SceneVerse: Scaling 3D Vision-Language Learning for Grounded Scene Understanding *Supplementary Material*

Baoxiong Jia*, Yixin Chen*, Huangyue Yu, Yan Wang, Xuesong Niu, Tengyu Liu, Qing Li, and Siyuan Huang

* indicates equal contribution
State Key Laboratory of General Artificial Intelligence, BIGAI
https://scene-verse.github.io

A The SceneVerse Dataset

A.1 3D Scenes

To address the scarcity of available 3D scene data, we construct SceneVerse by unifying 3D scene data from various existing datasets. The curation involves utilizing real-world scene datasets such as ScanNet [6], ARKitScenes [2], HM3D [16], 3RScan [18] and MultiScan [12], in conjunction with synthetic environments from Structured3D [22] and ProcTHOR [7]. The incorporation of these synthetic datasets is primarily driven by their potential as scalable data sources for 3D vision-language (3D-VL) alignment. To facilitate the training process, we conduct the following preprocessing steps.

Room Segmentation The 3D scenes in HM3D and ProcTHOR are released at the building level, encompassing multiple rooms and sometimes spanning over 50 meters. To align with existing benchmarks [1, 3], we leverage the associated metadata to segment the 3D point cloud at the room level, facilitating subsequent operations in scene graph construction and language description generation. Additionally, we implement a filtering process to exclude extremely large rooms and those with fewer than 4 objects in the scene.

Point Cloud Normalization To mitigate the data disparities arising from diverse capture devices across various data sources, we subsample each point cloud to a maximum of $240,000$ points. Each point cloud then undergoes a transformation centered on the central point on the floor, followed by rotation to align the room layout with the axis following the approach by Chen *et al.* [4].

Semantic Label Alignment Given the divergence in semantic label sets across different datasets, we undertake a comprehensive effort to map all the object class labels to the 607 semantic labels in ScanNet [6] to facilitate close-vocabulary object classification [14] in the existing model framework [23]. We construct the mapping in each dataset through LLM and manual verification. Note that the object-level grounding in GPS can directly deal with open-set object labels or captions, similar to CLIP [10].

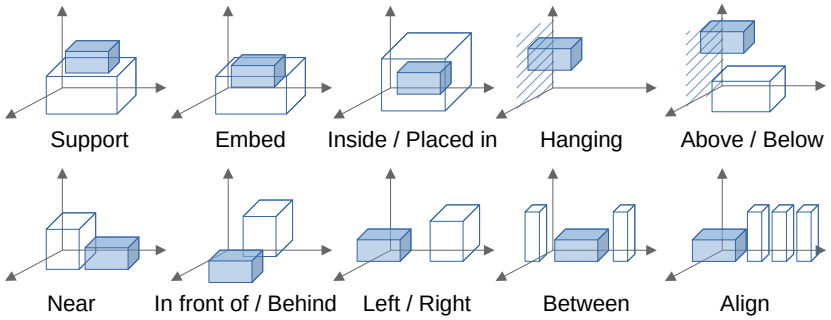

Fig. A.1: Overview of the relationships in SceneVerse. The target object is colored in blue.

After the preprocessing, each scan is represented by a point cloud $\mathrm{P} \in \mathbb{R}^{N \times 8}$, wherein each point is defined by its 3D coordinates, RGB color, instance id and semantic label. In total, we curate 68,406 3D scenes in SceneVerse.

A.2 3D Scene Graph

Here, we provide the definition of 3D scene graphs and relationships, as well as the implementation details of our proposed pipeline to construct 3D scene graphs from the point cloud.

3D Scene Graph Definition In our constructed 3D scene graph $\mathcal{G} = (\mathcal{V}, \mathcal{E})$, the nodes $\mathcal{V}$ comprises the union of node sets $\mathcal{V}_1 \bigcup \mathcal{V}_2 \bigcup \ldots \bigcup \mathcal{V}_K$, with $\mathcal{V}_k$ representing the set of nodes at a particular hierarchical level. The hierarchies are determined by the `support` relationship; for instance, objects supported by the floor constitute $\mathcal{V}_0$, while objects supported by the table will form $\mathcal{V}_1$, *etc.* Note that edges originating from one node $v \in \mathcal{V}_k$ may only terminate in nearby hierarchies $\mathcal{V}_k \cup \mathcal{V}_{k+1} \cup \mathcal{V}_{k+1}$. In other words, edges in the scene graph exclusively connect nodes within the same hierarchical level, or one level higher or lower.

Relationships Following prior work [1, 19], we consider the following spatial relationships between the object nodes:

- **Vertical proximity** This encompasses both in-contact relationships (*e.g.*, `support`, `inside`, `embed`), and non-contact ones (*e.g.*, `above`, `below`).
- **Horizontal proximity** Horizontal relationships describe the proximity relations like `in front of, next to, behind`, *etc.* Relationships like `left, right` are contextually dependent on a reference view, where another anchor object is used to establish the view direction. The distance between the two objects is calculated to describe if the objects are `far` or `near` in space.
- **Multi-object Relationships** This models the spatial arrangement of multiple objects, *e.g.*, `align` and `between`.

Edges originating from one node $v \in \mathcal{V}_k$ may only terminate in nearby hierarchies, *i.e.*, adjacent nodes in the scene graph. The node hierarchy is decided

Table A.1: Relationships in SceneVerse. The 3D scene graph captures 21 types of relationships ranging in 4 categories.

Category	Relation	
In-contact vertical	supported by placed in	embedded into inside
Non-contact vertical	hanging on mounted on higher than lower than	affixed on above below
Horizontal	near(far) to the left of is behind close to besides	near(far) to the right of is in front of adjacent to next to
Multi-object	between	aligned

by the `support` relationship. We traverse all the object nodes to calculate spatial relationships, which undergo an automatic verification procedure to rectify incorrect ones. Our 3D scene graph captures 21 types of relations as shown in Tab. A.1. We provide illustrations of how these relations are defined in the 3D space, as can be seen in Fig. A.1.

Scene Graph Construction Due to the inherent noise and incompleteness in the point cloud representation, automatically extracting precise and comprehensive relationships from the point clouds is a non-trivial task. Below we detail our 3D scene graph construction process, as outlined in Algorithm 1.

We first instantiate the graph nodes with the instance annotation from the point cloud and parameterize each node with object centroid $\mathrm{p_i} \in \mathbb{R}^3$ and size of the axis-aligned bounding box $\mathrm{b_i} = (b_x, b_y, b_z) \in \mathbb{R}^3$ (Line 1-3). Next, we traverse all the nodes to determine their spatial relationships (Line 4-22). Notably, in cases where an object node lacks any in-contact vertical relationships with other objects in the scene, we designate such objects as `hangable` and calculate their non-contact vertical relationships (Line 9-13). Examples of such objects include paintings, curtains, *etc*. Finally, we establish relationships between multiple objects (Line 23): i) When a target object is connected with two edges labeled `left` and `right`, the target object, along with the two neighboring nodes, forms a `between` relationship triplets. ii) If the offset of the center point coordinates of a group of objects in either the X-axis or Y-axis direction is smaller than a specified offset threshold δ, then this group of objects forms an `align` relationship. The offset threshold δ will be adjusted based on the size of the scene. In additional, we utilize an automatic verification procedure to validate the scene graph, further improving the quality of the scene graph we constructed (line 24). One of the verification operations involves manually maintaining a mapping between objects and relationship descriptions based on common sense. For example, people usually use `mounted on` to describe the relation between TV and wall, rather

than `hanging on`. Therefore, we would automatically refined (TV, hanging on, wall) to (TV, mounted on, wall).

Algorithm 1: Scene Graph Construction Pipeline

Input : M object point clouds $\{P_1, P_2, \ldots, P_m\}$
Output : 3D scene graph $\mathcal{G}(\mathcal{V}, \mathcal{E})$

for i from 1 to M **do**
  Create node $v_i \in \mathcal{V}$ using the centroid p_i and bounding box size b_i of object
  point cloud P_i
end for
for i from 1 to M **do**
  for j from $i+1$ to M **do**
    $\text{RelsType}_v \leftarrow \text{VerticalInContact}(v_i, v_j)$
    Add in-contact vertical relationship triplets $(v_i, v_j, e_{i,j})$ with RelsType_v to $\mathcal{G}$
  end for
  if No objects horizontally related to v_i **then**
    for k from 1 to M and $i \neq k$ **do**
      $\text{RelsType}_v \leftarrow \text{VerticalNonContact}(v_i, v_k)$
      Add non-contact vertical relationship triplets $(v_i, v_k, e_{i,k})$ with RelsType_v to $\mathcal{G}$
    end for
  end if
end for
for $v_i \in \mathcal{V}$ **do**
  let $\{v_{i_1}, v_{i_2}, ..., v_{i_N}\}$ be the N different nodes with the same in-contact vertical parent node v_i
  for j from 1 to N **do**
    $\text{RelsType}_h \leftarrow \text{Horizontal}(v_i, v_{i_j})$
    Add horizontal relationship triplets $(v_i, v_{i_j}, e_{i,i_j})$ with RelsType_h to $\mathcal{G}$
  end for
end for
Update $\mathcal{G} \leftarrow \text{MultiObjects}(\mathcal{G})$
Update $\mathcal{G}$ with automatic verification procedure

A.3 Object Captioning Pipeline

Object captions aim to provide detailed descriptions of an object's visual and physical properties, facilitating object-level grounding with its distinctive features. The detailed object captioning pipeline is outlined in Algorithm 2. Given the multi-view images $\{I_1, I_2, \ldots, I_n\}$, we utilize the point cloud P_o of the object o

to get the visible points $P_{o,v}^{vis}$ in the images v through rendering with the camera intrinsic and camera poses. The occlusion score $s_{o,v}^{occ}$ is calculated as the ratio between the number of visible points in the image and the full object point cloud. The image is then cropped with the rendered bounding box and processed through BLIP2 [11] to generate the initial object caption $C_{o,v}$. For each initial caption, we calculate its CLIP [15] similarity score between the text and the cropped image, denoted by $s_{o,v}^{clip}$. To get a refined object caption, we select the top 10 initial captions with the highest CLIP score and minimal occlusion. The selected sentences are fed into a LLM to obtain a coherent summary of the object captions. In this process, we explicitly instruct the language model to identify and correct the potential errors.

Algorithm 2: Object Captioning Pipeline

Input : M object point clouds $\{P_1, P_2, \ldots, P_m\}$; N multiview images $\{I_1, I_2, \ldots, I_n\}$

Output : Captions for each object in the scene $\{C_1, C_2, \ldots, C_m\}$

for $o = 1, 2, \ldots, M$ **do**
  for $v = 1, 2, \ldots, N$ **do**
    Project P_o on I_v to get visible points $P_{o,v}^{vis}$
    Crop I_v with the bounding box of $P_{o,v}^{vis}$ to get $I_{o,v}^{crop}$
    Get the image caption $C_{o,v}$ for $I_{o,v}^{crop}$ using BLIP2 [11]
    Calculate the similarity score $s_{o,v}^{clip}$ between $C_{o,v}$ and $I_{o,v}^{crop}$ with CLIP [15]
    Calculate the occlusion score $s_{o,v}^{occ} = \frac{\#P_{o,v}^{vis}}{\#P_o}$
  end for
  Select the top-10 $\{C_{o,v}\}$ with highest $s_{o,v}^{clip} * s_{o,v}^{occ}$
  Summary selected $\{C_{o,v}\}$ with GPT-3.5 to get C_o
end for

Fig. A.2: Examples of object captioning. We color the target object in **bold**.

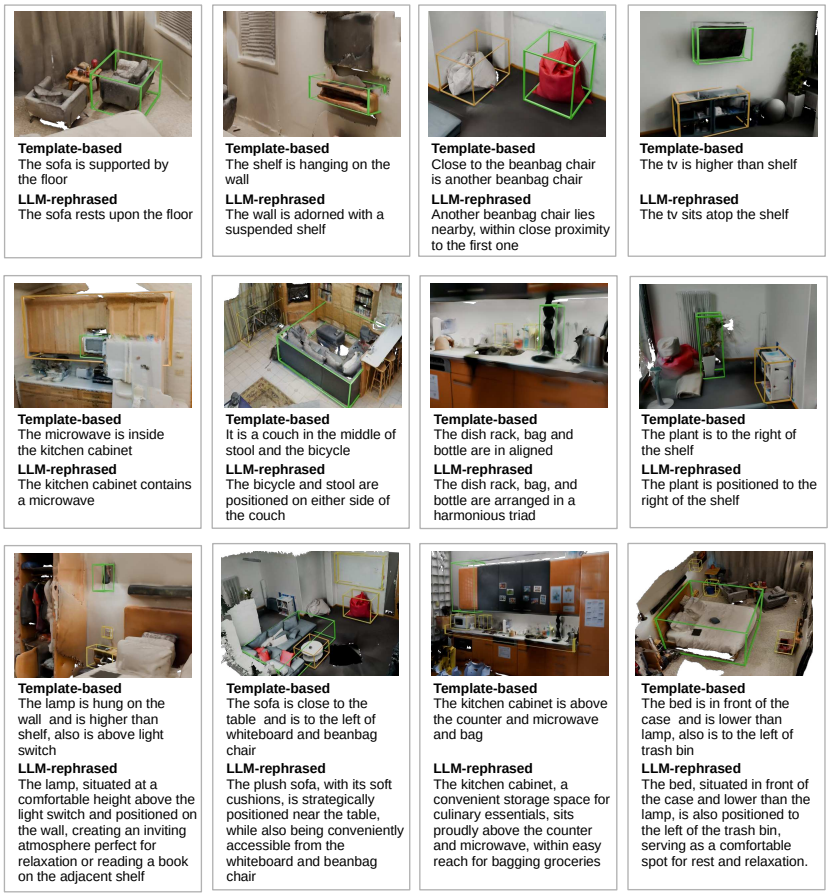

Fig. A.3: Examples of object referral. Note that the green bounding box indicates the target object and yellow bounding box indicates the anchor object(s).

A.4 Object Referral Generation from 3D Scene Graphs

1. **Template-based** We create diverse templates to generate descriptions for each type of relationship. We categorized the templates into three types based on the number of objects involved and their spatial relationships.
   - **Pair-wise**: The pair-wise templates are used to describe the positional relationship between the target object and the anchor object in the scene. We design various templates to enrich the templated-based descriptions, spanning active and passive tense, as well as inversion clauses. Typical examples are shown below:
     - The target-object (is) spatial-relation the anchor-object.
     - It is a target-object that (is) spatial-relation the anchor-object.
     - There is a target-object that (is) spatial-relation the anchor-object.
     - Spatial-relation the anchor-object is the target-object.
     - Spatial-relation the anchor-object, a target-object is placed.

  - **Multi-objects**: This is utilized when the target object forms a `between` or `align` relationship with multiple anchor objects in the scene. The templates follow the same construction rules as the **Pair-wise** templates.
  - **Star-reference**: To increase complexity in templated-based descriptions, we design "star-reference" to describe the target object and its relationship with 3 randomly selected anchor objects in the scene graph. In particular, we perform cluster analysis on the selected relationship triplets. Based on the diversity of the analysis, different templates will be chosen to generate descriptions. For example, when the relations between 3 anchor objects and the target object is the same, we prefer to use the template like: "The target-object (is) spatial-relation the anchor-object-1, anchor-object-2 and anchor-object-3". If 2 out of the 3 anchor objects have the same relations with the target object, we would use a template like: "The target-object (is) spatial-relation-1 the anchor-object-1 and anchor-object-2, and (is) spatial-relation-2 the anchor-object-3."
2. **Large Language Model (LLM)-rephrasing** To increase description diversity we use the GPT-3.5 [13] and Llama [17] for description rephrasing. This improves the diversity and naturalness of the template-based descriptions, as is shown by the statistics presented in the main paper. The detailed prompts are provided in Tab. A.2.

A.5 Are 3D Scene Graphs Necessary?

We compare our pipeline that utilizes the 3D scene graphs with directly employing spatial locations and classes (*POS*-based) in LLM prompting on GPT-3.5. Results on 10 ScanNet [6] scenes show our pipeline outperforms the *POS*-based in terms of both diversity (7.29 / 5.08 on Shannon Index) and correctness (90% / 32% through human validation). *POS*-based descriptions lack a sense of direction (*e.g.*, left/right) and hierarchy (*e.g.*, support), while scene graph provides hierarchical and precise depictions of object relations. The quality checks show the capability of our proposed scene-graph-based generation approach to produce high-quality language descriptions, laying a robust foundation for future scalability.

More examples of the scene-language pairs in SceneVerse are shown in Fig. A.2, Fig. A.4 and Fig. A.3.

B Model Details

B.1 Spatial-Attention Transformer

In our proposed model, we leverage a transformer architecture based on spatial attention to aggregate object-level point cloud features with spatial location information. In this section, we provide the detailed design of this proposed module.

Formally, given object features $\{\boldsymbol{f}_i^O\}_{i=1}^N$ and their locations $\{\boldsymbol{l}_i\}_{i=1}^N$, we first construct pair-wise spatial relationship feature via:

$$\boldsymbol{m}_{ij} = [d_{ij}, \sin(\theta_h), \cos(\theta_h), \sin(\theta_v), \cos(\theta_v)]\,,$$

Table A.2: Prompts used in SceneVerse.

Description type	Prompt
Object caption	Summarize **caption** below. The summary should be a description of the **target-object**. Focus on the **target-object**'s attribute, like color, shape and material, *etc.* Identify and correct the potential errors. **caption**: *A bed in a hotel room. A white comforter on a bed. A bed with a striped comforter...* **target-object**: *Bed*
Object referral	Rewrite the following **caption** using one random sentence structure. You should give me only one rewritten sentence without explanation. **caption**: *The bed is between desk and nightstand.*
	Rewrite the following **caption**. You should give me only one rewritten sentence about **target-object** without explanation. Make sure **target-object** is the subject of the sentence, not **anchor-object(s)**. If the sentence is in full inversion, keep the inversion. **caption**: *The armchair is next to the sofa.* **target-object**: *Armchair* **anchor-object(s)**: *Sofa*
	Rewrite the following **caption** using one random sentence structure. You need to focus on the location and relations of the **target-object** that appears in the sentence. If multiple **target-object** appear in the sentence, you need to focus on the first **target-object** that appears. You can also add the **target-object**'s function and comfort level based on the sentence, e.g., how the objects can be used by humans and human activities in the scene. You should give me only one rewritten sentence without explanation. **caption**: *Far from the bowl and peppershaker, the vase is to the left, it is also on the top of countertop.* **target-object**: *Vase*
Scene captioning	Your task is to provide a summary for a scene from a given **scene graph**. The scene contains some objects, which compose a scene graph in json format. There are 3 types of descriptions in scene graph: "scene type" denotes the type of the scene. "objects count" then listed the objects in the scene and their quantity, it should be noted that the actual objects in the room may be more than listed. "objects relations" describe the spatial relations with objects. Also describe the scene concerning commonsense, e.g., how the objects can be used by human and human activity in the scene. The description should conform to the given scene graph. The spatial relations between objects can only be inferred from the "objects relations" in scene graph. Don't describe each object in the scene, pick some objects of the scene for summary. Don't describe each relations in the scene, pick some relations of the scene for summary. You can also summarize the room's function, style, and comfort level based on the arrangement and count of objects within the room. The summary should be about the object types, object attributes, relative positions between objects. Your summary must not exceed 80 words. You must write using one random sentence structure. **scene graph**: `{ 'scene_type': 'Bedroom', 'object_count': {'nightstand':2, ...}, 'relation': {'nightstand', 'on', 'floor'}, {'backback', 'in front of', 'bed'}, ...}`

where d_{ij} denotes the Euclidean distance between objects and θ_h, θ_v are the horizontal and vertical angles of the line connecting the centers of objects i, j. We then use this pair-wise relationship feature $M = [\boldsymbol{m}_{ij}] \in \mathbb{R}^{N\times N\times 5}$ to modulate the attention weights of the self-attention layer in the transformer when aggregating object features as shown below:

$$\text{Attn}(Q, K, V, M) = \text{softmax}\left(\frac{QK^T}{\sqrt{d_h}} + \log\sigma(M\omega)\right)V,$$

where $\omega \in \mathbb{R}^5$ is a projection layer mapping spatial pair-wise features to the attention scores and σ denotes the sigmoid function. This process could be equivalently interpreted as using the spatial location of objects to adjust the self-

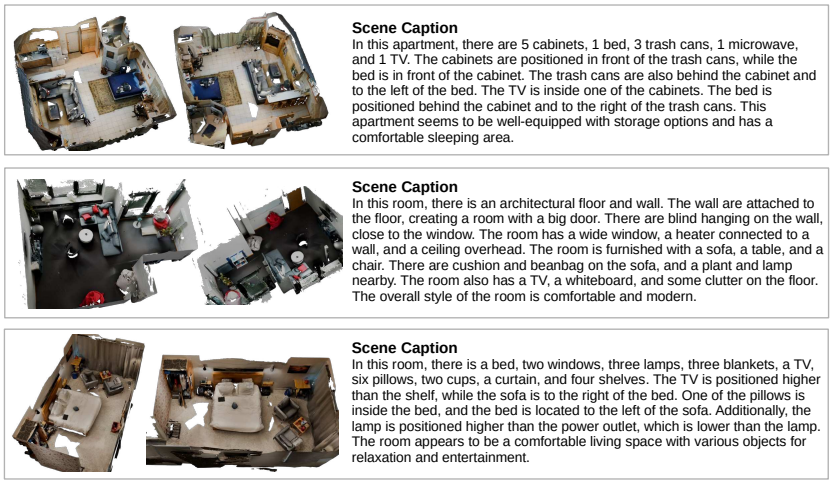

Fig. A.4: Examples of scene captioning.

attention feature aggregation between objects, making spatially related objects have more attention weights.

B.2 Pre-training Details

For training our model GPS, we conduct a two-stage training approach. We first pre-train the object point cloud encoder with the object-level grounding objective. Next, we freeze the object point cloud encoder during the second pre-training stage for scene-level pre-training that includes model training with scene-level grounding and referral object grounding objectives. This design is inspired by recent works like [5,23] that demonstrated a well-grounded initialization of object representations is beneficial for 3D scene grounding.

Object-level pre-training To correctly align objects in scenes with their captions, we utilize the ground-truth object bounding boxes provided with the datasets to segment all objects in the scenes. Next, we utilize a PointNet++ [14] encoder to encode and align these object point clouds with object captions provided in SceneVerse. For object instances with no object captions synthesized, we follow [15] and construct captions with their semantic class labels like "the point cloud of <`CLASS`>". Notably, as our model design sets no constraints on object point cloud encoders, the choice of object encoder mainly depends on the computing resources available.

Scene-level pre-training With pre-trained object feature extractors, we further use both scene captions and object-referring expressions for scene-level pre-training. We use a 4-layer BERT encoder for encoding both scene captions and object referrals. We apply a 4-layer spatial transformer to encode object features with their locations. For scene-level grounding, we adopt a max-pooling layer to

aggregate features from the spatial transformer and align with the [CLS] feature of the scene caption. For referral-object-level grounding, we further pass the obtained object features as well as the referral language features into a 4-layer self-attention transformer and use the grounding objective to match the referred object's feature and the [CLS] feature of the referring expression.

Training For object-level pre-training, we utilize an AdamW optimizer with a learning rate of 1×10^{-2} for 1500 epochs and no warm-up periods. During training, we use a batch size of 512 and leverage a cosine annealing scheme for learning rate scheduling with a minimum learning rate of 1×10^{-3}. For scene-level pre-training, we use an AdamW optimizer with a learning rate of 1×10^{-5} for the language encoder, a learning rate of 1×10^{-4} for the spatial transformer, a learning rate of 1×10^{-4} for the self-attention transformer, and a learning rate of 5×10^{-4} for all remaining learnable parameters (*e.g.*, projections). For all experiments, we train the model for 150 epochs with a warm-up period of 500 and also a cosine annealing scheme for learning rate with a minimum learning rate ratio of 0.1. All pre-training experiments are run on 8 NVIDIA-A100 GPUs with the longest pre-training on SceneVerse taking about 2 days.

C Experimental Details

In this section, we provide details on experimental settings, model implementation, and additional results.

C.1 3D Visual Grounding

Setting Following previous works [23], we report model performance on the validation set of all datasets. Notably, we used an off-the-shelf Mask3D segmentation model for generating object proposals with no optimization.

Implementation As briefly mentioned in the experiment section, we mainly considered three model settings in 3D visual grounding experiments, namely *scratch*, *pre-train*, and *fine-tuned*. For the *pre-train* setting, we follow the same setting mentioned in Appendix B.2. In the *scratch* and *fine-tuned* settings, to fairly compare with other dataset-specific fine-tuned models, we add an additional 2-layer MLP over the object features from the referral grounding self-attention transformer. During training, we fine-tune this grounding head together with all model weights for 100 epochs with a learning rate of 1×10^{-4} for the added projection layer and set all other settings the same as the implementation described in Appendix B.2.

C.2 Zero-shot Transfer

Setting In the zero-shot experiments, we first construct the held-out test set by aggregating scene-text pairs in SceneVerse from scenes in ScanNet and MultiScan. Specifically, we use the validation set of ScanRefer, Nr3D, and Sr3D.

For scene-text pairs in the SceneVerse-val, we construct the test set by randomly sampling $\frac{1}{5}$ of human-annotated object referrals in the MultiScan dataset. This results in a test set with around 1.7K object referrals randomly drawn from 8.5k human-annotated object referrals in the MultiScan dataset. In the *zero-shot* settings, we use all scene-text pairs from datasets in SceneVerse except for ScanNet and MultiScan. This includes both human-annotated and generated texts in ARKitScenes, 3RScan, and HM3D. This setting serves to test models' generalization capability in grounding objects with both unseen scenes and unseen texts. In the *zero-shot text* setting, we add generated scene-text pairs in ScanNet and MultiScan into the data used in the *zero-shot* setting, thereby making the held-out test contain mainly unseen object referrals.

Implementation In the zero-shot experiments, we mainly considered three model settings *scratch*, *zero-shot*, and *zero-shot text*. For the *zero-shot* setting, we pre-train the model following Appendix B.2 without additional grounding heads considering there is no additional training data available in the zero-shot transfer setting. In the *scratch* and *zero-shot text* setting, we follow the model implementation described in Appendix C.1 and add an additional 2-layer MLP over the object features from the self-attention transformer. We follow the same fine-tuning setting described in Appendix C.1.

C.3 3D question answering

Setting In the 3D question answering (3D-QA) experiments, we evaluate all models with only the training sets provided for fine-tuning. Following previous works [23], we report model performance on the validation and test sets of ScanQA and the test set of SQA3D.

Implementation In this experiment, we mainly considered the fine-tuned GPS when comparing it with existing methods. For all datasets, we initialize our GPS model from a checkpoint pre-trained with 3D visual grounding on SceneVerse. We follow 3D-VisTA and add an additional question-answering module over the pre-trained representations for the answer prediction. We fine-tune the model for 100 epochs with a learning rate of 1×10^{-4} for the added question answering head and set all other settings the same as the implementation described in Appendix B.2.

C.4 Open-vocabulary 3D semantic segmentation

Setting Following RegionPLC [20] proposed by Yang *et al.*, we conduct experiments to assess the performance of SceneVerse on open-vocabulary 3D semantic segmentation (OV-Seg). To establish a benchmark for open-vocabulary semantic segmentation, we adopt the experimental setup outlined in PLA [8], as per the methodology of RegionPLC. We utilize the annotation-free training setting, as described in RegionPLC, wherein semantic labels for all categories are omitted. This approach allows us to evaluate the effectiveness of SceneVerse in facilitating open-vocabulary segmentation without relying on predefined semantic

segmentation annotations. For evaluation, we compute the mean Intersection over Union (mIoU) and mean accuracy (mAcc) across 17 foreground categories, excluding "wall" and "floor" background classes, as well as the "other furniture" category due to its inherent ambiguity.

Implementation We employ SparseUNet [9] as our 3D backbone network for extracting point features. We utilize different variants of SparseUNet, varying the number of channels in the input layer to explore its impact on performance. For text feature extraction, we employ CLIP [10] text encoder. To align the extracted features from the 3D scene encoder and the text encoder, we incorporate a vision-language adapter. The only supervision comes from the point-discriminative contrastive loss proposed by RegionPLC. During training, we employ the AdamW optimizer to update model parameters. We train the model from scratch for 500 epochs, utilizing a learning rate of 1×10^{-3}. Additionally, we incorporate a warm-up period of 200 steps and a cosine annealing scheme for learning rate scheduling, with a minimum learning rate ratio of 1×10^{-5}.

D Additional Results

D.1 Semantic Segmentation

Setting To test if the scaling effect of SceneVerse is universally beneficial for 3D understanding tasks, we use 3D semantic segmentation as a signature task to illustrate the effectiveness of SceneVerse. Notably, a recent work that introduced the Swin3D model [21] has identified the importance of pre-training for 3D semantic segmentation [21]. Following the same setting, we test if the proposed Swin3D model could be further improved by substituting the pre-training data to SceneVerse. Specifically, we test models' performance on the ScanNet semantic segmentation task with 20 semantic categories and report the mean IoU and mean Acc on the validation set of ScanNet. As the original implementation of Swin3D pre-training requires surface normals as additional inputs, we reimplement the model and pre-train all models with only point coordinates and colors.

Table A.3: Semantic segmentation results on ScanNet validation set. † denotes model trained with surface normals as an additional input. S3D indicates models initialized with the original Swin3D model weights pre-trained on Structured3D provided by Yang *et al.* [21].

Methods	Init.	SceneVerse Pre.	mIoU	mAcc
Swin3D$_n$-S†	✗	✗	75.2	-
Swin3D$_n$-S†	S3D	✗	75.6	-
Swin3D-S	✗	✗	63.2	72.8
Swin3D-S	S3D	✗	64.1	75.1
Swin3D-S (*pre-train*)	✗	✓	67.7	78.0
Swin3D-S (*pre-train*)	S3D	✓	69.5	80.1
Swin3D-S (*fine-tuned*)	S3D	✓	**70.6**	**80.2**

Comparison As shown in Tab. A.3, we observe a significant model performance improvement (∼6%) by training SWIN3D-S model on our SCENEVERSE dataset. Comparing our pre-training set to Structured 3D, we also observe consistent model performance improvement, showcasing the benefit of scaling-effect in SCENEVERSE. Moreover, we fine-tune the model on ScanNet after pre-training on SCENEVERSE. This process further brings improvement in model performance on semantic segmentation. We believe these results serve as strong pieces of evidence validating the effectiveness of data scaling in SCENEVERSE and also its potential benefit for all 3D tasks in addition to 3D visual grounding.

D.2 Qualitative Results

We provide the qualitative results of 3D vision-language grounding in Fig. A.5 and the results of open-vocabulary semantic segmentation in Fig. A.6.

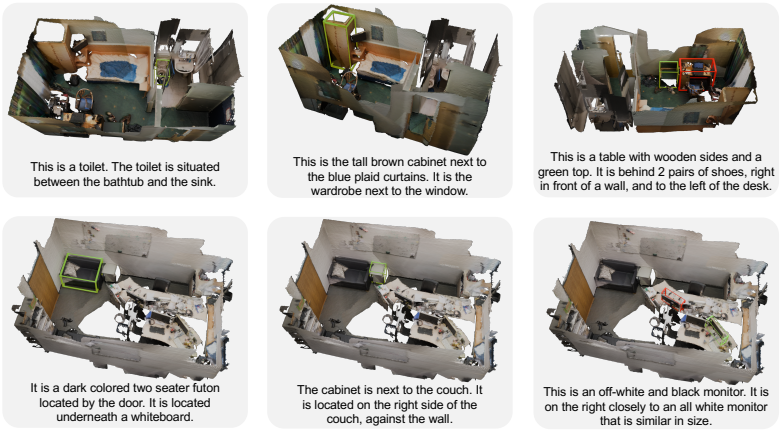

Fig. A.5: Qualitative results of GPS on 3D visual-language grounding. We visualize the incorrect predictions in red and the correct predictions or ground truths in green.

References

1. Achlioptas, P., Abdelreheem, A., Xia, F., Elhoseiny, M., Guibas, L.: Referit3d: Neural listeners for fine-grained 3d object identification in real-world scenes. In: Proceedings of European Conference on Computer Vision (ECCV) (2020)
2. Baruch, G., Chen, Z., Dehghan, A., Dimry, T., Feigin, Y., Fu, P., Gebauer, T., Joffe, B., Kurz, D., Schwartz, A., et al.: Arkitscenes: A diverse real-world dataset for 3d indoor scene understanding using mobile rgb-d data. In: Proceedings of Advances in Neural Information Processing Systems Datasets and Benchmarks (NeurIPS Datasets and Benchmarks Track) (2021)

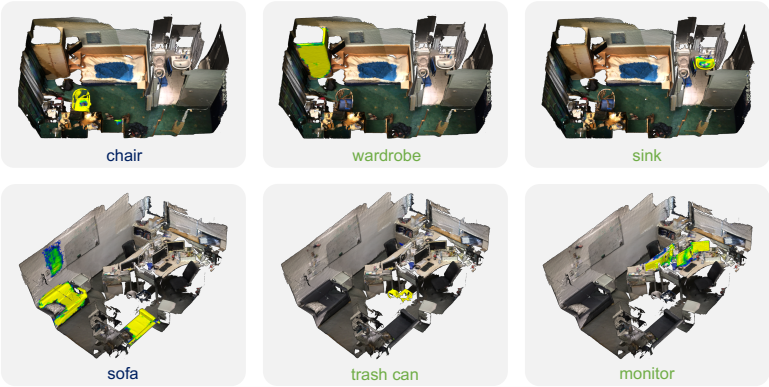

Fig. A.6: Qualitative results of open-vocabulary 3D semantic segmentation (OV-Seg). We label the object class in ScanNet-20's vocabulary in blue, and unseen class in ScanNet-20 in green.

3. Chen, D.Z., Chang, A.X., Nießner, M.: Scanrefer: 3d object localization in rgb-d scans using natural language. In: Proceedings of European Conference on Computer Vision (ECCV) (2020)
4. Chen, S., Guhur, P.L., Tapaswi, M., Schmid, C., Laptev, I.: Language conditioned spatial relation reasoning for 3d object grounding. In: Proceedings of Advances in Neural Information Processing Systems (NeurIPS) (2022)
5. Chen, Z., Hu, R., Chen, X., Nießner, M., Chang, A.X.: Unit3d: A unified transformer for 3d dense captioning and visual grounding. In: Proceedings of International Conference on Computer Vision (ICCV) (2023)
6. Dai, A., Chang, A.X., Savva, M., Halber, M., Funkhouser, T., Nießner, M.: Scannet: Richly-annotated 3d reconstructions of indoor scenes. In: Proceedings of Conference on Computer Vision and Pattern Recognition (CVPR) (2017)
7. Deitke, M., VanderBilt, E., Herrasti, A., Weihs, L., Ehsani, K., Salvador, J., Han, W., Kolve, E., Kembhavi, A., Mottaghi, R.: Procthor: Large-scale embodied ai using procedural generation. In: Proceedings of Advances in Neural Information Processing Systems (NeurIPS) (2022)
8. Ding, R., Yang, J., Xue, C., Zhang, W., Bai, S., Qi, X.: Pla: Language-driven open-vocabulary 3d scene understanding. In: Proceedings of Conference on Computer Vision and Pattern Recognition (CVPR) (2023)
9. Graham, B., Engelcke, M., Van Der Maaten, L.: 3d semantic segmentation with submanifold sparse convolutional networks. In: Proceedings of Conference on Computer Vision and Pattern Recognition (CVPR) (2018)
10. Hegde, D., Valanarasu, J.M.J., Patel, V.: Clip goes 3d: Leveraging prompt tuning for language grounded 3d recognition. In: Proceedings of International Conference on Computer Vision (ICCV) (2023)
11. Li, J., Li, D., Savarese, S., Hoi, S.: BLIP-2: bootstrapping language-image pre-training with frozen image encoders and large language models. In: Proceedings of International Conference on Machine Learning (ICML) (2023)
12. Mao, Y., Zhang, Y., Jiang, H., Chang, A., Savva, M.: Multiscan: Scalable rgbd scanning for 3d environments with articulated objects. In: Proceedings of Advances in Neural Information Processing Systems (NeurIPS) (2022)

13. OpenAI: Introducing chatgpt. `https://openai.com/blog/chatgpt` (2022)
14. Qi, C.R., Yi, L., Su, H., Guibas, L.J.: Pointnet++: Deep hierarchical feature learning on point sets in a metric space. In: Proceedings of Advances in Neural Information Processing Systems (NeurIPS) (2017)
15. Radford, A., Kim, J.W., Hallacy, C., Ramesh, A., Goh, G., Agarwal, S., Sastry, G., Askell, A., Mishkin, P., Clark, J., et al.: Learning transferable visual models from natural language supervision. In: Proceedings of International Conference on Machine Learning (ICML) (2021)
16. Ramakrishnan, S.K., Gokaslan, A., Wijmans, E., Maksymets, O., Clegg, A., Turner, J., Undersander, E., Galuba, W., Westbury, A., Chang, A.X., et al.: Habitat-matterport 3d dataset (hm3d): 1000 large-scale 3d environments for embodied ai. In: Proceedings of Advances in Neural Information Processing Systems Datasets and Benchmarks (NeurIPS Datasets and Benchmarks Track) (2021)
17. Touvron, H., Lavril, T., Izacard, G., Martinet, X., Lachaux, M.A., Lacroix, T., Rozière, B., Goyal, N., Hambro, E., Azhar, F., et al.: Llama: Open and efficient foundation language models. arXiv preprint arXiv:2302.13971 (2023)
18. Wald, J., Avetisyan, A., Navab, N., Tombari, F., Nießner, M.: Rio: 3d object instance re-localization in changing indoor environments. In: Proceedings of International Conference on Computer Vision (ICCV) (2019)
19. Wald, J., Dhamo, H., Navab, N., Tombari, F.: Learning 3d semantic scene graphs from 3d indoor reconstructions. In: Proceedings of Conference on Computer Vision and Pattern Recognition (CVPR) (2020)
20. Yang, J., Ding, R., Wang, Z., Qi, X.: Regionplc: Regional point-language contrastive learning for open-world 3d scene understanding. arXiv preprint arXiv:2304.00962 (2023)
21. Yang, Y.Q., Guo, Y.X., Xiong, J.Y., Liu, Y., Pan, H., Wang, P.S., Tong, X., Guo, B.: Swin3d: A pretrained transformer backbone for 3d indoor scene understanding. arXiv preprint arXiv:2304.06906 (2023)
22. Zheng, J., Zhang, J., Li, J., Tang, R., Gao, S., Zhou, Z.: Structured3d: A large photo-realistic dataset for structured 3d modeling. In: Proceedings of European Conference on Computer Vision (ECCV) (2020)
23. Zhu, Z., Ma, X., Chen, Y., Deng, Z., Huang, S., Li, Q.: 3d-vista: Pre-trained transformer for 3d vision and text alignment. In: Proceedings of International Conference on Computer Vision (ICCV) (2023)