

## Appendix

### Acknowledgements

This work was supported by JST CREST Grant Number JPMJCR19A1, Japan, and JSPS KAKENHI Grant Number JP21J13789, Japan. The experiments were partially conducted on AI Bridging Cloud Infrastructure (ABCI) provided by National Institute of Advanced Industrial Science and Technology (AIST). The training progress was tracked and the reports were created with Weight & Biases. Finally, we thank David Felices for helpful comments and discussion.

### A. UniQer Model Details

**Object Targeting Module (§4.3).** We show an overview of the Object Targeting Module (OTM) with an example when  $k = 3$  in Fig. 4. The OTM groups objects  $\mathcal{O}$  into three groups: the target object group  $\mathcal{O}_t$ , the distracter object group  $\mathcal{O}_d$ , and the masked object group  $\mathcal{O}_m$ . Examples of the object groups and the corresponding questions are shown in Fig. 5.

### B. Evaluation Metrics

**Perfect Address and Correct Address (§6.1).** Given a question  $q$  and an answer  $a$  from the *Oracle*,  $\mathcal{O}$  will be divided into two object groups in mutually exclusive and collectively exhaustive manner: a matched group  $\mathcal{O}_q$  for objects that matches  $(q, a)$  and an unmatched group  $\overline{\mathcal{O}}_q$  otherwise. In order for a question to be perfect or correct, the target objects  $\mathcal{O}_t$  and the distracter objects  $\mathcal{O}_d$  must be separately belongs to matched or unmatched group, *i.e.*, it must fulfill either  $\mathcal{O}_t \in \mathcal{O}_q \wedge \mathcal{O}_d \in \overline{\mathcal{O}}_q$  or  $\mathcal{O}_t \in \overline{\mathcal{O}}_q \wedge \mathcal{O}_d \in \mathcal{O}_q$ . What makes the difference between perfect and correct is whether the masked objects are mixed with the target objects. That is, when the target objects belong to the matched group  $\mathcal{O}_t \in \mathcal{O}_q$ , the case  $\mathcal{O}_q \cap \mathcal{O}_m = \{\phi\}$  is perfect and, otherwise, correct. Fig. 5 also gives examples of perfect questions (*ex.1* and *ex.4*) and correct questions (*ex.2* and *ex.3*). In *ex.3*, objects are grouped as  $\mathcal{O}_t = \{\text{green\_cylinder}\}$ ,  $\mathcal{O}_d = \{\text{green\_sphere}\}$ , and  $\mathcal{O}_m = \{\text{red\_cylinder}\}$ . The question in *ex.3* is correct, since it will divide the objects as  $\mathcal{O}_q = \{\text{green\_cylinder, red\_cylinder}\}$  and  $\overline{\mathcal{O}}_q = \{\text{green\_sphere}\}$ , while in *ex.4* is perfect, since the question addresses to the target objects without including the masked object as  $\mathcal{O}_q = \{\text{green\_cylinder}\}$  and  $\overline{\mathcal{O}}_q = \{\text{green\_sphere, red\_cylinder}\}$ .

### C. Learning Details

**Policy Gradient Optimization (§5.2).** In accordance with the existing studies, we adopt the policy gradient method [8]

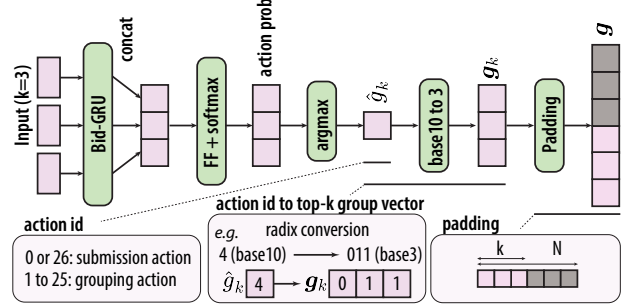


Figure 4: Overview of the Object Targeting Module (OTM). In this figure, an example process of when  $k = 3$  and  $N = 6$  is demonstrated. The input of the bidirectional GRU is the object features in order of decreasing confidence.

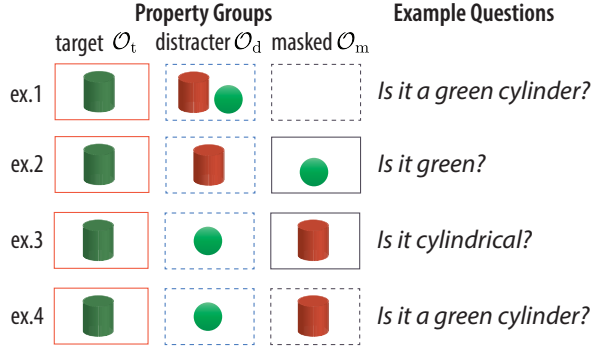


Figure 5: Example of possible questions given objects with three property groups; the target object group  $\mathcal{O}_t$ , the distracter object group  $\mathcal{O}_d$ , and the masked object group  $\mathcal{O}_m$ . Given such object groups from the OTM, the QDT is required to generate questions that discriminate target objects from distracter objects. Groups with solid line represent matched objects if an answer to a question is *yes* and groups with dotted lines represent unmatched objects. Here, *ex.1* and *ex.4* are the perfect questions, and *ex.2* and *ex.3* are the correct questions.

for optimizing the *OTM*. The objective function will be

$$J(\theta) = \mathbb{E}_{\pi_\theta} \left[ \sum_{t=1}^T r(S_t, A_t) \right], \quad (10)$$

where  $\pi_\theta$  represents the policy function parameterized with  $\theta$ .

Since policy gradient approach has a good convergence on high-dimensional action space, it is suitable for our purposes. The goal of policy gradient is to find the policy parameter  $\theta$  that gives better expected rewards via gradient ascent. With the episodic settings, the *Questioner* engages in multiple trajectories with length  $T$  defined as  $\tau = (S_0, A_0, R_0, \dots, S_{T-1}, A_{T-1}, R_{T-1}, R_T)$ . Given the

---

**Algorithm 1: UniQer** RL flow for a scene

---

**Data:**  $\mathcal{O}$

```
1 The Oracle picks an goal object  $o^* \in \mathcal{O}$ 
2  $q = [], a = []$  ▷ questions & answers
3 for  $t = 1$  to  $T$  do ▷ dialogue loop
4   ▷ Prepare embeddings ▷ see §4.1
5    $\mathbf{x}_v \leftarrow \mathcal{F}_v([\mathbf{o}_v(i), \boldsymbol{\sigma}_g^{\mathcal{N}}(i)]_{i \in \mathcal{N}})$ 
6    $\mathbf{x}_l^t \leftarrow [[\text{CLS}], w_1^t, w_2^t, \dots, w_{W_1}^t, a^1, w_1^t, \dots, a^{t-1}]$ 
7    $\mathbf{x}_h \leftarrow [\mathbf{x}_v, \mathbf{x}_l^t]$ 
8    $\mathbf{x}_e \leftarrow \mathbf{x}_h + \mathbf{x}_{\text{seg}} + \mathbf{x}_{\text{pos}}$ 
9   ▷ OET ▷ see §4.2
10   $\{\tilde{\mathcal{X}}_o, \tilde{\mathcal{X}}_{[\text{CLS}]}, \tilde{\mathcal{X}}_i\} \leftarrow \text{OET}(\mathbf{x}_e)$ 
11   $\sigma_o \leftarrow [\text{sigmoid}(\mathcal{F}_o(\tilde{\mathcal{X}}_o^i)) \cdot \mathcal{F}_c(\tilde{\mathcal{X}}_{[\text{CLS}]})]_{i \in \mathcal{N}}$ 
12   $P_{\hat{o}} \leftarrow \text{softmax}(\sigma_o)$ 
13  ▷ OTM ▷ see §4.3
14  Get top-k object ids:  $\mathcal{K} \leftarrow \text{argtopk}(P_{\hat{o}})$ 
15   $\mathbf{x}_k \leftarrow [\mathcal{F}_A(\mathbf{o}_v(i)), \mathcal{F}_B(\boldsymbol{\sigma}_g^{\mathcal{K}}(i)), \mathcal{F}_C(P_{\hat{o}}(i))]_{i \in \mathcal{K}}$ 
16   $A_t \leftarrow \text{sampling}(\mathcal{F}_{\text{RL}}(\mathbf{x}_k))$ 
17   $\mathbf{g}_k \leftarrow \text{ternary}(A_t)$  ▷  $\mathbf{g}_k \in \mathbb{R}^{3 \times k}$ 
18   $\mathbf{g} \leftarrow \text{padding}(\mathbf{g}_k)$  ▷  $\mathbf{g} \in \mathbb{R}^{3 \times \mathcal{N}}$ 
19  if  $A_t \in \{0, 3^k - 1\}$ : ▷ end of dialogue (EOD)
20     $r(S_t, A_t) \leftarrow \begin{cases} 1 - r_d(t) & \text{if } \text{argmax}(P_{\hat{o}}) = o_{id}^* \\ 0 & \text{otherwise} \end{cases}$ 
21    break
22  elif  $t == T$ : ▷ EOD should have been generated
23     $r(S_t, A_t) \leftarrow 0$ 
24  else:
25    ▷ QDT ▷ see §4.4
26     $\mathcal{M} \leftarrow \tilde{\mathcal{X}}_o + \mathcal{F}_s(\mathbf{g})$ 
27     $[w_1^t, w_2^t, \dots, w_{W_t}^t] \leftarrow \text{QDT}([\text{BOS}], \mathcal{M})$ 
28    append  $[w_1^t, w_2^t, \dots, w_{W_t}^t]$  to  $q$ ,  $a^t$  to  $a$ 
29  end
30 end
```

---

trajectories, the gradients of the objective function will be approximated by introducing **REINFORCE** algorithm [9] as follows

$$\nabla_{\theta} J(\theta) \approx \mathbb{E}_{\pi_{\theta}} \left[ \sum_{t=0}^{T-1} \nabla_{\theta} \ln \pi_{\theta}(A_t | S_t) (G(t) - b(S_t)) \right] \quad (11)$$

where  $G(t) = \sum_{t'=t}^{T-1} \gamma^{T-(t'+1)} R_{t'+1}$  is the return function with discount factor  $\gamma$  applied.

**Implementation Details (§6, §7).** *UniQer* was implemented in PyTorch and the *Oracle* was implemented using the engine proposed by [5]. All experiments were conducted using six Quadro RTX 8000 GPUs. The parameter setting for the encoder and the decoder was: `d_model=512`, `n_head=8`, `dim_feedforward=512`, `n_layers=3`, and `dropout=0.1`. As an image feature extractor, we used Imagenet pre-trained ResNet34 [4]. It took approximately two days for *UniQer* to train from scratch on a single GPU.

**Turn Discount Factor  $r_d$  (§7.1).** The turn discount factor  $r_d$ , which reduces the reward based on the number of questions, is introduced to promote a more efficient ques-

tioning strategy by trying to minimize the number of questions. Without this discount factor, the agent tends to repeat the same question it already asked during the dialogue even after it finds the correct goal object. We define the turn discount factor  $r_d$  as:

$$r_d(t) = \beta * \frac{t}{T}, \quad (12)$$

where  $\beta$  is a coefficient that determines the scale of the penalty,  $t$  is the number of generated questions, and  $T$  is the maximum number of questions allowed for the agent. Our experiments were conducted with  $\beta = 0.2$  and  $T = 5$ . Just to be sure, the agent should generate a special token, end of dialogue (EOD), to receive a reward before the number of questions reaches  $T$ . Therefore, our final reward function is described as:

$$r(S_t, A_t) = \begin{cases} 1 - r_d(t) & \text{if } \text{argmax}(P_{\hat{o}}) = o_{id}^* \\ 0 & \text{otherwise} \end{cases} \quad (13)$$

**RL algorithm (§5.2).** The overall procedure of reinforcement learning is shown in Algorithm 1. The detailed explanation of each equation is denoted in the corresponding section.

## D. Baseline Model Details (§7.1)

In a manner similar to that of the previous studies [3, 7], we divide the role of the questioner into the following four components:

- **Question and Answer Encoder (QAE):** An LSTM module encoding the current question and the answer given by the *Oracle*.
- **Dialogue State Encoder (DSE):** An LSTM module encoding past dialogues.
- **Question Generator (QGen):** An LSTM module generating a question based on the DSE’s output and the top-k object features with high  $P_{\hat{o}}$ .
- **Guesser:** An MLP module for outputting candidate probabilities  $P_{\hat{o}}$  using the features of an object as the input.

The Guesser, accompanied by the QAE and the DSE, and the QGen are trained separately in the upstream tasks in a supervised manner, and then merged into a single agent model to conduct reinforcement learning.

**Question and Answer Encoder (QAE)** The QAE encodes the question tokens generated by the QGen and the corresponding answer provided by the Guesser. The encoded features will be passed to the DSE. This function is implemented by a standard LSTM.

**Dialogue State Encoder (DSE)** The DSE generates a dialogue state vector  $\tilde{X}_{\mathcal{D}}$  that holds the history of questions and answers.  $\tilde{X}_{\mathcal{D}}$  is used to both generate the next question token in the QGen and compute the goal object probabilities  $P\hat{o}$  in the Guesser. This function is implemented by a standard LSTM.

**Question Generator (QGen)** Formally, the QGen can be thought of as the probabilistic language model, which sequentially generates a word  $w_l^t$  to compose a question  $q^t$ , given the previous word token  $w_{l-1}^t$ , the dialogue history vector  $\tilde{X}_{\mathcal{D}}$ , the context image feature  $x'_v$  extracted by the image feature extractor and tok-k object feature embedding  $x_k$ . This can be formalized as follows:

$$P(w_{l+1}^t | w_l^t, \tilde{X}_{\mathcal{D}}, x'_v, x_k). \quad (14)$$

We employed the LSTM to implement such functions.

**Guesser** The Guesser guesses the goal object  $o^*$  based on the current scene and the question answer history as:

$$P(\hat{o}^* | x_v, \tilde{X}_{\mathcal{D}}). \quad (15)$$

We implement this function as:

$$P_{\hat{o}} = \text{softmax}([\text{sigmoid}(\mathcal{F}_1(x'_v)) \cdot \mathcal{F}_2(\tilde{X}_{\mathcal{D}})])_{i \in \mathcal{N}}, \quad (16)$$

where  $\mathcal{F}_1$  and  $\mathcal{F}_2$  are the linear transformation functions. While this operation is similar to Eqs. (3, 4), the other object features are not compared when computing a probability for an object.

**Supervised Learning** The loss function of the Guesser is as same as the object prediction loss defined in Eq. (8), where the QAE and the DSE will be jointly trained.

The loss function of the QGen is described as:

$$L_{\text{gen}} = - \sum_{t=1}^T \sum_{l=1}^{W_t} \log p(w_{l+1}^t | w_l^t, \mathbf{h}), \quad (17)$$

where  $\mathbf{h}$  is the hidden vector of the model,  $T$  is the maximum number of questions in a dialogue, and  $W_t$  is the number of tokens included in the  $t$ -th question.

**Reinforcement Learning** Since the baseline model produces a word token on each iterative step, the global timestep for the baseline model  $t'$  is defined as  $t' = \sum_{\tau=1}^{t-1} |\mathcal{D}_{\tau}| + l$ , where  $|\mathcal{D}_{\tau}|$  is the length of a past dialogue and  $l$  is the step in the current dialogue. The set of actions  $A_{t'}$  corresponds to the tokens in the vocabulary  $\mathcal{V}$ . The state is defined as follows:

$$S_{t'} = (\mathcal{I}, (q^{\tau}, a^{\tau})_{1:t-1}, (w_1^t, \dots, w_l^t)). \quad (18)$$

The transition to the next state depends on the selected action:

- If  $A_{t'+1} = \langle \text{EOD} \rangle$ , the dialogue terminates. Thus,  $S_{t'+1}$  becomes the last state.

- If  $A_{t'+1} = \langle \text{EOS} \rangle$ , the current question generation terminates and the *Questioner* receives an answer  $a^t$ . The next state will be  $S_{t'+1} = (\mathcal{I}, (q^{\tau}, a^{\tau})_{1:t})$ .

- Otherwise, if the generation of the question continues, the next state will be  $S_{t'+1} = (\mathcal{I}, (q^{\tau}, a^{\tau})_{1:t-1}, (w_1^t, \dots, w_l^t, w_{l+1}^t))$

The model will be trained with policy gradient optimization as introduced in Eq. (10).

## E. Supplementary Results (§7)

**Average Vocabulary of Questions (§7.2, §7.3).** Since our CLEVR Ask task comprises images that include multiple identical objects, descriptive questions are a requirement for task success by nature. Therefore, the descriptiveness of the questions can be evaluated by how well the model performed in such an environment (*i.e.* the task success ratio).

The descriptiveness can be also measured as the number of a question’s unique attributes, such as colors, sizes, and spatial relations. We therefore introduced two metrics:  $n_{\text{vocab}}$ , which shows the average vocabulary size of the questions, and  $\bar{n}_{\text{vocab}}$ , which shows the average vocabulary size of the dialogues. They are defined as

$$n_{\text{vocab}} = \frac{1}{N_{\text{data}}} \sum_{\mathcal{D} \in \text{data}} \frac{1}{T_{\mathcal{D}}} \sum_{t=1}^{T_{\mathcal{D}}} |\{w_{\omega}^t\}_{\omega=1}^{W_t}|, \quad (19)$$

$$\bar{n}_{\text{vocab}} = \frac{1}{N_{\text{data}}} \sum_{\mathcal{D} \in \text{data}} \frac{1}{T_{\mathcal{D}}} \left| \bigcap_{t=1}^{T_{\mathcal{D}}} \{w_{\omega}^t\}_{\omega=1}^{W_t} \right|, \quad (20)$$

where  $N_{\text{data}}$  is a number of data samples and  $T_{\mathcal{D}}$  is a number of questions in a dialogue  $\mathcal{D}$ .

Summarized results are shown in Tab. 4. The question mean in Ask3 was 1.48 for *Baseline* and 3.32 for *Ours (full)*, while the dialogue mean was 3.15 for *Baseline* and 3.84 for *Ours (full)*. The results indicate that, when asking a question, *UniQer* uses more than three attributes in its questions, *e.g.* “*Is it to the LEFT of a RED CUBE?*”, while the baseline asks trivial questions with a single attribute, *e.g.* “*Is it a SPHERE?*” They also show that *UniQer* has the richest vocabulary in a dialogue. The results in Ask4 are less significant compared with Ask3, while *UniQer* still uses more than one attributes in a question. The question mean and the dialogue mean were highest with *Ours (v)* condition. However, as shown by the task success ratio presented in Tab. 2, the generated questions were not as effective as with *UniQer*.

**Force-Stop Conditions (§7.2, §7.3).** Additional results on the force-stop condition is presented in Tab. 5. The result shows that our model outperformed in the force-stop condition as well. Compared with the results obtained without force-stop conditions, which were presented in Tab. 2, the

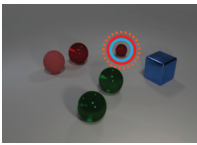
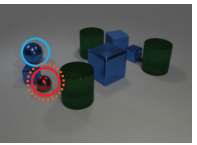
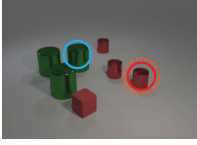
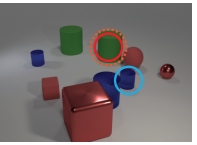
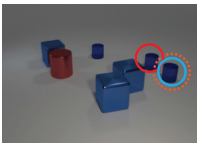
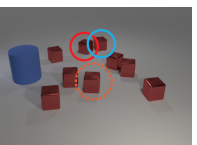
Baseline		UniQer		Baseline		UniQer	
	Is it behind a sphere? Y Is it to the right of a sphere? Y Is it a sphere? Y [Submit] Success	Is it a green glass sphere? N Is it to the left of a green glass sphere? N Is it a red sphere? Y Is it a red rubber sphere? N [Submit] Success			Is it behind a cylinder? Y Is it to the right of a cylinder? N Is it a cylinder? N [Submit] Fail	Is it green? N Is it behind a blue metal cube? N Is it to the right of a blue metal cube? Y Is it to the left of a blue metal cube? Y [Submit] Success	
	Is it behind a cylinder? Y Is it to the right of a cylinder? Y Is it a cylinder? Y [Submit] Fail	Is it a metal cylinder? Y Is it to the left of a red metal cylinder? N Is it behind a red metal cylinder? N Is it a green metal cylinder? N [Submit] Success			Is it behind a cylinder? Y Is it to the right of a cylinder? Y Is it a cylinder? Y [Submit] Fail	Is it to the right of a shiny cube? Y Is it a cylinder? Y Is it a blue glass cylinder? N Is it a green cylinder? Y [Submit] Success	
	Is it behind a cylinder? N Is it a cylinder? Y [Submit] Success	Is it blue? Y Is it in front of a blue glass cylinder? Y Is it a blue glass cylinder? Y Is it to the right of a blue glass cylinder? Y [Submit] Fail			Is it behind a cube? Y Is it to the right of a cube? Y Is it a cube? Y [Submit] Fail	Is it a shiny red cube? Y Is it to the left of a shiny red cube? Y Is it in front of a shiny red cube? Y Is it in front of a shiny red cube? Y [Submit] Fail	

Figure 6: Additional qualitative samples from *Baseline* and *UniQer*.

Model	Question Mean ( $n_{\text{vocab}}$ )		Dialogue Mean ( $\bar{n}_{\text{vocab}}$ )	
	New Img	New Obj	New Img	New Obj
Ask3				
Baseline	1.48±0.27	1.48±0.27	3.15±0.08	3.15±0.07
Ours(v)	2.62±0.04	2.58±0.08	3.54±0.13	3.54±0.16
Ours(num)	1.54±0.04	1.52±0.03	2.92±0.11	2.98±0.12
Ours(nu)	1.51±0.02	1.47±0.07	3.11±0.17	3.12±0.12
Ours(full)	3.32±0.16	3.35±0.08	3.82±0.12	3.84±0.20
Ask4				
Baseline	1.00±0.00	1.00±0.00	3.03±0.06	3.05±0.03
Ours(v)	2.13±0.05	2.10±0.04	3.44±0.24	3.47±0.22
Ours(num)	1.56±0.03	1.60±0.08	3.12±0.09	3.09±0.05
Ours(nu)	1.56±0.04	1.59±0.05	3.19±0.08	3.13±0.06
Ours(full)	1.49±0.02	1.54±0.05	3.06±0.08	3.06±0.04

Table 4: Question vocabulary for comparative models presented in Tab. 2. Here,  $n_{\text{vocab}}$  shows the average vocabulary size of the questions, and  $\bar{n}_{\text{vocab}}$ , which shows the average vocabulary size of the dialogues.

Model	Ask3		Ask4	
	New Img↑	New Obj↑	New Img↑	New Obj↑
Baseline(fs)	59.78±5.75	60.37±5.71	64.76±1.26	65.02±1.22
Ours(full-fs)	85.06±1.55	85.17±1.47	83.08±0.69	83.70±0.88

Table 5: Comparative results on the task success ratio for the baseline and *UniQer* in reinforcement learning on stop condition. In stop condition, a submission action is not required and the goal object prediction is automatically sent to the oracle at the end of the dialogue.

variance for *UniQer* decreased; however, there were no significant differences.

**Additional Qualitative Samples (§7.4).** Extensive qualitative examples are presented in Fig. 6. From the samples, we can see that *UniQer* made a full use of descriptive question in all scenes, while *Baseline* ended up with generating simple questions. We also find that the question strategy of *Baseline* was nearly fixed; most of the time it was asking relative location question two times in the beginning and simple material question at the end. The results also reveal the *UniQer*’s limitation. It tend to ask extra questions to make its prediction perfect, while the goal object is deemed to be found. This suggests some improvements on answer submission are required in the future works. Additionally, *UniQer* fails when the goal object is surrounded by too many identical objects. In our future work, more complex referring expressions such as “Is it second to the left of *sth*?” will be needed.

## F. GuessWhat?! Results

The supplemental results for *GuessWhat?!* in Tab 6. Surprisingly, *UniQer* achieved the on par results, even with the trivial modifications from *CLEVR Ask* settings. This implies that the model can robustly be applied to the variant datasets. Moreover, the qualitative inspection revealed that *UniQer* managed to generate descriptive questions which were not shown in the previous models, for example, “Is it the book that is closest to the left side of the picture?” and “Is it the second one from the left?”. In the future study, we are planning to report the improved results based on *UniQer* architecture.

## G. Dataset Details (§3)

**Statistics.** Both Ask3 and Ask4 datasets consist of 70K training, 7.5K validation, and 7.5K test images, each of which includes three to ten objects. The number of objects



Models	Task Success Ratio (%)
Baseline (RL)	58.40
[10] [Zhang+ ECCV18]	60.80
[2] [Abbasnejad+ CVPR19]	60.60
[1] [Abbasnejad+ CVPR20]	62.10
[6] [Pang+ AAAI20]	<b>64.44</b>
<b>Ours (UniQer)</b>	<u>62.17</u>

Table 6: **Results for *GuessWhat?!***. The task success ratio for the new image condition.

in Ask3 dataset is 455,216, 48,673, and 48,447, for training, validation and test sets respectively, while for Ask4 is 454,038, 48,317, and 48,674. Ten questions per image are generated yielding 700K, 75K, and 75K questions for training, validation, and test images, respectively. Questions are generated by sampling the question templates given the image scene graph. Each question is validated to ensure it is related to the scene by checking whether it is asking about something that exists in the scene. Note that these questions are only used in supervised learning. Additional statistics for the datasets are available as follows: a distribution of the number of objects Fig. 7, a distribution of the object attributes Fig. 8, and a distribution of the question attributes Fig. 9.

**Scene Examples.** Examples of scenes for both datasets are presented in Fig. 10.

## References

- [1] E. Abbasnejad, I. Abbasnejad, Q. Wu, J. Shi, and A. v. d. Hengel. Gold seeker: Information gain from policy distributions for goal-oriented vision-and-language reasoning. In *CVPR*, pages 13450–13459, 2020.
- [2] E. Abbasnejad, Q. Wu, Q. Shi, and A. v. d. Hengel. What’s to know? uncertainty as a guide to asking goal-oriented questions. In *CVPR*, pages 4155–4164, 2019.
- [3] H. De Vries, F. Strub, S. Chandar, O. Pietquin, H. Larochelle, and A. Courville. Guesswhat?! visual object discovery through multi-modal dialogue. In *CVPR*, pages 5503–5512, 2017.
- [4] K. He, X. Zhang, S. Ren, and J. Sun. Deep residual learning for image recognition. In *CVPR*, pages 770–778, 2016.
- [5] J. Johnson, B. Hariharan, L. van der Maaten, L. Fei-Fei, C. Lawrence Zitnick, and R. Girshick. Clevr: A diagnostic dataset for compositional language and elementary visual reasoning. In *CVPR*, pages 2901–2910, 2017.
- [6] W. Pang and X. Wang. Visual dialogue state tracking for question generation. In *AAAI*, pages 11831–11838, 2020.
- [7] F. Strub, H. De Vries, J. Mary, B. Piot, A. Courville, and O. Pietquin. End-to-end optimization of goal-driven and visually grounded dialogue systems. In *IJCAI*, pages 2765–2771, 2017.
- [8] R. S. Sutton, D. A. McAllester, S. P. Singh, and Y. Mansour. Policy gradient methods for reinforcement learning with function approximation. In *NIPS*, pages 1057–1063, 2000.
- [9] R. J. Williams. Simple statistical gradient-following algorithms for connectionist reinforcement learning. *Machine learning*, 8(3-4):229–256, 1992.
- [10] J. Zhang, Q. Wu, C. Shen, J. Zhang, J. Lu, and A. Van Den Hengel. Goal-oriented visual question generation via intermediate rewards. In *ECCV*, pages 186–201, 2018.

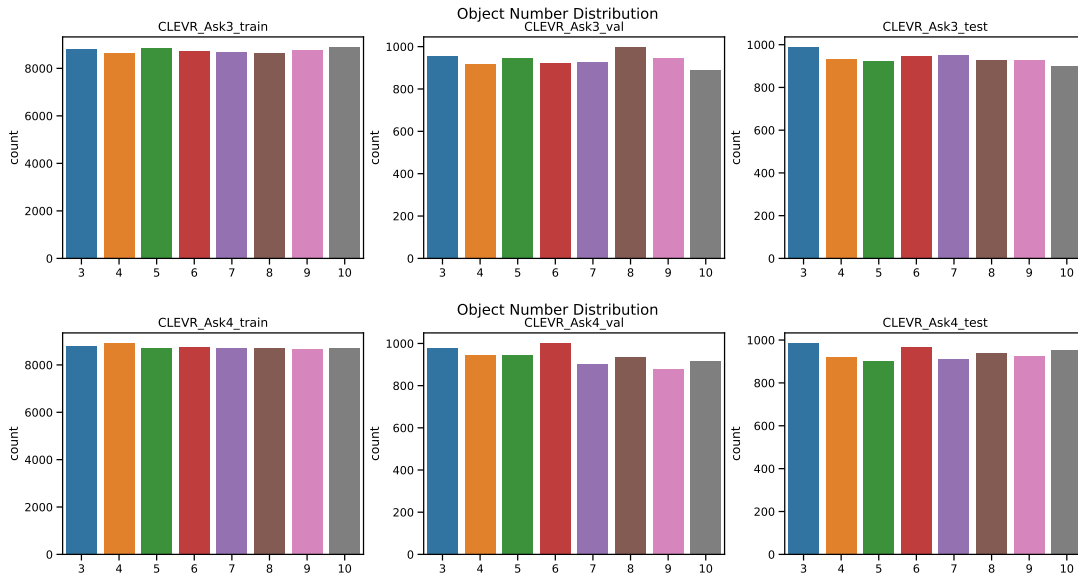


Figure 7: Distribution of the number of objects in Ask3 and Ask4 dataset.

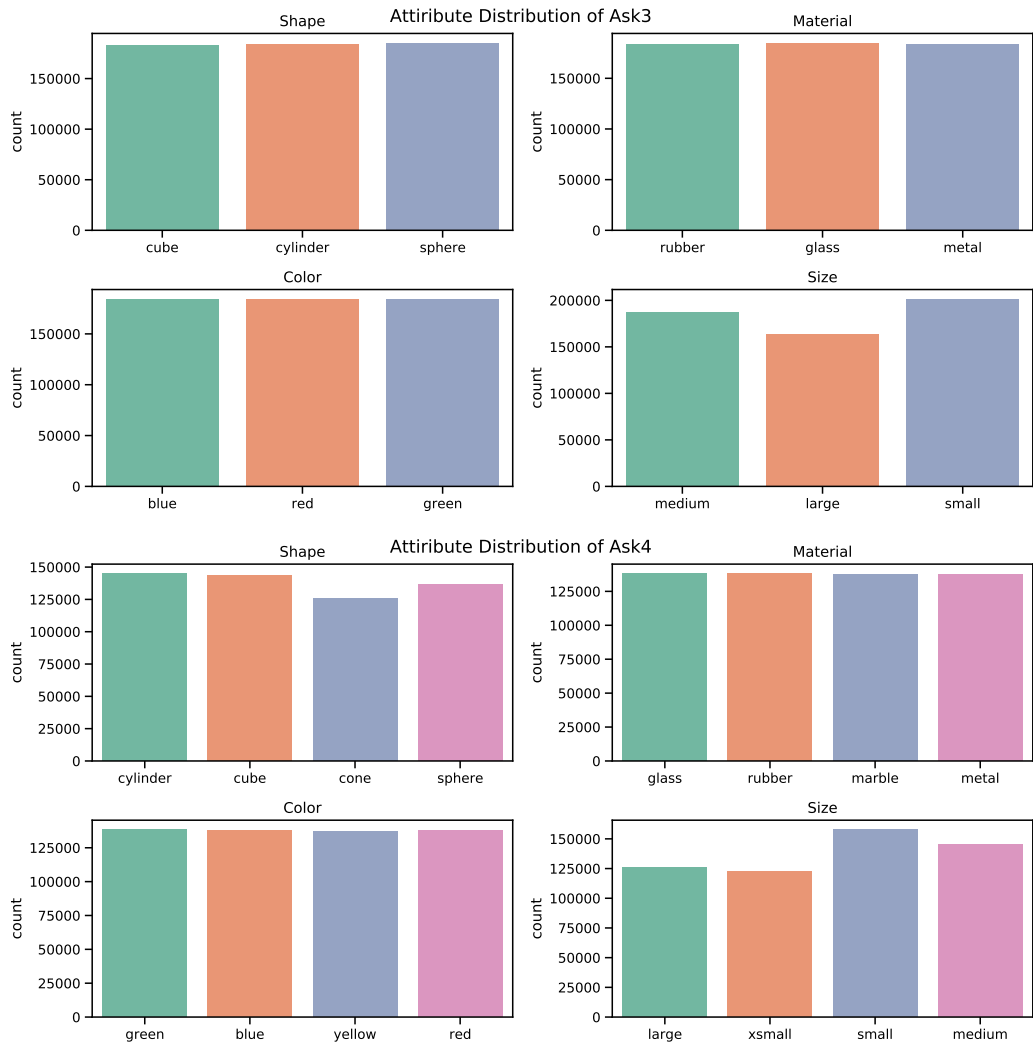
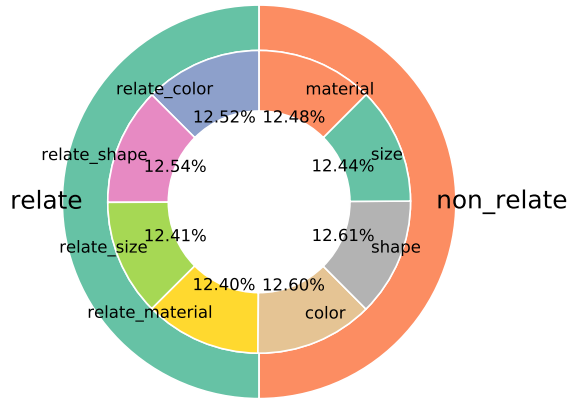


Figure 8: Distribution of the attributes of objects in Ask3 and Ask4 dataset.

Distribution of question types of Ask3



Distribution of question types of Ask4

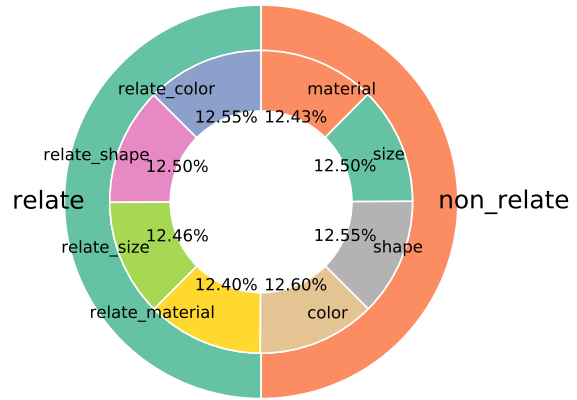


Figure 9: Distribution of question attributes in Ask3 and Ask4 dataset.

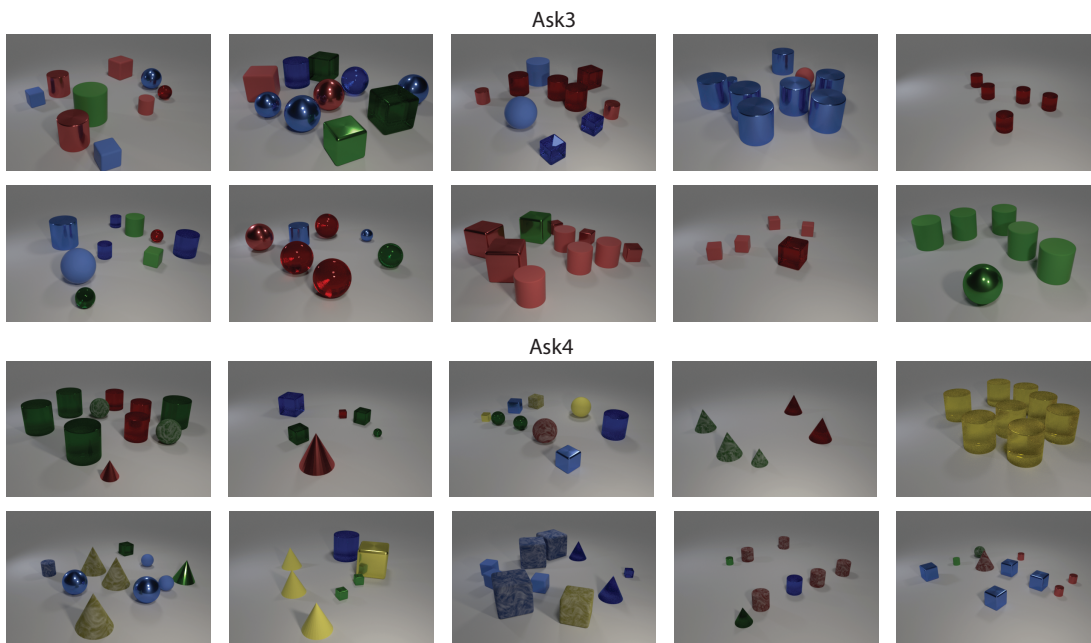


Figure 10: Sample scenes in Ask3 and Ask4 dataset.