3D Captioning for Multiple Objects with Relation

Anonymous ICCV submission

Paper ID 44

Abstract

Currently, the description of single 3D object is studied on the base of large language models. However, descriptions of multiple 3D objects are not well studied. Therefore, we introduce an automatic approach for generating descriptive captions with relation for multiple 3D objects. The framework contains three modules: a 3D object detection module, a vision-language bridge module and a multiple objects description and relation module. First, the 3D object detection module works to detect each 3D object. Second, the vision-language module works to transfer the features in the vision domain to the language domain. Third, the multiple objects description and relation module works to describe 3D objects and illustrate the connections among them. In the evaluation, we show the initial results of the proposed framework.

1. Introduction

The creation process of 3D assets will be revolutionized by the text-conditioned 3D synthesis [4], which will impact various sectors. However, the state-of-the-art model [8] only describes a single 3D object, which has certain limitations.

Therefore, we propose a new method to describe multiple 3D objects and their relation. The proposed method contains three main modules as shown in Figure 1.

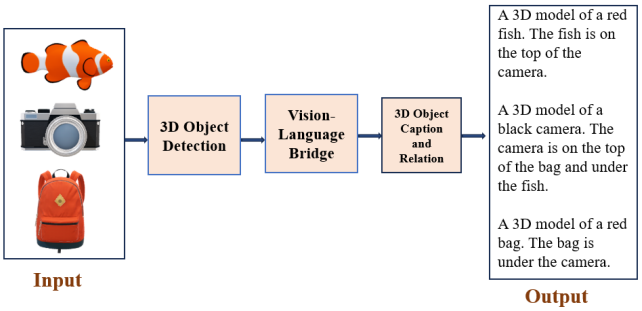

Figure 1. Overview of Multiple 3D Objects Description and Relation Framework.

2. Related Work

Until recently, the scale of 3D data was relatively small. Then the scale is enlarged [2]. Afterward, Cap3D[8] is developed to assign a single caption to each object. However, detailed caption with relation for multiple 3D objects is not studied. Therefore, We proposed a new framework to solve at some extend.

3. Method

The framework contains three modules: a 3D object detection module, a vision-language bridge module and a multiple objects description and relation module. The overview of the proposed method is illustrated in Figure 2. Firstly, the 3D object detection module works to detect each 3D object. Secondly, the vision-language bridge module works to transfer the features in the vision domain to the language domain. Thirdly, the multiple objects description and relation module works to describe 3D objects and illustrate the relation among them. The overview of the caption comparison is illustrated in Figure 4.

Firstly, the calculation of the first module is described as the following:

$$r_1 = f_{Detection}(i) \tag{1}$$

where $f_{Detection}$ denotes the detection module for 3D object detection, i denotes the input which contains multiple 3D objects. r_1 denotes the separate regions of multiple 3D objects.

After the region of each 3D object is obtained, the second module works to transfer features from the vision domain to the language domain. The computation is described as the following:

$$r_2 = f_{Bridge}(r_1) \tag{2}$$

where f_{Bridge} denotes the vision-language module for the domain transfer from the vision domain to the language domain, r_2 denotes the features of multiple 3D objects in the language domain.

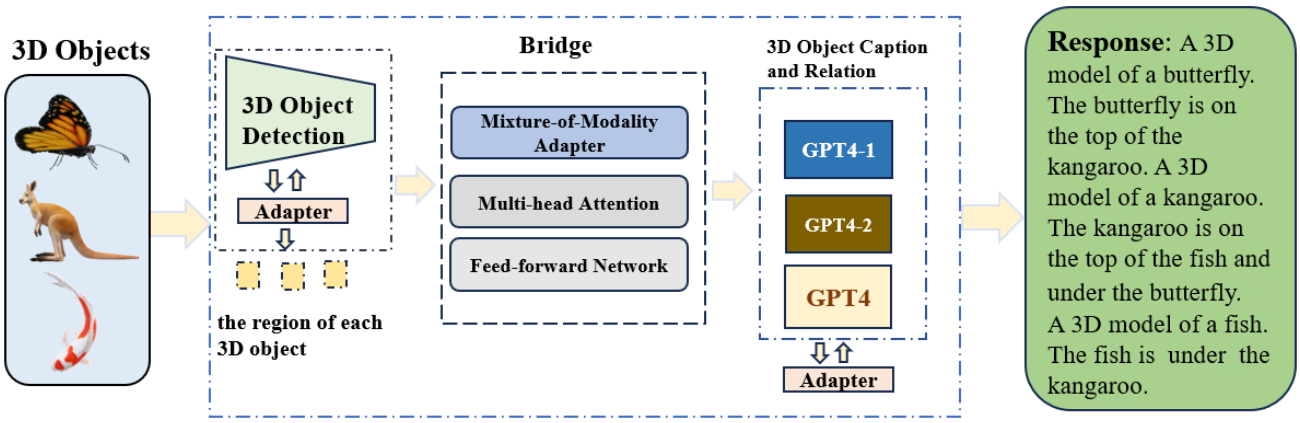

Figure 2. We provides detailed descriptions and relation of 3D objects by leveraging pretrained 3D Object Detection models ,Vision-language Bridge and GPT4. Input three objects are shown here.

Then, the third module works to describe 3D objects and explain the relation among them. The computation is described as the following:

$$r_3 = f_{Descrip-Relation}(i, r_1, r_2) \tag{3}$$

where $f_{Descrip-Relation}$ denotes the multiple objects description and relation module, r_3 denotes the captions of multiple 3D objects and the relation among them.

3.1. 3D Object Detection

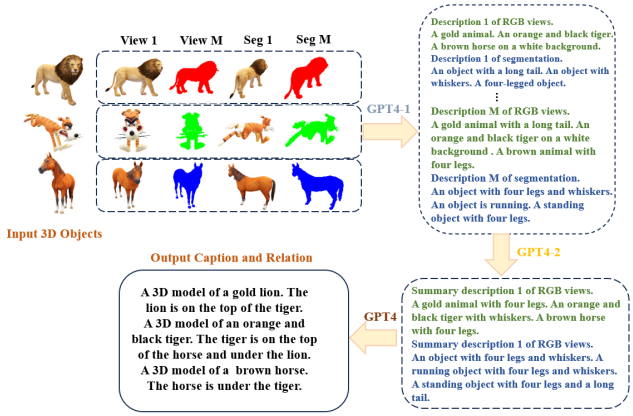

Figure 3. Overview of 3D Object Caption and Relation

In this module, multiple 3D objects i is seen as the input, the separate regions r_1 of multiple 3D objects are output [9]. In details, we apply a constraint on the update of a pre-trained weight matrix $W_{Detection} \in \mathbb{R}^{d\times k}$ by utilizing a low-rank decomposition. The update is represented as $W_{Detection} + \Delta W = W_{Detection} + BA$. The computation is shown as the following:

$$r_1 = W_{Detection}\ p + \Delta W\ p = W_{Detection}\ p + BAp \tag{4}$$

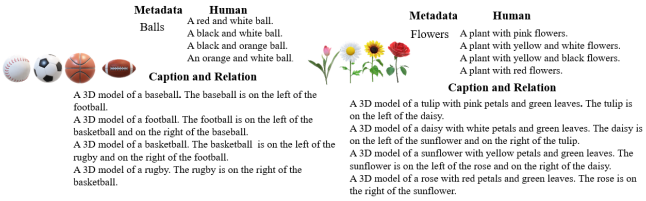

Figure 4. Caption Comparison

where $B \in \mathbb{R}^{d\times r}$, $A \in \mathbb{R}^{r\times k}$, and the rank $r \ll \min(d, k)$. $W_{Detection}$ remains frozen during training and does not receive gradient updates, while A and B include trainable parameters. It is important to note that both $W_{Detection}$ and $\Delta W = BA$ are multiplied by the same input, and their respective output vectors are summed coordinate-wise. p is denoted as an attention score estimated based on the token features [9].

3.2. Vision Language Bridge

In this module, we transfer the feature map from the vision domain to the language domain.

Firstly, we introduce a modality token $t_m \in \mathbb{R}^c$ to indicate the input modality. The computation is shown as the following:

$$t_m = mE_m \tag{5}$$

where $E_m \in \mathbb{R}^{2\times c}$ is the modality embedding. $m \in \mathbb{R}^2$ is a one-hot vector to represent the input modality.

Therefore, MM-Adapter[7] is applied and the computation is described as the following:

$$Z' = Z + s \cdot router(f_{a_1}(Z), f_{a_2}(Z); f_w(t_m)) \tag{6}$$

where the terms f_{a1} and f_{a2} refer to RepAdapters [6]. The scale factor s is used and the function $router(\cdot)$ determines the routing path for multiple adapters [7]. To minimize parameter costs, the downsampling projection of the two adapters is shared.

Then, the key to achieving dynamic adaptations lies in the design of the routing function, denoted as $router(\cdot)$. The formulation of this function is as follows:

$$router(f_{a_1}(Z), f_{a_2}(Z)) = \hat{w}_0 \cdot f_{a_1}(Z) + \hat{w}_1 \cdot f_{a_2}(Z) \quad (7)$$

$$where \quad \hat{w} = f_w(t_m) = softmax(\frac{t_m W_m + b_m}{\tau}) \quad (8)$$

where $W_m \in \mathbb{R}^{c\times 2}$ denotes the weight matrix and the $b_m \in \mathbb{R}^2$ denotes the bias. The routing weights are represented by $\hat{w}$. τ corresponds to the temperature parameter used in the $softmax$ function.

Then, the output is computed through the end-to-end optimization:

$$r_2 = \arg min \mathcal{L}(f_\phi(Z), R; \theta_a) \quad (9)$$

where R represents the ground-truth response [5]. $\mathcal{L}(\cdot)$ is the objective loss function. The LLM[7] is denoted by f_ϕ. The adaptation parameters are represented by θ_a. $I \in \mathbb{R}^{h\times w\times 3}$ denotes the input image and $T \in \mathbb{R}^l$ represents the text instruction.r_2 is the feature embedding in the language domain. To be noted, we use e to denote r_2 in the next section.

3.3. Multiple 3D Objects Caption and Relation

In this module, we apply the region of each 3D object from the previous detection module and the corresponding embedding from the previous domain adaption module to produce a detailed description with relations. Firstly, we finetune GPT4 to produce the descriptions of the 3D RGB views. Similarly, the descriptions of the 3D segmentation views are produced. Secondly, the GPT4 is applied to produce the summary of the descriptions of the RGB views and segmentation views respectively. Finally, the two descriptions are combined to produce a single description with the relation of 3D objects. The overview of this module is illustrated in Figure 3.

Firstly, the GPT4 is fine-tuned to produce the descriptions of the RGB views of each 3D object. The computation is represented as the following:

$$p^j_{RGB_i} = f^1_{GPT4}(I^j_i, e^j_i), i \in \{1, ..., N_i\}, j \in \{1, ..., N_j\} \quad (10)$$

where f^1_{GPT4} is the finetuned GPT4 module. I^j_i is the j-th RGB view of the i-th object, e^j_i is the corresponding embedding from the previous domain adaption module. $p^j_{RGB_i}$ is the description of the i-th object. N_i is the total number of objects, N_j is the total number of views.

Similarly, the descriptions of the 3D segmentation views are produced. The computation is represented as the following:

$$p^j_{seg_i} = f^1_{GPT4}(I_{seg}{}^j_i, e_{seg}{}^j_i), i \in \{1, ..., N_i\}, j \in \{1, ..., N_j\} \quad (11)$$

where f^1_{GPT4} is the finetuned GPT4 module. $I_{seg}{}^j_i$ is the j-th segmentation view of the i-th object, $e_{seg}{}^j_i$ is the corresponding embedding from the previous domain adaption module. $p^j_{seg_i}$ is the description of the i-th object. N_i is the total number of objects, N_j is the total number of views.

Secondly, the GPT4 is applied to produce the summary description of the RGB views and segmentation views respectively. The computation is represented as the following:

$$p^i_{summary} = f_{GPT4}(I_{RGB}, e_{RGB}), i \in \{1, ..., N\} \quad (12)$$

where f_{GPT4} is the GPT4 module for caption consolidation. I_{RGB} is the RGB view of the object, e_{RGB} is the corresponding embedding from the previous description module. $p^i_{summary}$ is the consolidated description of the i-th object. N is the total number of objects.

Similarly, the summary description of the 3D segmentation views is produced. The computation is represented as the following:

$$p^i_{summary} = f_{GPT4}(I_{seg}, e_{seg}), i \in \{1, ..., N\} \quad (13)$$

where f_{GPT4} is the module for caption consolidation. I_{seg} is the segmentation view of the object, e_{seg} is the corresponding embedding from the previous description module. $p^i_{summary}$ is the consolidated description of the i-th object. N is the total number of objects.

Finally, the two descriptions are combined to produce a single description with relation of 3D objects. The computation is represented as the following:

$$p^i_{Caption\ \ and\ \ Relation} = f_{GPT4}(D_{RGB}, D_{seg}), i \in \{1, ..., N\} \quad (14)$$

where GPT4 denotes the final step, which combined two captions into a final and output the relation among the multiple 3D objects. D_{RGB} is the consolidated description of RGB views of the object, D_{seg} is the consolidated description of segmentation views of the object.

3.4. Loss Function

In this paper, we applied a combined loss to train the network. The combined loss is shown as the following:

$$\mathcal{L}_{total} = \mathcal{L}_{FL} + \mathcal{L}_{mini} + \mathcal{L}_{GPT4}^{tune} \tag{15}$$

Focal Loss. We represent the loss of the point detection task as follows:

$$\mathcal{L}_{\mathrm{FL}} = \mathrm{FL}\left(\mathbf{g}^{m}\left(\mathbf{u_t}, \mathbf{v_t}\right), \mathbf{S}\right), \tag{16}$$

where FL is the Focal Loss, used to handle the issue of sample imbalance for key point labels. (u_t, v_t) represents the ground truth key point coordinate. g^m is the mapping function g^m. The score matrix S with a size of $(W_s \times H_s)$, is obtained by adding S_d and αS_s together. Here, S_d corresponds to the score matrix for the image feature map, while S_s corresponds to the score matrix for the segmentation task. The hyperparameter α controls the weighting of the segmentation task relative to the detection task. The matrix addition method adjusts the dimensions and adds content as required.

Mini training Loss. Then, we apply a mini training batch by randomly sampling text-only and text-image instructions. The overall training objective $\mathcal{L}$ can be defined as the following:

$$\mathcal{L}_{mini} = \sum_{i=1}^{m} \sum_{s=1}^{S+1} \log p\left(R_s^i \mid Z^i, R_{0:s-1}^i; \theta_a\right) \tag{17}$$

where m represents the batch size. S denotes the length of the response. The Loss $\mathcal{L}_{GPT4}^{tune}$ is the same loss to train GPT4 [1].

4. Experiments

4.1. Dataset

Objaverse Objaverse[3] contains around 800k 3D object assets from over 100k artists, covering 21k classes. All objects have corresponding captions.

4.2. Training Plan

We compare the caption generated by our method with Human and Metadata. We ablate the main components of 3D Object Caption and Relation into GPT4-1 , GPT4-2 and GPT4. GPT4-1 is used to produce the description of the 3D RGB views and 3D segmentation views. GPT4-2 is applied to produce the summary of the description of the RGB views and segmentation views respectively. Finally, the two descriptions are combined by GPT4 to produce a single description with relation of 3D objects. Currently, we train the third module and the result is shown in Figure 4, then we are going to train all the modules in an end-to-end training strategy.

4.3. Results

Now we have conducted the first experiment where the proposed framework was trained in the initial stage. The qualitative results are presented in Figure 4. We will proceed with the remaining experiments in the near future.

References

[1] Sébastien Bubeck, Varun Chandrasekaran, Ronen Eldan, Johannes Gehrke, Eric Horvitz, Ece Kamar, Peter Lee, Yin Tat Lee, Yuanzhi Li, Scott Lundberg, et al. Sparks of artificial general intelligence: Early experiments with gpt-4. *arXiv preprint arXiv:2303.12712*, 2023.

[2] Angel X Chang, Thomas Funkhouser, Leonidas Guibas, Pat Hanrahan, Qixing Huang, Zimo Li, Silvio Savarese, Manolis Savva, Shuran Song, Hao Su, et al. Shapenet: An information-rich 3d model repository. *arXiv preprint arXiv:1512.03012*, 2015.

[3] Matt Deitke, Dustin Schwenk, Jordi Salvador, Luca Weihs, Oscar Michel, Eli VanderBilt, Ludwig Schmidt, Kiana Ehsani, Aniruddha Kembhavi, and Ali Farhadi. Objaverse: A universe of annotated 3d objects. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 13142–13153, 2023.

[4] Heewoo Jun and Alex Nichol. Shap-e: Generating conditional 3d implicit functions. *arXiv preprint arXiv:2305.02463*, 2023.

[5] Pan Lu, Swaroop Mishra, Tanglin Xia, Liang Qiu, Kai-Wei Chang, Song-Chun Zhu, Oyvind Tafjord, Peter Clark, and Ashwin Kalyan. Learn to explain: Multimodal reasoning via thought chains for science question answering. *Advances in Neural Information Processing Systems*, 35:2507–2521, 2022.

[6] Gen Luo, Minglang Huang, Yiyi Zhou, Xiaoshuai Sun, Guannan Jiang, Zhiyu Wang, and Rongrong Ji. Towards efficient visual adaption via structural re-parameterization. *arXiv preprint arXiv:2302.08106*, 2023.

[7] Gen Luo, Yiyi Zhou, Tianhe Ren, Shengxin Chen, Xiaoshuai Sun, and Rongrong Ji. Cheap and quick: Efficient vision-language instruction tuning for large language models. *arXiv preprint arXiv:2305.15023*, 2023.

[8] Tiange Luo, Chris Rockwell, Honglak Lee, and Justin Johnson. Scalable 3d captioning with pretrained models. *arXiv preprint arXiv:2306.07279*, 2023.

[9] Yunsong Zhou, Hongzi Zhu, Quan Liu, Shan Chang, and Minyi Guo. Monoatt: Online monocular 3d object detection with adaptive token transformer. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 17493–17503, 2023.