APPENDIX OVERVIEW

We provide an overview to present a clear understanding of this section.

- In Sec. A, we present visual comparisons of recent VG methods under various instructions.
- In Sec. B and Sec. C, we describe the process of generating expressions from foundation models in both global prompt and local prompt pipelines, respectively.
- In Sec. D, we detail the aspects of visual prompting and visual-textual matching.
- In Sec. E, we explain our approach to designing prompts that enable LLaVA to extract commonalities among multiple objects for summarized expression generation.
- In Sec. F, we discuss our strategy for designing prompts that allow LLaMA to assign generated instructions to our predefined groups.
- In Sec. G, we provide an overview of our model architecture and implementation specifics.
- In Sec. H, we present ablation studies investigating LLaVA fine-tuning in local prompt pipeline and visual prompting selection.
- In Sec. I, we provide the prompt we used in post processing.
- In Sec. J, we add supplementary evaluation results to Table. 2 and Table. 3.
- In Sec. K, we add the time consumption of main steps in our instruction generation procedure.

A VISUAL COMPARISON RESULTS

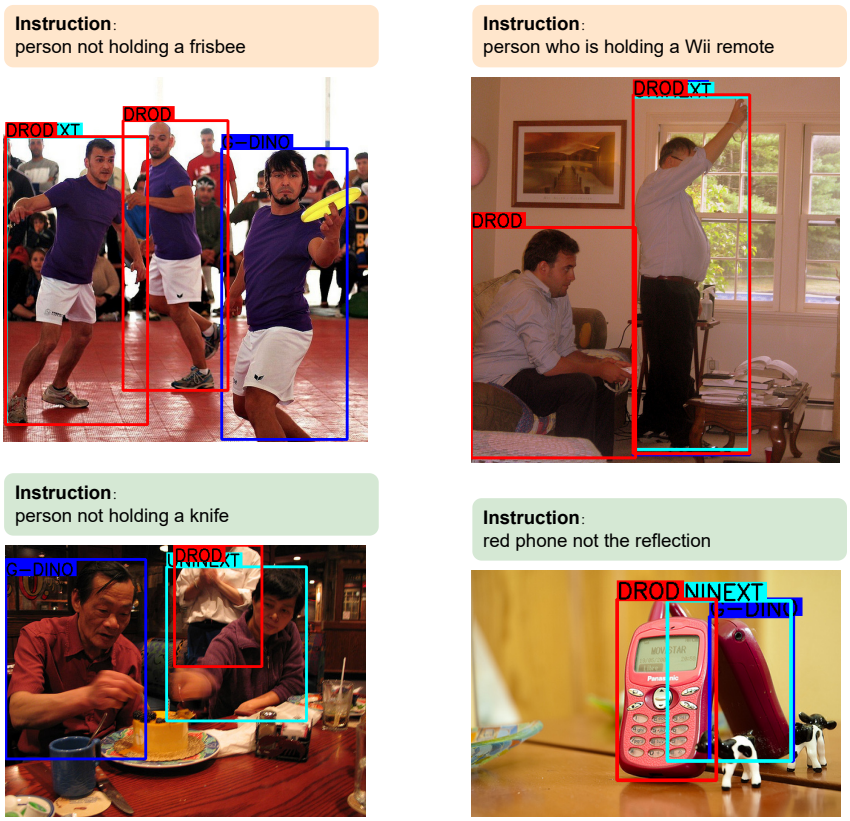

Figure 8: Visual comparison results. Our DROD well executes detection instruction compared to Grounding-DINO and UNINEXT.

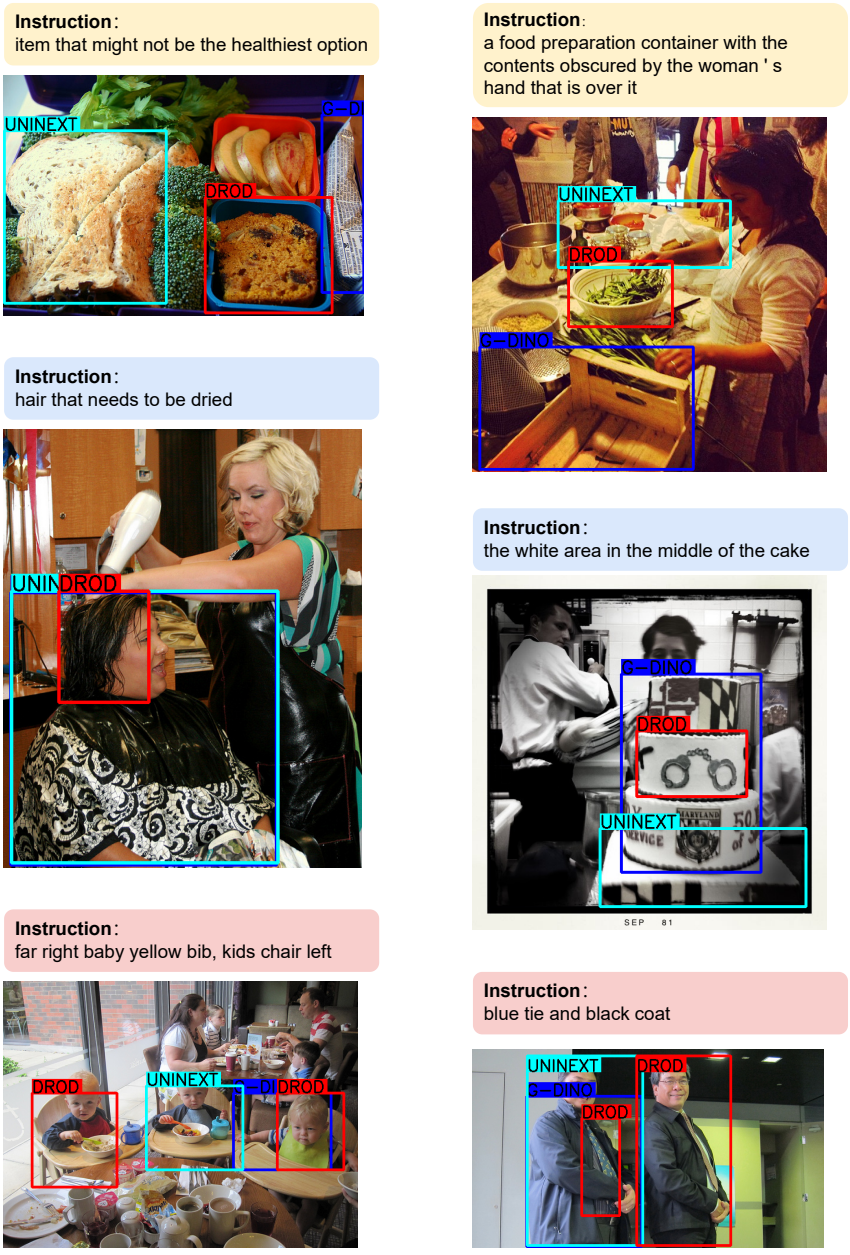

Figure 9: Visual comparison results. Our DROD well executes detection instruction compared to Grounding-DINO and UNINEXT.

B EXPRESSION GENERATION VIA LOCAL PROMPT PIPELINE

B.1 PROMPT AND GENERATION DISPLAY

We show the text prompts we use below in Table 4. For the LLaVA input, We randomly select one text prompt together with one image marked with object bbox. The inputs and samples generated from LLaVA are shown in Fig. 10.

- "Describe the objects in the red box."
- "Take a look at this image and describe What's in the red box."
- "Please provide a description of the object in the red box."
- "Could you describe the contents in the red box of the image for me?"
- "Use one sentence to index the objects in the red box."
- "Output a sentence describing the objects in the red box, so that people can locate the objects without ambiguity through this sentence."
- "Look carefully at the objects in the red box and describe them in one sentence to distinguish them from other objects."

Table 4: Text prompts we use for LLaVA inputs.

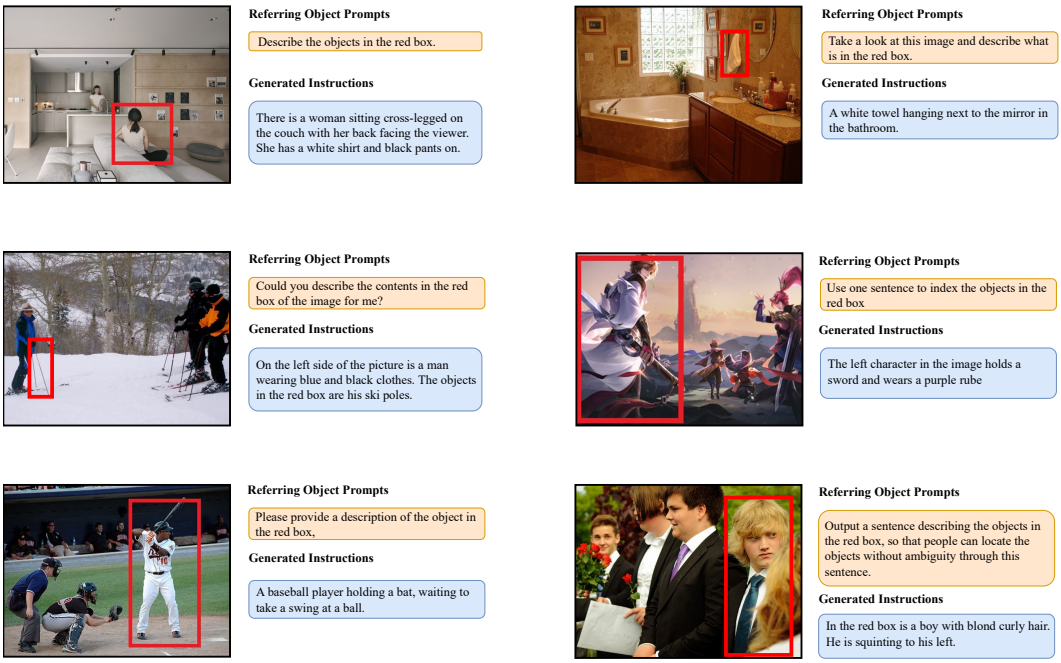

Figure 10: Examples of text prompts, images with marked object bboxs, and LLaVA outputs.

B.2 FINE-TUNING SETTINGS

Dataset Construction We employed the entire dataset from RefCOCO/+/g for our fine-tuning endeavors, which primarily comprises triplets of images, bounding boxes, and instructions. The detection boxes are overlaid onto the images, serving as the visual input for LLAVA, while the instructions from the dataset are used as supervised outputs. The prompts are randomly selected from Table 4. The overall dataset consists of 386k instructions paired with 66k images. We chose not to utilize additional REC datasets for fine-tuning, as our fine-tuning objective is solely directed

towards enabling LLAVA to comprehend our task and acquire the ability to describe targets within red detection boxes.

Training Process Firstly, it is crucial to emphasize that in this paper, LLAVA does not refer to the model presented in the original LLAVA paper. Instead, it broadly denotes a multimodal paradigm where image information is abstracted into several tokens and inputted into a Large Language Model (LLM) for cross-modal attention with textual tokens. Our LLAVA architecture utilizes Q-Former to extract 32 tokens from the VIT features of the image, and these 32 tokens are subsequently mapped to the same feature space as textual tokens through a linear layer. During fine-tuning, only the final mapping linear layer is trained. This fine-tuning approach enables the model to learn the salience of the target object within the red bounding box in the input image while ensuring that the overall model avoids catastrophic forgetting. We initialize our model with the weights of MiniGPT-4.

Hyperparameter Configuration and Implementation Details We employed Vicuna 13B as our Large Language Model (LLM). The batch size and number of epochs were set to 32 and 30, respectively. The initial learning rate was set to 3×10^{-5}. Model training was conducted using 4 threads on a single A100 80GB GPU.

B.3 Chain-of-Thought and Reflection

After fine-tuning, the LLAVA model still exhibits a certain degree of hallucinations, manifesting as the generation of imaginary objects. Taking inspiration from the concept of thought chains in large language models, we manually constructed a thought chain specifically tailored for the task of describing target objects. In Table 5, we illustrate the prompts at each step of the thought chain: 1) Inquire about the category of the target object. 2) Inquire about the attributes possessed by the target object. 3) Inquire about the objects surrounding the target object. 4) Inquire about the relationship between the target object and its surrounding objects. Through these four steps, we can attain a comprehensive understanding of LLAVA's perception of the target object. However, hallucinations may still persist in the thought chain; therefore, in step 5), we prompt the model to reexamine the image and correct any errors. Finally, in step 6), we prompt LLAVA to produce the ultimate description of the target object.

1) "What is the object inside the red bounding box?"
2) "What are the attributes of the object within the red bounding box?"
3) "Which objects are around the target object in the red box?"
4) "What is the relationship between the object inside the red bounding box and the surrounding objects?"
5) "Please review the image once again, and if there are any inaccuracies in your previous answers regarding the object's attributes and relationships, kindly correct them."
6) "Look carefully at the objects in the red box and describe them in one sentence to distinguish them from other objects."

Table 5: Text prompts for guiding chain-of-thought and reflection.

C EXPRESSION GENERATION VIA GLOBAL PROMPT PIPELINE

We choose an image from Objects365 as an example to illustrate the instruction generation pipeline via our global prompt pipeline. This pipeline consists of inputs, two steps, in-context samples, and LLaMA outputs, which are illustrated one-by-one in the following:

Inputs Our input is an example image with object bbox coordinates, which is shown in Fig. 11.

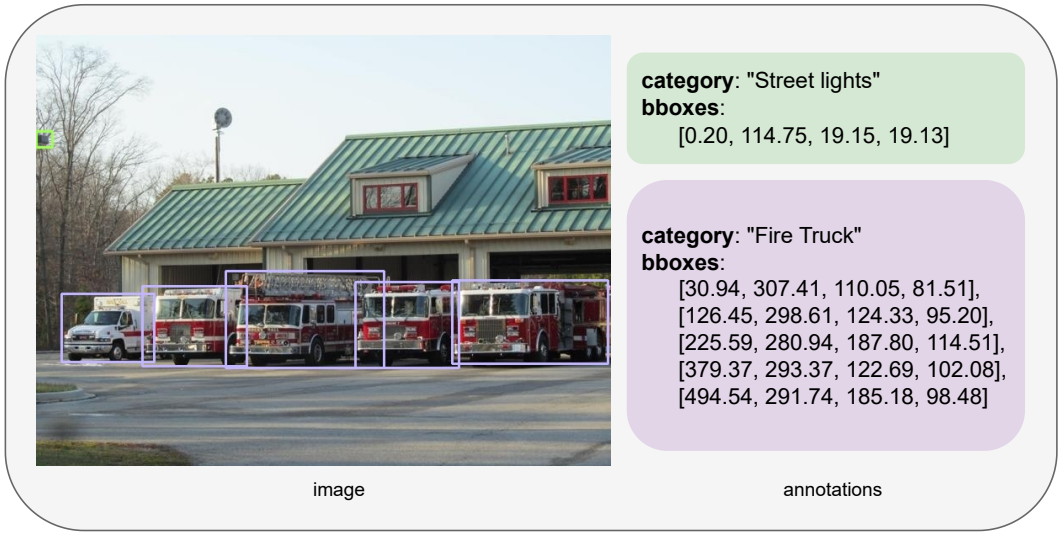

Figure 11: An input example. The left part is an image from the Objects365 dataset. The right part is the object category names and corresponding bounding boxes coordinates.

Output We send the final text prompt shown in Table 9 to LLaMA. Table 6 shows the output expression of the inputs shown in Fig. 11. The detailed steps to prepare text prompts are illustrated as follows:

[Fire Truck]
(1) vehicle, emergency vehicle, fire engine, parked outside the fire station, an essential part of the fire station's resources, essential part of the fire station's resources
(2) lined up in a neat row, ready for use, object parked in the row with other fire trucks, object with ladders and equipment
[Street Lights]
(1) light fixtures, outdoor lighting, two lights visible, providing illumination, source of illumination
(2) objects providing illumination, objects in the parking lot, objects providing a clear representation of the overall setting, objects providing light for the parking lot

Table 6: LLaMA output for the inputs shown in Fig. 11.

Figure 11 and Table 6 depict the original input and final output of the global prompt pipeline, with the intermediary processing steps outlined below. Initially, the generation of the desired caption description for images lacking dense captions is required (**Step 1**). Subsequently, with the image fully textualized, we employ a detailed prompt design and seed example design to instruct LLAMA in outputting instructions describing the target object (**Step 2**).

Step 1 We use LLaVA to generate image captions. Table 7 shows the text prompt list we use. One random text prompt from this list is combined with object content description (i.e., referring expression or category name) for LLaVA input. For each image, we send these prompts into LLaVA twice and obtain two diverse text descriptions. These two text descriptions are both utilized in the following steps. Examples of generated text descriptions are shown in Table 8.

- "Describe the following image in detail"
- "Provide a detailed description of the given image"
- "Give an elaborate explanation of the image you see"
- "Share a comprehensive rundown of the presented image"
- "Offer a thorough analysis of the image"
- "Explain the various aspects of the image before you",
- "Clarify the contents of the displayed image with great detail"
- "Characterize the image using a well-detailed description"
- "Break down the elements of the image in a detailed manner"
- "Walk through the important details of the image"
- "Portray the image with a rich, descriptive narrative"
- "Narrate the contents of the image with precision"
- "Analyze the image in a comprehensive and detailed manner"
- "Illustrate the image through a descriptive explanation"
- "Examine the image closely and share its details"
- "Write an exhaustive depiction of the given image"

Table 7: Text prompts we use for LLaVA. These prompts follow the author usage in the original LLaVA work.

Prompt for LLaVA
Provide a detailed description of the given image, including objects: *Street lights*, *Fire Truck*

Output from LLaVA
Output 1st The image displays a parking lot outside a fire station where several fire trucks are parked. In total, there are five fire trucks of varying sizes, all lined up neatly in the lot. Additionally, there are two street lights visible in the scene, providing illumination for the area. The fire trucks appear ready for use and are an bessential part of the fire station's resources.
Output 2nd The image depicts a parking lot outside a fire station, where several fire trucks are parked in a neat row. There are a total of five fire trucks of varying sizes, all aligned and ready for use. In addition to the fire trucks, there are two street lights visible in the scene, providing illumination for the parking lot. The overall setting gives a clear representation of the organized and prepared nature of the fire station.

Table 8: Generated examples of LLaVA based on one random text prompt from Table 7.

Step 2 We write text prompt for LLaMA. In this step, the text prompt contains three parts including task descriptions, in-context examples, and text description of image (i.e., Table 8). An example text prompt for LLaMA is shown in Table 9 where an in-context sample is shown in Table 10.

```
task_description = f"""
```

 # Task description prompt

Establishing fundamental roles and task localization for LLM.
You are an AI visual assistant that can analyze a single image.
Elucidating the form and significance of input information.
User will give you several sentences, describing the same image you are observing. In addition, specific interested object locations within the image are given, along with detailed coordinates. These coordinates are in the form of bounding boxes, represented as (x1, y1, x2, y2) with floating numbers ranging from 0 to 1. These values correspond to the left top x, left top y, right bottom x, and right bottom y.
Explicating the output content and the encapsulated information.
Using the provided caption and bounding box information, give descriptions about the visual content of each interested objects as if you are seeing the image, as an assistant:

(1) give descriptions about the object itself, including **object types, object functions, object counts, object locations, object actions**, etc.
(2) give descriptions about the object and other objects, including **the relative positions between objects, the interaction between objects in the image**, etc.
Emphasizing common output issues.
Descriptions should be a series of phrases, not whole sentence. Give descriptions for specific interested objects only, do not centered on other objects.
Again, give descriptions centered on specific objects only.

```
"""

image2text = f"""
```

 # Image description of LLaVA, the following two paragraphs are from Table 8

The image displays a parking lot outside a fire station where several fire trucks are parked. In total, there are five fire trucks of varying sizes, all lined up neatly in the lot. Additionally, there are two street lights visible in the scene, providing illumination for the area. The fire trucks appear ready for use and are an bessential part of the fire station's resources.

The image depicts a parking lot outside a fire station, where several fire trucks are parked in a neat row. There are a total of five fire trucks of varying sizes, all aligned and ready for use. In addition to the fire trucks, there are two street lights visible in the scene, providing illumination for the parking lot. The overall setting gives a clear representation of the organized and prepared nature of the fire station.

Street lights: [0.0, 0.23, 0.03, 0.26]
Fire truck: [0.05, 0.6, 0.21, 0.76], [0.19, 0.58, 0.37, 0.77], [0.33, 0.55, 0.61, 0.77], [0.56, 0.57, 0.74, 0.77], [0.72, 0.57, 1.0, 0.76]

```
"""
```

 # Python code together with above text prompts are directly sent to LLaMA

```
messages = [{"role": system, "content": task_description}]
for in_context_example in in_context_examples:
    messages.append({"role": user, "content": in_context_example["
    content"]})
    messages.append({"role": assistant, "content":
    in_context_example["response"]}
messages.append({"role": user, "content": image2text})
```

Table 9: An example of prompt generation for LLaMA input. For each image, we first obtain text description in Table 8. Then, we organize the above prompts and python code, together with in-context examples for LLaMA input. An in-context example is in Table 10.

Example image2text

Two children, a girl and a boy are practicing their writing.
Two children sit on a small seesaw in the sand.
Two children sitting on a teeter totter.
2 kids playing on a seesaw.
Two kids sit on a seesaw.
2 kids/two kids/two children: [0.09, 0.59, 0.49, 0.94]
girl: [0.09, 0.59, 0.23, 0.92]
boy: [0.34, 0.62, 0.49, 0.94]
seesaw/small seesaw/teeter totter: [0.06, 0.83, 0.57, 0.93]
sand: [0.01, 0.61, 1.0, 1.0]

Example Response

[2 kids/two kids/two children]
(1) persons, children, two kids, children learning, kids playing, kids sitting, children practicing their writing
(2) 2 kids playing on a seesaw, Two kids sitting on a seesaw, a girl and a boy, two person on the teeter totter, kids by the sea
[girl]
(1) girl, kid on the left of the image, girl sitting, girl playing
(2) kid playing with the boy, girl sitting on the small seesaw, girl playing in the sand, girl reading a book, a friend of the boy on the right, the taller kid
[boy]
(1) boy, kid on the right, boy sitting, boy playing, boy practicing his writing
(2) kid playing with the girl, boy sitting on a teeter totter, a friend of the girl on the left, boy playing in the sand
[seesaw/small seesaw/teeter totter]
(1) small seesaw, teeter totter, item to be played on, common facilities in parks and playground, game of two people
(2) seesaw in the sand, item the kids are sitting on, item the girl is sitting on, item the boy is playing on
[sand]
(1) common by the sea, the background of the scene
(2) item on which the seesaw is placed, item on which the kids are standing

Table 10: An in-context example. We manually prepare three in-context examples and show one here. On the top block there are image captions and object bbox coordinates. Note that the displayed image is only for image caption reference and is not used in practice. The bottom block shows our expected expressions for each object. The expressions listed in (1) focus on the object attribute itself, and listed in (2) focus on relationship between current object and other objects in this image.

The in-context example in Table 10 follows the task description in Table 9 and primarily serves to standardize the output format of LLAMA. The desired output format is intended to resemble the **Example Response** presented in Table 9. [name] denotes the target currently being described, where the descriptions in (1) are relatively simple, pertaining only to the inherent attributes, and those in (2) entail more complex descriptions involving surrounding objects. This facilitates accurate correspondence between each description and the object it pertains to when parsing the output descriptions.

It is noteworthy that the prompt design is manually crafted. It draws significant inspiration from the prompt framework used by LLAVA, with specific modifications and supplements tailored to the requirements of our task. The entire prompt design underwent numerous iterations based on the output results, ensuring the stability of the output format and style.

D VISUAL PROMPTING AND VISUAL-TEXTUAL MATCHING

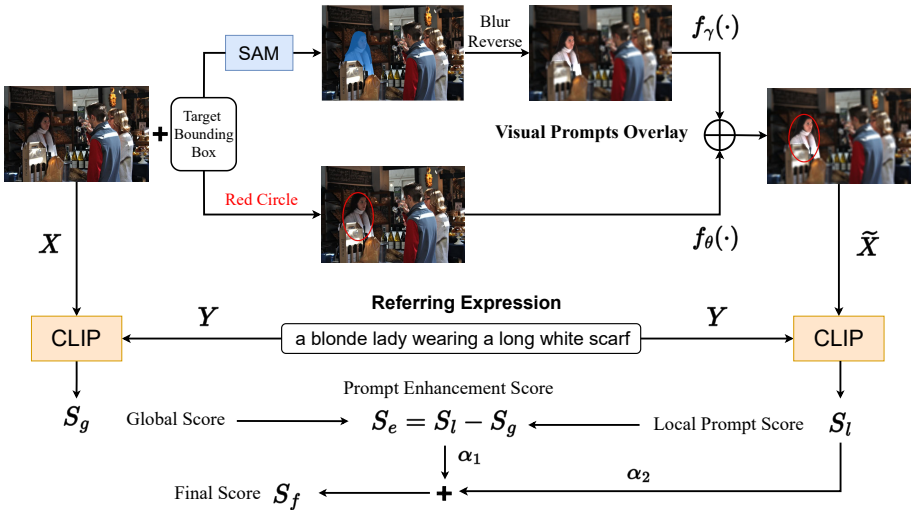

Figure 12: A detailed pipeline of visual prompting and visual-textual matching. We use visual prompting to emphasize target object in the image. Then, we use the CLIP model to compute the scores for expression filtering

.

Motivated by recent applications (Yang et al., 2023; Shtedritski et al., 2023) of visual language pretraining (VLP) models in solving zero-shot REC tasks, we discover that coupling VLP with visual prompting enables robust pairing of image regions and text on the basis of potent generalization. The efficacy of pairing is largely influenced by the configuration of visual prompting, which can be divided into two types as follows:

(i) Shape division of emphasized areas: box, circle and contour.

(ii) Highlighting methods for emphasized areas: crop-based, line-based, mask-based, grayscale reversion and blur reverion based image editing operations.

As shown in Fig. 12, after various combinations of experiments, we find the superposition of line-based circle prompt and blur reverse contour prompt yields the optimal visual prompting strategy. The image $\widetilde{X}$ after visual prompting can be expressed as:

$$\widetilde{X} = f_\theta(f_\gamma(X)) \quad \text{with} \quad f_\gamma(\cdot) = BR(\cdot, SAM(\cdot, B)) \quad f_\theta(\cdot) = RC(\cdot, B) \tag{2}$$

where B represents the box coordinates of the target, BR and RC represent the reverse Gaussian blur for the mask area and the inscribed ellipse for the target bounding box, respectively. SAM denotes Segment Anything Model (Kirillov et al., 2023), which makes visual prompting more refined.

Once we obtain image x and visually prompted $\widetilde{X}$, we send both of them to the CLIP model together with the generated expressions Y from global prompt and local prompt pipelines. Then, we can compute the global score S_g and local score S_l, and thus obtain the final score S_f as illustrated in equation 1 for expression filtering.

E INSTRUCTION GENERATION FOR MINING CLUSTERS OF MULTI-OBJECTS

In order to find and summarize the common attributes among multiple objects in the clusters from DBSCAN (Ester et al., 1996), we use LLaMA for a further analysis. Table 11 shows the text prompt of task description and in-context examples we use for the LLaMA inputs. Then LLaMA produces expressions for multi-objects as shown in Fig. 4.

Task Description
You are an AI language assistant that can analyze phrases and sentenses. User will give you descriptions of several objects in an image. Descriptions of each object are several phrases or short sentences.

The given objects are expected to have similar properties. Based on the descriptions, find the common properties between given objects and summerize precisely as an assistant: common properties between objects can include same types, same functions, same color components, same poses, same relationships with other objects, engaging in the same activity, etc.

If there are no common properties between given objects, just tell that there are no common properties. Your summery should also be phrases. Do not repeat.

Give similarity between all given objects, contrary properties like different positions or different colors of clothes should not be included together in your descriptions.

One In-context Example
Prompt: Objects and their descriptions:
object 2: girl sitting on bed, girl with toy, girl sitting on bed
object 3: man looking down, boy sitting on the bed, man sitting on bed
Please find an summarize the similar properties of given objects.
Response: Summary of common properties of given objects:
people on bed; person sitting on bed; people playing on bed; who sitting on bed;

Table 11: Task description and an in-context example for multi-objects instruction generation.

F INSTRUCTION GROUPING

We use LLaMA to analyze different extents of description based on object category, attribute, and relations. For each instruction of single object in the InDET dataset, we use LLaMA to assign it into one of the preset 4 groups. Table 12 shows the text prompts of task description and in-context examples we use for instruction grouping. For instructions of multiple objects, we assign them to G5 if there is the combination (e.g., "and") of single instructions, or we assign them to G6 if the instructions are generated without concatenation.

Task Description
You are an AI language assistant that can analyze the language complexity of sentences or phrases.
User will give you a phrase or sentence describing a specific object in a image, which could be composed of nouns, adjectives, verbs, etc.
Grade the description according to its language complexity as an AI language assistant.

The language complexity of a phrase or sentence describing a specific object in an image can be graded into four levels:
Level 0. A single noun is used to give the object's name or type.
Level 1. A phrase with one or more nouns, verbs and abjectives is used to describe simple attributes of the object itself, such as its color, its function or purpose, location in the image, or actions.
Level 2 A phrase with one or more nouns, verbs and abjectives is used to describe the object by referring to other objects in the image and describing their relationship, such as their relative positions or interactions.
Level 3. A long phrase or sentence is used to describe attributes of the object and also refer to a few other objects in detail, or describe a complicated or comprehensive/implicit relationship between multiple objects.
The level of descriptions increase as the language complexity and length increase, and also increase as the phrases or sentences become more descriptive and use more abjectives and nouns to describe the object.

One In-context Example
Prompt: Grade description: people who are sitting under an umbrella.
Response: My grading for description people who are sitting under an umbrella: This phrase is referring to the object of people, and gives simple object action of sitting and object relationship with the umbrella. The level of this description is: level 2.

Table 12: Task description and in-context examples for single-object instruction grouping.

G Model and Implementation Details

In this section, we illustrate our model architecture and training configurations.

G.1 Model Architecture

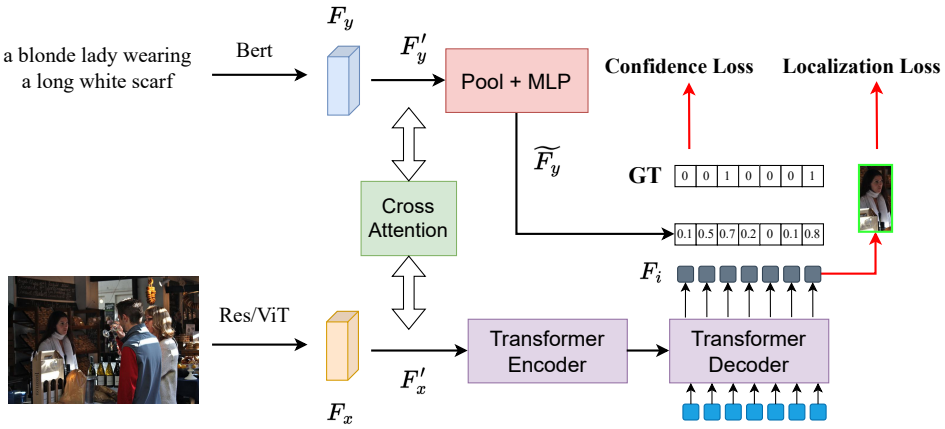

Figure 13: A detailed overview of our diversified referring object detection (DROD) model.

In the main text, We briefly introduced the DROD model architecture we use. Here, we introduce the overall model structure in Fig. 13.

Visual Textual Feature Fusion We use a general text encoder (Devlin et al., 2018) and vision encoder (Dosovitskiy et al., 2021) to obtain text features F_y and image features F_x. In order to improve the attention of visual contents on the described regions by the text expressions, we conduct a multi-modality feature fusion. Specifically, we use a bidirectional cross attention (Bi-XAtt) module to enhance image and text features through information transfer between modalities. The enhanced image and text features are then added to the original features. This process can be formulated as follows:

$$F'_x = F_x + F_{y2x} \; ; \; F'_y = F_y + F_{x2y} \quad \text{with} \quad F_{y2x}, F_{x2y} = \text{Bi-XAtt}(F_x, F_y) \tag{3}$$

Target Discovery and Retrieval With the enhanced visual and language representations, we need to extract the targets referred to by the expression from the image features. Our DROD model applies the encoder-decoder architecture of Deformable DETR (Zhu et al., 2020), which allows for more flexible query retrieving. Here we provide a detailed introduction to our internal module.

The transformer encoder receives multi-scale visual features after text enhancement. Multi-scale Deformable encoder utilizes flexible attention at different scales and spatial shapes to obtain hierarchical characteristics for instances. In addition, following the design of two-stage Deformable DETR, we add an auxiliary prediction head for reference points. The top N points are input into the decoder as the position prior.

The Transformer decoder uses learnable queries to retrieve instance-related information from encoded multi-scale visual features. The design of the query is critical for ensuring stability and efficiency in training, based on previous works (Meinhardt et al., 2022; Wang et al., 2021). The N reference points served by encoder act as the position priors of the N queries. The content part of each query is a static learnable vector. Moreover, following DINO, we add denoising queries to make the decoder's convergence more stable and faster. Through Deformable attention, N queries efficiently retrieve instance embedding $F_i \in \mathbb{R}^{N \times d}$ from expression-aware visual features.

Finally, we need to select the instances referred to by expression from the N instance proposals. As shown in Fig. 13, we use global average pooling and MLP to map the visual-aware expression feature F'_y to the semantic space of the query embedding to obtain $\widetilde{F_y} \in \mathbb{R}^{1 \times d}$. Thus, the matching

score between each query proposal and the expression can be expressed by the cosine similarity between vectors. $S = F_i \times \widetilde{F_y}^\top$. In the inference stage, proposals with scores above threshold θ are selected. This flexible selecting strategy allows our architecture to output any number of results that satisfy user requirements. In the training stage, the overall model is supervised by a combination of confidence loss and localization loss.

G.2 Training Configurations

Generation Engine α_1 in the expression filter responsible for balancing referentiality and semantic accuracy is set to 0.5. While generating multi-target expressions, DBSCAN clustering method has two hyperparameters: neighbourhood radius eps is 1.5 and minimum neighbors of the core point $minPts$ is 2. The temperatures of LLaMA are set to 0.7.

DROD Model We use ResNet-50 (He et al., 2016) and ViT-Huge (Dosovitskiy et al., 2021) as visual encoders and Bert (Devlin et al., 2018) as the text encoder. The transformer encoder-decoder architecture consists of a six-layer encoder and a six-layer decoder. The number of object queries N is set to 900. Our DROD model is initialized by weights pretrained on Objects365 released by UNINEXT (Yan et al., 2023). The optimizer we use is AdamW Loshchilov & Hutter (2019) with a learning rate of 2e-4, a weight decay of 0.05 and the warm-up steps are 400 with an initial learning rate of 4e-5. The model is trained on 32 and 16 V100 GPUs for pretraining and finetuning, respectively.

H ABLATION STUDIES

In this section, we study the necessity of finetuning LLaVA during expression generation from local prompt pipeline. Then, we investigate how different visual prompting strategies affect the CLIP model to maintain our expected expressions.

LLaVA Finetuning In our InstructDET, we do not finetune LLaMA in global prompt pipeline, but finetune LLaVA in local prompt pipeline. There are two main reasons illustrated as follows:

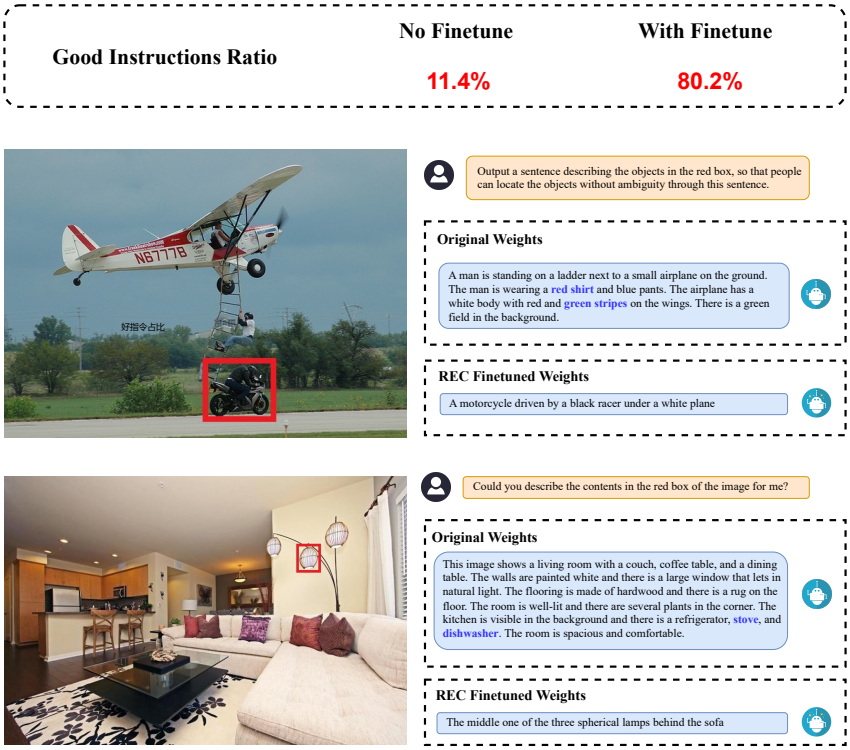

Figure 14: Comparison of expressions generated by LLaVA before and after our finetuning. Blue words indicate incorrect descriptions.

(i) Asking questions directly to LLaVA will get the answers with dense description, rather than the exact expressions we expect for the specific target objects.

(ii) Due to the limited number of visual tokens output by Q-Former, it is difficult to fully display all the visual features of the entire image. With limited visual information, LLaVA produces lengthy descriptions and thus lead to massive hallucinations (blue words in Fig. 14). The generated expressions in this way becomes unstable.

Based on above analysis, we partially finetune LLaVA on the linear layer that that transforms visual features into the semantic text embedding space. To this end, the model is updated to understand which area we aim to emphasize (i.e., marked in the red box in Fig. 14) and which visual features are important for target object description. Fig. 14 shows that after finetuning the linear layer using REC datasets, the probabilities of generating our expected high-quality expressions increases from 11.4% to 80.2%.

Visual Prompting Selection The selection of visual prompting (VP) is crucial for expression filtering. We evaluate the affect of various visual prompting methods based on the retrieval performance of CLIP model in Table 13. First, we define a new metric called expression retrieval ratio, which indicates the proportion of correct expressions that can be retrieved based on the visual-textual matching of CLIP model. During testing, each minibatch contains k correct target expressions, and the remaining expressions are negative samples. We take the expressions with the top k Clip score as

Table 13: Visual Prompting (VP) Evaluation. Different VP methods make differences in the ability of CLIP to retrieve high-quality expressions.

Strategy	Visual Prompting		Expression Retrieval (%)	
	Shape	Tool	Easy ↑	Hard ↑
1	Box	Crop	31.52	24.09
2		Gray	39.66	31.85
3		Line	50.12	42.30
4		Mask	48.29	39.93
5		Blur	52.75	44.30
6	Circle	Line	52.78	44.33
7		Mask	51.01	42.87
8		Blur	54.13	46.22
9	Contour	Line	52.34	44.05
10		Mask	51.90	43.44
11		Blur	55.79	47.78
12	VP3 + VP11		56.81	48.97
13	**VP6 + VP11**		**58.29**	**50.99**

the retrieved expressions. Finally, the proportion of correct expressions in the retrieved expressions is the expression retrieval ratio. Under the Easy setting, the negative samples in the minibatch come from other images. Under the Hard setting, the negative samples in the minibatch may come from different targets in the same image. Table 13 shows that cropping (1) and grayscale reversion (2) methods achieve poor results. Because cropping loses all surrounding environment information, and grayscale reversion loses all color features. The best single prompt is the contour blur reversion. The best combination prompt is the contour blur reversion and circle line. Circle line can indicate the rough areas that needs attention, and contour blur reversion can highlight the target object in a fine-grained manner that eliminates background interference. Inevitably, the prior knowledge in the CLIP model is also important to the difference in visual prompt effects. In CLIP's pre-trained web-scale dataset, images with red circles often indicate that the targets in the red circles are more important and need to be noticed. A large amount of photography also exists in the web-scale dataset, which employs "Bokeh" to blur the background and highlight the subject.

I SYNONYMOUS REWRITE

In post processing, we utilize LLaMA to do synonymous rewriting to further diversify the generated expressions. The prompt we used is shown in Table 14.

Synonymous Rewriting Prompt
I want you to act as a synonymous expression provider. I will give you a text of phrase or short sentence, which is an expression that describes a main object while mentioning some other objects. And you will reply to me with a new expression that have the same semantic meaning and describe the same main object as the provided expression. The new expressions should also be phrases or short sentences no longer than 25 words. Do not write explanations on replies. Do not repeat.

Table 14: Synonymous Rewriting Prompt for LLaMA.

J SUPPLEMENT COMPARISONS

J.1 EVALUATION ON INDET

Table 15: Evaluation results on our InDET. We show the object bbox average precision (AP) values (%).

Method	Backbone	AP	AP by Group					
			G1	G2	G3	G4	G5	G6
MDETR	ResNet101	34.86	47.44	46.79	34.14	23.22	25.91	28.17
G-DINO	SwinB	35.96	47.10	47.17	35.29	26.84	27.95	27.61
UNINEXT	ResNet50	43.37	54.49	54.52	44.49	37.17	31.41	32.01
DROD (Ours)[1]	ResNet50	62.24	67.14	67.34	60.89	55.10	70.15	74.22
DROD (Ours)[2]	ResNet50	62.34	67.22	68.04	61.09	55.42	68.60	72.91
DROD (Ours)	ViT-H	66.90	72.53	72.47	66.42	59.86	73.34	75.92

[1] For fair comparison, our DROD model in Table 2 only utilizes RefCOCO/+/g datasets but with diversified instructions, which is partial of InDET.
[2] Here we add evaluation results of DROD model trained on full InDET.

J.2 EVALUATION ON REFCOCO/G/+

Table 16: Supplementary Evaluation results on the RefCOCO/g/+ datasets. We follow evaluation protocols to report AP values (%) of comparing methods. We use the notations "CC", "VG", "OI", "O365", "RIGame", for COCO, Visual Genome, OpenImage, Objects365, ReferItGame, respectively.

Method	Backbone	Data	RefCOCO			RefCOCO+			RefCOCOg	
			val	testA	testB	val	testA	testB	val-u	test-u
RefTR	ResNet101	VG	85.65	88.73	81.16	77.55	82.26	68.99	79.25	80.01
SeqTR	DarkNet53	VG,RIGame,Flickr,RefC	87.00	90.15	83.59	78.69	84.51	71.87	82.69	83.37
MDETR	ResNet101	GoldG,CC,RefC	86.75	89.58	81.41	79.52	84.09	70.62	81.64	80.89
G-DINO	SwinB	O365,CC,RefC,GoldG,etc	83.95	87.79	79.16	72.91	80.91	62.96	76.98	76.76
PolyFormer	SwinL	GoldG,ReferIt,RefC	90.38	92.89	87.16	84.98	89.77	77.97	85.83	85.91
UNINEXT	ResNet50	O365,CC,RefC	87.64	90.35	83.49	78.14	83.22	68.71	80.96	81.86
DROD (Ours)[1]	ResNet50	O365,CC,InDET	88.92	90.86	85.57	78.27	83.39	71.04	83.01	82.91
DROD (Ours)[2]	ResNet50	O365,CC,InDET	89.85	92.03	87.24	80.50	85.87	73.61	83.93	84.73
DROD (Ours)[3]	ViT-H	O365,CC,InDET	92.93	94.47	91.13	86.20	89.82	80.86	88.62	89.46

[1] For fair comparison, our DROD model in Table 3 only utilizes RefCOCO/+/g datasets but with diversified instructions, which is a part of InDET.
[2] Here we add evaluation results of DROD model trained on full InDET.
[3] DROD here with ViT-H as backbone also utilizes RefCOCO/+/g datasets with diversified instructions.

We have supplemented the comparisons with more specialized models, such as Polyformer(Liu et al., 2023b). With comparable magnitudes in the backbone network, our model continues to exhibit advantages.

K TIME CONSUMPTION

Table 17: Time Consumption Statistics. We use "instr." as the short for "instruction".

Pipeline	Step	Main Model	BatchSize	Time	Avg
Global Prompt	Image Caption	LLaVA	4	10.71 s/batch	2.68 s/image
	Instr. Generation	LLaMA	1	17.25 s/image	0.63 s/instr.
Local Prompt	Instr. Generation	LLaVA	16	2.47 s/batch	0.15 s/instr.
Post-Processing	Instr. filtration	CLIP	1	0.186 s/image	0.016 s/box
	Multi-Object Instr.	DBSCAN	1	0.053 s/image	-
	Instr. Grouping	LLaMA	1	1.43e-06 s/instr.	-

Table 17 shows the time consumption of the main steps in our instruction generation procedure. The data is obtained from NVIDIA-v100-32G machines. Obviously, the most time consuming steps are those who depend on large models. Except for the limited hardware performance, another reason why the instruction generation step in global prompt pipeline is very time consuming is that we have very long LLaMA prompts which include three in-context examples and we also desire long response which could include more diversified instruction for each object in image. So this step requires multiple machines to generate simultaneously. While the text received and generated in each forward pass is relatively short in the local prompt pipeline, the generation efficiency is significantly constrained by the fact that only one instruction can be generated per forward pass. Therefore, we address this limitation by maximizing parallelism through the increase in batch size, allowing the simultaneous generation of multiple instructions.