Logical Closed Loop: Uncovering Object Hallucinations in Large Vision-Language Models

Anonymous ACL submission

Abstract

Object hallucination has been an Achilles' heel which hinders the broader applications of large vision-language models (LVLMs). Object hallucination refers to the phenomenon that the LVLMs claim non-existent objects in the image. To mitigate the object hallucinations, instruction tuning and external model-based detection methods have been proposed, which either require large-scare computational resources or depend on the detection result of external models. However, there remains an under-explored field to utilize the LVLM itself to alleviate object hallucinations. In this work, we adopt the intuition that the LVLM tends to respond logically consistently for existent objects but inconsistently for hallucinated objects. Therefore, we propose a Logical Closed Loop-based framework for Object Hallucination Detection and Mitigation, namely **LogicCheckGPT**. In specific, we devise logical consistency probing to raise questions with logical correlations, inquiring about attributes from objects and vice versa. Whether their responses can form a logical closed loop serves as an indicator of object hallucination. As a plug-and-play method, it can be seamlessly applied to all existing LVLMs. Comprehensive experiments conducted on three benchmarks across four LVLMs have demonstrated significant improvements brought by our method, indicating its effectiveness and generality.

1 Introduction

With the great advancement of large language models (LLMs) (Ouyang et al., 2022; Touvron et al., 2023; Zhao et al., 2023), they have showcased impressive abilities, such as text generation, instruction following. Recent studies have been devoted to introduce the general artificial intelligence of LLMs to the field of multimodal models. Empowered by LLMs, large vision-language models (LVLMs) (Liu et al., 2023; Ye et al., 2023; Zhu et al., 2023; Li et al., 2023; Bai et al., 2023; Dai et al., 2023) are facilitated to perform strong multimodal understanding and reasoning.

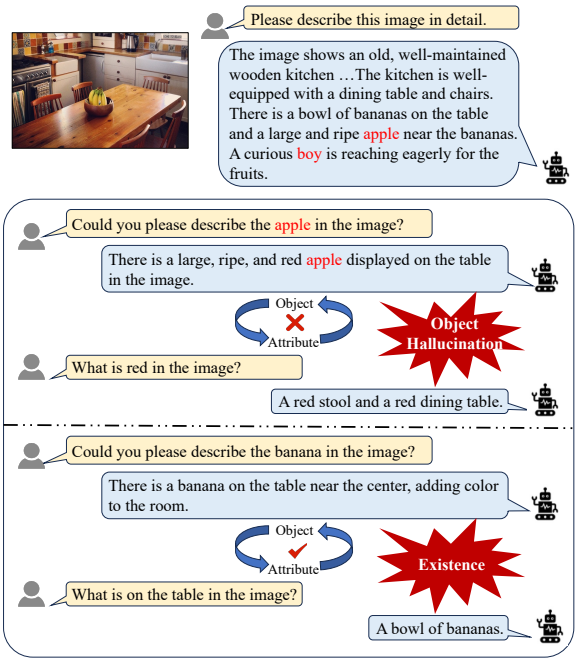


Figure 1: An example of object hallucinations. Hallucinated objects are highlighted in red. The LVLM shows different logical consistency to hallucinated object "apple" and existent object "banana".

Despite the exciting breakthrough in LVLMs, they all suffer from hallucination issues inevitably, particularly object hallucination. Object hallucination refers to the phenomenon that the LVLMs generate inconsistent descriptions of the given image, for example, making up non-existent objects. Taking Fig. 1 as an example, the LVLM hallucinates several objects which are not existent in the image, including an apple and a boy. This issue hinders more widespread application of LVLMs. In safety-related scenarios, the consequences of hallucinations would be unbearable.

There have been several efforts made to alleviate hallucinations in LVLMs. Generally, these meth-

ods can be categorized into three groups. The first as well as the most popular approach (Liu et al., 2024; Lee et al., 2023) mainly resorts to instruction tuning or specific retraining to facilitate LVLMs to generate less hallucinated contents. The second approach (Yin et al., 2023; Zhou et al., 2024) incorporates external detection models or specific LVLMs to enhance visual understanding, thereby refining the outputs of the original LVLMs. The last approach (Huang et al., 2023; Leng et al., 2023) investigates the decoding process of LVLMs and device novel decoding strategies to avoid hallucinations. Although these methods have achieved some effectiveness in alleviating hallucinations, there are still some drawbacks: requiring significant computational resources, depending on external models, or necessitating access to the internal parameters of the model.

However, we argue that the logical consistency of LVLM behaviors have the potential to elucidate the underlying hallucination it encapsulates. As illustrated in Fig. 1, we first obtain the attributes "red" for "apple" and "on the table" for "banana" by inquiring the LVLM. Subsequently, when we inquire about which object possesses these attributes, the LVLM can correctly respond with "banana" but fails to answer for the hallucinated object "apple". It demonstrates that when we pose a series of logically connected questions about a particular object, the LVLM exhibits better logical consistency for existing objects, while its performance tends to degrade for hallucinated objects. It is reasonable because the described attributes of hallucinated objects primarily originate from two sources: attributes from other objects in the image, or fabricated attributes absent in the image. Consequently, the model may fail to answer the hallucinated object when we question what possesses these attributes.

Inspired by this observation, we propose a novel and effective framework called Logic Closed Loop for Object Hallucination Detection and Mitigation, namely **LogicCheckGPT**, which is training-free and only requires language interaction. Our aim is to formulate two types of questions in two stages: the first stage involves inquiring attributes based on objects, followed by inquiring objects based on attributes. Whether their responses can form a logical closed loop serves as an indicator of object hallucination.

In specific, according to the different stages of questioning, we divide our framework into 5 steps: (1) *Object extraction* extracts objects in the responses of LVLMs. (2) *Object-to-Attribute inquiring* inquiries into the detailed attributes of the target objects. (3) *Attribute-to-Object inquiring* further formulates follow-up questions to inquire what object possesses the attributes mentioned in previous answers. (4) *Logic closed loop check* examines whether the logical relationships from objects to attributes and attributes to objects can form a closed loop. (5) *Hallucination detection and mitigation* rectifies hallucinated objects if the ratio of closed loops to the total number of questions exceeds a certain threshold. Our method is a plug-and-play approach that can be applied to various LVLMs without training or relying on external detection models. Furthermore, the question-answer process in natural language enhances its interpretability.

We evaluated the effectiveness of our framework across multiple advanced LVLMs on several benchmarks (Yifan Li and Wen, 2023; Fu et al., 2023), as well as GPT-4v assisted evaluation(Liu et al., 2024; Yin et al., 2023). Our method demonstrates significant improvements across state-of-the-art LVLMs, including a 31.33%/10.00% improvement on the POPE dataset for mPLUG-Owl (Ye et al., 2023)/MiniGPT-4 (Zhu et al., 2023).

Overall, our main contributions can be summarized as follows:

- We are the first to adopt the logical closed loop in the context of object hallucination alleviation in LVLMs.
- We propose a novel framework LogicCheckGPT for detecting and mitigating object hallucinations in LVLMs, which is training-free and offers language interaction for user-friendly interpretation.
- Comprehensive experiments are conducted to validate the effectiveness of our method, where the results demonstrate the superiority and universality.

2 Related Work

2.1 Large Vision-Language Models

With the surge in the capabilities of large language models (LLMs) (Ouyang et al., 2022; Zhao et al., 2023; Brown et al., 2020), there is currently a growing interest in how to integrate the general artificial intelligence of LLMs into the multimodal domains. In consequence, large vision-language

models (LVLMs) powered by LLMs are proposed (Ye et al., 2023; Zhu et al., 2023; Liu et al., 2023; Li et al., 2023; Dai et al.; Bai et al., 2023), which can understand multimodal contents and perform multimodal tasks under instructions. In general, existing LVLMs follow the following paradigm: leveraging a multimodal alignment module to comprehend multimodal inputs, followed by utilizing a LLM to generate responses. Therefore, the training process of LVLMs typically involves modalities alignment pre-training and instruction tuning. Specifically, mPLUG-Owl (Ye et al., 2023) pre-trains the encoder and alignment module, and then finetunes LLaMa (Touvron et al., 2023) by low-rank adaption. In contrast, LLaVA (Liu et al., 2023) only pre-trains the alignment network and finetunes the alignment network and Vicuna (Chiang et al., 2023) on constructed instructions. MiniGPT-4 (Zhu et al., 2023) only finetunes the cross-modal alignment network with other modules frozen.

2.2 Hallucination in LVLMs

Despite the strong capabilities of these LVLMs, they all grapple with hallucination issues unexpectedly. To tackle with this issue, several benchmarks (Fu et al., 2023; Xu et al., 2023; Yifan Li and Wen, 2023; Lovenia et al., 2023) have been proposed to provide detailed evaluations of the hallucination degree exhibited by LVLMs.

Existing hallucination mitigation strategies for LVLMs can be roughly divided into three groups. The first and most widely adopted approach (Liu et al., 2024; Gunjal et al., 2023; Lee et al., 2023; Wang et al., 2023) primarily relies on instruction tuning and retraining. LRV-Instruction (Liu et al., 2024) introduces a comprehensive instruction tuning dataset encompassing positive and negative instructions. (Wang et al., 2023) adopts an iterative instruction generation strategy to improve diversity and accuracy of instructions. Volcano (Lee et al., 2023) facilitates the model with the ability to utilize self-feedback to self-revise responses through training. However, these methods heavily depend on the quality of instruction data construction and require substantial computational resources.

The second group of approaches, exemplified by Woodpecker (Yin et al., 2023) and LURE (Zhou et al., 2024), aim to integrate external detection models or specific LVLMs as revisors to enhance accurate visual understanding, thereby refining base LVLMs' hallucinated generation. Nevertheless, these approaches rely on external models and fail to explore the intrinsic capabilities of the base model.

The third group of approaches aim to devise decoding strategies to mitigate hallucinations during the process of decoding. OPERA (Huang et al., 2023) propose a penalty-based decoding method along with roll-back strategy to avoid over-trust during decoding. On the other hand, VCD (Leng et al., 2023) introduces contrastive decoding to reduce over-reliance on spurious bias and learned priors. However, obtaining internal states of LVLMs during the decoding process poses a challenge for common users.

Compared to existing approaches, our proposed method is training-free and mitigates hallucinations solely through language interactions. It not only explores the potential of LVLMs to alleviate hallucinations but also offers better interpretability.

2.3 Consistency Checking for Hallucination Detection

There have also been some works on hallucination detection in LLMs (Manakul et al., 2023; Kuhn et al., 2022; Lin et al., 2023), which view the consistency of responses reflects the model's uncertainty. (Kuhn et al., 2022) propose semantic entropy to measure the degree of semantic divergence among responses to accommodate semantic equivalence in free-form text. SelfCheckGPT (Manakul et al., 2023) extend the method to black-box LLMs, eliminating the need for tokens' probability. (Lin et al., 2023) introduce and compare various uncertainty estimation metrics for black-box LLMs. In contrast to prior works that focus on consistency among responses to the same question, we propose LogicCheckGPT, a logic consistency-based method that involves logic-related questions and answers. LogicCheckGPT is more capable of delving deeper into the internal uncertainty and the degree of hallucination within the LVLMs.

3 Method

In this section, we first introduce the overall framework of LogicCheckGPT, and then elaborate each component. Our framework is shown in Fig. 2.

3.1 Overall

To alleviate object hallucinations, we delve into the logical consistency of LVLMs' responses. Specifically, we examine whether the responses demonstrate logical coherence. For each object mentioned

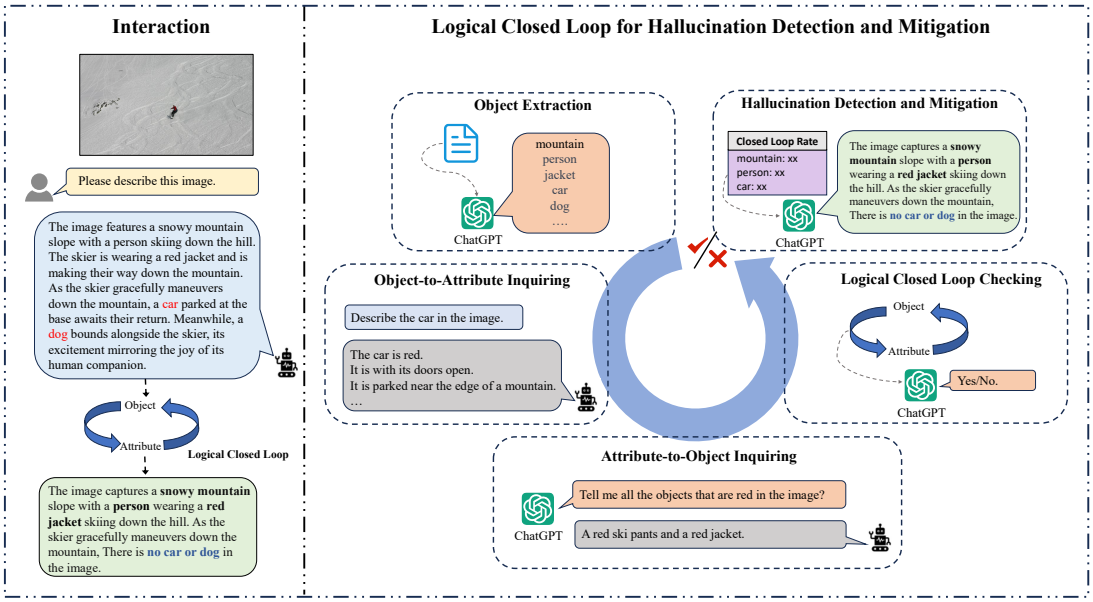


Figure 2: The proposed framework LogicCheckGPT. For LVLM responses to multimodal instructions, LogicCheckGPT employs the following five steps to alleviate object hallucinations: object extraction, object-to-attribute inquiring, attribute-to-object inquiring, logical close loop checking, and hallucination detection and mitigation.

in a response, we pose two types of sequential questions: one regarding the attributes of the object, and another about which object possesses those attributes. The logical consistency of the LVLM's responses, i.e., whether they form a closed loop of logical reasoning, serves as an indicator of object hallucination. Here, the "logical closed loop" refers to the subsequent object answered being consistent with the initial object.

This process is decomposed into the following steps: object extraction, object-to-attribute inquiring, attribute-to-object inquiring, logical closed loop checking, and hallucination detection and mitigation.

3.2 Object Extraction

To determine the object hallucination, we first need to extract candidate objects from the responses of LVLMs for further querying and checking. For simplicity and versatility, we adopt an LLM to complete each sub-task, including object extraction. Specifically, we employ GPT-3.5 [1] as our LLM because of its strong capabilities. The prompt we used can be referred in Appendix C.1.

3.3 Object-to-Attribute Inquiring

As discussed in Section 3.1, the primary step involves constructing object-to-attribute questions to inquire about the attributes of the object. However, it is impractical to enumerate all possible attributes, given their infinite nature. Additionally, it is challenging to create generic attribute question templates applicable to all objects, as different objects typically possess their own specific attributes. For example, attributes related to material for a "dining table" differ from those related to clothing for a "person".

For better flexibility and adaptability, we prompt the LVLM to provide a detailed description of the object in free-form text. The template question is formulated as follows: *"Could you please describe the object in the image?"* We ask the LVLM to respond multiple times. This approach allows us to obtain detailed and specific attribute descriptions from the LVLM regarding the object, thereby facilitating the construction of attribute-to-object questions in subsequent steps.

3.4 Attribute-to-Object Inquiring

In the attribute-to-object inquiring stage, our goal is to formulate questions from the attributes to the object, in contrast to Section 3.3. After obtaining the attribute descriptions of the object, follow-up questions can be raised based on them. Rather than directly prompting the LLM to formulate questions from the descriptions, we break down this task into two subtasks: attribute extraction and question formulation. This approach avoids potential issues of misleading the LLM to not follow instructions and

[1] https://platform.openai.com/docs/models/gpt-3-5-turbo

inadvertently revealing object identity in the questions. It has also been observed that decomposing a task into several simple sub-tasks for LLMs to fulfill yields better performance (Wei et al., 2022; Zhao et al., 2023).

In specific, we prompt the LLM to extract attributes of the target object from the description, wherein the target object is represented as "The object". For instance, extracted attributes such as *"The object is made of wood in the image"*, *"The object is red in color"*. Detailed prompting instructions can be found in Appendix C.2.1.

Subsequently, we instruct the LLM to convert these extracted attributes into questions that inquire about what object possesses the specific attributes. However, we have noticed that asking questions like *"What is/has {attribute} in the image?"* often yield answers about the most obvious objects with the target attributes, potentially leading to the omission of other less conspicuous yet existent objects. To deal with this issue, we frame our questions in the format *"Could you tell me all the objects that {attribute} in the image?"*. The ablation study of this prompt design can be referred in Section 4.3. This format enables the LVLM to comprehensively cover objects that meet the specified attribute constraints. Detailed prompting instructions can be found in Appendix C.2.2.

3.5 Logical Closed Loop Checking

After obtaining the answers from the attribute-to-object inquiring stage, we can assess whether each answer forms a logical closed loop, meaning that the object mentioned in the answer is consistent with the examinee object. In specific, we prompt the LLM to check whether the examinee object is covered in each LVLM's response. The judgement is limited to "Yes" and "No". For the i-th examinee object, the judgement of the j-th answer is mapped into score x_j^i through the mapping {Yes: 1.0, No: 0.0}. The prompt is listed in Appendix C.3.

3.6 Hallucination Detection and Mitigation

Finally, the logical closed loop rate for each examinee object, defined as the number of logical closed loops divided by the total number of attribute-to-object question-answer pairs N, can be formulated as

$$\mathcal{S}(i) = \frac{1}{N} \sum_{j}^{N} x_j^i \tag{1}$$

where $\mathcal{S}(i)$ denotes the logical closed loop rate for the i-th object, N signifies the total number of attribute-to-object question-answer tuples.

As discussed before, the object is likely to be hallucinated when $\mathcal{S}(i)$ is low. On the other hand, when it's close to 1, it's more likely that the object truly exists. Therefore, $\mathcal{S}(i)$ serves as an indicator of object hallucination. Through a valid hallucination threshold λ, our method can effectively detect objects that are likely to be hallucinated, i.e., $\mathcal{S}(i)$ is below λ.

After identifying hallucinated objects, we guide the LLM to eliminate contents related to hallucinated objects, aiming to minimize hallucinations. The detailed prompt can be found in Appendix C.4.

4 Experiment

4.1 Experimental Setup

4.1.1 Dataset

POPE (Yifan Li and Wen, 2023) is proposed to provide a detailed evaluation of object hallucination in LVLMs, by querying the models about the presence of specific objects in given images. POPE adopts three sampling settings to construct negative samples: random, popular, and adversarial. The random setting selects non-present objects at random, whereas the popular setting chooses from a list of frequently occurring but absent objects, and the adversarial method selects based on common co-occurrence in contexts despite the absence in the target image. For each sampling setting, we sample 50 images and 6 questions for each image, with an even distribution of positive and negative samples (50% - 50%). Accuracy and F1-score are employed as the evaluation metrics.

MME (Fu et al., 2023) serves as a comprehensive benchmark for evaluating the perecptual and cognitive capabilities of LVLMs across a wide spectrum of tasks. For the purpose of this study, we only utilize the *existence* subset to evaluate the phenomenon of object-level hallucination within these models. This approach mirrors the methodology employed in the POPE framework, wherein each subset consists of binary "Yes-or-No" questions. We use accuracy and accuracy+ as metrics, where the former is calculated based on each question, while the latter is based on each image, requiring both questions to be answered correctly.

GPT-4v Assisted Evaluation To assess the effectiveness of our method for hallucination mitigation

Model	Method	Adversarial		Popular		Random	
		Acc	F1	Acc	F1	Acc	F1
mPLUG-Owl	vanilla	50.67	66.81	51.66	67.26	55.33	68.98
	LRV-Instruction	59.67	69.21	68.33	74.11	74.33	77.94
	SelfCheck	66.67	74.09	72.00	77.29	70.66	75.82
	LURE	72.40	77.60	78.60	80.29	79.67	78.75
	LogicCheckGPT	**82.00**	**82.23**	**84.66**	**84.45**	**91.00**	**90.84**
MiniGPT-4	vanilla	72.67	75.88	78.33	79.87	84.33	84.59
	LRV-Instruction	74.00	71.11	80.33	78.70	81.67	80.97
	SelfCheck	73.00	72.72	76.67	75.86	76.00	73.53
	LURE	76.20	78.04	80.67	80.67	83.67	84.14
	LogicCheckGPT	**82.67**	**80.59**	**83.67**	**81.51**	**86.67**	**85.29**
LLaVA-1.5	vanilla	83.33	84.84	84.67	85.89	93.00	93.02
	SelfCheck	88.67	88.27	88.67	88.59	90.33	89.53
	LURE	84.54	85.04	85.51	84.32	89.67	89.70
	LogicCheckGPT	**90.00**	**89.58**	**91.67**	**91.40**	**93.33**	**93.00**
QWEN-VL-Chat	vanilla	86.67	86.67	86.33	86.37	90.67	90.28
	SelfCheck	87.67	87.54	87.67	87.87	91.13	**91.45**
	LURE	87.00	87.62	87.33	87.16	88.67	88.28
	LogicCheckGPT	**89.00**	**88.00**	**89.67**	**88.64**	**91.33**	90.71

Table 1: The performance comparison between our proposed method LogicCheckGPT and baselines on POPE. The best result is highlighted in boldface.

in open-ended generation, we adopted the GPT-4v Assisted Evaluation, inspired by Yin et al. (2023) and Liu et al. (2024). Our evaluation samples a set of 50 images from the COCO 2014 validation dataset, and asks the model to generate detailed descriptions. Subsequently, we prompt GPT-4v to score original outputs and our outputs in accuracy and relevancy, based on the image and instruction. The detailed prompt example is shown in Appendix C.5.

4.1.2 Baselines

We selected several widely used open-source LVLMs as backbones to evaluate the effectiveness of our LogicCheckGPT, including mPLUG-Owl (mplug-owl-llama-7b) (Ye et al., 2023), LLaVA (llava-1.5-7b) (Liu et al., 2023), MiniGPT-4 (vicuna-13b) (Zhu et al., 2023), QWEN-VL-Chat (Bai et al., 2023).

We also compare our method with advanced hallucination detection and mitigation methods, including LRV-Instruction (Liu et al., 2024), LURE (Zhou et al., 2024) and SelfCheckGPT (Manakul et al., 2023). IRV-Instruction constructs a comprehensive instruction dataset for instruction tuning. However, as they only released the checkpoints of fine-tuned mPLUG-Owl and MiniGPT-4, we report the results of these two models. LURE trains a LVLM revisor to post-hoc rectify the hallucinated outputs of base models. SelfCheckGPT employs semantic uncertainty to detect hallucinations. We integrate it into our framework to fulfill hallucination mitigation. In addition, the base LVLMs are referred to as vanilla.

4.2 Experimental Results

Results on POPE The overall performance of our proposed method LogicCheckGPT on POPE is shown in Table 1, from which we have the following observations:

Firstly, our LogicCheckGPT consistently demonstrates significant performance improvements across various LVLMs under different settings. It can be observed that there is a significant performance decline of LVLMs as we transition from random to popular and adversarial settings, indicating that LVLMs tend to hallucinate non-existent objects that are related to the input image. In specific, mPLUG-Owl only achieved accuracies of 50.67%, 51.66%, and 55.33% across the three

Model	Method	Acc	Acc+
mPLUG-Owl	vanilla	65.00	35.00
	LRV-Instruction	83.33	66.67
	SelfCheck	85.00	73.33
	LURE	80.00	60.00
	LogicCheckGPT	**96.67**	**93.33**
MiniGPT-4	vanilla	78.33	56.67
	LRV-Instruction	83.33	66.67
	SelfCheck	80.00	60.00
	LURE	85.00	70.00
	LogicCheckGPT	**86.67**	**73.33**
LLaVA-1.5	vanilla	96.67	93.33
	SelfCheck	96.67	93.33
	LURE	93.33	86.67
	LogicCheckGPT	96.67	93.33
QWEN-VL-Chat	vanilla	88.33	80.00
	SelfCheck	93.33	86.67
	LURE	90.00	80.00
	LogicCheckGPT	**95.00**	**90.00**

Table 2: The performance comparison between our proposed method LogicCheckGPT and baselines on MME Existence subset. The best result on each dataset is highlighted in boldface.

Model	Method	Acc	Rel
mPLUG-Owl	vanilla	3.44	**8.78**
	LogicCheckGPT	**4.32**	8.74
MiniGPT-4	vanilla	5.00	7.96
	LogicCheckGPT	**6.02**	**8.38**
LLaVA-1.5	vanilla	5.22	7.24
	LogicCheckGPT	**6.50**	**7.64**
QWEN-VL-Chat	vanilla	8.36	9.96
	LogicCheckGPT	**8.58**	9.96

Table 3: The performance of our proposed method LogicCheckGPT over base LVLMs on GPT-4v assisted evaluation. The best result on each dataset is highlighted in boldface.

settings. For MiniGPT-4, its performance in adversarial settings is over 10% lower compared to that in random settings. However, with the help of LogicCheckGPT, all LVLMs outperform the vanilla ones significantly under all settings. More specifically, mPLUG-Owl equipped with LogicCheckGPT achieves an accuracy enhancement exceeding 30% across three settings. Despite the already promising performance of LLaVA-1.5, our method still brings a substantial improvement to it, achieving about a 6.67% increase in accuracy. This demonstrates the effectiveness and robustness of our approach.

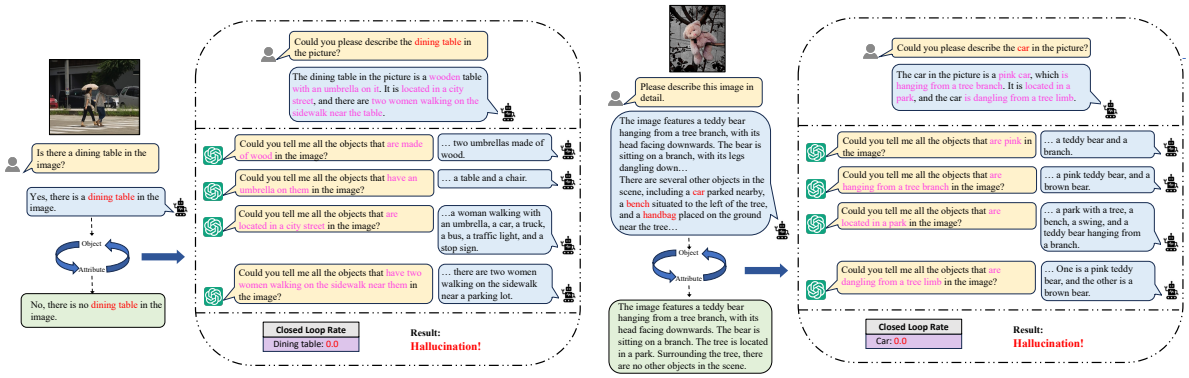
Secondly, our proposed LogiCheckGPT outperforms other competitors by a significant margin. While LRV-Instruction facilitates robust instruction tuning of LVLMs, it remains challenging to completely eliminate potential hallucination issues within the model. Due to that SelfCheckGPT relies on semantic consistency in model responses, it is difficult to detect hallucinated objects when the LVLM is overconfident. LURE demonstrates significant improvements on models like mPLUG-Owl and MiniGPT-4, but shows only marginal enhancements on more powerful models such as LLaVA-1.5 and QWEN-VL-Chat. We attribute this to the fact that LURE is built upon MiniGPT-4, thus inheriting the limitations of the underlying model. By contrast, our LogicCheckGPT yields the most substantial improvements across all LVLMs, indicating the superiority of our method.

Results on MME Existence Subset As shown in Table 2, we also evaluate our method on MME existence subset, which focuses on the object existence hallucination. It can be observed that, although mPLUG-Owl performs worst, our method results in substantial increases in both accuracy and accuracy+, by 31.67% and 58.33%, respectively, achieving remarkably high levels of performance. It also demonstrates mPLUG-Owl inherently contains information pertaining to object hallucinations, which can be unearthed by our method. In addition, LogicCheckGPT consistently brings significant improvement for MiniGPT-4 and QWEN-VL-Chat, while maintaining the strong performance of LLaVA-1.5.

Results on GPT-4v Assisted Evaluation Apart from "Yes" or "No" questions in POPE and MME, we employed GPT-4v to assist in evaluating our LogicCheckGPT for open-text generation, following previous works (Liu et al., 2024; Yin et al., 2023). The results are summarized in Table 3. It can be observed that our method significantly enhances the accuracy of each model, demonstrating the effectiveness of our approach in object hallucination mitigation. In addition, as LogicCheckGPT can remove irrelevant hallucinated information and preserving fluent language structures, it can maintain or even improve relevancy.

4.3 Ablation Study

We conduct ablation study for several variants, as illustrated in Fig. 4. *vanilla* adopts no halluci-



(a) Yes-or-No question. (b) Open-ended question.

Figure 3: The visualization of two representative examples of our LogicCheckGPT for mPLUG-Owl. The hallucinated objects are highlighted in red and attributes are highlighted in magenta.

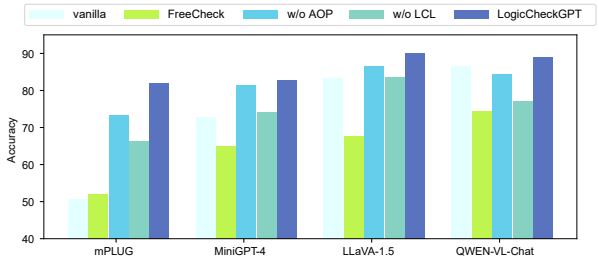


Figure 4: The performance comparison between LogicCheckGPT and several variants, vanilla, FreeCheck, LogicCheckGPT w/o AOP (w/o AOP) and LogicCheckGPT w/o LCL (w/o LCL) across LVLMs on POPE adversarial setting.

nation mitigation method. *FreeCheck* employs an LLM (GPT-3.5) to autonomously interrogate LVLMs through multi-turn interactions, similiar to (Cohen et al., 2023). *LogicCheckGPT w/o AOP* refers to replacing the Attribute-to-Object prompt in Section 3.4 to inquire only object instead of covering all objects, e.g. *What is/has {attribute} in the image?*. *LogicCheckGPT w/o LCL* employs an LLM to determine the logical consistency directly without Logical Closed Loop rate.

We can observe that *FreeCheck* performs worse for most LVLMs, primarily because the LLM fails to raise valuable questions to detect hallucinations. LogicCheckGPT w/o AOP brings significant improvements to several models but still falls below our method, indicating that covering a sufficient number of objects is beneficial. Though LogicCheckGPT w/o LCL falls between vanilla and w/o AOP, demonstrating the effectiveness of calculating logical closed loop rate. For the powerful QWEN-VL-Chat, which has already achieved impressive performance, only our LogicCheckGPT can enhance its capabilities.

4.4 Case Study

In this section, we have selected two representative examples for mPLUG-Owl covering distinct types of questions, including a binary question, "Is there a dining table in the image?" and an open-ended query, "Please describe this image in detail.", as illustrated in Fig 3. In the first case 3(a), the model answers "yes" to the binary question, while our method detects that dining table is hallucinated and corrects the output. As for open-ended text generation, the models tends to hallucinate nonexistent objects as the length of the generated sequence increases. However, our method is capable of individually verifying the existence of objects, thereby mitigating hallucinations.

5 Conclusion

We propose a novel logical closed loop-based framework LogicCheckGPT for object hallucination mitigation in LVLMs. Our motivation stems from the observation that LVLMs often exhibit logically inconsistent responses to hallucinated objects. Therefore, we devise logic consistency probing, which involves asking questions with logical correlations, such as inquiring about attributes from objects and vice versa. Specifically, we break down this process into several steps: object extraction, object-to-attribute inquiring, attribute-to-object inquiring, logical closed-loop checking, and hallucination detection and mitigation. Comprehensive experiments conducted on several benchmarks demonstrate the superiority of our framework.

Limitations

In this work, we propose a logical closed loop-based framework for object hallucination mitigation. However, our method still has the following two limitations. Firstly, adopting our framework inevitably incurs costs, as we rely on the GPT-3.5 API. Secondly, our work only focuses on addressing object hallucinations. There are also other types of hallucinations, including attribute hallucinations and knowledge hallucinations. Therefore, extending our framework to encompass a broader range of hallucination mitigation represents our future directions.

References

Jinze Bai, Shuai Bai, Shusheng Yang, Shijie Wang, Sinan Tan, Peng Wang, Junyang Lin, Chang Zhou, and Jingren Zhou. 2023. Qwen-vl: A frontier large vision-language model with versatile abilities. *arXiv preprint arXiv:2308.12966*.

Tom Brown, Benjamin Mann, Nick Ryder, Melanie Subbiah, Jared D Kaplan, Prafulla Dhariwal, Arvind Neelakantan, Pranav Shyam, Girish Sastry, Amanda Askell, et al. 2020. Language models are few-shot learners. *Advances in neural information processing systems*, 33:1877–1901.

Wei-Lin Chiang, Zhuohan Li, Zi Lin, Ying Sheng, Zhanghao Wu, Hao Zhang, Lianmin Zheng, Siyuan Zhuang, Yonghao Zhuang, Joseph E Gonzalez, et al. 2023. Vicuna: An open-source chatbot impressing gpt-4 with 90%* chatgpt quality. *See https://vicuna. lmsys. org (accessed 14 April 2023)*.

Roi Cohen, May Hamri, Mor Geva, and Amir Globerson. 2023. Lm vs lm: Detecting factual errors via cross examination. *arXiv preprint arXiv:2305.13281*.

W Dai, J Li, D Li, AMH Tiong, J Zhao, W Wang, B Li, P Fung, and S Hoi. Instructblip: Towards general-purpose vision-language models with instruction tuning. arxiv 2023. *arXiv preprint arXiv:2305.06500*.

Wenliang Dai, Junnan Li, Dongxu Li, Anthony Meng Huat Tiong, Junqi Zhao, Weisheng Wang, Boyang Li, Pascale Fung, and Steven Hoi. 2023. Instructblip: Towards general-purpose vision-language models with instruction tuning. *arXiv preprint arXiv:2305.06500*.

Chaoyou Fu, Peixian Chen, Yunhang Shen, Yulei Qin, Mengdan Zhang, Xu Lin, Jinrui Yang, Xiawu Zheng, Ke Li, Xing Sun, et al. 2023. Mme: A comprehensive evaluation benchmark for multimodal large language models. *arXiv preprint arXiv:2306.13394*.

Anisha Gunjal, Jihan Yin, and Erhan Bas. 2023. Detecting and preventing hallucinations in large vision language models. *arXiv preprint arXiv:2308.06394*.

Qidong Huang, Xiaoyi Dong, Pan Zhang, Bin Wang, Conghui He, Jiaqi Wang, Dahua Lin, Weiming Zhang, and Nenghai Yu. 2023. Opera: Alleviating hallucination in multi-modal large language models via over-trust penalty and retrospection-allocation. *arXiv preprint arXiv:2311.17911*.

Lorenz Kuhn, Yarin Gal, and Sebastian Farquhar. 2022. Semantic uncertainty: Linguistic invariances for uncertainty estimation in natural language generation. In *The Eleventh International Conference on Learning Representations*.

Seongyun Lee, Sue Hyun Park, Yongrae Jo, and Minjoon Seo. 2023. Volcano: mitigating multimodal hallucination through self-feedback guided revision. *arXiv preprint arXiv:2311.07362*.

Sicong Leng, Hang Zhang, Guanzheng Chen, Xin Li, Shijian Lu, Chunyan Miao, and Lidong Bing. 2023. Mitigating object hallucinations in large vision-language models through visual contrastive decoding. *arXiv preprint arXiv:2311.16922*.

Bo Li, Yuanhan Zhang, Liangyu Chen, Jinghao Wang, Jingkang Yang, and Ziwei Liu. 2023. Otter: A multi-modal model with in-context instruction tuning. *arXiv preprint arXiv:2305.03726*.

Zhen Lin, Shubhendu Trivedi, and Jimeng Sun. 2023. Generating with confidence: Uncertainty quantification for black-box large language models. *arXiv preprint arXiv:2305.19187*.

Fuxiao Liu, Kevin Lin, Linjie Li, Jianfeng Wang, Yaser Yacoob, and Lijuan Wang. 2024. Mitigating hallucination in large multi-modal models via robust instruction tuning. *ICLR*.

Haotian Liu, Chunyuan Li, Qingyang Wu, and Yong Jae Lee. 2023. Visual instruction tuning. In *NeurIPS*.

Holy Lovenia, Wenliang Dai, Samuel Cahyawijaya, Ziwei Ji, and Pascale Fung. 2023. Negative object presence evaluation (nope) to measure object hallucination in vision-language models. *arXiv preprint arXiv:2310.05338*.

Potsawee Manakul, Adian Liusie, and Mark JF Gales. 2023. Selfcheckgpt: Zero-resource black-box hallucination detection for generative large language models. *arXiv preprint arXiv:2303.08896*.

Long Ouyang, Jeffrey Wu, Xu Jiang, Diogo Almeida, Carroll Wainwright, Pamela Mishkin, Chong Zhang, Sandhini Agarwal, Katarina Slama, Alex Ray, et al. 2022. Training language models to follow instructions with human feedback. *Advances in Neural Information Processing Systems*, 35:27730–27744.

Adam Paszke, Sam Gross, Francisco Massa, Adam Lerer, James Bradbury, Gregory Chanan, Trevor Killeen, Zeming Lin, Natalia Gimelshein, Luca Antiga, et al. 2019. Pytorch: An imperative style, high-performance deep learning library. *Advances in neural information processing systems*, 32.

Hugo Touvron, Thibaut Lavril, Gautier Izacard, Xavier Martinet, Marie-Anne Lachaux, Timothée Lacroix, Baptiste Rozière, Naman Goyal, Eric Hambro, Faisal Azhar, et al. 2023. Llama: Open and efficient foundation language models. *arXiv preprint arXiv:2302.13971*.

Bin Wang, Fan Wu, Xiao Han, Jiahui Peng, Huaping Zhong, Pan Zhang, Xiaoyi Dong, Weijia Li, Wei Li, Jiaqi Wang, et al. 2023. Vigc: Visual instruction generation and correction. *arXiv preprint arXiv:2308.12714*.

Jason Wei, Xuezhi Wang, Dale Schuurmans, Maarten Bosma, Fei Xia, Ed Chi, Quoc V Le, Denny Zhou, et al. 2022. Chain-of-thought prompting elicits reasoning in large language models. *Advances in Neural Information Processing Systems*, 35:24824–24837.

Thomas Wolf, Lysandre Debut, Victor Sanh, Julien Chaumond, Clement Delangue, Anthony Moi, Pierric Cistac, Tim Rault, Rémi Louf, Morgan Funtowicz, et al. 2019. Huggingface's transformers: State-of-the-art natural language processing. *arXiv preprint arXiv:1910.03771*.

Peng Xu, Wenqi Shao, Kaipeng Zhang, Peng Gao, Shuo Liu, Meng Lei, Fanqing Meng, Siyuan Huang, Yu Qiao, and Ping Luo. 2023. Lvlm-ehub: A comprehensive evaluation benchmark for large vision-language models. *arXiv preprint arXiv:2306.09265*.

Qinghao Ye, Haiyang Xu, Guohai Xu, Jiabo Ye, Ming Yan, Yiyang Zhou, Junyang Wang, Anwen Hu, Pengcheng Shi, Yaya Shi, et al. 2023. mplug-owl: Modularization empowers large language models with multimodality. *arXiv preprint arXiv:2304.14178*.

Kun Zhou Jinpeng Wang Wayne Xin Zhao Yifan Li, Yifan Du and Ji-Rong Wen. 2023. Evaluating object hallucination in large vision-language models. In *The 2023 Conference on Empirical Methods in Natural Language Processing*.

Shukang Yin, Chaoyou Fu, Sirui Zhao, Tong Xu, Hao Wang, Dianbo Sui, Yunhang Shen, Ke Li, Xing Sun, and Enhong Chen. 2023. Woodpecker: Hallucination correction for multimodal large language models. *arXiv preprint arXiv:2310.16045*.

Wayne Xin Zhao, Kun Zhou, Junyi Li, Tianyi Tang, Xiaolei Wang, Yupeng Hou, Yingqian Min, Beichen Zhang, Junjie Zhang, Zican Dong, et al. 2023. A survey of large language models. *arXiv preprint arXiv:2303.18223*.

Yiyang Zhou, Chenhang Cui, Jaehong Yoon, Linjun Zhang, Zhun Deng, Chelsea Finn, Mohit Bansal, and Huaxiu Yao. 2024. Analyzing and mitigating object hallucination in large vision-language models. *ICLR*.

Deyao Zhu, Jun Chen, Xiaoqian Shen, Xiang Li, and Mohamed Elhoseiny. 2023. Minigpt-4: Enhancing vision-language understanding with advanced large language models. *arXiv preprint arXiv:2304.10592*.

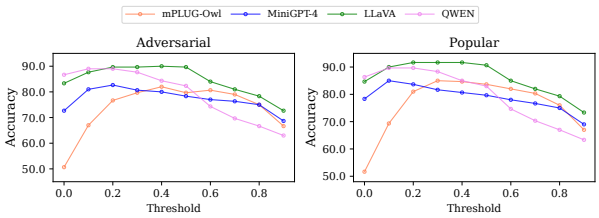


Figure 5: The performance of different threshold λ.

A Implementation Details

In this work, LogicCheckGPT was constructed utilizing the PyTorch framework (Paszke et al., 2019), incorporating capabilities from HuggingFace's Transformers library (Wolf et al., 2019). The large language model (LLM) we adopted is GPT-3.5-turbo, to help fulfill each subtask of hallucination mitigation. All base LVLMs and hallucination mitigation methods are re-implemented according to their literature. We maintain the default hyperparameter settings for all backbone LVLMs and baselines. The experiments were conducted using an NVIDIA A100 GPU and an AMD EPYC 7763 CPU. The hallucination threshold λ is searched within $[0.0, 0.9]$.

B Hyperparameter Analysis

In this section, we conduct experiments on POPE under adversarial and popular setting to analyze the performance fluctuation of LogicCheckGPT with different values of threshold λ ranging from 0.0 to 0.9, as shown in Figure 5. There is a significant increase when threshold λ is increasing from 0.0 initially for all models with LogicCheckGPT under both settings. It indicates that our method can significantly distinguish existent objects and hallucinated objects by valid threshold, as models tend to provide logically consistent responses for existent objects.

Subsequently, these these models reach their respective performance peaks. For instance, the performance of mPLUG-Owl achieves the best when λ is 0.3 under both settings, while the best λ for LLaVA is 0.5. For MiniGPT-4, the λ at which it reached its performance peak varied across different settings, 0.2 and 0.1 respectively. With the continued increase of the threshold, a noticeable decrease in performance can be observed. This is reasonable, as it leads to the misclassification of a large number of existent objects as non-existent.

C Prompts

C.1 Object Extraction

The prompt for object extraction is illustrated in Fig. 6.

C.2 Attribute-to-Object Inquiring

C.2.1 Attribute Extraction

The prompt for attribute extraction in attribute-to-object inquiring is illustrated in Fig. 7.

C.2.2 Question Formulation

The prompt for question formulation in attribute-to-object inquiring is illustrated in Fig. 8.

C.3 Logical Closed Loop Checking

The prompt for logical closed loop checking is illustrated in Fig. 9.

C.4 Hallucination Detection and Mitigation

The prompt for hallucination dection and mitigation is illustrated in Fig. 10.

C.5 GPT-4V Assisted Evaluation

The prompt for hallucination dection and mitigation is illustrated in Fig. 11.

System prompt

You are a language assistant that helps to extract information from given sentences.

Prompt

You are given a sentence, extract the entities within the sentence for me.

[Task]
Your task is to extract the common objects and summarize them as general categories without repetition, merging essentially similar objects. Avoid extracting abstract or non-specific entities. Extract entity in the singular form. Output all the extracted types of items in one line and separate each object type with a period. If there is nothing to output, then output a single "None". DO NOT RESPOND WITH ANYTHING ELSE.

Here are some examples:
`{In-context examples}`

Now complete the following:

[Sentence]
`{sentence}`

[Response]

Figure 6: Prompt template of object extraction.

System prompt

You are a language assistant that helps to extract information from given sentences.

Prompt

You will receive a piece of text that describes an object, and the given object.

[Task]
Your task is to accurately identify and extract every attribute associated with the given object in the provided text. Each claim should be concise (less than 15 words) and self-contained, corresponding to only one attribute. You MUST only respond in the format as required. Each line should contain the original claim and the modified claim with all the mentions of the given object being replaced with "the object". DO NOT RESPOND WITH ANYTHING ELSE. ADDING ANY OTHER EXTRA NOTES THAT VIOLATE THE RESPONSE FORMAT IS BANNED.

[Response Format]
original claim and modified claim

Here are some examples:
`{In-context examples}`

Now complete the following:

[Text]
`{sentence}`

[Entity]
`{entity}`

[Response]

Figure 7: Prompt template of attribute-to-object question (1).

System prompt

You are a language assistant that helps to extract information from given sentences.

Prompt

You will receive a list of statements of objects in an image.

[Task]
Your task is to rephrase each line of statement into a question following the below question template.
In specific, extract attributes of the object to fill in the attribute slot of the template to form questions.
DO NOT RESPOND WITH ANYTHING ELSE. DO NOT CHANGE THE QUESTION TEMPLATE.

[Response Format]
Could you tell me all the objects that `{ATTRIBUTE SLOT}` in the image?

Here are some examples:
`{In-context examples}`

Now complete the following:

[Statements]
`{statement}`

[Response]

Figure 8: Prompt template of attribute-to-object question (2).

System prompt

You are a language assistant that helps to extract information from given sentences.

Prompt

You are given a statement and a question.

[Task]
Your task is to answer the question based on the statement. The statement is about some objects. The question is to ask whether some specific object exists.
1. Your response should be limited to one of the following two choices: `"Yes"/"No"`.
2. Note that instances of a certain category can also belong to its super-categories. For example, a baseball is a subclass of the sports ball.
3. Note that the table is equivalent to the dining table here.
4. DO NOT RESPOND WITH ANYTHING ELSE.

[Response Format]
`Yes/No`

Here are some examples:
`{In-context examples}`

Now complete the following:

[Statement]
`{statement}`

[Question]
Is there a `{object}` in the statement?

[Response]

Figure 9: Prompt template of logic closed loop check.

System prompt

You are a language assistant that helps to extract information from given sentences.

Prompt

You are given a query, a passage and supplementary information.

[Task]
You are required to correct and output the refined passage in a fluent and natural style, following these rules:
1. Correct the sentences in the passage if they are inconsistent with the supplementary information.
2. Do not modify correct sentences and introduce additional information.
3. When giving refined passage, also pay attention to the given query. The refined passage should be reasonable answers to the query.
4. Note the dining table is equivalent to the table.
Output only the corrected passage, without introducing extra contents.

Here are some examples:
`{In-context examples}`

Now complete the following:

[Query]
`{query}`

[Passage]
`{passage}`

[Supplementary Information]
`{supplementary_information}`

[Response]

Figure 10: Prompt template of refinement.

System prompt

You are required to score the performance of two AI assistants in describing a given image.

Prompt

You should pay extra attention to the hallucination, which refers to the part of descriptions that are inconsistent with the image content, such as claiming the existence of something not present in the image or describing incorrectly in terms of the counts, positions, or colors of objects in the image. Please rate the responses of the assistants on a scale of 1 to 10, where a higher score indicates better performance, according to the following criteria:

1: Accuracy:
whether the response is accurate with respect to the image content. Responses with fewer hallucinations should be given higher scores.

2: Relevancy:
whether the response directly follows the instruction.

Please output the scores for each criterion, containing only two values indicating the scores for Assistant 1 and 2, respectively. The two scores are separated by a space. Following the scores, please provide an explanation of your evaluation, avoiding any potential bias and ensuring that the order in which the responses were presented does not affect your judgment.

[Assistant 1]
`{answer_1}`
[End of Assistant 1]

[Assistant 2]
`{answer_2}`
[End of Assistant 2]

Output format:

Accuracy: `<Scores of the two answers>`
Reason:

Relevancy: `<Scores of the two answers>`
Reason:

Figure 11: Prompt template of GPT-4v Assisted Evaluation.