Ground-V: Teaching VLMs to Ground Complex Instructions in Pixels

Supplementary Material

A. Implementation Details

A.1. Datasets

For a fair comparison, we adopt datasets originally used in LISA [13] and PSALM [53] for training. They are listed in Table 9 and 10 respectively.

Task	Datasets
Semantic Segmentation	ADE20K [54], COCO-Stuff [3], PACO-LVIS [34], PartImageNet [11]
Referring Segmentation	refCOCO/+/g [28, 50], refCLEF [17]
VQA	LLaVA Instruct-150k [20]
Reasoning Segmentation	ReasonSeg [13]

Table 9. Training datasets of LISA.

Task	Datasets
Generic Segmentation	COCO-Panoptic [18]
Referring Segmentation	refCOCO/+/g [28, 50]
VQA	LLaVA-v1.5-665k [21]

Table 10. Training datasets of PSALM.

A.2. Training Configurations

We follow the official implementation and training configurations of LISA[5] and PSALM[6] for training. The hyper-parameters of LISA and PSALM training are listed in Table 11 and 12 respectively. For ablation experiments in Section B.2, we train models for 3 epochs and keep the other settings the same. All models are trained on a single 8×A100 40GB machine.

B. Experiments

B.1. Evaluation on ReasonSeg and MUSE

We further evaluate the reasoning-based segmentation benchmarks ReasonSeg [13] and MUSE [39] using LISA and LISA-G5 trained on Ground-V. The test set results are presented in Table 13 and Table 14. Models trained on Ground-Vachieve a 1.8% gIoU improvement on ReasonSeg and a 4.9% gIoU improvement on MUSE. Notably, the evaluation on MUSE is conducted in a zero-shot setting, and the observed improvement highlights the effectiveness of training on Ground-V, particularly the multi-object and reasoning subsets, as MUSE involves scenarios with multiple objects.

[5]https://github.com/dvlab-research/LISA

[6]https://github.com/zamling/PSALM

Parameters	Value
Optimizer	AdamW [26]
Learning Rate	3×10^{-4}
Batch Size Per GPU	10
Gradient Accumulation Steps	2
Number of Epochs	10
Learning Rate Schedule	WarmupDecayLR
Weight Decay	0.0
Warmup Ratio	0.03
LoRA α	64
Image Size	224×224

Table 11. Training hyper-parameters for LISA.

Parameters	Value
Optimizer	AdamW [26]
Learning Rate	4×10^{-5}
Batch Size Per GPU	8
Gradient Accumulation Steps	1
Number of Epochs	10
Learning Rate Schedule	Cosine Decay
Weight Decay	0.0
Warmup Ratio	0.03
β_1	0.9
β_2	0.999
Image Size	1024×1024

Table 12. Training hyper-parameters for PSALM.

Models	gIoU	cIoU
LISA	50.3	49.8
LISA-G5	52.1 (1.8↑)	51.9 (2.1↑)

Table 13. Performance comparison on ReasonSeg test set.

Models	gIoU	cIoU
LISA	12.6	27.2
LISA-G5	17.5 (4.9↑)	30.4 (3.2↑)

Table 14. Performance comparison on MUSE test set (zero-shot).

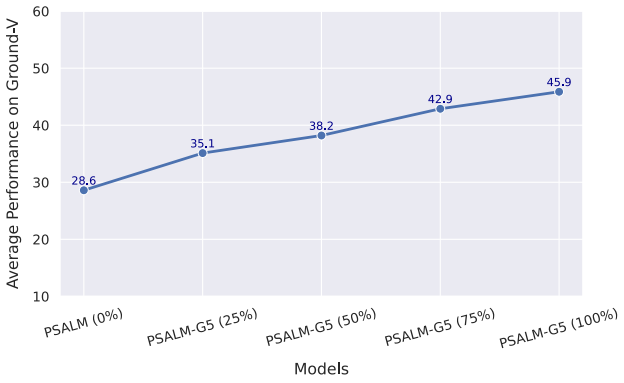

Figure 5. Performance of models trained with different proportions of Ground-V. The average performance is calculated by averaging the gIoU of all subsets.

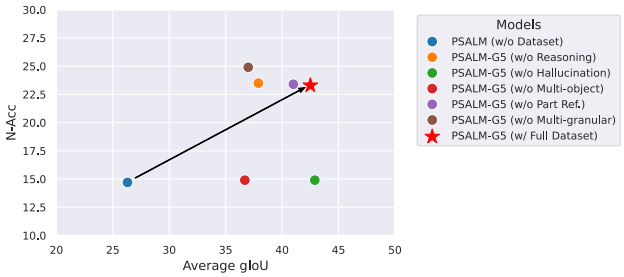

Figure 6. Performance of models trained on different subsets. X-axis measures how accurately the models can segment the objects and Y-axis measures how well the models can abstain the non-existing objects. The model trained on full Ground-V is on the Pareto curve and obtains significantly improved abilities to handle both scenarios.

B.2. Ablations on Data Components

We conduct two ablation studies to further understand the introduced Ground-V. We conduct experiments with the PSALM model. Note that we only train 3 epochs for ablation studies and thus the performance is not comparable with the performance in the main text.

Scaling of the data samples We first examines the impact of dataset scaling on model performance. Models are trained on subsets comprising 25%, 50%, 75%, and 100% of the data to assess the effectiveness of varying data volumes. The trained models are assessed on the Ground-V test set, with the average performance across different subsets presented in Figure5, and detailed results provided in Table 15. As shown in Figure 5, increasing the amount of data used during training consistently improves evaluation performance, highlighting the effectiveness of our dataset.

Contribution of different subsets Next we explore the importance of different subsets within Ground-V. This is achieved by training five models, each with one subset removed, alongside models trained with and without the full Ground-V, to assess their contributions to overall performance. As shown in Figure 6, the model trained on the full Ground-V lies on the Pareto curve, demonstrating significantly improved capabilities in both accurate segmentation and effective abstention from non-existent objects.

While decreased performance on the corresponding test set when a subset is removed from training is expected, we observe several interesting patterns: (1) Most subsets are mutually beneficial: For example, removing the multi-object subset also degrades performance on the reasoning subset, as the reasoning subset often involves multiple objects, and training on the multi-object subset enhances the model's ability to handle such scenarios—and vice versa. (2) Some subsets can introduce trade-offs: For instance, the performance on the fine-grained subset decreases when hallucination examples are included during training. This suggests that the hallucination subset can occasionally make the model overly cautious, leading to the rejection of existing objects. However, despite this trade-off, including hallucination examples in training remains valuable, as it significantly improves overall performance and helps mitigate hallucination-related errors. The detailed results are shown in Table 16.

C. Dataset Details

C.1. Statistics

We visualize the word frequencies of all subsets of Ground-V in Figure 7 to 11. As it can be seen from these figures, the word distributions of our dataset are highly diverse across subsets.

C.2. Human Annotation Process

The human annotation process was meticulously designed to ensure the creation of high-quality, consistent class labels and accurate segmentation mask annotations for the test partition of Ground-V. To minimize bias, the annotation was carried out by 20 external annotators who were independent of the authors. Prior to the full annotation phase, all annotators participated in training sessions and pilot trials to familiarize themselves with the task. Continuous quality control measures, including spot-checks and feedback loops, were implemented to address corner cases and enhance annotators' understanding of the task.

The annotation interface, illustrated in Figs. 12 to 14, was specifically designed to support human annotations. Each annotation instance included an image, a question, and one or more segmentation masks, each associated with an object name. Annotators were presented with the image, question, and all segmentation masks along with their corresponding object names simultaneously, providing complete

Method	Multi-granular		Multi-object	Hallucination	Reasoning		Part Reference	
	Abstract (gIoU)	**Fine-grained** (gIoU)	gIoU	N-Acc	gIoU	cIoU	gIoU	cIoU
PSALM (0%)	27.3	29.9	17.5	14.7	43.5	41.1	12.3	24.9
PSALM-G5 (25%)	34.1	37.3	23.7	16.9	50.2	50.8	14.6	28.7
PSALM-G5 (50%)	38.6	39.5	25.8	19.7	54.3	53.9	17.1	29.4
PSALM-G5 (75%)	45.9	43.6	28.9	22.5	60.3	59.8	18.8	31.2
PSALM-G5 (100%)	47.8	51.9	30.5	23.3	62.2	61.9	20.9	33.1

Table 15. Performance comparison of models trained with different proportions of Ground-V, evaluated on the disjoint test set of Ground-V.

Method	Multi-granular		Multi-object	Hallucination	Reasoning		Part Reference	
	Abstract (gIoU)	**Fine-grained** (gIoU)	gIoU	N-Acc	gIoU	cIoU	gIoU	cIoU
PSALM (w/o Ground-V)	27.3	29.9	17.5	14.7	43.5	41.1	12.3	24.9
PSALM-G5 (w/o Reasoning)	46.5	50.0	25.4	23.5	44.1	41.9	20.1	33.0
PSALM-G5 (w/o Hallucination)	46.7	**52.6**	29.4	14.9	60.1	61.0	**21.4**	**34.5**
PSALM-G5 (w/o Multi-object)	40.4	48.3	18.2	14.9	52.1	55.3	19.6	33.0
PSALM-G5 (w/o Part Ref.)	**48.0**	52.1	29.8	23.4	61.9	61.4	12.5	25.2
PSALM-G5 (w/o Multi-granular)	34.6	36.7	27.3	**24.9**	62.0	61.7	20.8	32.4
PSALM-G5 (w/ full Ground-V)	47.8	51.9	**30.5**	23.3	**62.2**	**61.9**	20.9	33.1

Table 16. Performance comparison of models trained with different subsets of Ground-V, evaluated on the disjoint test set of Ground-V. We remove one subset at a time for training.

Figure 7. Word Cloud visualizations for training (top) and evaluation (bottom) sets of hallucination mitigation subsets. We show word clouds for object, relation, and attribute hallucination.

contextual information. Annotators examined each segmentation mask individually, following the order of the mask index, and determined whether the object accurately and relevantly answered the question by selecting one of three options: YES, NO, or UNSURE. They adhered to comprehensive guidelines, which covered definitions of the five challenges, examples for clarification, and detailed instructions for resolving ambiguous cases. If annotators were unsure about the correctness of a label (e.g., unfamiliar with a "Maine Coon cat"), they were instructed to select UNSURE by default. Additionally, if any polygon node in a segmentation mask deviated by more than 5% of the object's size from its actual boundaries, they selected NO for segmentation verification. In practice, the 5% rule was applied based on the annotators' judgment and visual assessment. However, all annotators underwent training to ensure their evaluations aligned consistently with the requirements.

Each image-question pair was independently reviewed by two annotators. To maintain consistency and validate the process, periodic random audits were conducted by a third annotator, with different annotators auditing different instances. These audits revealed that when both initial annotators selected YES, there was a 95%+ probability that the third annotator would also select YES. Based on this finding and budget constraints, we used two annotators for each annotation instance. Only instances where both an-

Figure 8. Word Cloud visualizations for training (top) and evaluation (bottom) sets of multi-granular subsets. We show word clouds for coarse-grained, original, fine-grained categories.

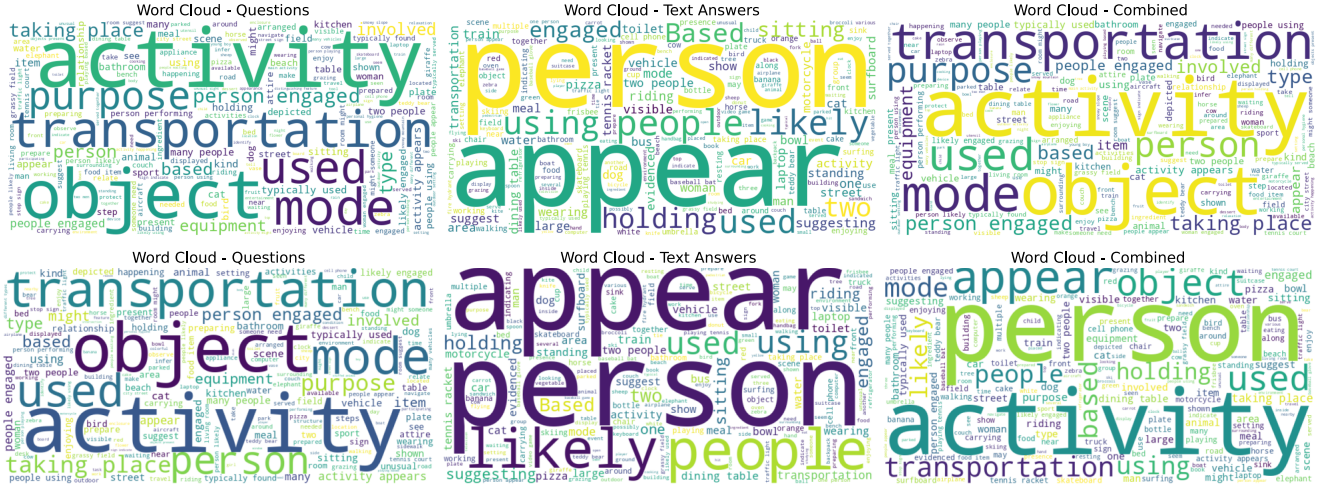

Figure 9. Word Cloud visualizations for training (top) and evaluation (bottom) sets of reasoning subsets. We show word clouds for questions, answers, as well as the combination of both.

Figure 10. Word Cloud visualizations for training (left) and evaluation (right) sets of part reference subsets.

notators agreed on YES were included in the final test set, ensuring high confidence in the data's correctness. As a re-

Figure 11. Word Cloud visualizations for training (left) and evaluation (right) sets of multi-object subsets.

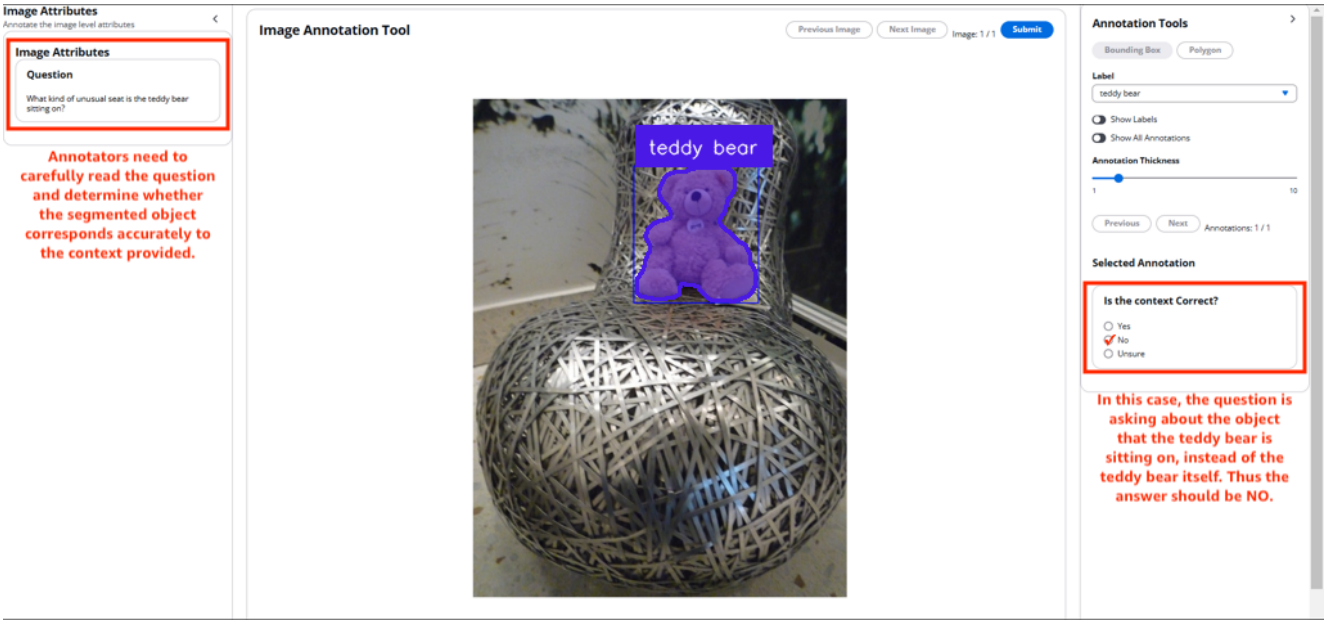

Figure 12. Human annotation interface for Ground-Vtest data that resemble reasoning segmentation. In this case, the question is asking about the object that the teddy bear is sitting on instead of the teddy bear itself. Thus the answer should be NO.

sult, 23.1% of the original test data was excluded due to erroneous or UNSURE labels. The remaining data is used as the test partition of Ground-V.

D. Prompt Details

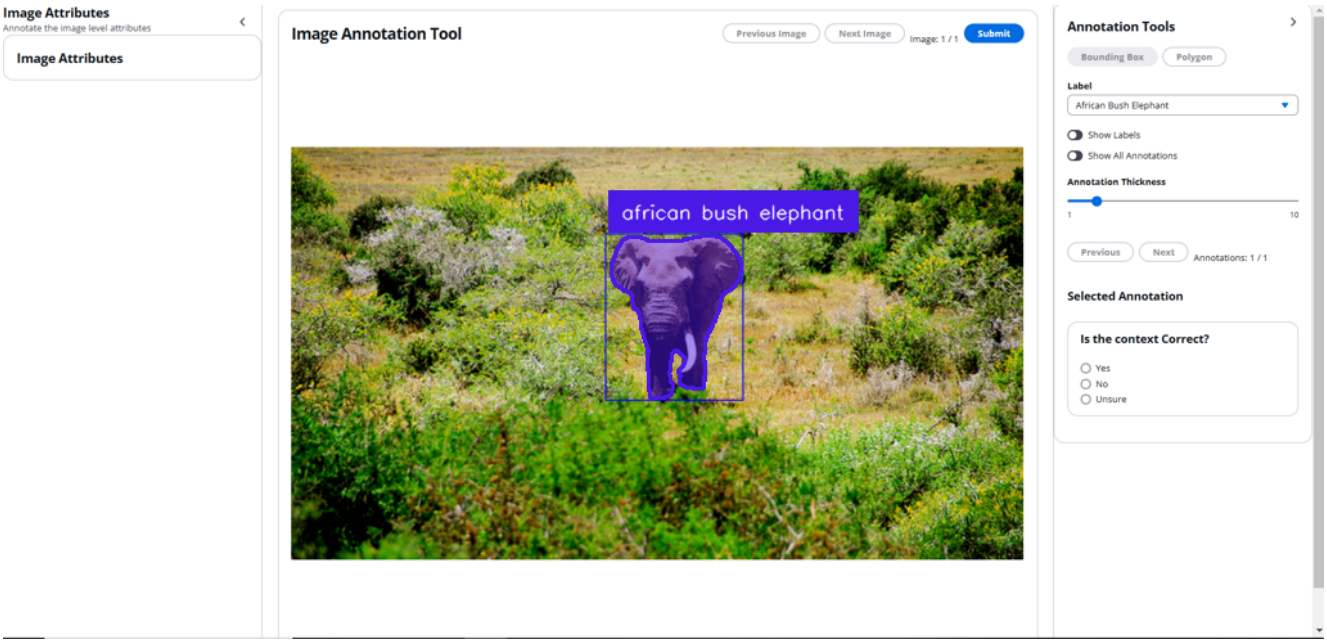

Figure 13. Human annotation interface for Ground-Vtest data that resemble multi-granularity referring. In this example, the label "African Bush Elephant" is more specific than the general label "elephant" that most people would typically use. The data annotator needs to evaluate whether this finer-grained label accurately matches the segmented object.

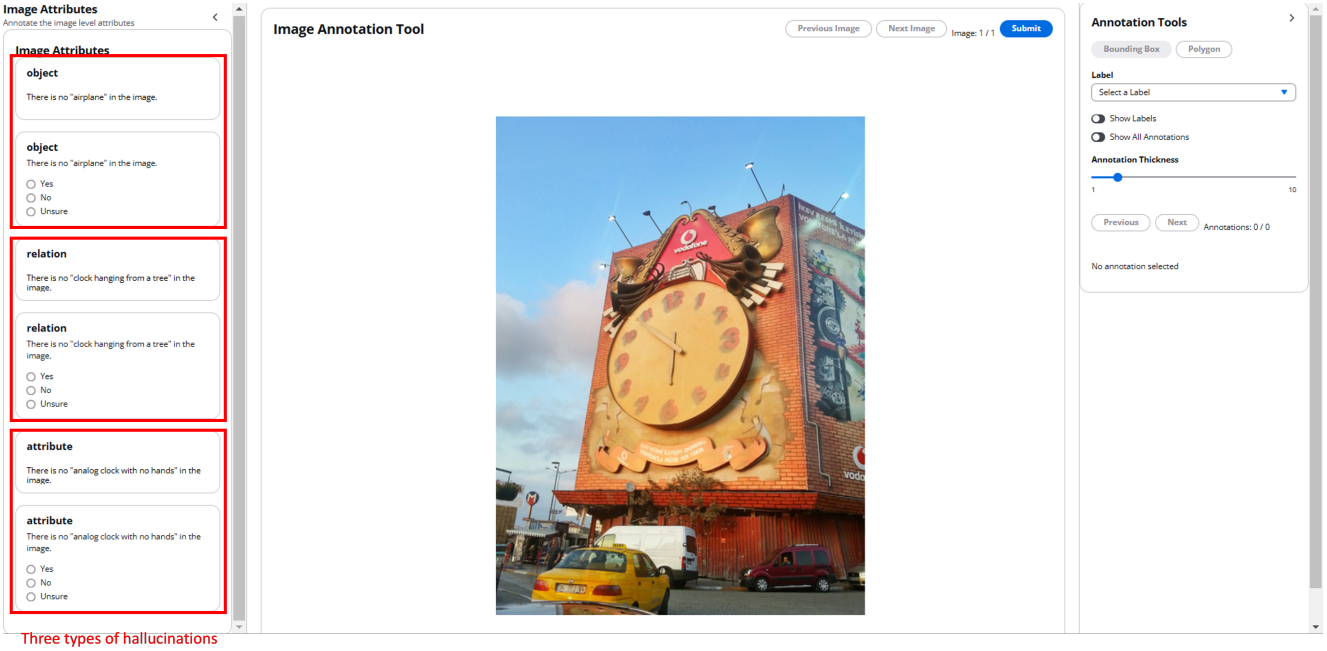

Figure 14. Human annotation interface for Ground-Vtest data that resemble complex referrings that that could lead to hallucinations in segmentation. In this example, the annotator is asked to assess three types of hallucinations for each image: 1) if the mentioned object itself is absent from the image, 2) if the relationship between objects is not present in the image, and 3) if the object's attribute incorrectly assigned or not present in the image. Each hallucination question is answered individually.

D.1. Prompts for Data Curation

We present the detailed prompts used for data curation below.
Prompt for attribute-level hallucination generation

System: You are a helpful assistant who can help with analyze image. You will be provided with an image containing certain objects and a set of object classes. You need to analyze the attributes of the objects and generate an exiting object (from the given classes) with different attribute that is not present in the image.

Few-shot Examples:

Image 1
- Present objects: Lemon.
- Attribute (you need to analyze the image): Yellow lemon.
- Negative sample: Green lemon.

Image 2
- Present objects: person
- Attribute (you need to analyze the image): Person wearing a blue shirt.
- Negative sample: Person wearing a red shirt.
Note: in this case, you need to ensure there is no person wearing red shirt in the image.

Image 3
- Present objects: car, road, person
- Attribute (you need to analyze the image): A shiny silver car parked in a driveway.
- Negative sample: A matte black car.

User: Now for the following image, generate an attribute-level negative sample. Only provide a attribute that is not present in the image, which would be clearly incorrect to request for segmentation. Do not generate reasoning process. Generate a Json dictionary in the format of "attribute": "the generated negative class". Make sure the generated negative sample is realistic. If you think you cannot think of a good example in terms of attribute or it is unrealistic, output "attribute": "Unknown". Generate Json only, strictly no other texts. Be concise in the generated class.

Query Image
- Present objects: [objects]
Assistant:

Prompt for relation-level hallucination generation

System: You are a helpful assistant who can help with analyze image. You will be provided with an image containing certain objects and a set of object classes. Note that the object classes may not be comprehensive and you should also pay attention to the image. You need to analyze the relations between objects and generate a relation that is not present in the image with existing object(s). It can also be spatial relation. Generate the object first, and then generate the relation, i.e., we should clearly know the main object from the generated phase.

Few-shot Examples:

Image 1
- Present objects: child, horse
- Relation (you need to analyze the image): A child sitting beside a sleeping horse.
- Negative sample: Child who is riding the horse.

Image 2
- Present objects: person, person
- Relation (you need to analyze the image): Two people walking in opposite directions
- Negative sample: Two people shaking hands.

Image 3
- Present objects: dog
- Relation (you need to analyze the image): dog on the left.
- Negative sample: dog on the right.

Note: in this case, you need to ensure there is no dog on the right in the image.

User: Now for the following image, generate an relation-level negative sample. Only provide a relation that is not present in the image, which would be clearly incorrect to request for segmentation. Do not generate reasoning process. Generate a Json dictionary in the format of "relation": "the generated negative class". Make sure the generated negative sample is realistic. If you think you cannot think of a good example in terms of relation or it is unrealistic, output "relation": "Unknown". Generate Json only, strictly no other texts. Be concise in the generated class.
Query Image
- Present objects: [objects]

Assistant:

Prompt for object-level hallucination generation

System: You are a helpful assistant who can help with analyze image. You will be provided with an image containing certain objects and a set of object classes. Note that the object classes may not be comprehensive and you should also pay attention to the image. Your task is to generate a negative class that is not present in the image.

Few-shot Examples:

Image 1
- Present objects: Airplane, Cloud, Runway
- Negative sample: Laptop

Image 2
- Present objects: Chair, Laptop, Cup, Plate
- Negative sample: Television

Image 3
- Present objects: boots, businessman
- Negative sample: basketball shoes, businesswoman

User: Now for the following image, generate an relation-level negative sample. Only provide a object that is not present in the image, which would be clearly incorrect to request for segmentation. Do not generate reasoning process. Generate a Json dictionary in the format of "object": "the generated negative class". Make sure the generated negative sample is realistic. If you think you cannot think of a good example in terms of object or it is unrealistic, output "object": "Unknown". Generate Json only, strictly no other texts. Be concise in the generated class. **Query Image**
- Present objects: [objects]
Assistant:

Prompt for multi-granular fine-grained category generation

System: You are a helpful assistant who can help with analyze image. You will be provided with an image containing certain objects and a set of object classes. Your task is to generate corresponding specifc-level class names to the given classes. Important: the number of abstract classes you generated should be strictly same to the number of input classes.
3-Layer Hierarchical Structure:
1. Fine-grained Level:
Combines the specific instance and subcategory levels.
Example: "Corgi," "Macbook," "SUV"
2. General Category Level:
Standard categories of objects.
Example: "Dog," "Computer," "Car"
3. Abstract Level:
Broad, overarching categories.
Example: "Animal," "Electronic Device," "Transportation"

Few-shot Examples:

**Image 1
- Given objects: fish, dog
- fine-grained labels: ["gold fish", "corgi"]

**Image 2
- Present objects: laptop, container, box
- "fine-grained labels": ["Macbook", "mug", "mailer box"]

**Image 3
- Present objects: Car, Road, Tree
- "fine-grained labels": ["Tesla", "Highway", "Oak"]

User: Now for the following image, generate granular-level object class names for the input. Generate a Json dictionary in the format of "fine-grained labels": ["the generated fine-grained labels class name", ...]. Only output the Json dictionary. Strictly no other output. Important: If there're multiple objects belong to the same abstract, be sure to give them the same abstract-level names. Note that the specific level object must present in the image. If you are not sure about the specific level or cannot distinguish, just output "fine-grained labels": "Unknown" only.
Query Image
- Present objects: [objects]
Assistant:

Prompt for reasoning subset generation

System: You are an intelligent chatbot designed to generate question-answer pairs according to the given image and a list of objects, each describing an object in the image you are observing. Your task is to return a question-answer pair where the question requires reasoning and the answer can correctly answer the question.
The provided image has a height of [height] and a width of [width]. The image, image caption, objects in the image, and their respective bounding box coordinates are as follows:
[**image**]
object at [bounding boxes]...
Coordinates represent (top-left x, top-left y, bottom-right x, bottom-right y).
The question must be framed to require image reasoning for a response. Additional requirements for the generated question include:
1. The answer must reference the given object class or its equivalent and should not imply other potential objects.
2. The question should require some reasoning to answer and cannot be too broad.

3. The question should describe a activity or incorporate world knowledge.
4. You will output two forms of answers:
(1) objects: it contains the objects required to answer the question, each object should be in a form of object at [bounding box].
(2) text answer: it should be one or a few coherent sentences connecting the objects in (1) that answer the question. In the sentence, [seg-X] should be used to indicate the objects in (1), and X corresponds to the No.X objects in the list (0-indexed).

Caption: a comfy room with table and sofa, and there is a laptop on the table.
Objects:
table at [120, 50, 240, 90];
pizza at [260, 80, 275, 100];
fork at [160, 30, 170, 35];
Plant at [315, 208, 320, 245]
Question: What steps do I need to take if I want to enjoy my meal?
Object answer: [”pizza at [160, 30, 180, 50]”, ”table at [120, 50, 240, 90]”, ”fork at [160, 30, 170, 35]”]
Text answer: ”You can pick up the metal fork [seg-2] on the wooden table [seg-1] to slice the pizza [seg-0] and then enjoy.”

Image 2:
Caption: a crowded road with bus and taxi.
Objects:
person at [335.52, 359.7, 347.85999999999996, 387.71];
boat at [285, 90, 337, 154];
person at [353.84, 390.06, 371.75, 412.82];
tree at [387.85, 3.42, 415.88, 40.67];
dog at [215, 362, 225.475, 381.84];
Question: What are people taking to reach the other side and how many people are there?
Object answer: [”boat at [285, 90, 337, 154]”, ”person at [335.52, 359.7, 347.85999999999996, 387.71]”, ”person at [353.84, 390.06, 371.75, 412.82]”]
Text answer: They are taking a boat [seg-0] to go to the other side of the river, and there are two persons [seg-1] [seg-2].

Query Image
Question: question
Answer: answer

User: Return the generated question and answer in a Json dictionary only.
Query Image
- Present objects: [objects]...
Assistant: