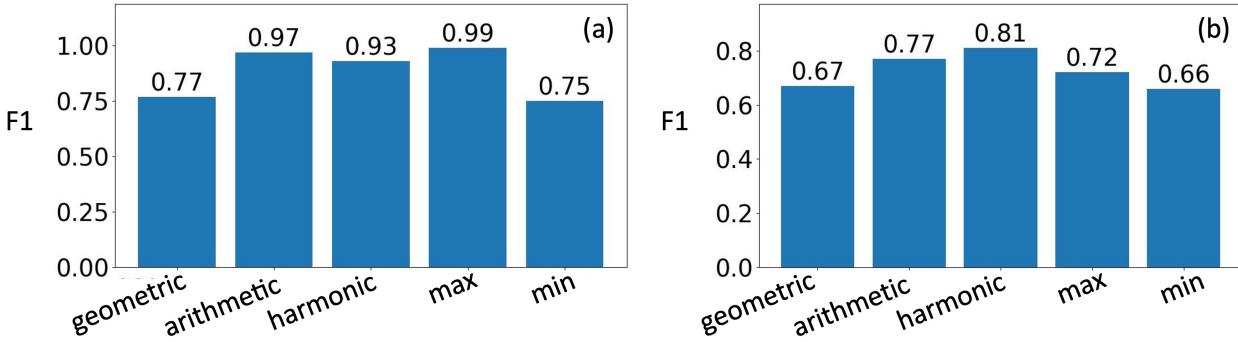


Fig. 5. Experiment results on different means algorithms in detecting correct responses from: (a) wrong, (b) partial.

- $P(\text{yes})$: Prompt SLM for the whole response for the probability to be "YES" using Qwen2;
- Qwen2: Similar to the proposed, but using only Qwen2 as the SLM.
- MiniCPM: Similar to the proposed, but using only MiniCPM as the SLM.

The approach using ChatGPT is for those LLMs that are only available through API, while $P(\text{yes})$ is the approach without a splitter. Approaches using Qwen2 and MiniCPM demonstrate the effectiveness of utilizing multiple SLMs.

D. Results

An experiment is conducted to compare the $F1$ scores of different approaches in detecting "correct" responses from "partial" or "wrong" ones. If the score in Eq. 6 exceeds a threshold, the response is labeled as "correct"; otherwise, it is not. The results are shown in Fig. 3. Fig. 3 (a) shows the results of detecting "correct" responses from "wrong" responses, while Fig. 3 (b) illustrates detecting "correct" responses from "partial" responses, with the thresholds yielding the highest $F1$ scores selected. In Fig. 3 (a), all approaches achieve high $F1$ scores, with $P(\text{yes})$ being the lowest at 0.89. The results indicate that all approaches can detect responses that are completely contradictory to the contexts. Fig. 3 (b) shows lower $F1$ scores, with the proposed method achieving the highest score of 0.81. These results suggest a general decrease in performance from detecting "wrong" to detecting "partial" responses, as the latter is much more challenging. Utilizing SLMs is better than using ChatGPT and $P(\text{yes})$ by 11% and 6.6%, respectively. Additionally, employing multiple SLMs outperforms the use of a single model.

A second test is conducted to measure the best precision p and the corresponding recall r of different approaches. As a question-and-answering system, it is desirable to have a high p and reasonable r , meaning a system that answers only those questions it is confident about, without providing incorrect information. Fig. 4 shows the p and r for different approaches, detecting correct responses from wrong versus detecting correct responses from partial, in Fig. 4 (a) and Fig. 4 (b), respectively. Note that r must be at least 0.5 while

selecting the p , to prevent selecting a very high p with a very low r . In Fig. 4 (a), the p score for "Qwen2" is 0.89, with MiniCPM at 0.92. However, they have relatively low r values of 0.56 and 0.53, respectively. Conversely, the proposed method has a comparable p but a much higher r , implying that using multiple SLMs could enhance performance. Similarly, Fig. 4 (b) shows the results for detecting correct responses from partial responses. All approaches exhibit lower p and r , yet the same conclusion can be drawn: utilizing multiple SLMs improves performance, and the proposed method is superior to both $P(\text{yes})$ and ChatGPT.

E. Means

It is interesting to investigate how the mean calculation affects the results. Fig. 5 presents two bar charts comparing the $F1$ scores across different methods labeled as "geometric," "arithmetic," "max," and "min." The calculations for these means are as follows:

$$s_i(S, m = \text{'ari'}) = \frac{1}{|S(r_i)|} \sum_{j=1}^{|S(r_i)|} s_{i,j} \quad (7)$$

where $|S(r_i)|$ is the number of elements in S . The geometric mean can be calculated as:

$$s_i(S, m = \text{'geo'}) = \exp \left(\frac{1}{|S(r_i)|} \sum_{j=1}^{|S(r_i)|} \log(s_{i,j}) \right), \quad s_{i,j} > 0 \quad (8)$$

To find the minimum value in the dataset, we use:

$$s_i(S, m = \text{'min'}) = \min(S(r_i)) \quad (9)$$

This represents the minimum value among all sentences. The maximum value is calculated as:

$$s_i(S, m = \text{'max'}) = \max(S(r_i)) \quad (10)$$

This represents the maximum value among all sentences. Hence, Fig. 5 shows the results of testing different mean calculations. In Fig. 5 (a), the $F1$ scores range from 0.75 to 0.99, with the highest score observed for the "max" method (0.99). Fig. 5 (b) shows lower $F1$ scores, with the highest score of 0.81 for the "harmonic" method and the lowest at

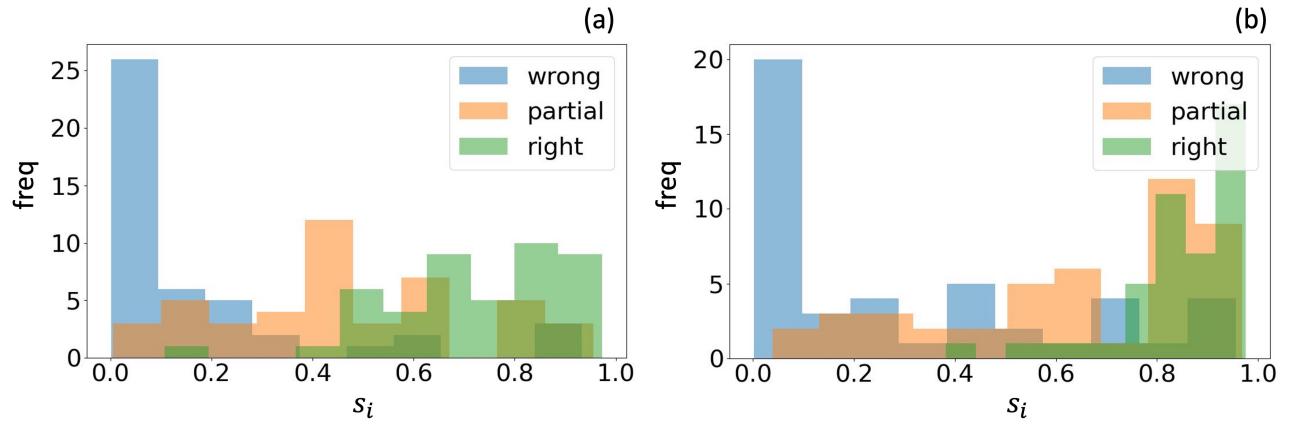


Fig. 6. Distributions of approaches: (a) proposed, (b) $P(\text{yes})$.

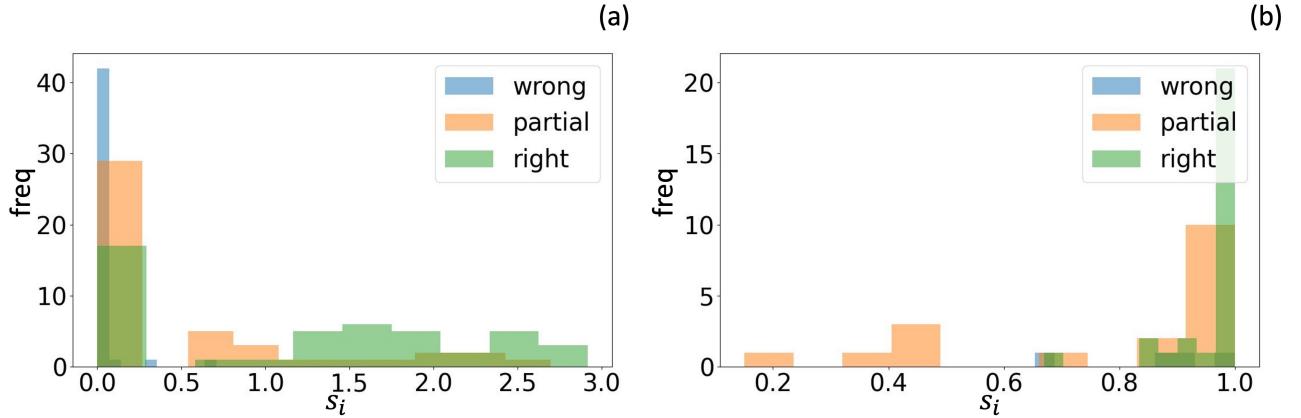


Fig. 7. Distributions of two means: (a) geometric, (b) harmonic.

0.66 for the "min" method. It is observed that the "max" method does not work well for "partial" responses, while there are good correct and hallucination sentences in one response. The results indicate that the harmonic mean yields the best outcomes.

F. Distributions

It is interesting to investigate why and how the proposed method works. Hence, Fig. 6 presents two histograms illustrating the frequency of responses categorized as "wrong," "partial," and "correct" across different values of s_i . Fig. 6 (a) and (b) show the histograms of s_i for the proposed method and $P(\text{yes})$, respectively. In both figures, "wrong" responses yield lower values of s_i , while "correct" responses consistently show higher s_i values and a higher frequency at elevated values of s_i . The "partial" responses are consistently present across both histograms but are less frequent than the "wrong" and "correct" categories. The "partial" responses spread between the two categories, which is one reason that partial responses are difficult to detect using s_i . Fig. 6 (a) exhibits a more pronounced peak for "wrong" responses compared to Fig. 6 (b). Although most "correct" responses in $P(\text{yes})$ have a

high s_i , they are inseparable from "partial" responses, while the proposed method can distinguish them and achieve better performance.

A similar conclusion can be drawn from using different means. Fig. 7 (a) and (b) present two histograms illustrating the frequency distribution of correctness categories (wrong, partial, correct) against the variable s for geometric and harmonic means. In Fig. 7 (a), there is a high frequency of "wrong" responses at lower values of s , while "partial" and "correct" responses show lower frequencies. Additionally, most "correct" responses have high values. Fig. 7 (b) indicates that the "correct" responses are found at higher values of s . The overall trend suggests that as s increases, the frequency of correct responses rises while the frequency of wrong responses decreases, highlighting a positive correlation between s and correctness. Note that Fig. 7 (b) only shows responses with values greater than 0, and more "wrong" responses are not depicted.

VI. CONCLUSION

This paper presents a framework for utilizing SLMs in answer verification for a given set of context, answers, and questions.

A real dataset is created to demonstrate their effectiveness in ensuring contextual relevance in responses. The experiments show that the proposed approach is 11% and 6.6% better than those using ChatGPT and P(yes), by utilizing two SLMs, Qwen2 and MiniCPM, within the proposed framework. The findings suggest that SLMs are a viable alternative for applications demanding efficient and scalable answer verification, particularly in resource-constrained environments. Future research could focus on optimizing the framework's performance on different types of data or on better integration of SLMs, such as adding gating mechanisms [37]. Another direction is to integrate with verification frameworks [38] to extract additional information online for checking general context.

REFERENCES

- [1] L. Huang, W. Yu, W. Ma, W. Zhong, Z. Feng, H. Wang, Q. Chen, W. Peng, X. Feng, B. Qin *et al.*, “A survey on hallucination in large language models: Principles, taxonomy, challenges, and open questions,” *arXiv preprint arXiv:2311.05232*, 2023.
- [2] C.-Y. Lin, “Rouge: A package for automatic evaluation of summaries,” in *Text summarization branches out*, 2004, pp. 74–81.
- [3] S. Kadavath, T. Conery, A. Askell, T. Henighan, D. Drain, E. Perez, N. Schieber, Z. Hatfield-Dodds, N. DasSarma, E. Tran-Johnson *et al.*, “Language models (mostly) know what they know,” *arXiv preprint arXiv:2207.05221*, 2022.
- [4] P. Lewis, E. Perez, A. Piktus, F. Petroni, V. Karpukhin, N. Goyal, H. Küttler, M. Lewis, W.-t. Yih, T. Rocktäschel *et al.*, “Retrieval-augmented generation for knowledge-intensive nlp tasks,” *Advances in Neural Information Processing Systems*, vol. 33, pp. 9459–9474, 2020.
- [5] Z. Jing, Y. Su, Y. Han, B. Yuan, H. Xu, C. Liu, K. Chen, and M. Zhang, “When large language models meet vector databases: A survey,” *arXiv preprint arXiv:2402.01763*, 2024.
- [6] Z. Lu, X. Li, D. Cai, R. Yi, F. Liu, X. Zhang, N. D. Lane, and M. Xu, “Small language models: Survey, measurements, and insights,” *arXiv preprint arXiv:2409.15790*, 2024.
- [7] L. Reynolds and K. McDonell, “Prompt programming for large language models: Beyond the few-shot paradigm,” in *Extended Abstracts of the 2021 CHI Conference on Human Factors in Computing Systems*, 2021, pp. 1–7.
- [8] J. Kaddour, J. Harris, M. Mozes, H. Bradley, R. Raileanu, and R. McHardy, “Challenges and applications of large language models,” *arXiv preprint arXiv:2307.10169*, 2023.
- [9] Y. Xie, C. Yu, T. Zhu, J. Bai, Z. Gong, and H. Soh, “Translating natural language to planning goals with large-language models,” *arXiv preprint arXiv:2302.05128*, 2023.
- [10] M. Cheung, “A reality check of the benefits of llm in business,” *arXiv preprint arXiv:2406.10249*, 2024.
- [11] H. Salehinejad, S. Sankar, J. Barfett, E. Colak, and S. Valaei, “Recent advances in recurrent neural networks,” *arXiv preprint arXiv:1801.01078*, 2017.
- [12] A. Graves and A. Graves, “Long short-term memory,” *Supervised sequence labelling with recurrent neural networks*, pp. 37–45, 2012.
- [13] A. Gillioz, J. Casas, E. Mugellini, and O. Abou Khaled, “Overview of the transformer-based models for nlp tasks,” in *2020 15th Conference on Computer Science and Information Systems (FedCSIS)*. IEEE, 2020, pp. 179–183.
- [14] A. Vaswani, N. Shazeer, N. Parmar, J. Uszkoreit, L. Jones, A. N. Gomez, Ł. Kaiser, and I. Polosukhin, “Attention is all you need,” *Advances in neural information processing systems*, vol. 30, 2017.
- [15] R. Dale, “Gpt-3: What’s it good for?” *Natural Language Engineering*, vol. 27, no. 1, pp. 113–118, 2021.
- [16] J. Manyika, “An overview of bard: an early experiment with generative ai,” Technical report, Google AI, Tech. Rep., 2023.
- [17] N. F. Liu, T. Zhang, and P. Liang, “Evaluating verifiability in generative search engines,” *arXiv preprint arXiv:2304.09848*, 2023.
- [18] L. Ouyang, J. Wu, X. Jiang, D. Almeida, C. Wainwright, P. Mishkin, C. Zhang, S. Agarwal, K. Slama, A. Ray *et al.*, “Training language models to follow instructions with human feedback,” *Advances in neural information processing systems*, vol. 35, pp. 27730–27744, 2022.
- [19] T. Brown, B. Mann, N. Ryder, M. Subbiah, J. D. Kaplan, P. Dhariwal, A. Neelakantan, P. Shyam, G. Sastry, A. Askell *et al.*, “Language models are few-shot learners,” *Advances in neural information processing systems*, vol. 33, pp. 1877–1901, 2020.
- [20] P. N. Venkit, T. Chakraborti, V. Gupta, H. Biggs, M. Srivastava, K. Goswami, S. Rajtmajer, and S. Wilson, ““confidently nonsensical?”: A critical survey on the perspectives and challenges of ‘hallucinations’ in nlp,” *arXiv preprint arXiv:2404.07461*, 2024.
- [21] C. Guo, G. Pleiss, Y. Sun, and K. Q. Weinberger, “On calibration of modern neural networks,” in *International conference on machine learning*. PMLR, 2017, pp. 1321–1330.
- [22] K. Papineni, S. Roukos, T. Ward, and W.-J. Zhu, “Bleu: a method for automatic evaluation of machine translation,” in *Proceedings of the 40th annual meeting of the Association for Computational Linguistics*, 2002, pp. 311–318.
- [23] Y. Chen, Q. Fu, Y. Yuan, Z. Wen, G. Fan, D. Liu, D. Zhang, Z. Li, and Y. Xiao, “Hallucination detection: Robustly discerning reliable answers in large language models,” in *Proceedings of the 32nd ACM International Conference on Information and Knowledge Management*, 2023, pp. 245–255.
- [24] K. Filippova, “Controlled hallucinations: Learning to generate faithfully from noisy data,” *arXiv preprint arXiv:2010.05873*, 2020.
- [25] H. Orgad, M. Toker, Z. Gekhman, R. Reichart, I. Szpektor, H. Kotek, and Y. Belinkov, “Llms know more than they show: On the intrinsic representation of llm hallucinations,” *arXiv preprint arXiv:2410.02707*, 2024.
- [26] A. Havrilla, S. Raparthy, C. Nalmpantis, J. Dwivedi-Yu, M. Zhuravskiy, E. Hambro, and R. Railneau, “Glore: When, where, and how to improve llm reasoning via global and local refinements,” *arXiv preprint arXiv:2402.10963*, 2024.
- [27] S. Welleck, X. Lu, P. West, F. Brahman, T. Shen, D. Khashabi, and Y. Choi, “Generating sequences by learning to self-correct,” *arXiv preprint arXiv:2211.00053*, 2022.
- [28] S. Farquhar, J. Kossen, L. Kuhn, and Y. Gal, “Detecting hallucinations in large language models using semantic entropy,” *Nature*, vol. 630, no. 8017, pp. 625–630, 2024.
- [29] Y. A. Yadkori, I. Kuzborskij, A. György, and C. Szepesvári, “To believe or not to believe your llm,” *arXiv preprint arXiv:2406.02543*, 2024.
- [30] Z. Sprague, F. Yin, J. D. Rodriguez, D. Jiang, M. Wadhwa, P. Singhhal, X. Zhao, X. Ye, K. Mahowald, and G. Durrett, “To cot or not to cot? chain-of-thought helps mainly on math and symbolic reasoning,” *arXiv preprint arXiv:2409.12183*, 2024.
- [31] A. Radford, K. Narasimhan, T. Salimans, I. Sutskever *et al.*, “Improving language understanding by generative pre-training,” 2018.
- [32] (2023) Llama-2-70b. (accessed: 12.11.2023). [Online]. Available: <https://huggingface.co/meta-llama/Llama-2-70b>
- [33] E. J. Hu, Y. Shen, P. Wallis, Z. Allen-Zhu, Y. Li, S. Wang, L. Wang, and W. Chen, “Lora: Low-rank adaptation of large language models,” *arXiv preprint arXiv:2106.09685*, 2021.
- [34] P. Béchard and O. M. Ayala, “Reducing hallucination in structured outputs via retrieval-augmented generation,” *arXiv preprint arXiv:2404.08189*, 2024.
- [35] J. Bai, S. Bai, Y. Chu, Z. Cui, K. Dang, X. Deng, Y. Fan, W. Ge, Y. Han, F. Huang *et al.*, “Qwen technical report,” *arXiv preprint arXiv:2309.16609*, 2023.
- [36] S. Hu, Y. Tu, X. Han, C. He, G. Cui, X. Long, Z. Zheng, Y. Fang, Y. Huang, W. Zhao *et al.*, “Minicpm: Unveiling the potential of small language models with scalable training strategies,” *arXiv preprint arXiv:2404.06395*, 2024.
- [37] Y. Zhou, T. Lei, H. Liu, N. Du, Y. Huang, V. Zhao, A. M. Dai, Q. V. Le, J. Laudon *et al.*, “Mixture-of-experts with expert choice routing,” *Advances in Neural Information Processing Systems*, vol. 35, pp. 7103–7114, 2022.
- [38] J. Chen, G. Kim, A. Sriram, G. Durrett, and E. Choi, “Complex claim verification with evidence retrieved in the wild,” *arXiv preprint arXiv:2305.11859*, 2023.