

Supplemental Material

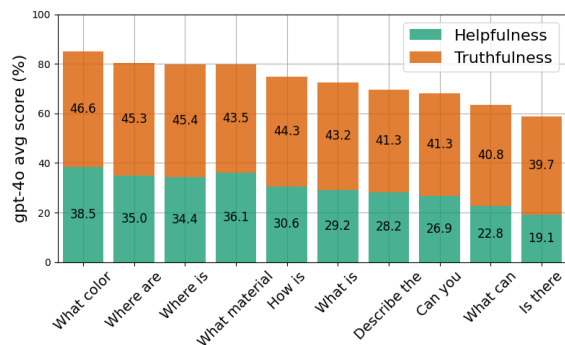


Figure 1. GPT-4o performance by question type on PROVE.

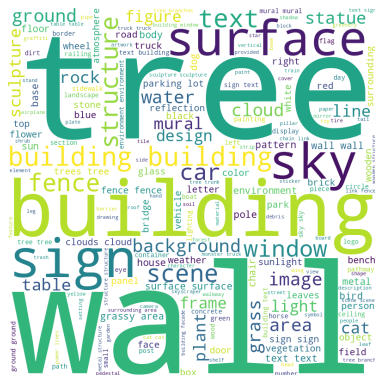


Figure 2. Word cloud of hallucinated objects in model-generated responses.

A. Additional implementation details

Tab. 1 provides a Python implementation of the SceneGraph class used to represent scene graphs in PROVE. The class provides methods to generate subgraphs, describe subgraphs in natural language, and query entities, attributes, and relationships. We include the prompts used for generating verifiable question-answer pairs as well as the post-processing prompt used to filter out low-quality QA pairs in Figs. 3-5.

B. Additional performance analysis

Fig. 1 presents a fine-grained performance analysis of

Consider the SceneGraph class defined below, which takes as input an image caption and a dictionary of tuples (entities, attributes, and relations) parsed from the image caption, and builds a directed graph representation with attributed entities as nodes and relations as edges.

You will be provided with such an image caption, tuple dictionary, and corresponding SceneGraph object. Your task is to generate a set of:

1. **Free-form question-answer pairs** that test non-trivial image understanding and reasoning capabilities.
2. **A Python function** that receives as input the SceneGraph object and can be executed to answer the query by reasoning over the scene graph.

Guidelines. The generated questions should be:

- **Clear and conversational**, in the tone of a person who is asking another person about the scene. You may paraphrase where appropriate to improve clarity (eg. “Can you describe the dog?” is better than “What is the state of the dog?”). Note that the tuples in the scene graph are generally accurate but not necessarily precise, and so may require rephrasing to generate meaningful questions from.
- **Diverse**, both in question type (e.g. starting with “is”, “where”, “what”, “when”, “how”, “which”, “why”, etc.) and length.
- **Non-trivial** (eg. avoid “What color are the green trees?”) and **unambiguous** (eg. avoid “What is the color of the puppy?” for an image with multiple puppies).

The generated Python functions should be:

- **Executable:** The code should run without requiring modifications.
- **General:** The code should generalize to similar scene graphs as the one provided. Do NOT hard-code specific attributes or relations.

For each image, generate 10-15 such question-answer pairs and corresponding Python functions.

Figure 3. LLM prompt for generating visual question-answer pairs along-with verification programs.

GPT-4o on PROVE. We break down helpfulness and truthful-

```

1 #####
2 # SceneGraph class API
3 #####
4 class SceneGraph(nx.DiGraph):
5     def __init__(self, caption, sg_dict, *args, **kwargs):
6         """Init scene graph from entity-attribute-relationship dict"""
7         super().__init__(*args, **kwargs)
8         for source_ent, metadata in sg_dict.items():
9             self.add_node(source_ent, **metadata["attributes"])
10            for target_ent, rel_info in metadata["relations_to"].items():
11                self.add_edge(source_ent, target_ent, **rel_info)
12            self.caption = caption
13            self.sg_dict = sg_dict
14
15        def generate_subgraph(self, node_list) -> "SceneGraph":
16            """Generates a subgraph with nodes in node_list"""
17            return nx.subgraph(self, node_list)
18
19        def describe(self, subgraph): -> str:
20            """Generate a natural language description of a subgraph"""
21            return generate_description(subgraph)
22
23        def get_entities(self) -> List[str]:
24            """Returns a list of entities in the scene graph."""
25            return list(self.nodes)
26
27        def get_attributes(self, ent_name) -> dict[List]:
28            """
29            Returns a list of attributes for ent_name in the scene graph
30            Format: { "att_type": f"att_value_1, att_value_2, ..." }
31            """
32            return self.nodes.get(ent_name, {})
33
34        def get_outgoing_relations(self, ent_name) -> dict:
35            """
36            Returns a dict of relations for which ent_name is the source.
37            Format: {target_ent_1: { rel_type_1: [rel_val_1, ...]} ... }
38            """
39            out_edges = list(self.out_edges(ent_name, data=True))
40            out_edges = { tup[1]: {**tup[2]} for tup in out_edges }
41            return out_edges
42
43        def get_incoming_relations(self, ent_name) -> dict:
44            """
45            Returns a dict of relations for which ent_name is the target
46            Format: {source_ent_1: { rel_type_1: [rel_val_1, ...]} ...}
47            """
48            in_edges = list(self.in_edges(ent_name, data=True))
49            in_edges = { tup[0]: {**tup[2]} for tup in in_edges }
50            return in_edges
51
52

```

Listing 1. Python API for the SceneGraph class

ness scores by question type, and display the top-10 most common question types sorted by performance. As seen, the model performs particularly well on questions that require reasoning about spatial relationships (where are/is), object attributes (what color), and generating image descriptions.

Fig. 2 shows a word cloud of the most commonly hallucinated objects in answers to questions from PROVE across all models. As seen, models commonly hallucinate common

objects such as “tree”, “building”, “wall”, and “sign”.

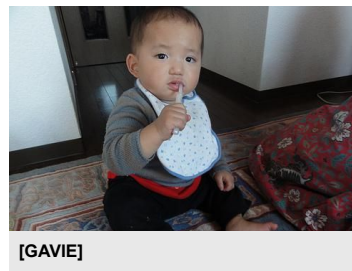
C. Additional comparison with existing benchmarks

In Figure 4 we include real examples from existing hallucination benchmarks that qualitatively highlight the limitations of LLM judges in open-ended response evaluation.

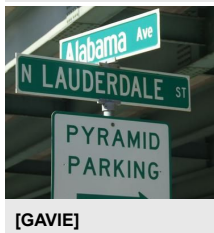
Ambiguous / Incorrect
Question or Answer



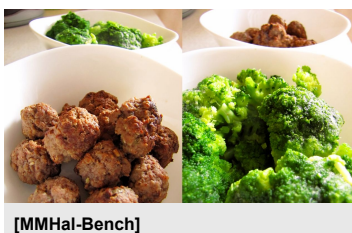
Which sheep is in front of the other sheep?
GT Answer: White sheep



Notice the bookshelf in the corner of the room filled with children's books.
GT Answer: There is no mention of a bookshelf or children's books in the image description, making this instruction misleading and unhelpful.



Is there a particular facial expression seen in any of the signs in the image?
GT Answer: No facial expressions are present.



Is the bowl with broccoli on top of the bowl with meatballs?
GT Answer: No, the bowl with broccoli is next to the bowl of meatballs

Insufficient image
context for LLM as judge



Describe the person playing tennis in this photo.

VLM: There is no person visible in the photo. The image only shows a tennis racket and a tennis ball on a tennis court. ... The racket and ball are positioned on the left side of the image, with the ball resting on the strings of the racket. The court appears to be a hard surface court, as indicated by the green color and white boundary lines. ...

GPT-4 Judge: The LMM's response is somewhat informative, but it contains a hallucination. The response provides a detailed description of the tennis racket and ball, as well as the court surface. However, it hallucinates the position of the ball, claiming it is resting on the strings of the racket, which is not mentioned in the image contents. Rating: 1/5, somewhat informative, with hallucination



What might be the intended effect of this painting?

VLM: The intended effect of this painting is to create a humorous and whimsical scene by combining a portrait of a dog with a human-like appearance, wearing a Renaissance style dress and a hat. The artist has painted the dog with a human-like face, complete with a nose and a mouth, and has given it a Renaissance style dress, which adds a touch of irony and humor to the scene. The painting likely aims to entertain viewers and provoke amusement by combining the innocence and playfulness of a dog with the sophistication and elegance of a Renaissance portrait.

GPT-4 judge rating: 4/10

Figure 4. **Failure modes of existing VLM Evaluation Benchmarks.** Existing free-form VLM evaluation benchmarks [? ? ?] frequently contain ambiguous, incorrect, or unanswerable questions, often due to LLM generation without additional verification. Further, these benchmarks typically rely on pure LLM-as-judge evaluation of free-form VLM responses at test-time, which often leads to unreliable scoring due to the LLM judge having insufficient image context or arbitrary scoring due to the lack of a scoring rubric.

You will be provided with a list of question-answer pairs about an image. Your task is to identify whether each pair has any of the following issues:

1. **Trivial question.** A trivial question can be answered directly from information provided in the question or using common-sense, without requiring looking at the image. Examples:

Question: What is the material of the stadium's horizontal concrete bar?

Answer: The stadium's horizontal concrete bar is made of concrete.

Judgement: Trivial (The question already mentions that the bar is made of concrete)

Question: What text rendering is found on the stop sign?

Answer: The stop sign has white text rendering of the word "STOP".

Judgement: Trivial (Stop signs almost always have the word "STOP" on them)

Question: What feature of the scene reflects sunlight?

Answer: The hard surfaces reflect sunlight.

Judgement: Trivial (Hard surfaces are known to reflect sunlight)

2. **Incomplete answer.** An incomplete answer does not completely answer the question. It may be missing key details or may not provide a full description, or may also be entirely irrelevant. Examples:

Question: What is between the red neon light and the frame?

Answer: The red neon light is behind the metal construction frame.

Judgement: Incomplete (does not answer the question)

Question: How would you describe the trees surrounding the green lake?

Answer: The trees surrounding the green lake are large in size.

Judgement: Incomplete ("large" is not a sufficiently detailed description)

Question: In which part of the image is there no visible cloud coverage?

Answer: The rest of the image has the clear blue sky with no visible cloud coverage.

Judgement: Incomplete ("the rest of the image" is meaningless without context)

3. **Unnatural-sounding.** The question-answer pair may sound awkward, ambiguous, or unnatural. This could be due to its phrasing, structure, or grammar. Examples:

Question: Can you describe the role of the stones in relation to the anemones?

Answer: The stones are covered with anemones and line the bottom of the tank and go up its left side.

Judgement: Unnatural ("role" of stones is odd phrasing and the overall question is ambiguous)

Question: What is the overall shape of the section of grass?

Answer: The section of grass is small in shape.

Judgement: Unnatural ("small" is not a shape)

Question: What kind of state is the sign experiencing due to the brightness?

Answer: Due to the brightness, the sign is experiencing a duller state.

Judgement: Unnatural ("experiencing a state" is awkward phrasing)

Figure 5. LLM text-based post-processing prompt.