Spot the Difference: A Cooperative Object-Referring Game in Non-Perfectly Co-Observable Scene

Anonymous ACL submission

Abstract

Visual dialogue has witnessed great progress after introducing various vision-oriented goals into the conversation, especially such as GuessWhich and GuessWhat, where the only image is visible by either and both of the questioner and the answerer, respectively. Researchers explore more on visual dialogue tasks in such kind of single- or perfectly co-observable visual scene, while somewhat neglect the exploration on tasks of non-perfectly co-observable visual scene, where the images accessed by two agents may not be exactly the same, often occurred in practice. Although building common ground in non-perfectly co-observable visual scene through conversation is significant for advanced dialogue agents, the lack of such dialogue task and corresponding large-scale dataset makes it impossible to carry out in-depth research. To break this limitation, we propose an object-referring game in non-perfectly co-observable visual scene, where the goal is to spot the difference between the similar visual scenes through conversing in natural language. The task addresses challenges of the dialogue strategy in non-perfectly co-observable visual scene and the ability of categorizing objects. Correspondingly, we construct a large-scale multimodal dataset, named *SpotDiff*, which contains 49k Virtual Reality images and 97k dialogues generated by self-play. Finally, we give benchmark models for this task, and conduct extensive experiments to evaluate its performance as well as analyze its main challenges[1].

1 Introduction

Building a dialogue agent that can intelligently communicate with people through comprehending and reasoning in vision and natural language is a challenging task in AI research (Strub et al., 2017a; Niu et al., 2018). Such visual dialogue agents have broad prospects in social services and commercial applications, e.g., assisting the visually impaired people to understand the surroundings (Bigham et al., 2010), recommending products by dialogue-based image retrieval (Guo et al., 2018), so that related researches (Das et al., 2017a; de Vries et al., 2017; Gan et al., 2019; Haber et al., 2019; Ilinykh et al., 2019; Chen et al., 2020; Wang et al., 2020; Cogswell et al., 2020; Takmaz et al., 2020; Liang et al., 2021; Kottur et al., 2021) have attracted increasing attention.

In recent years, researchers have proposed many visual dialogue tasks for different scene settings, including single-observable scene and perfectly co-observable scene. In single-observable scene, the scene is only visible to one interlocutor. For example, Das et al. (2017a) propose the task of Visual Dialogue, which requires the dialogue agent to answer questions given an image and dialogue history while the questioner can not see the image. On the basis of the above task, GuessWhich (Das et al., 2017b; Murahari et al., 2019; Zhou et al., 2019; Lee et al., 2019) introduces an image-guessing game. This task aims at enabling the questioner imagine the invisible target image and finally guess it, through conversing with the answerer who could access the target image. In co-observable scene, the scene is fully observed by all interlocutors. For example, GuessWhat?! (Zhang et al., 2017; Zhao and Tresp, 2018; Strub et al., 2017b; Shekhar et al., 2019; Shukla et al., 2019; Xu et al., 2020) focuses on locating the target object in an image, which is visible by both the questioner and the answerer, through dialogue between them. Moon et al. (2020a) introduce the task of SIMCC, which addresses the task-oriented dialogue scenario on shopping domain where a system dialogue agent and a user share the co-observable scene.

However, in actual applications, there are many situations where the visual scenes accessed by two people are similar but not be exactly the same. Take the remote abnormal troubleshooting as an exam-

[1]The dataset and codes will be released upon publication.

ple, the user can access the problem machine, while the quality inspector can access intact machine. They determine the fault location through conversation online or by telephone. At this time, it is more important to help each other to understand the partner's scene and clarify the differences, through dialogue interaction. Therefore, some researchers turn to investigate the visual dialogue in such non-perfectly co-observable scene with the provision of a small-scale dataset. Lopes et al. (2018) study the dialogue phenomenon under the setting of making two interlocutors to find differences between two similar scenes. They collect a dataset, which only contains 54 dialogues in 8 different cartoon scenes. More than that, lacking deeply analyzed challenges and corresponding solutions also makes its contribution to research community limited.

Two key challenges of the visual dialogue in non-perfectly co-observable scene are not covered by the above tasks: 1) Difference-oriented dialogue strategy. The two interlocutors participating in the dialogue can only access their own part of the visual scenes, so they can only clarify the difference through the dialogue. Therefore, to complete the goal of the dialogue, the dialogue interaction needs to constantly overcome the difficulty brought by differentiated visual information. 2) Categorization-oriented question strategy. Human understands the world usually through categorization, which requires subjective generalization and classification of objects (Rosch and Lloyd, 1978). Such ability can be necessary for advanced agents. Therefore, finding a question strategy that can efficiently categorize the objects in the scene may be a critical path to quickly locating the difference. Although categorization has been mentioned in GuessWhat?!, all the questions in it are Yes/No questions, such as 'is it a decoration?'. It ignores that an important purpose of categorization is induction, which often requires the abilities of accurate counting and clearly pointing out these objects, such as 'I have three decorations, and you?', 'what are they?'.

Obviously, the ability to deal with these challenges is significant for advanced dialogue agents. To develop these capabilities of machines, in this paper, we propose an object-referring game – *Spot the Difference*. As shown in Figure 1, the goal of *Spot the Difference* is to spot the different object between two similar images via conversing in natural language between a questioner and an answerer in a non-perfectly co-observable visual scene. To this end, we construct a large-scale multimodal dataset, named *SpotDiff*, which contains 49k images and 94k dialogues. First, we generate the images of *SpotDiff* with an elaborately designed scene simulator, taking into account the coherence of the real world. Then, based on the generated images, we generate the dialogues of *SpotDiff* through a well-designed two-stage dialogue generation algorithm. Finally, we propose benchmark models for *Spot the Difference*, which are based on the multimodal pre-trained model LXMERT (Tan and Bansal, 2019). We evaluate the performance of the dialogue system and the answerer agent, and analyze the model's ability in dialogue strategy and categorization.

Our main contributions are concluded as follows:

- We propose a new visual dialogue task – *Spot the Difference*, which mainly addresses challenges of the dialogue strategy in non-perfectly co-observable visual scene and the ability of categorizing objects.
- We construct the *SpotDiff* dataset, which consists of 49k Virtual Reality images and 95k programmatically simulated dialogues.
- We provide strong benchmark models for *Spot the Difference*. Experimental results show that the task performance can be improved by designing difference-oriented dialogue strategy and categorization-oriented question strategy, both of which are the challenges that the task of non-perfectly co-observable scene hope to address. These provide insights for developing more intelligent visual dialogue agents.

2 *Spot the Difference* Game

As illustrated in Figure 1, *Spot the Difference* is an object-referring game conducted by a questioner and an answerer. The questioner and answerer can see images I^Q and I^A, respectively.

The goal of questioner is to spot the difference from I^Q to I^A, i.e., the object in I^Q that is not in I^A (marked with a green box in Figure 1). The questioner constantly asks questions based on the image I^Q, such as asking the number of objects with specific conditions, the referential content of the previous round, and the object at a specific location, e.g., q_1 – 'There are four white objects?', q_3 – 'What are they?', q_4 – 'What is the leftmost thing on the tea table?'. After the questioner has located the different object, it terminates the conversation and makes a guess on the correct object list of I^Q.

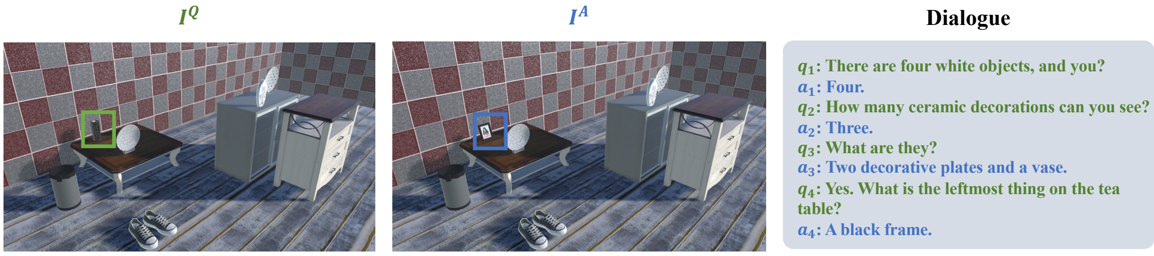


Figure 1: An example of *Spot the Difference*. The green box indicates the different object from I^Q to I^A.

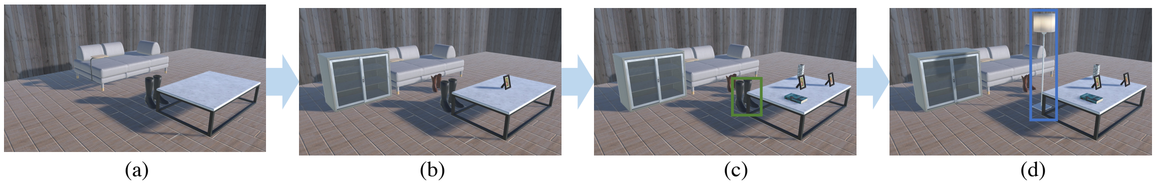


Figure 2: The generation process of *SpotDiff* images. (a)–(c) show the object-by-object generation process of scene. (c) and (d) are a pair of similar images. The green box in (c) represents the different object from (c) to (d).

Based on the image I^A and the question, the answerer gives the answer, which may be a number, a description of one or multiple objects, e.g., a_1 – 'Four', a_3 – 'Two decorative plates and a vase'.

3 *SpotDiff* Dataset

In this section, we first describe how images and dialogues of the *SpotDiff* are generated in Section 3.1 and Section 3.2, respectively. Then we present the dataset analysis in Section 3.3.

3.1 Image Generation

First, we develop a scene simulator to generate *SpotDiff* images in Virtual Reality (VR) environment. Then for each image, we construct its scene graph, serving as the input to dialogue generation.

3.1.1 Scene Simulator

The scene simulator generates similar image pairs with only one object different and the generation process of *SpotDiff* images is illustrated in Figure 2. First, the scene simulator generates a random scene by placing objects item by item. Then, it randomly selects one object from all the objects that can be replaced in the scene, and replace it with a random object of a different category. Finally, the scene simulator renders the scene in Unity3D (Unity Technologies, 2019) and takes screenshots with Unity Perception[2] (Unity Technologies, 2020).

Real world scenes appear as a composite of coherent objects (Galleguillos et al., 2008). To make VR scenes more reality, the scene simulator adopts an elaborately designed search algorithm, mainly considering the following aspects:

Richness of Objects. To generate richer scenes, more diverse objects are needed. We use 207 digital assets[3] which belong to 87 different categories.

Placement Relationship. A directed graph (please refer to Appendix A.2 for details) is defined to describe the placement relationship between categories. For example, bread can be placed on a plate, but not directly on the floor.

Spatial Arrangement[4]. The scene should neither be too evacuated nor too compact. The former may cause the pixels of objects in the image to be too small, and the latter may cause mutual occlusion between objects.

Object Co-Occurrence[4]. Related objects co-occur with high probability. For example, computers, mice and keyboards often appear together because they are all office supplies.

3.1.2 Image Scene Graph

The scene graph[5] contains the information of all objects in an image, including:

1) Attribute: Each Object is annotated with color and material.

2) Taxonomy: Taxonomy information is depicted by a predefined hierarchical tree of categories, which is in Appendix A.1. For example, pizza belongs to {*pizza, baked food, food*}.

[2]A toolkit provided by Unity Corporation for generating large-scale computer vision datasets.

[3]We obtain them from `https://assetstore.unity.com/` and `https://www.turbosquid.com/`

[4]We present the implementation details of spatial arrangement and object co-occurrence in Appendix A.3 and A.4, respectively.

[5]We show an example of scene graph in Appendix D.1.

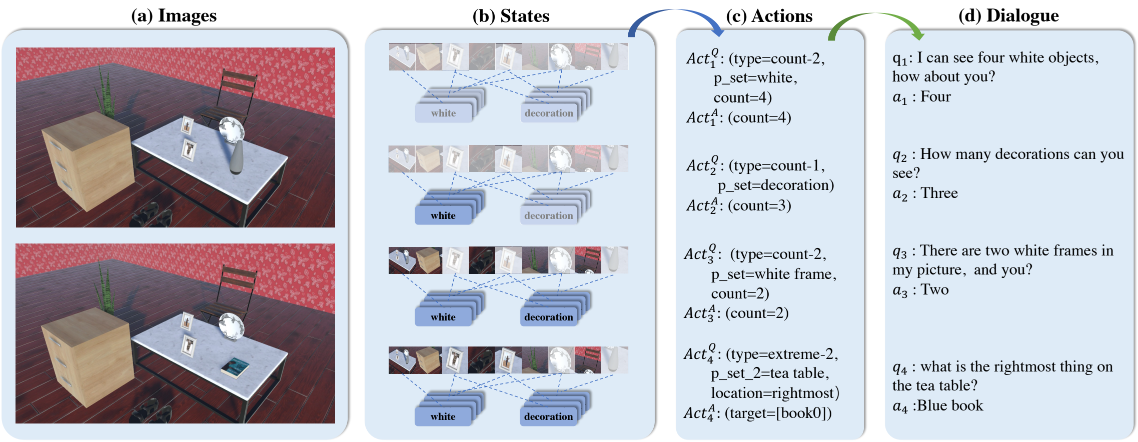


Figure 3: The generation process of a *SpotDiff* dialogue. (a) The questioner simulator and answerer simulator look at the top and bottom images, respectively. (b) The states give part of the visual state tracking information under current dialogue, where the transparent node means that an object or property has not been fully confirmed. (c) Act_t^Q and Act_t^A are respectively question action and answer action generated at time t. (d) The dialogue consists of a series of question-answer pairs, where the question q_t and answer a_t are mapped from Act_t^Q and Act_t^A, respectively.

3) Position: Position information is described by 2D bounding box and 3D bounding box, which are annotated when generating images with Unity Perception (Unity Technologies, 2020).

Color, material and categories are regarded as atomic properties of an object.

3.2 Dialogue Generation

With the image scene graph as input, we design a two-stage dialogue generation approach as shown in Figure 3. In the first stage, a questioner simulator and an answerer simulator are used to generate a dialogue action sequence through self-play (Section 3.2.1). In the second stage, the dialogue action sequence is mapped to natural language through manually defined templates (Section 3.2.2).

3.2.1 Dialog Action Generation

Inspired by previous works (Moon et al., 2020b; Kottur et al., 2021), the dialogue action sequence consists of question actions and answer actions, both of which are composed of a series of slot-value pairs. The dialogue action sequence is interactively generated by the questioner simulator and answerer simulator. In concrete, at each round, the questioner simulator produces a question action and the answerer simulator gives the corresponding answer action. The interaction is repeated until the questioner simulator could locate the target object.

Question actions are divided into seven subtypes (see Table 1), which belong to three types: 1) **Count** type (count-1 and -2) asks the number of objects with specific properties. Comparing with count-1 type (e.g., 'how many white objects can you see?'), count-2 type adds a hint for counting, e.g., 'I have four white objects, how about you?'. 2) **Extreme** type (extreme-1, -2 and -3) asks for a specific description of the object at a positional extreme among conditioned objects. For extreme-1 type, the conditioned objects are all objects in the image, e.g., 'what is the rightmost thing?'. For extreme-2 and -3 types, the conditioned objects are objects placed on a given object, e.g., 'what is the rightmost thing on the tea table?'. 3) **Ref** type (ref-1 and -2) follows the count type, and requires the answerer to give a specific description of the objects referred to in the previous round. Ref-1 type asks one object while ref-2 asks multiple, e.g., 'what is it?' and 'what are they?'.

At each round, the questioner simulator tracks visual state according to dialogue history, then selects an appropriate question action based on the tracked visual state and question strategy. A good questioner simulator can achieve the above goal by answering the following questions. $\mathbf{Q_1}$: How to accurately track the state of each object in an image, taking into account entailment relationships between properties of the object. For example, one won't ask 'is there any fruit?' after knowing there is an apple; $\mathbf{Q_2}$: How to efficiently guide the conversation to avoid object-by-object mechanical enumeration.

$\mathbf{Q_1}$**: Visual State Tracking.** To maintain the state of the image as the dialogue proceeds, the

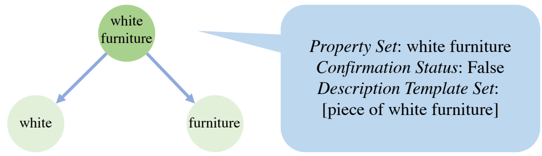


Figure 4: A simplified instance of object state graph.

questioner simulator constructs an object state graph for each object. Considering that there are many combinations of atomic properties, we define a property set. For example, for the white furniture in Figure 4, its property sets include *furniture*, *white* and *white furniture* (only retain *white* and *furniture* as atomic properties here for simplicity). Obviously, there are entailment relationships between property sets, inspiring us to describe the state of an object with a directed graph in the process of dialogue.

To this end, we construct the object state graph for each object, where nodes represent property sets and edges represent entailment relationships between them. To clarify which property sets of an object are known or not, each node maintains a boolean value initialized to False, which we name as confirmation status. When a node is confirmed (its confirmation status is True), all reachable nodes from it are also confirmed. Conversely, for two confirmed nodes, whose property sets are denoted as A and B respectively, the node corresponding to the property set $|A \cup B|$ is also confirmed. In addition, each node corresponds to a description template set, which is used to map the property set to a phrase in the natural language generation stage, such as *white furniture* $\rightarrow$ *piece of white furniture* .

In addition, the questioner simulator also track a candidate object set S_{cand}, which includes objects whose existence has not been confirmed. In the beginning, S_{cand} includes all objects in the image. As the dialogue proceeds, an object will be removed from S_{cand} after all nodes in its corresponding object state graph have been confirmed.

Q$_2$: Categorization-Based Question Strategy. For efficient questioning, we propose a categorization-based question strategy whose main idea is to gather more information by generalizing half of the remaining objects as much as possible. As illustrated in Figure 3, q_2 – 'How many decorations can you see' generalizes the decorations in the image to confirm whether a decoration has been replaced. Therefore, we design an approach to simulate such a strategy. In concrete, the questioner simulator maintains a list of question types that could be performed. The count type is always in the list. When the size of the candidate object set S_{cand} is less than n, the extreme type is added to the list[6]. When the question type of the previous round is count and the corresponding answer is less than m, the ref type is added to the list[6] The final question type is sampled from the list. After the question type is determined, the slot-value pairs are heuristically obtained as follows:

- Count type: First, the questioner simulator counts frequencies of all property sets, which are defined as the number of unconfirmed nodes corresponding to the property set for objects in the candidate object set S_{cand}. Then, it chooses the property set whose frequency is closet to $\frac{|S_{cand}|}{2}$, to produce a question action.
- Extreme type: First, the questioner simulator maintains a candidate list to store all valid slot-value pairs. Then, it enumerates all slot-value pairs, and put slot-value pairs that could be used to retrieve an object from S_{cand} into the list. The final slot-value pairs are sampled from the list.

Given a question action, the answerer simulator retrieves the corresponding information on the image scene graph, and produce the answer action (Please refer to Appendix B.1 for details).

3.2.2 Natural Language Generation

At this stage, each action is mapped to a natural language sentence. Taking question action as an example, we randomly select a question template according to the question subtype and fill the slot values into the question template to produce a question. Table 1 shows all the question subtypes, corresponding slots, and some natural language templates. Notably, the property sets are mapped to natural language phrases with description template sets. In addition, to make dialogues more fluid, we design transition sentences to concatenate adjacent rounds of dialogue.

3.3 Dataset Analysis

For each *SpotDiff* image pair, we generate 4 dialogues by changing the order between images (2 dialogues in positive order and 2 dialogues in reverse order), and the dialogues that fail to complete the task within 10 rounds are discarded. The *SpotDiff* dataset contains 94k dialogues and 49k *SpotDiff*

[6] n and m are hyper-parameters, which are empirically set to 5 and 4, respectively. .

count-1	S	(p_set=X)
	T	*How many [f(X)] can you see?*
	E	*How many plastic products can you see?*
count-2	S	(p_set=X,count=C)
	T	*I have [C] [f(X)], how about you?*
	E	*I have two carpets, how about you?*
extreme-1	S	(p_set=X, location=L)
	T	*There is [f(X)] on the far [L] of the image, and you?*
	E	*There is a white floor lamp on the far left of the image, and you?*
extreme-2	S	(p_set_1=X1, p_set_2=X2, location=L)
	T	*There is [f(X1)] on the far [L] of the [f(X2)], and you?*
	E	*There is a gray plastic cup on the far left of the table, and you?*
extreme-3	S	(p_set_1=X1, p_set_2=X2, location_1=L1, location_2=L2)
	T	*The [L1]smost one on the [L2]smost [f(X2)] is [f(X1)], what about you?*
	E	*The rightmost one on the leftmost nightstand is a frame, what about you?*
ref-1	T	*What is it?*
ref-2	T	*What are they?*

Table 1: The slot-value pairs, templates, and examples for question subtypes. The first column gives the question subtype, while the second and third column mean the key and corresponding value (S=slot-value pairs, T=template, E=example), respectively. $f(\cdot)$ means the mapping function from property sets to natural language phrases.

	SpotDiff	**SIMCC2**	**CLEVR**
# Dialogues	94k	11k	425k
# Images	49k	1.5k	85k
# Turn	5.0	5.2	10
# Unique Q	27k	-	73k
# Unique A	5.8k	-	29
Avg # Q len	9.8	-	10.6

Table 2: Comparison of *SpotDiff* to similar datasets.

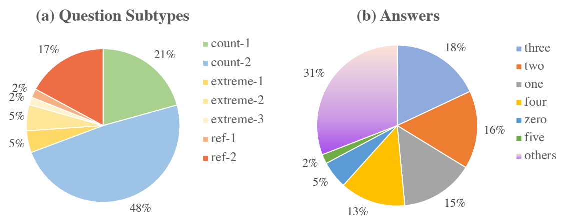


Figure 5: Distribution of question subtypes and answers.

images, and is splited by randomly assigning 80%, 10% and 10% of image pairs and its corresponding dialogues to train, valid and test set. Table 2 shows the comparison results of *SpotDiff* with SIMCC 2.0 (Kottur et al., 2021) and CLEVR Dialogue (Kottur et al., 2019). The SpotDiff dataset has much more unique answers than CLEVR Dialog (5.8k vs 29), indicating the answerer in our task has a higher degree of freedom.

Figure 5 (a) shows the distribution on question subtypes. More than 69% of the questions in the *SpotDiff* dataset need to count objects with specific properties. Figure 5 (b) presents the distribution of answers. There are a total of 5.8k unique answers, of which the 6 most frequent unique answer account for 69% of the total answers, while remaining unique answers (descriptions for one or more objects) account for 31%, making the distribution a long-tailed distribution.

4 Task Formulation

Following previous work (de Vries et al., 2017), the Questioner Bot (Q-Bot) consists of Question Generator (QGen) and Guesser, which are responsible for asking questions and guessing the target object, respectively. The Answerer Bot (A-Bot) is a visual question answering model.

QGen. At round t, QGen asks a question q_t given the dialogue history $H_{t-1} = \{(q_1, a_1), \cdots, (q_{t-1}, a_{t-1})\}$ and the image I^Q, which could be formulated as:

$$q_t \sim P(q|H_{t-1}, I^Q). \tag{1}$$

A-Bot. A-Bot predicts the answer a_t from the candidate answer set, based on the question q_t, dialogue history H_{t-1}, and the image I^A, which could be denoted as:

$$a_t \sim P(a|q_t, H_{t-1}, I^A). \tag{2}$$

Guesser. After T rounds of dialogue, Guesser makes a guess on the correct object list $O_{correct}$

#	GT-Q	GT-A	GT-V	SUCC ↑
1	-	-	-	35.17
2	-	-	√	44.40
3	√	-	-	65.62
4	√	√	-	74.97

Table 3: The performance of the dialogue system[8]. GT-Q: ground truth question, GT-A: ground truth answer, GT-V: visual features extracted by ground truth box, SUCC: task success rate (%). ↑: higher is better.

of I^Q given the full dialogue history H_T as follow:

$$o^* \sim P(o|H_T, O_{correct}), \quad (3)$$

where T is the maximum number of dialogue rounds, $O_{correct} = \{(c_1, p_1), \cdots, (c_M, p_M)\}$, c_i and p_i are the correct category and relative bounding box of the i-th object, respectively.

5 Experiments

To explore the challenges arising from the task, we first train benchmark models and evaluate their performance on *SpotDiff* dataset. Then we conduct extensive experiments to analyze two main challenges: categorization and dialogue strategy.

5.1 Benchmark Models

We train benchmark models on *SpotDiff* dataset: 1) **QGen**[7]: An encoder-decoder model where encoder adopts multimodal pre-trained model LXMERT (Tan and Bansal, 2019) and decoder is initialized by BERT (Devlin et al., 2019). 2) **A-Bot**[7]: A VQA model with the multimodal pre-trained model LXMERT as encoder and a classification head to predict the answer. 3) **Guesser**: A BERT (Devlin et al., 2019) encoder with a classification head to predict the target object.

Formally, input sentences are tokenized by WordPiece (Wu et al., 2016) from BERT (Devlin et al., 2019). We follow Tan and Bansal (2019) to represent the visual features as a series of object representations, where objects are detected by the Faster-RCNN (Ren et al., 2016) pre-trained on Visual Genome (Krishna et al., 2016). For each object, its representation is a concatenation of pooling features provided by (Anderson et al., 2018; Yu et al., 2020) and 4-dim vector of relative bounding box.

[7]We also adopts GPT-2 (Radford et al., 2019) and UpDn (Anderson et al., 2018) as QGen and A-Bot, respectively. Please refer to Appendix C.3 and C.2 for details.

[8]There is no ground truth answer when the question is generated by the model.

5.2 Quantitative Results

Dialogue System Performance. We investigate the performance of the dialogue system under the setting of *Spot the Difference*. Specifically, QGen and A-Bot first interactively generate a 5-round Q-Bot-A-Bot dialogue, and then Guesser makes a guess on the correct object list given the generated dialogue. Table 3 shows the task success rate under different settings. GT-Q and GT-A indicate whether the ground truth question and ground truth answer are used, respectively. GT-V indicates whether the visual features are extracted by the ground truth bounding box. Comparing row 1 and row 2, it can be seen that correct object detection could improve task success rate. Comparing row 1 and row 3, it shows that the questioner model greatly limits the task success rate and the main challenge of the task lies in the modeling of QGen. Comparing row 1 and 4, there is still a large gap between the Q-Bot-A-Bot dialogue and the ground truth data.

A-Bot Performance. Following previous works (Goyal et al., 2016; Anderson et al., 2018; Cadène et al., 2019; Si et al., 2021) on VQA, we verify A-Bot performance in term of accuracy on various question subtypes under classification setting. We observe that: 1) Count-2 (85.03%) surpasses count-1 (81.96%) due to the hints about counting in count-2 questions. 2) Extreme-1 (84.60%) outperforms extreme-2 (78.90%) and extreme-3 (72.74%) because the model's spatial reasoning ability is more urgently required for extreme-2 and extreme-3. 3) Ref-1 (89.02%) is better than ref-2 (81.96%), considering that ref-2 questions ask multiple objects while ref-1 questions ask one.

5.3 Effect of Categorization

We name the question that could generalize at least two different kinds of objects on an image as a Cate-Q. To investigate the effect of categorization ability on task success rate, we first obtain Q-Bot-A-Bot dialogues on the test set, and then group these dialogues in different ways, i.e., the accuracy rate, recall rate, and number of Cate-Q in the dialogue[9].

Accuracy Rate of Cate-Q. We first extract quantifiers and property sets in Cate-Q, and calculate the accuracy of Cate-Q in a dialogue by matching with objects on the scene graph. Then we divide the generated dialogues according to the counting

[9]The relationship between the number of cate-Q and task success rate is in Appendix C.4.

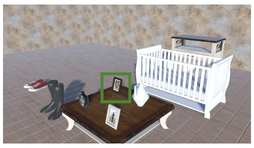
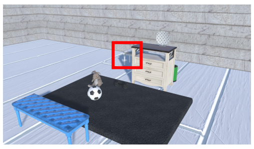
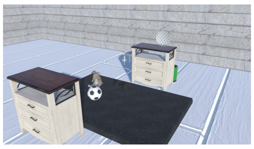
I^Q	I^A	*SpotDiff* Dialogue	Q-Bot-A-Bot Dialogue
		q_1: I can see four decorations, and you? a_1: Three q_2: This is different from mine. there are two plastic frames in my picture, what about you? a_2: One q_3: Mine is different from yours. There is a white plastic frame on the far front of the tea table, and you? a_3: White plastic frame	q_1: I can see four decorations, and you? a_1: Three q_2: I have two white decorations, how about you? a_2: Two q_3: Yes. can you give me more information about them? a_3: A frame and a vase q_4: There is a black ceramic vase with white patterns on the far front of the tea table, and you? a_4: White plastic frame q_5: There is a black plastic frame in my picture, what about you? a_5: Zero
		q_1: I can see four white objects, and you? a_1: Five q_2: Mine is less than yours. I want to know the number of black objects in your picture. a_2: Two q_3: Can you tell me what they are? a_3: A carpet and a dumbbell q_4: There is a gray bunny toy with a white belly on the far behind of the carpet, and you? a_4: Gray bunny toy with a white belly q_5: What is on the far front? a_5: White bedside table with three drawers	q_1: I can see four furnitures, and you? a_1: Four q_2: I can see four white objects, and you? a_2: Four q_3: Yes. There are three leather products in my picture, what about you? a_3: Two q_4: The leftmost one on the rug is a black-white soccer, what about you? a_4: Black-white soccer q_5: I can see a black-white soccer, and you? a_5: Zero

Figure 6: Examples of *SpotDiff* dialogues and Q-Bot-A-Bot dialogues. Green sentences: questions, blue sentences: answers, red box: the wrong guess, green box: the correct guess.

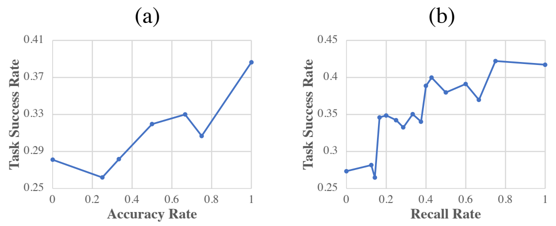


Figure 7: (a)/(b) shows the relationship between the task success rate and the accuracy/recall of Cate-Q.

Action Transition			SUCC↑
white	→	white ceramic	41.57
furniture	→	cloth carpet	46.04
	. . .		
deepening action transition			36.92
white	→	furniture	33.86
furniture	→	brown	30.98
	. . .		
converting action transition			34.42

Table 4: The relationship between action transition and task success rate. SUCC: task success rate (%).

accuracy of Cate-Q, and examine the task success rate within each group. As shown in Figure 7 (a), as the counting accuracy of Cate-Q increases, the task success rate shows an increasing trend, demonstrating that accurate counting for Cate-Q could help to complete the task.

Recall Rate of Cate-Q. For a pair of similar images, we extract the property sets of the Cate-Q in the corresponding Q-Bot-A-Bot dialogue and ground truth dialogue, which are denoted as A and B, respectively. We define the recall rate of Cate-Q as $\frac{|A \cap B|}{|B|}$. The Figure 7 (b) shows the task success rate increases as the the recall rate increases, indicating the importance of selecting appropriate property sets to raise Cate-Q for successfully completing the task.

5.4 Effect of Dialogue Strategy

Action Transition. To investigate the relationship between action transition and task success rate, we group Q-Bot-A-Bot dialogues according to adjacent question action transitions. Question action transitions could be divided into deepening action transitions and converting action transitions according to whether the latter question deepens the previous one. Table 4 shows that dialogues with deepening action transitions achieve higher task success rate (36.92% vs 34.42%) because the deepening action transitions could help Q-Bot to narrow the scope of the target object.

Case Study. We conduct case studies to investigate the effect of dialogue strategies in Figure 6. In the first Q-Bot-A-Bot dialogue, the questioner successfully complete the task by gradually narrowing down the candidates. In the second Q-Bot-A-Bot dialogue, q_4 and q_5 repeatedly confirm the existence of soccer, failing to collect more information.

6 Conclusion

In this paper, we propose a cooperative object-referring game – *Spot the Difference*, where the goal is to locate the different object between two similar images via conversing between questioner and answerer. The task addresses two challenges at visual dialogue in non-perfectly co-observable scene, including the difference-oriented dialogue strategy and the ability of categorization. We construct a multimodal large-scale dataset *SpotDiff*, which contains 49k VR images and 94k dialogues. Additionally, we provide strong benchmark models and conduct extensive experiments to analyze the two key challenges.

References

Peter Anderson, Xiaodong He, Chris Buehler, Damien Teney, Mark Johnson, Stephen Gould, and Lei Zhang. 2018. Bottom-up and top-down attention for image captioning and visual question answering.

Jeffrey P. Bigham, Chandrika Jayant, Hanjie Ji, Greg Little, Andrew Miller, Rob Miller, Rob Miller, Aubrey Tatarowicz, Brandyn Allen White, Samuel White, and Tom Yeh. 2010. Vizwiz: nearly real-time answers to visual questions. *Proceedings of the 23nd annual ACM symposium on User interface software and technology*.

Rémi Cadène, Hedi Ben-younes, Matthieu Cord, and Nicolas Thome. 2019. MUREL: multimodal relational reasoning for visual question answering. *CoRR*, abs/1902.09487.

Feilong Chen, Fandong Meng, Jiaming Xu, Peng Li, Bo Xu, and Jie Zhou. 2020. DMRM: A dual-channel multi-hop reasoning model for visual dialog. *Thirty-Fourth AAAI Conference on Artificial Intelligence*.

Michael Cogswell, Jiasen Lu, Rishabh Jain, Stefan Lee, Devi Parikh, and Dhruv Batra. 2020. Dialog without dialog data: Learning visual dialog agents from VQA data. *CoRR*, abs/2007.12750.

Abhishek Das, Satwik Kottur, Khushi Gupta, Avi Singh, Deshraj Yadav, José M. F. Moura, Devi Parikh, and Dhruv Batra. 2017a. Visual dialog. In *2017 IEEE Conference on Computer Vision and Pattern Recognition, CVPR 2017, Honolulu, HI, USA, July 21-26, 2017*, pages 1080–1089. IEEE Computer Society.

Abhishek Das, Satwik Kottur, José M. F. Moura, Stefan Lee, and Dhruv Batra. 2017b. Learning cooperative visual dialog agents with deep reinforcement learning. In *IEEE International Conference on Computer Vision, ICCV 2017, Venice, Italy, October 22-29, 2017*, pages 2970–2979. IEEE Computer Society.

Harm de Vries, Florian Strub, Sarath Chandar, Olivier Pietquin, Hugo Larochelle, and Aaron C. Courville. 2017. Guesswhat?! visual object discovery through multi-modal dialogue. In *2017 IEEE Conference on Computer Vision and Pattern Recognition, CVPR 2017, Honolulu, HI, USA, July 21-26, 2017*, pages 4466–4475. IEEE Computer Society.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2019. Bert: Pre-training of deep bidirectional transformers for language understanding.

Carolina Galleguillos, Andrew Rabinovich, and Serge Belongie. 2008. Object categorization using co-occurrence, location and appearance. In *2008 IEEE Conference on Computer Vision and Pattern Recognition*, pages 1–8.

Zhe Gan, Yu Cheng, Ahmed El Kholy, Linjie Li, Jingjing Liu, and Jianfeng Gao. 2019. Multi-step reasoning via recurrent dual attention for visual dialog. *CoRR*, abs/1902.00579.

Yash Goyal, Tejas Khot, Douglas Summers-Stay, Dhruv Batra, and Devi Parikh. 2016. Making the V in VQA matter: Elevating the role of image understanding in visual question answering. *CoRR*, abs/1612.00837.

Xiaoxiao Guo, Hui Wu, Yu Cheng, Steven Rennie, Gerald Tesauro, and Rogerio Schmidt Feris. 2018. Dialog-based interactive image retrieval.

Janosch Haber, Tim Baumgärtner, Ece Takmaz, Lieke Gelderloos, Elia Bruni, and Raquel Fernández. 2019. The PhotoBook dataset: Building common ground through visually-grounded dialogue. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 1895–1910, Florence, Italy. Association for Computational Linguistics.

Nikolai Ilinykh, Sina Zarrieß, and David Schlangen. 2019. Meetup! a corpus of joint activity dialogues in a visual environment.

Satwik Kottur, Seungwhan Moon, Alborz Geramifard, and Babak Damavandi. 2021. Simmc 2.0: A task-oriented dialog dataset for immersive multimodal conversations.

Satwik Kottur, José M. F. Moura, Devi Parikh, Dhruv Batra, and Marcus Rohrbach. 2019. Clevr-dialog: A diagnostic dataset for multi-round reasoning in visual dialog.

Ranjay Krishna, Yuke Zhu, Oliver Groth, Justin Johnson, Kenji Hata, Joshua Kravitz, Stephanie Chen, Yannis Kalantidis, Li-Jia Li, David A. Shamma, Michael S. Bernstein, and Fei-Fei Li. 2016. Visual genome: Connecting language and vision using crowdsourced dense image annotations.

Sang-Woo Lee, Tong Gao, Sohee Yang, Jaejun Yoo, and Jung-Woo Ha. 2019. Large-scale answerer in questioner's mind for visual dialog question generation.

Zujie Liang, Huang Hu, Can Xu, Chongyang Tao, Xiubo Geng, Yining Chen, Fan Liang, and Daxin Jiang. 2021. Maria: A visual experience powered conversational agent.

José Lopes, Nils Hemmingsson, and Oliver Åstrand. 2018. The spot the difference corpus: a multi-modal corpus of spontaneous task oriented spoken interactions.

Seungwhan Moon, Satwik Kottur, Paul A. Crook, Ankita De, Shivani Poddar, Theodore Levin, David Whitney, Daniel Difranco, Ahmad Beirami, Eunjoon Cho, Rajen Subba, and Alborz Geramifard. 2020a. Situated and interactive multimodal conversations. *CoRR*, abs/2006.01460.

Seungwhan Moon, Satwik Kottur, Paul A. Crook, Ankita De, Shivani Poddar, Theodore Levin, David Whitney, Daniel Difranco, Ahmad Beirami, Eunjoon Cho, Rajen Subba, and Alborz Geramifard. 2020b. Situated and interactive multimodal conversations.

Vishvak Murahari, Prithvijit Chattopadhyay, Dhruv Batra, Devi Parikh, and Abhishek Das. 2019. Improving generative visual dialog by answering diverse questions.

Yulei Niu, Hanwang Zhang, Manli Zhang, Jianhong Zhang, Zhiwu Lu, and Ji-Rong Wen. 2018. Recursive visual attention in visual dialog. *CoRR*, abs/1812.02664.

Alec Radford, Jeff Wu, Rewon Child, David Luan, Dario Amodei, and Ilya Sutskever. 2019. Language models are unsupervised multitask learners.

Shaoqing Ren, Kaiming He, Ross Girshick, and Jian Sun. 2016. Faster r-cnn: Towards real-time object detection with region proposal networks.

Eleanor Rosch and Barbara Bloom Lloyd. 1978. Cognition and categorization.

Ravi Shekhar, Aashish Venkatesh, Tim Baumgärtner, Elia Bruni, Barbara Plank, Raffaella Bernardi, and Raquel Fernández. 2019. Beyond task success: A closer look at jointly learning to see, ask, and GuessWhat. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pages 2578–2587, Minneapolis, Minnesota. Association for Computational Linguistics.

Pushkar Shukla, Carlos Elmadjian, Richika Sharan, Vivek Kulkarni, Matthew Turk, and William Yang Wang. 2019. What should I ask? using conversationally informative rewards for goal-oriented visual dialog. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 6442–6451, Florence, Italy. Association for Computational Linguistics.

Qingyi Si, Zheng Lin, Mingyu Zheng, Peng Fu, and Weiping Wang. 2021. Check it again: Progressive visual question answering via visual entailment. *CoRR*, abs/2106.04605.

Florian Strub, Harm de Vries, Jérémie Mary, Bilal Piot, Aaron C. Courville, and Olivier Pietquin. 2017a. End-to-end optimization of goal-driven and visually grounded dialogue systems. *CoRR*, abs/1703.05423.

Florian Strub, Harm de Vries, Jérémie Mary, Bilal Piot, Aaron C. Courville, and Olivier Pietquin. 2017b. End-to-end optimization of goal-driven and visually grounded dialogue systems. In *Proceedings of the Twenty-Sixth International Joint Conference on Artificial Intelligence, IJCAI 2017, Melbourne, Australia, August 19-25, 2017*, pages 2765–2771. ijcai.org.

Ece Takmaz, Mario Giulianelli, Sandro Pezzelle, Arabella Sinclair, and Raquel Fernández. 2020. Refer, reuse, reduce: Generating subsequent references in visual and conversational contexts.

Hao Tan and Mohit Bansal. 2019. Lxmert: Learning cross-modality encoder representations from transformers.

Unity Technologies. 2019. Unity. `https://unity.com/`.

Unity Technologies. 2020. Unity Perception package. `https://github.com/Unity-Technologies/com.unity.perception`.

Yue Wang, Shafiq R. Joty, Michael R. Lyu, Irwin King, Caiming Xiong, and Steven C. H. Hoi. 2020. VD-BERT: A unified vision and dialog transformer with BERT. *CoRR*, abs/2004.13278.

Yonghui Wu, Mike Schuster, Zhifeng Chen, Quoc V. Le, Mohammad Norouzi, Wolfgang Macherey, Maxim Krikun, Yuan Cao, Qin Gao, Klaus Macherey, Jeff Klingner, Apurva Shah, Melvin Johnson, Xiaobing Liu, Łukasz Kaiser, Stephan Gouws, Yoshikiyo Kato, Taku Kudo, Hideto Kazawa, Keith Stevens, George Kurian, Nishant Patil, Wei Wang, Cliff Young, Jason Smith, Jason Riesa, Alex Rudnick, Oriol Vinyals, Greg Corrado, Macduff Hughes, and Jeffrey Dean. 2016. Google's neural machine translation system: Bridging the gap between human and machine translation.

Z. Xu, F. Feng, X. Wang, Y. Yang, and H. Jiang. 2020. Answer-driven visual state estimator for goal-oriented visual dialogue. In *ACM MM '20: The 28th ACM International Conference on Multimedia*.

Zhou Yu, Jing Li, Tongan Luo, and Jun Yu. 2020. A pytorch implementation of bottom-up-attention. `https://github.com/MILVLG/bottom-up-attention.pytorch`.

Junjie Zhang, Qi Wu, Chunhua Shen, Jian Zhang, Jianfeng Lu, and Anton van den Hengel. 2017. Asking the difficult questions: Goal-oriented visual question generation via intermediate rewards. *CoRR*, abs/1711.07614.

Rui Zhao and Volker Tresp. 2018. Learning goal-oriented visual dialog via tempered policy gradient. In *2018 IEEE Spoken Language Technology Workshop (SLT)*, pages 868–875.

Mingyang Zhou, Josh Arnold, and Zhou Yu. 2019. Building task-oriented visual dialog systems through alternative optimization between dialog policy and language generation.

A Image Generation

A.1 Taxonomy Information

We present the taxonomy information with a predefined hierarchical tree structure, which is illustrated in Table 7.

A.2 Placement Relationship

We empirically construct a directed graph of object placement relationship. As shown in Table 8, it describes which category of objects could be placed on a object of specific category.

A.3 Spatial Arrangement

Given an object o (the object to be placed), a rectangular area and existing objects, we put the unplaced object on the area as follows:

1) Randomly sample T points on the area.
2) Filter out the points whose distance from any existing object is less than X or that cross the boundary.
3) Select the point with the minimum L1 distance to its closet existing object, and place the object o at the point.

After taking screenshots, we retain the image where the pixels of objects in the image are all larger than Y, to avoid the serious mutual occlusion between objects in the image.

A.4 Object Co-Occurrence

Considering the hierarchical tree structure of categories, we define the degree of divergence d_u for category u as:

$$d_u = \sum_{v \in child(u)} [\exists o \in O \wedge v \in f(o)], \quad (4)$$

where $child(u)$ means the child categories of the category u (e.g., $child(fruit) = \{apple, banana\}$), O is the object list of the image, $f(o)$ is the category set corresponding the object o (e.g., for an apple, its category set is $\{apple, fruit, food\}$), $[\cdot]$ is 1 if and only the expression in the bracket is True.

To make the related objects co-occur with high probability, for each category u, we limit d_u not to exceed K=3.

B Dialog Generation

B.1 Answer Action

The answer is divided into two types: 1) Count answer, which corresponds to the count question, gives the number of objects with specific conditions in the image. 2) Description answer responds to the extreme and ref questions, and describes one or multiple objects in natural language, e.g., ‘Black frame’, ‘A decorative plate, a nightstand and a plant’.

#	QGen	A-Bot	SUCC ↑
1	GPT-2	UpDn	27.39
2	GPT-2	LXMERT	28.21
3	LXMERT	UpDn	30.82
4	LXMERT	LXMERT	35.17

Table 5: The performance of the dialogue system. SUCC: task success rate (%). ↑: higher is better.

C Experiments

C.1 Implementation Details

We implement our method with Pytorch and conduct all experiments on four NVIDIA Tesla V100 GPU. For all models, we use Adam optimizer with a learning rate of 5e-5 and a mini-batch size of 32. We train QGen, A-Bot, Guesser for 10, 8, 30 epochs. For A-Bot and Guesser, we select the models with best accuracy on the val set. For QGen, we select the best performed model on the val set, under the game setting.

C.2 Dialogue System Performance Comparison

We implement different models for this task.

QGen. 1) GPT-2 (Radford et al., 2019): A decoder-only model with the pretrained language model GPT-2 as the backbone; 2) LXMERT (Tan and Bansal, 2019): Our benchmark QGen.

A-Bot. 1) UpDn (Anderson et al., 2018): A representative VQA model with attention mechanism; 2) LXMERT: Our benchmark A-Bot.

As shown in Figure C.2, row 4 (QGen: LXMERT, A-Bot: LXMERT) achieves the best performance among all comparing methods, demonstrating the superiority of multimodal pretrained model.

C.3 A-Bot Performance Comparison

We compare UpDn (Anderson et al., 2018) to LXMERT (Tan and Bansal, 2019) under VQA setting. As shown in Table C.3, LXMERT outperforms UpDn on all question subtypes, demonstrating the superiority of multimodal pretrained model.

QTYPE	count-1	count-2	extreme-1	extreme-2	extreme-3	ref-1	ref-2	all
UpDn	70.32	76.07	77.19	63.96	62.02	86.49	70.67	73.40
LXMERT	81.96	85.03	84.60	78.90	72.74	89.02	81.96	83.18

Table 6: A-Bot performance on various question subtypes. QTYPE means the question subtype.

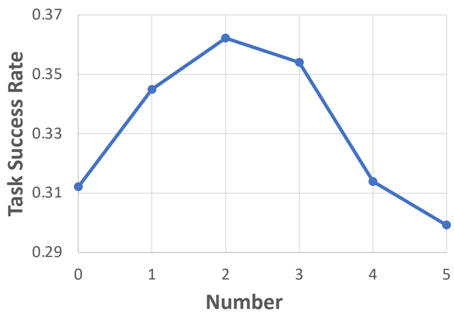


Figure 8: The relationship between the task success rate and the number of Cate-Q.

C.4 Number of Cate-Q

We divide the generated dialogues according to the number of Cate-Q in a dialogue, and examine the task success rate within each group. The results are shown in Figure 8. When the number increases from 0 to 2, the task success rate shows an increasing trend, demonstrating that Cate-Q could help model improve the performance; when the number increases from 2 to 5, the task success rate shows a downward trend. This may due to the failure of the Cate-Q to narrow down the candidate, resulting in waste of dialogue rounds.

D Examples

D.1 Image Scene Graph

We present an example of image scene graph in Figure 9.

D.2 *SpotDiff* Examples

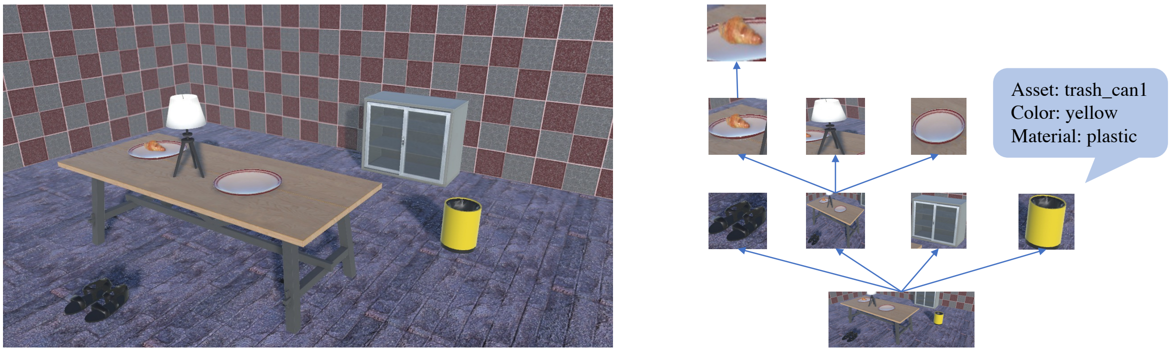
As shown in Figure 10, we give more random examples of *SpotDiff*.

category	subcategories
home appliance	large household appliance, small household appliance
large household appliance	fridge, television, floor lamp, washing machine
small household appliance	coffee machine, desk lamp
furniture	table, chair, bench, sofa, nightstand, baby bed, cabinet, carpet, cloth tree, bed
table	dining table, tea table, study table
toy	animal toy, toy model
animal toy	teddy bear, elephant toy, bunny toy, giraffe toy
toy model	car model, airplane model
food	fruit, drink, meat product, baked food
fruit	apple, banana
drink	cola, milk, tea, beer
baked food	bread, pizza
meta product	chicken leg, chicken nugget
sporting goods	ball, sports equipment
ball	soccer, basketball, tennis, bowling pin
sports equipment	bow, dumbbell, baseball bat, archery target, skateboard
kitchenware	tableware, kettle
tableware	plate, cup, fork, spoon
office supply	stationery, office equipment, paper product
stationery	pencil, palette
paper product	paperbox, notebook
office equipment	computer, mouse, keyboard, headphone, plug plate, phone
computer	laptop, desktop
decoration	vase, decorative plate, frame
fashion item	fashion accessory, shoes, backpack
fashion accessory	glasses, hat
shoes	boots, sandals, canvas shoes
hat	cotton cap, top hat, baseball cap

Table 7: The taxonomy information. The first column gives the category while the second column gives its corresponding subcategories.

floor	furniture, shoes, fridge, floor lamp, trash can, plant, table
dining table	kitchenware, drink, pizza, small household appliance, plate
tea table	decoration, book, television, cup
study table	book, office supply, toy, sporting goods
carpet	tea table, toy, sporting goods, backpack
plate	fruit, bread, meat product
nightstand	decoration, cup, glasses, hat
cabinet	decoration

Table 8: The placement relationship. The first column represents the category of objects, and the second column represents the categories that could be placed on objects of the category (in the first column).



(a) *SpotDiff* Image (b) Image Scene Graph

Figure 9: An Example of image scene graph. (a) gives a *SpotDiff* image and (b) displays its corresponding scene graph, where blue lines indicate placement relationships between objects.

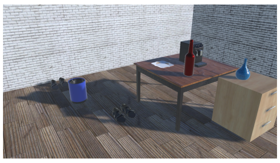
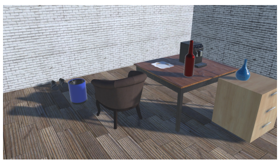
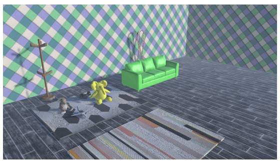
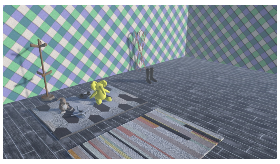
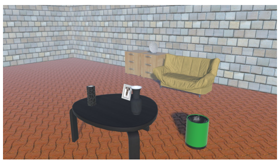
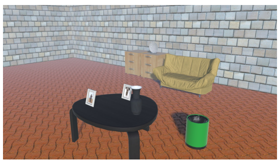
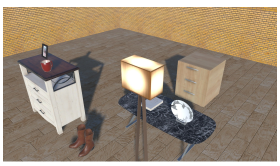
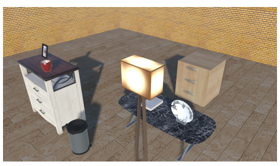
I^Q	I^A	*SpotDiff* Dialogue
		q_1: I can see three black objects, and you? a_1: Two q_2: I have two pairs of black sandals, how about you? a_2: One q_3: There is a pair of black sandals on the far behind of the image, and you? a_3: Pair of black sandals
		q_1: I can see five furnitures, and you? a_1: Four q_2: I want to know the number of cloth rugs in your picture. a_2: Two q_3: I can see a rose gold metal hanger, and you? a_3: One q_4: I can see a green leather sofa that can accommodate three people, and you? a_4: Zero
		q_1: I want to know the number of decorations in your picture. a_1: Four q_2: I have four furnitures, how about you? a_2: Four q_3: How many black objects do you have? a_3: Two q_4: I have one more than you. there is an irregularly shaped black wooden table on the far left of the image, and you? a_4: Irregularly shaped black wooden table q_5: I have a black straight vase, how about you? a_5: Zero
		q_1: How many furnitures are there? a_1: Three q_2: So do I. There are three white objects in my picture, what about you? a_2: Three q_3: Can you tell me what they are? a_3: A book, a decorative plate and a nightstand q_4: There are three brown objects in my picture, what about you? a_4: Two q_5: There are some differences. can you tell me what they are? a_5: A floor lamp and a nightstand

Figure 10: Random Examples of *SpotDiff* test set. Green sentences: questions, blue sentences: answers.