This CVPR paper is the Open Access version, provided by the Computer Vision Foundation.
Except for this watermark, it is identical to the accepted version;
the final published version of the proceedings is available on IEEE Xplore.

PartSLIP: Low-Shot Part Segmentation for 3D Point Clouds via Pretrained Image-Language Models

Minghua Liu[1] Yinhao Zhu[2] Hong Cai[2] Shizhong Han[2] Zhan Ling[1] Fatih Porikli[2] Hao Su[1]
[1]UC San Diego [2]Qualcomm AI Research*

Figure 1. We propose PartSLIP, a zero/few-shot method for 3D point cloud part segmentation by leveraging pretrained image-language models. The figure shows text prompts and corresponding semantic segmentation results (zoom in for details). Our method also supports part-level instance segmentation. See Figure 5 and Figure 7 for more results.

Abstract

Generalizable 3D part segmentation is important but challenging in vision and robotics. Training deep models via conventional supervised methods requires large-scale 3D datasets with fine-grained part annotations, which are costly to collect. This paper explores an alternative way for low-shot part segmentation of 3D point clouds by leveraging a pretrained image-language model, GLIP, which achieves superior performance on open-vocabulary 2D detection. We transfer the rich knowledge from 2D to 3D through GLIP-based part detection on point cloud rendering and a novel 2D-to-3D label lifting algorithm. We also utilize multi-view 3D priors and few-shot prompt tuning to boost performance significantly. Extensive evaluation on PartNet and PartNet-Mobility datasets shows that our method enables excellent zero-shot 3D part segmentation. ***Our few-shot version not only outperforms existing few-shot approaches by a large margin but also achieves highly competitive results compared to the fully supervised counterpart.*** *Furthermore, we demonstrate that our method can be directly applied to iPhone-scanned point clouds without significant domain gaps.*

1. Introduction

Human visual perception can parse objects into parts and generalize to unseen objects, which is crucial for understanding their structure, semantics, mobility, and functionality. 3D part segmentation plays a critical role in empowering machines with such ability and facilitates a wide range of applications, such as robotic manipulation, AR/VR, and shape analysis and synthesis [2, 31, 39, 69].

Recent part-annotated 3D shape datasets [40, 67, 72] have promoted advances in designing various data-driven approaches for 3D part segmentation [34, 44, 65, 73]. While standard supervised training enables these methods to achieve remarkable results, they often struggle with out-of-distribution test shapes (e.g., unseen classes). However, compared to image datasets, these 3D part-annotated datasets are still orders of magnitude smaller in scale, since building 3D models and annotating fine-grained 3D object parts are laborious and time-consuming. It is thus challenging to provide sufficient training data covering all object categories. For example, the recent PartNet dataset [40] contains only 24 object categories, far less than what an intelligent agent would encounter in the real world.

To design a generalizable 3D part segmentation module, many recent works have focused on the few-shot setting, assuming only a few 3D shapes of each category during training. They design various strategies to learn better representations, and complement vanilla supervised learning [33, 53, 54, 60, 80]. While they show improvements over the original pipeline, there is still a large gap between what these models can do and what downstream applica-

*Qualcomm AI Research is an initiative of Qualcomm Technologies, Inc.

tions need. The problem of generalizable 3D part segmentation is still far from being solved. Another parallel line of work focuses on learning the concept of universal object parts and decomposing a 3D shape into a set of (hierarchical) fine-grained parts [37, 64, 74]. However, these works do not consider the semantic labeling of parts and may be limited in practical use.

In this paper, we seek to solve the low-shot (zero- and few-shot) 3D part segmentation problem by leveraging pretrained image-language models, inspired by their recent striking performances in low-shot learning. By pretraining on large-scale image-text pairs, image-language models [1,22,29,45,46,50,76] learn a wide range of visual concepts and knowledge, which can be referenced by natural language. Thanks to their impressive zero-shot capabilities, they have already enabled a variety of 2D/3D vision and language tasks [10, 16, 20, 47, 49, 51, 77].

As shown in Figure 1, our method takes a 3D point cloud and a text prompt as input, and generates both 3D semantic and instance segmentations in a zero-shot or few-shot fashion. Specifically, we integrate the GLIP [29] model, which is pretrained on 2D visual grounding and detection tasks with over 27M image-text pairs and has a strong capability to recognize object parts. To connect our 3D input with the 2D GLIP model, we render multi-view 2D images for the point cloud, which are then fed into the GLIP model together with a text prompt containing part names of interest. The GLIP model then detects parts of interest for each 2D view and outputs detection results in the form of 2D bounding boxes. Since it is non-trivial to convert 2D boxes back to 3D, we propose a novel 3D voting and grouping module to fuse the multi-view 2D bounding boxes and generate 3D instance segmentation for the input point cloud. Also, the pretrained GLIP model may not fully understand our definition of parts only through text prompts. We find that an effective solution is prompt tuning with few-shot segmented 3D shapes. In prompt tuning, we learn an offset feature vector for the language embedding of each part name while fixing the parameters of the pretrained GLIP model. Moreover, we propose a multi-view visual feature aggregation module to fuse the information of multiple 2D views, so that the GLIP model can have a better global understanding of the input 3D shape instead of predicting bounding boxes from each isolated 2D view.

To better understand the generalizability of various approaches and their performances in low-shot settings, we propose a benchmark PartNet-Ensembled (PartNetE) by incorporating two existing datasets PartNet [40] and PartNetMobility [67]. Through extensive evaluation on PartNetE, we show that our method enables excellent zero-shot 3D part segmentation. With few-shot prompt tuning, our method not only outperforms existing few-shot approaches by a large margin but also achieves highly competitive performance compared to the fully supervised counterpart. We also demonstrate that our method can be directly applied to iPhone-scanned point clouds without significant domain gaps. In summary, our contributions mainly include:

- We introduce a novel 3D part segmentation method that leverages pretrained image-language models and achieves outstanding zero-shot and few-shot performance.
- We present a 3D voting and grouping module, which effectively converts multi-view 2D bounding boxes into 3D semantic and instance segmentation.
- We utilize few-shot prompt tuning and multi-view feature aggregation to boost GLIP's detection performance.
- We propose a benchmark PartNetE that benefits future work on low-shot and text-driven 3D part segmentation.

2. Related Work

2.1. 3D Part Segmentation

3D part segmentation involves two main tasks: semantic segmentation and instance segmentation. Most 3D backbone networks [43, 44, 56, 65] are capable of semantic segmentation by predicting a semantic label for each geometric primitive (e.g., point or voxel). Existing learning-based approaches solve instance segmentation by incorporating various grouping [9, 15, 23, 30, 58, 62, 63, 75] or region proposal [17,70,73] strategies into the pipeline. Different from standard training with per-point part labels, some works leverage weak supervision, such as bounding box [8, 35], language reference game [26], or IKEA manual [61]. Instead of focusing on single objects, [4,42] also consider part segmentation for scene-scale input. Moreover, unlike the two classical tasks of semantic and instance segmentation, another parallel line of works decomposes a 3D shape into a set of (hierarchical) fine-grained parts but without considering semantic labels [37, 64, 74], which differs from our objective. Recently, some works also propose to learn a continuous implicit semantic field [25, 81].

2.2. Data-Efficient 3D Segmentation

In order to train a generalizable 3D part segmentation network with low-shot data, many existing efforts focus on leveraging various pretext tasks and auxiliary losses [3, 12, 14, 52, 55]. In addition, [13, 41] studies the compositional generalization of 3D parts. [60] deforms input shapes to align with few-shot template shapes. [53] leverages 2D contrastive learning by projecting 3D shapes and learning dense multi-view correspondences. [7] leverages branched autoencoders to co-segment a collection of shapes. Also, some works aim to learn better representations by utilizing prototype learning [80], reinforcement learning [33], and data augmentation [54]. Moreover, there is a line of work investigating label-efficient 3D segmentation [18, 32, 36, 68, 71, 78, 78, 79], assuming a small portion of training data is annotated (e.g., 0.1% point labels). While

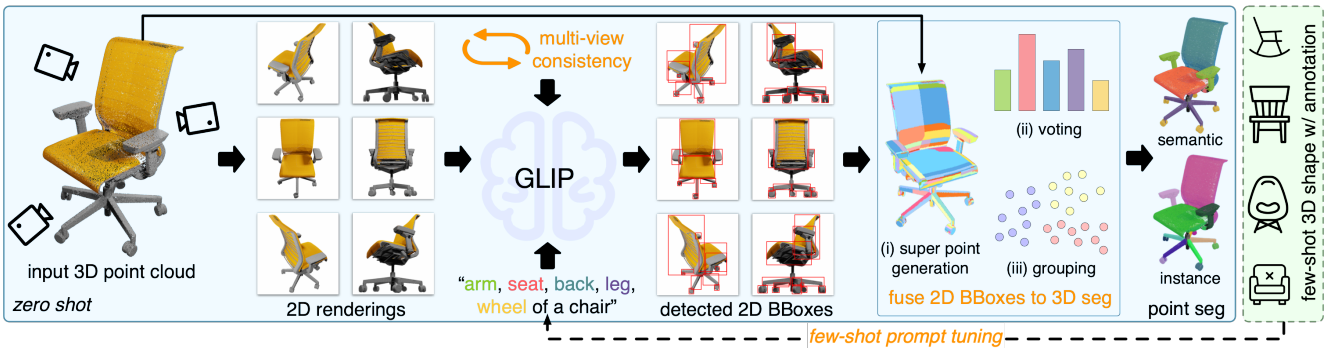

Figure 2. The figure shows our overall pipeline. Our proposed components are highlighted in orange.

the setting may be useful in indoor and autonomous driving scenarios, it is not aligned with our goal since the number of training shapes is already limited in our setup.

2.3. 3D Learning with Image-Language Models

Pretrained image-language models have recently made great strides by pretraining on large-scale image-text pairs [1, 22, 29, 45, 46, 50, 76]. Due to their learned rich visual concepts and impressive zero-shot capabilities, they have been applied to a wide range of 3D vision tasks, such as 3D avatar generation and manipulation [5, 16, 21], general 3D shape generation [19, 24, 38, 51], low-shot 3D shape classification [77], neural radiance fields [20, 59], 3D visual grounding [10, 57], and 3D representation learning [49]. To the best of our knowledge, we are one of the first to utilize pretrained image-language models to help with the task of 3D part segmentation.

3. Proposed Method: PartSLIP

3.1. Overview: 3D Part Segmentation with GLIP

We aim to solve both semantic and instance segmentation for 3D object parts by leveraging pretrained image-language models (ILMs). There are various large-scale ILMs emerged in the past few years. In order to enable generalizable 3D object part segmentation, the pre-trained ILM is expected to be capable of generating region-level output (e.g., 2D segmentation or 2D bounding boxes) and recognizing object parts. After comparing several released pretrained ILMs (e.g., CLIP [45]), we find that the GLIP [29] model is a good choice. The GLIP [29] model focuses on 2D visual grounding and detection tasks. It takes as input a free-form text description and a 2D image, and locates all phrases of the text by outputting multiple 2D bounding boxes for the input image. By pretraining on large-scale image-text pairs (e.g., 27M grounding data), the GLIP model learns a wide range of visual concepts (e.g., object parts) and enables open-vocabulary 2D detection.

Figure 2 shows our overall pipeline, where we take a 3D point cloud as input. Here, we consider point clouds from unprojecting and fusing multiple RGB-D images, which is a common setup in real-world applications and leads to dense points with color and normal. To connect the 2D GLIP model with our 3D point cloud input, we render the point cloud from K predefined camera poses. The camera poses are uniformly spaced around the input point cloud, aiming to cover all regions of the shape. Since we assume a dense and colored point cloud input[1], we render the point cloud by simple rasterization without introducing significant artifacts. The K rendered images are then fed separately into the pretrained GLIP model along with a text prompt. We format the text prompt by concatenating all part names of interest and the object category. For example, for a chair point cloud, the text prompt could be "arm, back, seat, leg, wheel of a chair". Please note that unlike the traditional segmentation networks, which are limited to a closed set of part categories, our method is more flexible and can include any part name in the text prompt. For each 2D rendered image, the GLIP model is expected to predict multiple bounding boxes, based on the text prompt, for all part instances that appear. We then fuse all bounding boxes from K views into 3D to generate semantic and instance segmentation for the input point cloud (Section 3.2).

The above pipeline introduces an intuitive zero-shot approach for 3D part segmentation without requiring any 3D training. However, its performance may be limited by the GLIP predictions. We thus propose two additional components, which could be incorporated into the above pipeline to encourage more accurate GLIP prediction: (a) prompt tuning with few-shot 3D data, which enables the GLIP model to quickly adapt to the meaning of each part name (Section 3.3); (b) multi-view feature aggregation, which allows the GLIP model to have a more comprehensive visual understanding of the input 3D shape (Section 3.4).

3.2. Detected 2D BBoxes to 3D Point Segmentation

Although the correspondence between 2D pixels and 3D points are available, there are still two main challenges when converting the detected 2D bounding boxes to 3D

[1]Recent commodity-grade 3D scanning devices (e.g., iPhone 12 Pro) can already capture high-quality point clouds (see Figure 7).

point segmentation. First, bounding boxes are not as precise as point-wise labels. A 2D bounding box may cover points from other part instances as well. Also, although each bounding box may indicate a part instance, we are not provided with their relations across views. It's not very straightforward to determine which sets of 2D bounding boxes indicate the same 3D part instance.

Therefore, we propose a learning-free module to convert the GLIP predictions to 3D point segmentation, which mainly includes three steps: (a) oversegment the input point cloud into a collection of super points; (b) assign a semantic label for each super point by 3D voting; and (c) group super points within each part category into instances based on their similarity of bounding box coverage.

3D Super Point Generation: We follow the method in [28] to oversegment the input point cloud into a collection of super points. Specifically, we utilize point normal and color as features and solve a *generalized minimal partition problem* with an l_0-cut pursuit algorithm [27]. Since points in each generated super point share similar geometry and appearance, we assume they belong to one part instance. The super point partition serves as an important 3D prior when assigning semantic and instance labels. It also speeds up the label assignment, as the number of super points is orders of magnitude smaller than the number of 3D points.

3D Semantic Voting: While a single bounding box may cover irrelevant points from other parts, we want to leverage information from multiple views and the super point partition to counteract the effect of irrelevant points. Specifically, for each pair of super point and part category, we calculate a score $s_{i,j}$ measuring the proportion of the ith super point covered by any bounding box of part category j:

$$s_{i,j} = \frac{\sum_k \sum_{p \in SP_i} [\mathrm{VIS}_k(p)][\exists b \in BB_k^j : \mathrm{INS}_b(p)]}{\sum_k \sum_{p \in SP_i} [\mathrm{VIS}_k(p)]}, \quad (1)$$

where SP_i indicates the ith super point, $[\cdot]$ is the Iverson bracket, $\mathrm{VIS}_k(p)$ indicates whether the 3D point p is visible in view k, BB_k^j is a list of predicted bounding boxes of category j in view k, and $\mathrm{INS}_b(p)$ indicates whether the projection of point p in view k is inside the bounding box b.

Note that for each view, we only consider visible points since bounding boxes only contain visible portions of each part instance. Both $\mathrm{VIS}_k(p)$ and $\mathrm{INS}_b(p)$ can be computed based on the information from point cloud rasterization. After that, for each super point i, we assign part category j with the highest score $s_{i,j}$ to be its semantic label.

3D Instance Grouping: In order to group the super points into part instances, we first regard each super point as an individual instance and then consider whether to merge each pair of super points. For a pair of super points SP_u and SP_v, we merge them if: (a) they have the same semantic label, (b) they are adjacent in 3D, and (c) for each bounding box, they are either both included or both excluded.

Specifically, for the second criterion, we find the k nearest neighbors for all points within each super point. If any point in SP_v is among the k nearest neighbors of a point in SP_u, or vice versa, we consider the super points to be adjacent. For the third criterion, we consider bounding boxes from views where both of them are visible:

$$B = \{b \in BB_k | \, \mathrm{VIS}_k(SP_u) \wedge \mathrm{VIS}_k(SP_v)\}, \quad (2)$$

where $\mathrm{VIS}_k(SP_u)$ indicates whether the super point SP_u can be (partially) visible in view k and BB_k indicates all predicted bounding boxes of view k. Suppose B contains n bounding boxes. We then construct two n dimensional vectors I_u and I_v, describing the bounding box coverage of SP_u and SP_v. Specifically, $I_u[i]$ is calculated as:

$$I_u[i] = \frac{\sum_{p \in SP_u} [\mathrm{VIS}_{B[i]}(p)][\mathrm{INS}_{B[i]}(p)]}{\sum_{p \in SP_u} [\mathrm{VIS}_{B[i]}(p)]}, \quad (3)$$

where $B[i]$ indicates the ith bounding box of B, $\mathrm{VIS}_{B[i]}(p)$ indicates whether p is visible in the corresponding view of $B[i]$, and $\mathrm{INS}_{B[i]}(p)$ indicates whether the projection of p is inside $B[i]$. If $\frac{|I_u - I_v|_1}{max(|I_u|_1, |I_v|_1)}$ is smaller then a predefined threshold τ, we consider they satisfy the third criterion.

After checking all pairs of super points, the super points are divided into multiple connected components, each of which is then considered to be a part instance. We found that our super point-based module works well in practice.

3.3. Prompt Tuning w/ Few-Shot 3D Data

In our method, we utilize natural language to refer to a part. However, natural language can be flexible. An object part can be named in multiple ways (e.g., spout and mouth for kettles; caster and wheel for chairs), and the definition of some parts may be ambiguous (see the dispenser in Figure 1). We thus hope to finetune the GLIP model using a few 3D shapes with ground truth part segmentation, so that the GLIP model can quickly adapt to the actual definition of the part names in the text prompt.

Figure 3 shows the overall architecture of the GLIP model. It first employs a language encoder and an image encoder to extract language features and multi-scale visual features, respectively, which are then fed into a vision-language fusion module to fuse information across modalities. The detection head then takes as input the language-aware image features and predicts 2D bounding boxes. During pretraining, the GLIP network is supervised by both detection loss and image-language alignment loss.

It is not desirable to change the parameters of the visual module or the entire GLIP model since our goal is to leverage only a few 3D shapes for finetuning. Instead, we follow the prompt tuning strategy introduced in GLIP [29] to finetune only the language embedding of each part name while freezing the parameters of the pretrained GLIP model. Specifically, we perform prompt tuning for each object category separately. Suppose the input text of an object category

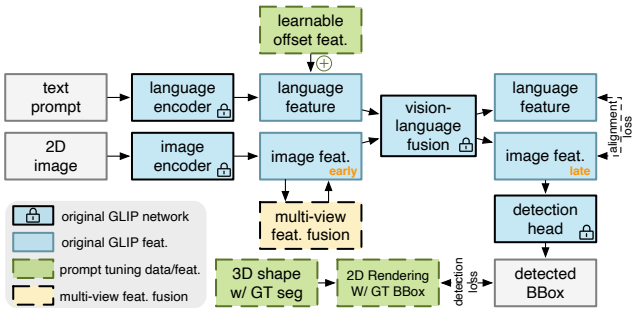

Figure 3. The original GLIP pipeline and our additional modules: few-shot prompt tuning and multi-view feature aggregation. We find that early fusion leads to better performance than late fusion.

includes l tokens and denote the extracted language features (before VL fusion) as $f_l \in \mathbb{R}^{l\times c}$, where c is the number of channels. We aim to learn offset features $f_o \in \mathbb{R}^{l\times c}$ for f_l and feed their summation $f_l + f_o$ to the remaining GLIP pipeline. The offset features f_o consist of constant vectors for each token (part name), which can be interpreted as a local adjustment of the part definition in the language embedding space. Note that f_o is not predicted by a network but is directly optimized as a trainable variable during prompt tuning. Also, f_o will be fixed for each object category after prompt tuning.

In order to utilize the detection and alignment losses for optimization, we convert the few-shot 3D shapes with ground truth instance segmentation into 2D images with bounding boxes. Specifically, for each 3D point cloud, we render K 2D images from the predefined camera poses. For generating corresponding 2D ground-truth bounding boxes, we project each part instance from 3D to 2D. Note that, after projection, we need to remove occluded points (i.e., invisible points of each view) and noisy points (i.e., visible but isolated in tiny regions) to generate reasonable bounding boxes. We find that by prompt tuning with only one or a few 3D shapes, the GLIP model can quickly adapt to our part definitions and generalize to other instances.

3.4. Multi-View Visual Feature Aggregation

The GLIP model is sensitive to camera views. For example, images taken from some unfamiliar views (e.g., the rear view of a cabinet) can be uninformative and confusing, making it difficult for the GLIP model to predict accurately. However, unlike regular 2D recognition tasks, our input is a 3D point cloud, and there are pixel-wise correspondences between different 2D views. Therefore, we hope the GLIP model can leverage these 3D priors to make better predictions instead of focusing on each view in isolation.

In order to take full advantage of the pretrained GLIP model, we propose a training-free multi-view visual feature aggregation module that could be plugged into the original GLIP network without changing any existing network weights. Specifically, the feature aggregation module takes

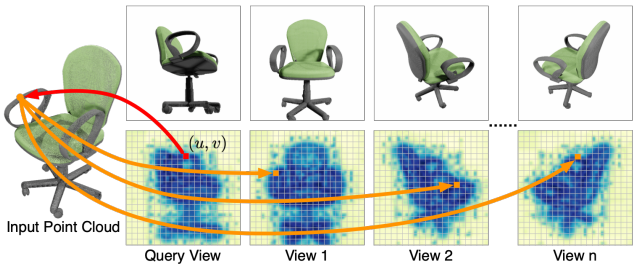

Figure 4. Multi-view 2D renderings (first row) and their feature maps (second row). For a feature cell (red), we aggregate all its corresponding feature cells (orange) across views.

K feature maps $\{f_k \in \mathbb{R}^{m\times m\times c}\}$ as input, where m is the spatial resolution of the feature map and c is the number of channels. The input feature maps $\{f_k\}$ are generated by the GLIP module separately for each 2D view of the input point cloud. Our feature aggregation module fuses them and generates K fused feature maps $\{f'_k\}$ of the same shape, which are then used to replace the original feature maps and fed into the remaining layers of the GLIP model.

As shown in Figure 4, for each cell (u, v) of feature map f_i, we find its corresponding cell $(u^{i\to k}, v^{i\to k})$ in each feature map f_k and use their weighted average to serve as the fused feature of the cell:

$$f'_i[u, v] = \frac{1}{\sum_k w^{i\to k}_{u,v}} \sum_k w^{i\to k}_{u,v} f_k[u^{i\to k}, v^{i\to k}]. \quad (4)$$

Specifically, we define $P_i(u, v)$ as the set of 3D points that are visible in view i and whose projections lie within cell (u, v). We then choose the cell in view k with the most overlapping 3D points as the corresponding cell: $(u^{i\to k}, v^{i\to k}) = \arg\max_{(x,y)} |P_i(u, v) \cap P_k(x, y)|$ and define the weights $w^{i\to k}_{u,v}$ as $\frac{|P_i(u,v)\cap P_k(u^{i\to k}, v^{i\to k})|}{|P_i(u,v)|}$. Note that if all 3D points in $P_i(u, v)$ are not visible in a view k, then feature map f_k will not contribute to $f'_i[u, v]$. Since the GLIP model generates multi-scale visual features, our aggregation module fuses features of each scale level separately.

There are various options for which visual features to fuse (see Figure 3). One intuitive choice is to fuse the final visual features before the detection head, and we denote this choice as *late fusion*. We find that the late fusion does not improve or even degrade the original performance. This is mainly because the final visual features contain too much shape information of the predicted 2D bounding boxes. Directly averaging the final visual features can somehow be seen as averaging bounding boxes in 2D, which does not make sense. Instead, we choose to fuse the visual features before the vision-language fusion (denoted as *early fusion*). Since the text prompt is not involved yet, the visual features mainly describe the geometry and appearance of the input shape. Fusing these features across views with the 3D priors can thus lead to a more comprehensive visual understanding of the input shape.

4. Experiments

4.1. Datasets and Metrics

To evaluate the generalizability of various approaches and their performances in the low-shot setting, we curate an ensembled dataset named PartNet-Ensembled (PartNetE), which consists of shapes from existing datasets PartNet [40] and PartNet-Mobility [67]. Note that PartNet-Mobility contains more object categories but fewer shape instances, and PartNet contains more shape instances but fewer object categories. We thus utilize shapes from PartNet-Mobility for few-shot learning and test, and use shapes from PartNet to serve as additional large-scale training data for transfer learning. As a result, the test set of PartNetE contains 1,906 shapes covering 45 object categories. In addition, we randomly reserve 8 shapes from each of the 45 object categories for few-shot training. Also, we may utilize the additional 28,367 shapes from PartNet for training, which cover 17 out of 45 object categories and have consistent part annotations as the test set. Some of the original part categories in PartNet (e.g., "back_frame_vertical_bar" for chairs) are too fine-grained and ambiguous to evaluate unsupervised text-driven part segmentation approaches. We thus select a subset of 103 parts when constructing the PartNetE dataset, which covers both common coarse-grained parts (e.g., chair back and tabletop) and fine-grained parts (e.g., wheel, handle, button, knob, switch, touchpad) that may be useful in downstream tasks such as robotic manipulation. See supplementary for more details of the dataset.

We follow [40] to utilize category mIoU and mAP (50% IoU threshold) as the semantic and instance segmentation metrics, respectively. We first calculate mIoU/mAP50 for each part category across all test shapes, and then average part mIoUs/mAP50s that belong to each object category to compute the object category mIoU/mAP50.

4.2. Implementation Details

For each 3D shape (i.e., ShapeNet [6] mesh), we use BlenderProc [11] to render 6 views of RGB-D images and segmentation masks with a resolution of 512×512. We unproject the images to the world space to obtain a fused point cloud with colors, normals, and ground truth part labels. The fused point clouds are used as the input for both our method and baseline approaches.

For our method, we render each input point cloud into $K = 10$ color images with Pytorch3D [48]. In few-shot experiments, we utilize 8 point clouds (8×10 rendered images with 2D bounding boxes) of each object category for prompt tuning. The threshold τ in part instance grouping is empirically set to 0.3.

4.3. Comparison with Existing Methods

4.3.1 Low-Shot Settings and Baseline Methods

We consider three low-shot settings: (a) zero-shot: no 3D training/finetuning involved; (b) few-shot (45×8): utilize only 8 shapes for each object category during training; (c) few-shot with additional data ($45 \times 8 + 28k$): utilize 28,367 shapes from PartNet [40] in addition to the 45×8 shapes during training. The 28k shapes cover 17 of the 45 object categories. Here, the last setting ($45 \times 8 + 28k$) describes a realistic setup, where we have large-scale part annotations for some common categories (17 categories in our case) but only a few shapes for the other categories. We aim to examine whether the 28k data of the 17 categories can help the part segmentation of the other 28 underrepresented categories. All settings are tested on the same test set.

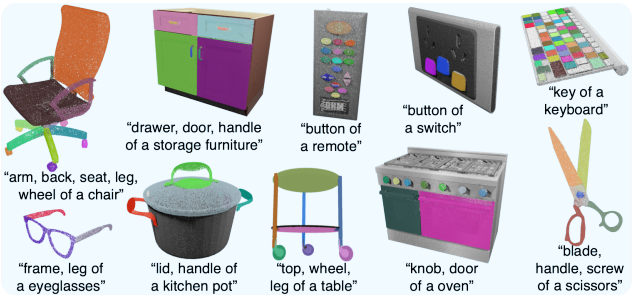

Figure 5. Instance segmentation results of our method (8-shot) on the PartNetE dataset. Different part instances are in different colors (zoom in for details).

We compare with PointNet++ [43] and PointNext [44] for semantic segmentation, and compare with PointGroup [23] and SoftGroup [58] for instance segmentation. We train four baseline approaches on the PartNetE dataset by taking point clouds with normals as input. For semantic segmentation, we follow [40] to sample 10,000 points per shape as network input. For instance segmentation, we sample up to 50,000 points per shape. For each pair of baseline and setting, we train a single network.

In addition to the four baselines mentioned above, we compare against two methods dedicated to few-shot 3D semantic segmentation: ACD [12] and Prototype [80]. In ACD, we decompose the mesh of each 3D shape into approximate convex components with CoACD [66] and utilize the decomposition results for adding an auxiliary loss to the pipeline of PointNet++. In Prototype, we utilize the learned point features (by PointNext backbone) of few-shot shapes to construct 100 prototypes for each part category, which are then used to classify each point of test shapes. See supplementary for more details of baseline approaches.

4.3.2 Evaluation Results

Table 1 shows the results of semantic segmentation. Our method achieves impressive zero-shot performance on some common object categories (such as bottle, chair, and table), but also poor performances on certain categories (e.g., kettle). This is mainly due to the pretrained GLIP model may not understand the meaning of the text prompt (e.g., spout for kettles). After prompt tuning with 8-shot 3D data, our method achieves a 59.4% mIoU and outperforms

Table 1. Semantic segmentation results on the PartNetE dataset. Object category mIoU(%) are shown. For 17 overlapping object categories, baseline models leverage additional 28k training shapes in the 45x8+28k setting. For the other 28 non-overlapping object categories, there are only 8 shapes per object category during training. Please refer to the supplementary for the full table of all 45 categories.

#3D data	method	Overlapping Categories									Non-Overlapping Categories									
		Bottle	Chair	Display	Door	Knife	Lamp	Storage Furniture	Table	Overall (17)	Camera	Cart	Dis-Penser	Kettle	Kitchen-Pot	Oven	Suit-case	Toaster	Overall (28)	Overll (45)
few-shot w/ extra data (45x8+28k)	PointNet++ [43]	48.8	84.7	78.4	45.7	35.4	68.0	46.9	**63.7**	55.6	6.5	6.4	12.1	20.9	15.8	34.3	40.6	14.7	25.4	36.8
	PointNext [44]	68.4	**91.8**	**89.4**	43.8	58.7	64.9	**68.5**	52.1	**58.5**	33.2	36.3	26.0	45.1	57.0	37.8	13.5	8.3	**45.1**	**50.2**
	SoftGroup [58]	41.4	88.3	62.1	**53.1**	31.3	**82.2**	60.2	54.8	50.2	23.6	23.9	18.9	57.4	45.5	13.6	18.3	26.4	30.7	38.1
few-shot (45x8)	PointNet++ [43]	27.0	42.2	30.2	20.5	22.2	10.5	8.4	7.3	18.1	9.7	11.6	7.0	28.6	31.7	19.4	3.3	0.0	21.8	20.4
	PointNext [44]	67.6	65.1	53.7	46.3	59.7	55.4	20.6	22.1	39.2	26.0	47.7	22.6	60.5	66.0	36.8	14.5	0.0	41.5	40.6
	SoftGroup [58]	20.8	80.5	39.7	16.3	38.3	38.3	18.9	24.9	32.8	28.6	40.8	42.9	60.7	54.8	35.6	29.8	14.8	41.1	38.0
	ACD [12]	22.4	39.0	29.2	18.9	39.6	13.7	7.6	13.5	19.2	10.1	31.5	19.4	40.2	51.8	8.9	13.2	0.0	25.6	23.2
	Prototype [80]	60.1	70.8	67.3	33.4	50.4	38.2	30.2	25.7	41.1	32.0	36.8	53.4	62.7	63.3	36.5	35.5	10.1	46.3	44.3
	Ours	**83.4**	85.3	84.8	40.8	**65.2**	66.0	53.6	42.4	**56.3**	**58.3**	**88.1**	**73.7**	**77.0**	**69.6**	**73.5**	**70.4**	**60.0**	**61.3**	**59.4**
zero-shot	**Ours**	76.3	60.7	43.8	2.7	46.8	37.1	29.4	47.7	31.8	21.4	87.7	16.5	20.8	4.7	33.0	40.2	13.8	24.4	27.2

Table 2. Instance segmentation results on the PartNetE dataset. Category mAP50 (%) are shown. See supplementary for the full table.

#3D data	method	Overlapping Categories									Non-Overlapping Categories									
		Bottle	Chair	Display	Door	Knife	Lamp	Storage Furniture	Table	Overall (17)	Camera	Cart	Dis-Penser	Kettle	Kitchen-Pot	Oven	Suit-case	Toaster	Overall (28)	Overll (45)
45x8+28k	PointGroup [23]	38.2	87.6	65.1	**23.4**	19.3	62.7	**49.1**	**46.4**	41.7	8.6	29.2	24.0	61.3	59.4	13.8	15.6	7.0	24.6	31.0
	SoftGroup [58]	43.9	**89.1**	68.7	21.2	27.2	63.3	**49.1**	46.2	**42.4**	0.7	28.4	26.4	63.8	59.3	16.4	13.5	7.5	**25.6**	**31.9**
few-shot (45x8)	PointGroup [23]	8.0	77.2	16.7	3.7	15.6	9.8	0.0	0.0	14.6	4.7	28.5	30.7	52.1	57.0	0.0	0.0	0.0	16.8	16.0
	SoftGroup [58]	22.4	87.7	27.5	5.6	10.3	19.4	11.6	14.2	21.3	11.2	29.8	37.8	63.4	65.7	10.4	8.0	10.7	28.4	25.7
	Ours	**79.4**	84.4	**82.9**	17.9	**43.9**	**68.3**	32.8	32.3	**42.5**	**36.8**	**83.3**	**63.5**	**75.4**	**70.5**	**64.5**	**44.9**	**38.4**	**46.2**	**44.8**
zero-shot	**Ours**	75.5	54.5	32.9	1.3	22.1	35.8	10.9	36.6	20.9	8.4	79.3	9.3	18.3	1.1	25.9	34.2	4.5	16.2	18.0

all baseline methods from the few-shot setting and even the $45 \times 8 + 28k$ setting. For the $45 \times 8 + 28k$ setting, baseline methods are trained with additional 28k shapes covering 17 categories. **For these overlapping categories, it's a fully-supervised setting, but our 8-shot version can achieve highly competitive overall mIoU (56.3% vs. 58.5%).** Note that the 28k training data is of limited help for the baselines to generalize to non-overlapping categories. Our method outperforms all baselines on non-overlapping categories by a large margin. The two few-shot strategies ACD and Prototype improve the performance of the original backbone, but there are still large gaps compared to our method. Please see Figure 1 for example results of our methods and see supplementary for qualitative comparison.

Table 2 shows the results of instance segmentation. We observe similar phenomena as semantic segmentation. Our method achieves 18.0% mAP50 for the zero-shot setting and 44.8% mAP50 for the 8-shot setting, which outperforms all baseline approaches from both 45×8 and $45 \times 8 + 28k$ settings. See Figure 5 for qualitative examples.

4.4. Ablation Studies

Proposed Components: We ablate the proposed components, and the results are shown in Table 3. For the first row, we only utilize the pretrained GLIP model. In order to get 3D semantic segmentation, we assign part labels to all visible points within bounding boxes. The numbers indicate that this strategy is less effective than our proposed 3D voting and grouping module (second row). Moreover, without our proposed module, we are not able to get 3D instance segmentation. The second and third rows compare the impact of (8-shot) prompt tuning. We observe significant improvements, especially on the Kettle category, as the zero-shot GLIP model fails to understand the meaning of "spout" but it adapts to the definition after few-shot prompt tuning. The second and fourth rows compare our multi-view feature aggregation module. Without utilizing any extra data for finetuning, we leverage multi-view 3D priors to help the GLIP model better understand the input 3D shape and thus improve performance. After integrating all three modules, we achieve the final good performance (last row).

Table 3. Ablation study of the proposed components. We show the performances of both GLIP 2D detection (category mAP50) and 3D semantic segmentation (category mIoU) on three categories. *3D semantic segmentation is generated by assigning part labels to all visible points in bounding boxes.

BBox2 3DSeg	Prompt Tuning	Feat Aggre.	Chair 2D	Chair 3D	Kettle 2D	Kettle 3D	Suitcase 2D	Suitcase 3D	All 3D
			50.4	50.6*	26.4	7.5*	31.9	21.1*	
✓			50.4	60.7	26.4	20.8	31.9	40.2	27.2
✓	✓		80.7	83.8	82.1	72.7	65.6	65.1	58.0
✓		✓	52.3	64.5	32.2	25.9	36.4	49.1	27.7
✓	✓	✓	82.4	85.3	84.3	77.0	68.9	70.4	59.4

Variations of Input Point Clouds: Table 4 evaluates the robustness of our method about variations of input point clouds. We observe that when the input point cloud is par-

Table 4. Ablation study of various input point clouds. We show the semantic segmentation results of the Chair category.

setting	# views	image reso.	texture	Chair mIoU (%)
original	6	512×512	w/	85.3
partial pc	2	512×512	w/	84.3
no texture	6	512×512	w/o	84.0
sparse pc	6	128×128	w/	82.4
sparse pc	6	64×64	w/	68.3

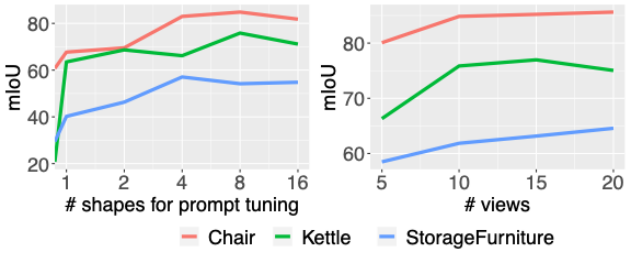

Figure 6. Ablation study of the number of shapes in prompt tuning and the number of 2D views (K). Category mIoU of 3D semantic segmentation on the PartNetE dataset are shown.

tial and does not cover all regions of the object, our method still performs well (second row). Also, we find that after removing the textures of the ShapeNet models and generating the input point cloud by using gray-scale images, our method can achieve good performance as well, suggesting that textures are less important in recognizing object parts. However, we find that the performance of our method may degrade when the input point cloud becomes sparse. On the one hand, sparse point clouds cause a larger domain gap for 2D renderings of point clouds. On the other hand, the sparsity makes it hard for our super point generation algorithm to produce good results. That being said, we want to point out that dense point clouds are already mostly available in our daily life (see Section 4.5).

Number of Shapes in Prompt Tuning: We ablate the number of shapes used for prompt tuning, and the results are shown in Figure 6 (left). We observe that only using one single shape for prompt tuning can already improve the performance of the pretrained GLIP model a lot in some categories (e.g., Kettle). Also, after using more than 4 shapes, the gain from increasing the number of shapes slows down. We also find that prompt tuning is less effective for object categories that have richer appearance and structure variations (e.g., StorageFurniture).

Number of 2D Views: We render $K = 10$ 2D views for each input point cloud in our main experiments. We ablate the value of K, and the results are shown in Figure 6 (right). We observe a significant performance drop when K is reduced to 5 and also a mild gain when using a larger K.

Early Fusion vs. Late Fusion: In the last paragraph of Section 3.4, we discuss two choices for multi-view feature aggregation: early fusion and late fusion. Table 5 compares these two choices and verifies that late fusion will even degrade the performance while early fusion is helpful.

Table 5. Early vs. late fusion in multi-view feature aggregation. We compare GLIP detection (mAP50) on the Suitcase category.

w/o fusion	early fusion	late fusion
65.6	68.9	47.3

Figure 7. Each pair shows a captured point cloud by iPhone (left) and the semantic segmentation result of our method (right).

GLIP vs. CLIP: We have also considered using other pretrained vision-language models, such as CLIP [45]. However, we find that the pretrained CLIP model fails to recognize fine-grained object parts and has difficulty generating region-level output. See supplementary for details.

4.5. Real-World Demo

Thanks to the strong generalizability of the GLIP model, our method can be directly deployed in the real world without a significant domain gap. As shown in Figure 7, we use an iPhone 12 Pro Max, equipped with a LiDAR sensor, to capture a video and feed the fused point cloud to our method. We observe similar performances as in our synthetic experiments. Please note that existing 3D networks are sensitive to the input format. For example, they assume objects are normalized in per-category canonical poses. Also, they need to overcome the significant domain gap, making it hard to deploy them directly in real scenarios. See supplementary for more details.

5. Discussion and Limitations

The current pipeline utilizes predicted bounding boxes from the GLIP model. We notice that GLIPv2 [76] has 2D segmentation capabilities, but their pretrained model is not released at the time of submission. We admit that it will be more natural to use 2D segmentation results, which are more accurate than bounding boxes, from pretrained models. However, we want to point out that it is still non-trivial to get 3D instance segmentation even from multi-view 2D segmentation, and all components of our proposed method would still be useful (with necessary adaptations). A bigger concern is that our method cannot handle the interior points of objects. It also suffers from long running time due to point cloud rendering and multiple inferences of the GLIP model. Therefore, using our method to distill the knowledge of 2D VL models and train 3D foundation models is a promising future direction, which may lead to more efficient inferences.

References

[1] Jean-Baptiste Alayrac, Jeff Donahue, Pauline Luc, Antoine Miech, Iain Barr, Yana Hasson, Karel Lenc, Arthur Mensch, Katie Millican, Malcolm Reynolds, et al. Flamingo: a visual language model for few-shot learning. *arXiv preprint arXiv:2204.14198*, 2022. 2, 3

[2] Jacopo Aleotti and Stefano Caselli. A 3d shape segmentation approach for robot grasping by parts. *Robotics and Autonomous Systems*, 60(3):358–366, 2012. 1

[3] Antonio Alliegro, Davide Boscaini, and Tatiana Tommasi. Joint supervised and self-supervised learning for 3d real world challenges. In *2020 25th International Conference on Pattern Recognition (ICPR)*, pages 6718–6725. IEEE, 2021. 2

[4] Alexey Bokhovkin, Vladislav Ishimtsev, Emil Bogomolov, Denis Zorin, Alexey Artemov, Evgeny Burnaev, and Angela Dai. Towards part-based understanding of rgb-d scans. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 7484–7494, 2021. 2

[5] Zehranaz Canfes, M Furkan Atasoy, Alara Dirik, and Pinar Yanardag. Text and image guided 3d avatar generation and manipulation. *arXiv preprint arXiv:2202.06079*, 2022. 3

[6] 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. 6

[7] Zhiqin Chen, Kangxue Yin, Matthew Fisher, Siddhartha Chaudhuri, and Hao Zhang. Bae-net: Branched autoencoder for shape co-segmentation. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 8490–8499, 2019. 2

[8] Julian Chibane, Francis Engelmann, Tuan Anh Tran, and Gerard Pons-Moll. Box2mask: Weakly supervised 3d semantic instance segmentation using bounding boxes. In *European Conference on Computer Vision*, pages 681–699. Springer, 2022. 2

[9] Ruihang Chu, Yukang Chen, Tao Kong, Lu Qi, and Lei Li. Icm-3d: Instantiated category modeling for 3d instance segmentation. *IEEE Robotics and Automation Letters*, 7(1):57–64, 2021. 2

[10] Rodolfo Corona, Shizhan Zhu, Dan Klein, and Trevor Darrell. Voxel-informed language grounding. *arXiv preprint arXiv:2205.09710*, 2022. 2, 3

[11] Maximilian Denninger, Martin Sundermeyer, Dominik Winkelbauer, Youssef Zidan, Dmitry Olefir, Mohamad Elbadrawy, Ahsan Lodhi, and Harinandan Katam. Blenderproc. *arXiv preprint arXiv:1911.01911*, 2019. 6

[12] Matheus Gadelha, Aruni RoyChowdhury, Gopal Sharma, Evangelos Kalogerakis, Liangliang Cao, Erik Learned-Miller, Rui Wang, and Subhransu Maji. Label-efficient learning on point clouds using approximate convex decompositions. In *European Conference on Computer Vision*, pages 473–491. Springer, 2020. 2, 6, 7

[13] Songfang Han, Jiayuan Gu, Kaichun Mo, Li Yi, Siyu Hu, Xuejin Chen, and Hao Su. Compositionally generalizable 3d structure prediction. *arXiv preprint arXiv:2012.02493*, 2020. 2

[14] Kaveh Hassani and Mike Haley. Unsupervised multi-task feature learning on point clouds. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 8160–8171, 2019. 2

[15] Tong He, Dong Gong, Zhi Tian, and Chunhua Shen. Learning and memorizing representative prototypes for 3d point cloud semantic and instance segmentation. In *European Conference on Computer Vision*, pages 564–580. Springer, 2020. 2

[16] Fangzhou Hong, Mingyuan Zhang, Liang Pan, Zhongang Cai, Lei Yang, and Ziwei Liu. Avatarclip: Zero-shot text-driven generation and animation of 3d avatars. *arXiv preprint arXiv:2205.08535*, 2022. 2, 3

[17] Ji Hou, Angela Dai, and Matthias Nießner. 3d-sis: 3d semantic instance segmentation of rgb-d scans. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 4421–4430, 2019. 2

[18] Ji Hou, Benjamin Graham, Matthias Nießner, and Saining Xie. Exploring data-efficient 3d scene understanding with contrastive scene contexts. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 15587–15597, 2021. 2

[19] Ajay Jain, Ben Mildenhall, Jonathan T Barron, Pieter Abbeel, and Ben Poole. Zero-shot text-guided object generation with dream fields. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 867–876, 2022. 3

[20] Ajay Jain, Matthew Tancik, and Pieter Abbeel. Putting nerf on a diet: Semantically consistent few-shot view synthesis. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 5885–5894, 2021. 2, 3

[21] Nikolay Jetchev. Clipmatrix: Text-controlled creation of 3d textured meshes. *arXiv preprint arXiv:2109.12922*, 2021. 3

[22] Chao Jia, Yinfei Yang, Ye Xia, Yi-Ting Chen, Zarana Parekh, Hieu Pham, Quoc Le, Yun-Hsuan Sung, Zhen Li, and Tom Duerig. Scaling up visual and vision-language representation learning with noisy text supervision. In *International Conference on Machine Learning*, pages 4904–4916. PMLR, 2021. 2, 3

[23] Li Jiang, Hengshuang Zhao, Shaoshuai Shi, Shu Liu, Chi-Wing Fu, and Jiaya Jia. Pointgroup: Dual-set point grouping for 3d instance segmentation. In *Proceedings of the IEEE/CVF conference on computer vision and Pattern recognition*, pages 4867–4876, 2020. 2, 6, 7

[24] Nasir Khalid, Tianhao Xie, Eugene Belilovsky, and Tiberiu Popa. Text to mesh without 3d supervision using limit subdivision. *arXiv preprint arXiv:2203.13333*, 2022. 3

[25] Amit Pal Singh Kohli, Vincent Sitzmann, and Gordon Wetzstein. Semantic implicit neural scene representations with semi-supervised training. In *2020 International Conference on 3D Vision (3DV)*, pages 423–433. IEEE, 2020. 2

[26] Juil Koo, Ian Huang, Panos Achlioptas, Leonidas J Guibas, and Minhyuk Sung. Partglot: Learning shape part segmentation from language reference games. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 16505–16514, 2022. 2

[27] Loic Landrieu and Guillaume Obozinski. Cut pursuit: Fast algorithms to learn piecewise constant functions on general weighted graphs. *SIAM Journal on Imaging Sciences*, 10(4):1724–1766, 2017. 4

[28] Loic Landrieu and Martin Simonovsky. Large-scale point cloud semantic segmentation with superpoint graphs. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 4558–4567, 2018. 4

[29] Liunian Harold Li, Pengchuan Zhang, Haotian Zhang, Jianwei Yang, Chunyuan Li, Yiwu Zhong, Lijuan Wang, Lu Yuan, Lei Zhang, Jenq-Neng Hwang, et al. Grounded language-image pre-training. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 10965–10975, 2022. 2, 3, 4

[30] Jinxian Liu, Minghui Yu, Bingbing Ni, and Ye Chen. Self-prediction for joint instance and semantic segmentation of point clouds. In *European Conference on Computer Vision*, pages 187–204. Springer, 2020. 2

[31] Minghua Liu, Xuanlin Li, Zhan Ling, Yangyan Li, and Hao Su. Frame mining: a free lunch for learning robotic manipulation from 3d point clouds. *arXiv preprint arXiv:2210.07442*, 2022. 1

[32] Minghua Liu, Yin Zhou, Charles R Qi, Boqing Gong, Hao Su, and Dragomir Anguelov. Less: Label-efficient semantic segmentation for lidar point clouds. In *European Conference on Computer Vision*, pages 70–89. Springer, 2022. 2

[33] Xueyi Liu, Xiaomeng Xu, Anyi Rao, Chuang Gan, and Li Yi. Autogpart: Intermediate supervision search for generalizable 3d part segmentation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 11624–11634, 2022. 1, 2

[34] Yongcheng Liu, Bin Fan, Shiming Xiang, and Chunhong Pan. Relation-shape convolutional neural network for point cloud analysis. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 8895–8904, 2019. 1

[35] Yan Liu, Qingyong Hu, Yinjie Lei, Kai Xu, Jonathan Li, and Yulan Guo. Box2seg: Learning semantics of 3d point clouds with box-level supervision. *arXiv preprint arXiv:2201.02963*, 2022. 2

[36] Zhengzhe Liu, Xiaojuan Qi, and Chi-Wing Fu. One thing one click: A self-training approach for weakly supervised 3d semantic segmentation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 1726–1736, 2021. 2

[37] Tiange Luo, Kaichun Mo, Zhiao Huang, Jiarui Xu, Siyu Hu, Liwei Wang, and Hao Su. Learning to group: A bottom-up framework for 3d part discovery in unseen categories. *arXiv preprint arXiv:2002.06478*, 2020. 2

[38] Oscar Michel, Roi Bar-On, Richard Liu, Sagie Benaim, and Rana Hanocka. Text2mesh: Text-driven neural stylization for meshes. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 13492–13502, 2022. 3

[39] Kaichun Mo, Paul Guerrero, Li Yi, Hao Su, Peter Wonka, Niloy Mitra, and Leonidas J Guibas. Structurenet: Hierarchical graph networks for 3d shape generation. *arXiv preprint arXiv:1908.00575*, 2019. 1

[40] Kaichun Mo, Shilin Zhu, Angel X Chang, Li Yi, Subarna Tripathi, Leonidas J Guibas, and Hao Su. Partnet: A large-scale benchmark for fine-grained and hierarchical part-level 3d object understanding. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 909–918, 2019. 1, 2, 6

[41] Muhammad Ferjad Naeem, Evin Pınar Örnek, Yongqin Xian, Luc Van Gool, and Federico Tombari. 3d compositional zero-shot learning with decompositional consensus. In *European Conference on Computer Vision*, pages 713–730. Springer, 2022. 2

[42] Alexandr Notchenko, Vladislav Ishimtsev, Alexey Artemov, Vadim Selyutin, Emil Bogomolov, and Evgeny Burnaev. Scan2part: Fine-grained and hierarchical part-level understanding of real-world 3d scans. *arXiv preprint arXiv:2206.02366*, 2022. 2

[43] Charles Ruizhongtai Qi, Li Yi, Hao Su, and Leonidas J Guibas. Pointnet++: Deep hierarchical feature learning on point sets in a metric space. *Advances in neural information processing systems*, 30, 2017. 2, 6, 7

[44] Guocheng Qian, Yuchen Li, Houwen Peng, Jinjie Mai, Hasan Hammoud, Mohamed Elhoseiny, and Bernard Ghanem. Pointnext: Revisiting pointnet++ with improved training and scaling strategies. *arXiv:2206.04670*, 2022. 1, 2, 6, 7

[45] Alec Radford, Jong Wook Kim, Chris Hallacy, Aditya Ramesh, Gabriel Goh, Sandhini Agarwal, Girish Sastry, Amanda Askell, Pamela Mishkin, Jack Clark, et al. Learning transferable visual models from natural language supervision. In *International Conference on Machine Learning*, pages 8748–8763. PMLR, 2021. 2, 3, 8

[46] Aditya Ramesh, Prafulla Dhariwal, Alex Nichol, Casey Chu, and Mark Chen. Hierarchical text-conditional image generation with clip latents. *arXiv preprint arXiv:2204.06125*, 2022. 2, 3

[47] Yongming Rao, Wenliang Zhao, Guangyi Chen, Yansong Tang, Zheng Zhu, Guan Huang, Jie Zhou, and Jiwen Lu. Denseclip: Language-guided dense prediction with context-aware prompting. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 18082–18091, 2022. 2

[48] Nikhila Ravi, Jeremy Reizenstein, David Novotny, Taylor Gordon, Wan-Yen Lo, Justin Johnson, and Georgia Gkioxari. Accelerating 3d deep learning with pytorch3d. *arXiv preprint arXiv:2007.08501*, 2020. 6

[49] David Rozenberszki, Or Litany, and Angela Dai. Language-grounded indoor 3d semantic segmentation in the wild. *arXiv preprint arXiv:2204.07761*, 2022. 2, 3

[50] Chitwan Saharia, William Chan, Saurabh Saxena, Lala Li, Jay Whang, Emily Denton, Seyed Kamyar Seyed Ghasemipour, Burcu Karagol Ayan, S Sara Mahdavi, Rapha Gontijo Lopes, et al. Photorealistic text-to-image diffusion models with deep language understanding. *arXiv preprint arXiv:2205.11487*, 2022. 2, 3

[51] Aditya Sanghi, Hang Chu, Joseph G Lambourne, Ye Wang, Chin-Yi Cheng, Marco Fumero, and Kamal Rahimi Malekshan. Clip-forge: Towards zero-shot text-to-shape generation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 18603–18613, 2022. 2, 3

[52] Gopal Sharma, Bidya Dash, Aruni RoyChowdhury, Matheus Gadelha, Marios Loizou, L Cao, Rui Wang, EG Learned-Miller, Subhransu Maji, and Evangelos Kalogerakis. Prifit: Learning to fit primitives improves few shot point cloud segmentation. In *Computer Graphics Forum*, volume 41, pages 39–50. Wiley Online Library, 2022. 2

[53] Gopal Sharma, Kangxue Yin, Subhransu Maji, Evangelos Kalogerakis, Or Litany, and Sanja Fidler. Mvdecor: Multi-view dense correspondence learning for fine-grained 3d segmentation. *arXiv preprint arXiv:2208.08580*, 2022. 1, 2

[54] Chun-Yu Sun, Yu-Qi Yang, Hao-Xiang Guo, Peng-Shuai Wang, Xin Tong, Yang Liu, and Heung-Yeung Shum. Semi-supervised 3d shape segmentation with multilevel consistency and part substitution. *arXiv preprint arXiv:2204.08824*, 2022. 1, 2

[55] Ali Thabet, Humam Alwassel, and Bernard Ghanem. Mortonnet: Self-supervised learning of local features in 3d point clouds. *arXiv preprint arXiv:1904.00230*, 2019. 2

[56] Hugues Thomas, Charles R Qi, Jean-Emmanuel Deschaud, Beatriz Marcotegui, François Goulette, and Leonidas J Guibas. Kpconv: Flexible and deformable convolution for point clouds. In *Proceedings of the IEEE/CVF international conference on computer vision*, pages 6411–6420, 2019. 2

[57] Jesse Thomason, Mohit Shridhar, Yonatan Bisk, Chris Paxton, and Luke Zettlemoyer. Language grounding with 3d objects. In *Conference on Robot Learning*, pages 1691–1701. PMLR, 2022. 3

[58] Thang Vu, Kookhoi Kim, Tung M Luu, Thanh Nguyen, and Chang D Yoo. Softgroup for 3d instance segmentation on point clouds. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 2708–2717, 2022. 2, 6, 7

[59] Can Wang, Menglei Chai, Mingming He, Dongdong Chen, and Jing Liao. Clip-nerf: Text-and-image driven manipulation of neural radiance fields. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 3835–3844, 2022. 3

[60] Lingjing Wang, Xiang Li, and Yi Fang. Few-shot learning of part-specific probability space for 3d shape segmentation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 4504–4513, 2020. 1, 2

[61] Ruocheng Wang, Yunzhi Zhang, Jiayuan Mao, Ran Zhang, Chin-Yi Cheng, and Jiajun Wu. Ikea-manual: Seeing shape assembly step by step. In *Thirty-sixth Conference on Neural Information Processing Systems Datasets and Benchmarks Track*. 2

[62] Weiyue Wang, Ronald Yu, Qiangui Huang, and Ulrich Neumann. Sgpn: Similarity group proposal network for 3d point cloud instance segmentation. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 2569–2578, 2018. 2

[63] Xinlong Wang, Shu Liu, Xiaoyong Shen, Chunhua Shen, and Jiaya Jia. Associatively segmenting instances and semantics in point clouds. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 4096–4105, 2019. 2

[64] Xiaogang Wang, Xun Sun, Xinyu Cao, Kai Xu, and Bin Zhou. Learning fine-grained segmentation of 3d shapes without part labels. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 10276–10285, 2021. 2

[65] Yue Wang, Yongbin Sun, Ziwei Liu, Sanjay E Sarma, Michael M Bronstein, and Justin M Solomon. Dynamic graph cnn for learning on point clouds. *Acm Transactions On Graphics (tog)*, 38(5):1–12, 2019. 1, 2

[66] Xinyue Wei, Minghua Liu, Zhan Ling, and Hao Su. Approximate convex decomposition for 3d meshes with collision-aware concavity and tree search. *arXiv preprint arXiv:2205.02961*, 2022. 6

[67] Fanbo Xiang, Yuzhe Qin, Kaichun Mo, Yikuan Xia, Hao Zhu, Fangchen Liu, Minghua Liu, Hanxiao Jiang, Yifu Yuan, He Wang, et al. Sapien: A simulated part-based interactive environment. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 11097–11107, 2020. 1, 2, 6

[68] Xun Xu and Gim Hee Lee. Weakly supervised semantic point cloud segmentation: Towards 10x fewer labels. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 13706–13715, 2020. 2

[69] Xianghao Xu, Yifan Ruan, Srinath Sridhar, and Daniel Ritchie. Unsupervised kinematic motion detection for part-segmented 3d shape collections. In *ACM SIGGRAPH 2022 Conference Proceedings*, pages 1–9, 2022. 1

[70] Bo Yang, Jianan Wang, Ronald Clark, Qingyong Hu, Sen Wang, Andrew Markham, and Niki Trigoni. Learning object bounding boxes for 3d instance segmentation on point clouds. *Advances in neural information processing systems*, 32, 2019. 2

[71] Cheng-Kun Yang, Ji-Jia Wu, Kai-Syun Chen, Yung-Yu Chuang, and Yen-Yu Lin. An mil-derived transformer for weakly supervised point cloud segmentation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 11830–11839, 2022. 2

[72] Li Yi, Vladimir G Kim, Duygu Ceylan, I-Chao Shen, Mengyan Yan, Hao Su, Cewu Lu, Qixing Huang, Alla Sheffer, and Leonidas Guibas. A scalable active framework for region annotation in 3d shape collections. *ACM Transactions on Graphics (ToG)*, 35(6):1–12, 2016. 1

[73] Li Yi, Wang Zhao, He Wang, Minhyuk Sung, and Leonidas J Guibas. Gspn: Generative shape proposal network for 3d instance segmentation in point cloud. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 3947–3956, 2019. 1, 2

[74] Fenggen Yu, Kun Liu, Yan Zhang, Chenyang Zhu, and Kai Xu. Partnet: A recursive part decomposition network for fine-grained and hierarchical shape segmentation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 9491–9500, 2019. 2

[75] Biao Zhang and Peter Wonka. Point cloud instance segmentation using probabilistic embeddings. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 8883–8892, 2021. 2

[76] Haotian Zhang, Pengchuan Zhang, Xiaowei Hu, Yen-Chun Chen, Liunian Harold Li, Xiyang Dai, Lijuan Wang, Lu Yuan, Jenq-Neng Hwang, and Jianfeng Gao. Glipv2: Unifying localization and vision-language understanding. *arXiv preprint arXiv:2206.05836*, 2022. 2, 3, 8

[77] Renrui Zhang, Ziyu Guo, Wei Zhang, Kunchang Li, Xupeng Miao, Bin Cui, Yu Qiao, Peng Gao, and Hongsheng Li. Pointclip: Point cloud understanding by clip. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 8552–8562, 2022. 2, 3

[78] Yachao Zhang, Zonghao Li, Yuan Xie, Yanyun Qu, Cuihua Li, and Tao Mei. Weakly supervised semantic segmentation for large-scale point cloud. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 35, pages 3421–3429, 2021. 2

[79] Yachao Zhang, Yanyun Qu, Yuan Xie, Zonghao Li, Shanshan Zheng, and Cuihua Li. Perturbed self-distillation: Weakly supervised large-scale point cloud semantic segmentation. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 15520–15528, 2021. 2

[80] Na Zhao, Tat-Seng Chua, and Gim Hee Lee. Few-shot 3d point cloud semantic segmentation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 8873–8882, 2021. 1, 2, 6, 7

[81] Shuaifeng Zhi, Tristan Laidlow, Stefan Leutenegger, and Andrew J Davison. In-place scene labelling and understanding with implicit scene representation. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 15838–15847, 2021. 2