SNAP: Towards Segmenting Anything in Any Point Cloud

Aniket Gupta[1,*] Hanhui Wang[1,*] Charles Saunders[2] Aruni RoyChowdhury[2]
Hanumant Singh[1] Huaizu Jiang[1]

[1]Northeastern University [2]The Mathworks, Inc.

[*] Equal contribution

Abstract

Interactive 3D point cloud segmentation enables efficient annotation of complex 3D scenes through user-guided prompts. However, current approaches are typically restricted in scope to a single domain (indoor or outdoor), and to a single form of user interaction (either spatial clicks or textual prompts). Moreover, training on multiple datasets often leads to negative transfer, resulting in domain-specific tools that lack generalizability. To address these limitations, we present ***SNAP*** *(**S**egment a**N**ything in **A**ny **P**oint cloud), a unified model for interactive 3D segmentation that supports both point-based and text-based prompts across diverse domains. Our approach achieves cross-domain generalizability by training on 7 datasets spanning indoor, outdoor, and aerial environments, while employing domain-adaptive normalization to prevent negative transfer. For text-prompted segmentation, we automatically generate mask proposals without human intervention and match them against CLIP embeddings of textual queries, enabling both panoptic and open-vocabulary segmentation. Extensive experiments demonstrate that SNAP consistently delivers high-quality segmentation results. We achieve state-of-the-art performance on 8 out of 9 zero-shot benchmarks for spatial-prompted segmentation and demonstrate competitive results on all 5 text-prompted benchmarks. These results show that a unified model can match or exceed specialized domain-specific approaches, providing a practical tool for scalable 3D annotation. Project page is at https://neu-vi.github.io/SNAP/*

1. Introduction

Inspired by the success of SAM [1] for 2D images, we study interactive segmentation for 3D point clouds in this paper, allowing users to create high-quality annotations at scale with minimal effort. While supervised deep learning has fueled significant progress in visual learning, its reliance on vast labeled datasets presents a major bottleneck in the 3D domain, where manual annotation is notoriously difficult and time consuming.

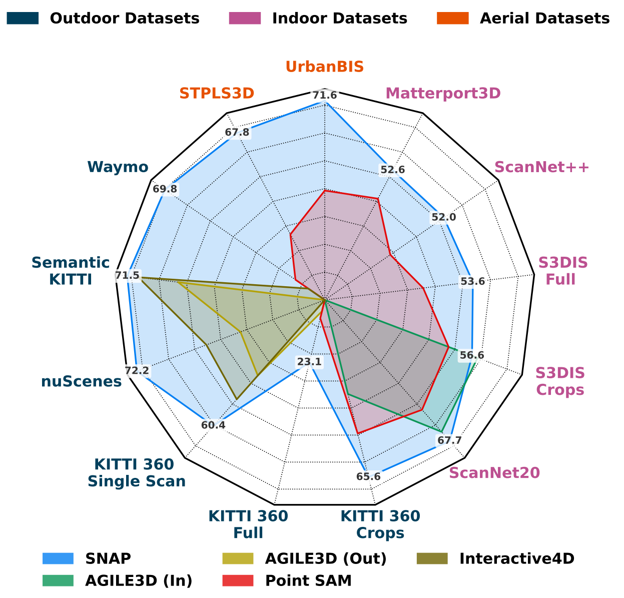

Figure 1. **Comparison of models on IoU@1 Click across multiple domains.** SNAP is a unified interactive point cloud segmentation model trained on multiple datasets spanning multiple domains. It generalizes robustly across a wide array of benchmarks.

Current interactive point cloud segmentation methods suffer from critical limitations that hinder their adoption as general-purpose annotation tools. Most existing approaches lack *generalizability*, being designed for specific domains such as indoor scenes [2–4] or outdoor environments [5]. This domain-specific design stems from the significant statistical differences between point cloud types: indoor scenes feature dense, structured environments with clear object boundaries, while outdoor scenes contain sparse, large scale structures with varying point densities. Training models across these diverse domains introduces negative transfer effects that degrades performance without careful architectural considerations [6]. Additionally, current methods lack *flexibility* in prompt modalities, typically supporting either spatial inputs (*e.g.*, clicks) [2–5] or text descriptions[7], but rarely both. This limitation restricts users to specific annotation workflows, preventing adaptation to different labeling needs and use cases.

To address these limitations, we present **SNAP** (Segment Anything in aNy Point cloud), a unified model that supports both spatial and text-based prompts To achieve cross-domain generalizability, we train on seven diverse datasets [8–14] spanning indoor, outdoor, and aerial domains, employing domain-wise normalization to mitigate negative transfer effects caused by statistical shifts between datasets [15]. For text-prompted segmentation, we first introduce a a simple iterative algorithm to automatically generate prompt points without human intervention. These prompt points are used via the spatial-prompted segmentation pipeline to generate mask proposals, which are then matches against CLIP [16] embeddings of textual prompts to run text-prompted segmentation. During inference, this approach supports both panoptic segmentation with predefined categories and open-vocabulary segmentation with novel classes.

Extensive experiments demonstrate that SNAP consistently predicts high-quality spatial masks and correct class labels across a diverse range of indoor [11, 17–19], outdoor [8–10, 20–22], and aerial [13, 14, 23, 24] point clouds. For point-prompted segmentation, SNAP sets a new state-of-the-art on 8 out of 9 zero-shot benchmarks. For text-prompted segmentation, SNAP shows competitive results on all 5 evaluated benchmarks. These results demonstrate that a single, unified model can match or exceed the performance of specialized, domain-specific approaches, shown in **Fig. 1**. In summary, our contributions are as follows:

- We introduce SNAP, a model for interactive point cloud segmentation that works across indoor, outdoor, and aerial domains.
- SNAP offers flexibility to use multi-modal prompts like points and text to segment objects of interest, and predicts classes as well as spatial masks.
- We introduce an automatic prompt points generation algorithm that bootstraps the panoptic labeling process without human intervention.
- SNAP achieves state-of-the-art performance across multiple datasets across various domains and is usable as an out-of-the-box semi-automated labeling tool.

2. Related Work

Interactive 3D segmentation with spatial prompts. Interactive segmentation has become well established in 2D since the introduction of SAM [1], but remains comparatively underexplored in 3D. Early works such as InterObject3D[2] and AGILE3D[3] focused on indoor point clouds, using positive/negative clicks for single or multi-object segmentation. Point-SAM[4] leveraged SAM-generated pseudo-labels to support indoor and part-level segmentation. Interactive4D [5] adopted a 4D setup on outdoor LiDAR sequences. However, these works are generally restricted in their usability to either indoor or outdoor scenes and show limited generalizability to out of domain point clouds. SNAP, in contrast, is designed to perform robustly across different domains.

Text-based 3D segmentation. Another line of research introduces natural language as a flexible interface for 3D segmentation. Within this area, *open-vocabulary segmentation* methods [25–28] aim to recognize novel or user-specified categories beyond a fixed label set. OpenScene [25] achieves this directly in the 3D domain by predicting per-point CLIP embeddings without relying on images. In contrast, [26–28] utilize either original RGB images or rendered multiview images to extract CLIP image features, which are aligned with text embeddings to enable text-based segmentation. A complementary direction is *panoptic segmentation*, where the goal is to provide instance-level predictions across all categories. [7, 29] exemplify this approach by predicting per-instance CLIP tokens and aligning them with a predefined class vocabulary to support class-specific instance segmentation. SNAP is able to accommodate both *open-vocabulary* and *panoptic* settings within a single framework.

Lifting 2D foundation models for 3D segmentation. Due to the limited availability of annotated 3D data, recent works like SAM3D [30] and SAMPro3D [31] focus on lifting robust 2D foundation models into the 3D domain. However, they require paired image and point cloud data, which limits their usability in real-world use cases with LiDAR only datasets or legacy datasets. [4, 7, 32] employs SAM [1] to generate pseudo-labels provided RGB images and use them to train 3D models on indoor scenes, with Point-SAM [4] further addressing part-level segmentation. While this process helps generate significant amounts of training data, this pseudo-labeling process invariably introduces label noise. SNAP avoids this problem altogether by pooling publicly available datasets for training and establishes state-of-the-art performance on several unseen datasets.

3. Method

In this section, we present the overall framework of SNAP, which is organized into 4 parts: (1) Point Cloud Encoding, (2) Spatial-prompted Segmentation, (3) Text-prompted Segmentation, and (4) Training. **Fig. 2** provides an overview of our approach.

3.1. Point Cloud Encoding

The input point cloud is represented by its XYZ coordinates, denoted as $\mathbf{P} = \{\mathbf{p}_i \in \mathbb{R}^3\}_{i=1}^N$, where N is the number of points. We use the Point Transformer V3 (PTv3 [33])

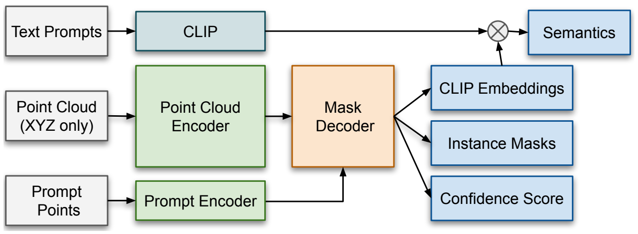

Figure 2. **Overview of SNAP.** SNAP encodes point clouds and prompts separately, then uses a Mask Decoder to generate segmentation masks. Text prompts are handled by matching CLIP embeddings with predicted mask embeddings for semantic classification.

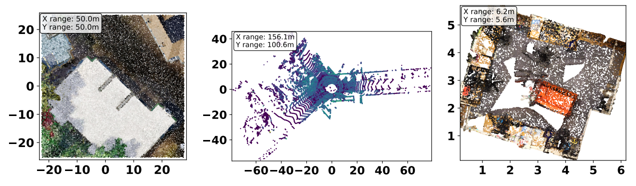

Figure 3. **Point clouds from different domains vary significantly in their properties. (a)** STPLS3D provides dense point clouds from an aerial view using RGB photogrammetry in the 50m range, **(b)** KITTI provides lidar data in the 150m range, **(c)** ScanNet provides point clouds in the 10m range.

to extract point-wise embeddings $\mathbf{F}_{\text{pc}} \in \mathbb{R}^{N \times D}$. To support cross-domain generalization, we replace the regular batch normalization in PTv3 with domain normalization.

Domain Normalization for Multi-Dataset Training. Training a *single* model on multiple point cloud datasets often leads to lower performance than *multiple per-dataset* models due to negative transfer caused by significant distributional differences between various datasets [6], as shown in **Fig. 3**. A straightforward approach to mitigate this is *dataset-specific* normalization [6], which learns unique normalization parameters for each dataset in the training set. While effective at addressing distributional differences among *known* datasets, this method presents practical challenges: users need to select appropriate normalization parameters at test time, which can be ambiguous when the source of a point cloud at test-time differs from the training datasets. Moreover, dataset-specific normalization may limit knowledge transfer between related datasets that could benefit from shared representations.

To address the limitation, we propose *domain-specific* normalization, which groups datasets into broader *domains* with similar statistical properties (*e.g.*, indoor, outdoor, or aerial) and learns a separate set of normalization parameters for each domain. This strategy allows our model to effectively adapt to different data distributions while maintaining the flexibility to be applied to new datasets by identifying their general domain, a more intuitive decision than selecting specific dataset parameters as shown in **Fig. 4**.

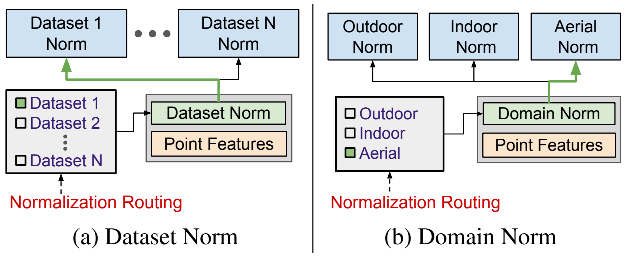

Figure 4. **Dataset Norm vs Domain Norm**. Domain-norm simplifies the overall architecture and improves zero-shot generalization.

Formally, we denote the set of domains as $\mathbb{D} = \{\mathcal{D}_j\}$. Domain normalization for the k-th layer's activations from the j-th domain, $\bar{\mathbf{x}}_j^k$, is expressed as:

$$\bar{\mathbf{x}}_j^k = \frac{\mathbf{x}_j^k - \mathbb{E}[\mathbf{x}_j^k]}{\sqrt{\text{Var}[\mathbf{x}_j^k] + \epsilon}} \cdot \gamma_j^k + \beta_j^k, \tag{1}$$

where γ_j^k and β_j^k are learned domain-specific scale and shift parameters. Compared with standard batch and dataset-specific normalizations, this domain-specific design allows the model to effectively adapt the normalization to the distinct statistics of each domain while sharing the core model weights across all domains.

3.2. Spatial-prompted Segmentation

Spatial Prompt Encoder. Given a spatial prompt point $\mathbf{p}_{\text{sp}} \in \mathbb{R}^3$, we identify its nearest neighbor in the point cloud: $i^* = \arg\min_i \|\mathbf{p}_{\text{sp}} - \mathbf{p}_i\|_2$, and retrieve the corresponding point embedding, $\mathbf{f}_{\text{sem}} = \mathbf{f}_{i^*}$, from the point-wise PTv3 embeddings $\mathbf{F}_{\text{pc}}$. We also compute positional encodings of the prompt point: $\mathbf{f}_{\text{pos}} = \phi(\mathbf{p}_{\text{sp}})$, where $\phi(\cdot)$ is the Fourier encoding function as in [34]. The final spatial prompt embedding combines semantic features and the positional encodings, which help preserve spatial information: $\mathbf{F}_{\text{sp}} = \Phi_{\text{sp}}([\mathbf{f}_{\text{sem}} \| \mathbf{f}_{\text{pos}}]) \in \mathbb{R}^D$ where $\Phi_{\text{sp}}(\cdot)$ is a learned projection function, and $[\cdot \| \cdot]$ is feature concatenation. Given P prompt points (clicks) on M objects, we get $\mathbf{F}_{\text{sp}} \in \mathbb{R}^{M \times P \times D}$.

Mask Decoder. The mask decoder module takes in two inputs: $\mathbf{F}_{\text{pc}} \in \mathbb{R}^{N \times D}$ for the point cloud embeddings, and $\mathbf{F}_{\text{sp}} \in \mathbb{R}^{M \times P \times D}$ for the prompt embeddings. Inspired by the design in SAM [1], we additionally introduce *three* task-specific learnable tokens per object. These tokens are responsible for predicting the final mask, its confidence score, and CLIP embeddings, respectively, resulting in $\mathbf{F}_{\text{sp}} \in \mathbb{R}^{M \times (P+3) \times D}$. Following the approach of SAM [1], we first tile the point embeddings to match the prompt dimension: $\tilde{\mathbf{F}}_{\text{pc}} \in \mathbb{R}^{M \times N \times D}$. The decoder then uses a series of transformer blocks to iteratively refine the embeddings. The process is designed to first incorporate contextual information from the point cloud into the prompt embeddings, and then use the refined prompts to condition the point cloud embeddings. Within each block, the updates occur in the following sequence:

- Prompt Self-Attention: $\mathbf{Z}_1 = \Psi_1(\mathbf{F}_{\text{sp}}, \mathbf{F}_{\text{sp}})$.
- Prompt-to-point Cross-Attention: $\mathbf{Z}_2 = \Psi_2(\mathbf{Z}_1, \tilde{\mathbf{F}}_{\text{pc}})$.
- Feedforward Network (FFN): $\mathbf{Z}_{\text{sp}} = \Phi(\mathbf{Z}_2)$.
- Point-to-Prompt Cross-Attention: $\mathbf{Z}_{\text{pc}} = \Psi_3(\tilde{\mathbf{F}}_{\text{pc}}, \mathbf{Z}_{\text{sp}})$.

Here, each $\Psi_{(\cdot)}$ denotes a multi-head attention block. Each of them takes two inputs, where the first denotes the query and the second indicates the key and value. Φ is a position-wise feedforward network. The final output of the entire decoder module is a set of refined prompt embeddings $\mathbf{Z}_{\text{sp}} \in \mathbb{R}^{M\times(P+3)\times D}$ and conditioned point cloud embeddings $\mathbf{Z}_{\text{pc}} \in \mathbb{R}^{M\times N\times D}$.

To generate predictions, we first separate the learnable token embeddings from the refined prompt embeddings $\mathbf{Z}_{\text{sp}}$. Recall that during prompt encoding, we appended three learnable tokens to each object's prompt sequence. We extract the following embeddings accordingly: mask token embeddings $\mathbf{Z}_{\text{mask}} \in \mathbb{R}^{M\times 1\times D}$, CLIP token embeddings $\mathbf{Z}_{\text{CLIP}} \in \mathbb{R}^{M\times 1\times D}$, mask confidence score token embeddings $\mathbf{Z}_{\text{S}} \in \mathbb{R}^{M\times 1\times D}$, and auxiliary mask token embeddings $\mathbf{Z}_{\text{aux}} \in \mathbb{R}^{M\times P\times D}$ and retain the original prompt point embeddings as auxiliary embeddings $\mathbf{Z}_{\text{aux}} \in \mathbb{R}^{M\times P\times D}$. The first three token embeddings are fed into dedicated prediction heads to generate mask, mask confidence scores, and CLIP embeddings, while the auxiliary embeddings are used for an additional supervision described in Sec. 3.4.

- **Mask Head.** We pass the mask token embeddings $\mathbf{Z}_{\text{mask}} \in \mathbb{R}^{M\times 1\times D}$ through a small MLP and then compute the dot product between the mask token embeddings and point cloud embeddings to get M segmentation logits $\mathbf{L}_{\text{mask}} = \mathbf{Z}_{\text{pc}} \cdot (\mathbf{F}_{\text{mask}})^\top \in \mathbb{R}^{M\times N}$. The final segmentation masks can then be obtained by applying a sigmoid function σ to the logits: $\mathcal{M}_{\text{mask}} = \sigma(\mathbf{L}_{\text{mask}})$.
- **Confidence Score Head.** Mask confidence score predictions can be similarly obtained by passing the mask score token embeddings through a MLP: $\mathbf{L}_{\text{S}} = \Phi_{\text{S}}(\mathbf{Z}_{\text{S}})$ and applying a sigmoid function σ: $\mathcal{S} = \sigma(\mathbf{L}_{\text{S}})$.
- **CLIP Embedding Head.** Motivated by SAL [7], we learn to predict CLIP embeddings for each mask using a MLP Φ_{CLIP}: $\mathbf{L}_{\text{CLIP}} = \Phi_{\text{CLIP}}(\mathbf{Z}_{\text{CLIP}})$. We show in **Tab. 6** that this also improves segmentation accuracy.

3.3. Text-prompted Segmentation

Our text-prompted segmentation pipeline consists of two stages: (1) automatic generation of prompt points to produce mask proposals via our spatial-prompted segmentation module, and (2) matching of predicted CLIP embeddings from each mask to the input textual prompt.

Automatic Prompt Points Generation. To comprehensively segment the input point cloud without manual prompts, we employ an iterative coarse-to-fine strategy. Starting with coarse voxelization, we use voxel centers as prompt points to generate initial mask proposals. We then identify unsegmented regions and generate new prompts using progressively smaller voxel sizes in the unsegmented regions, continuing for a fixed number of iterations. Finally, non-maximum suppression removes redundant overlapping masks. This approach ensures comprehensive coverage while avoiding redundant computation in well-segmented regions, striking a better balance between accuracy and efficiency compared to uniform grid sampling as in the 2D counterpart of SAM [1]. See § 7.3 for more details.

Text Prompt Encoder and Matching. To classify the generated mask proposals, we encode input text prompts using the CLIP text encoder [16]. During training, we use category names as text prompts to train SNAP. Specifically, we wrap category names in full sentences (*e.g.*, `a photo of a class_name`). At inference, the generated mask proposals are matched to text queries by comparing their predicted CLIP embeddings with encoded text prompts. This enables both panoptic segmentation using predefined vocabularies and open-vocabulary segmentation for novel object classes.

3.4. Training

Click Sampling Strategy. Following AGILE3D [3] and Interactive4D [5], we use an iterative click sampling strategy to simulate user clicks. Unlike their computationally intensive process of calculating and then ranking error regions for additional clicks, we adopt a simpler approach by randomly sampling clicks from the unsegmented regions of each object.

Training Losses. In line with prior works [1, 3, 5], we supervise our mask predictions using the ground truth annotated instance segmentation labels using Focal [35] and Dice [36] loss. Following [5], we also incorporate weights on these losses to modulate them based on proximity to the user clicks. Additionally, to encourage each click to independently yield a plausible mask prediction, we use an auxiliary mask loss ($\mathcal{L}_{\text{aux}}$), which adopts the auxiliary mask token embeddings (as defined in Sec. 3.2) to generate additional mask predictions. To improve the reliability of mask confidence score estimation, we also introduce a score prediction loss ($\mathcal{L}_{\text{score}}$). Specifically, the target score for each mask is defined as the Intersection-over-Union (IoU) between the predicted mask and its corresponding ground-truth mask. To supervise the predicted class labels, we use $\mathcal{L}_{\text{text}}$ to penalize incorrect alignment between the CLIP embeddings of predicted text tokens for each mask and the class vocabulary, using a cosine distance loss. Our overall loss function can be written as:

$$\mathcal{L}_{\text{SNAP}} = \mathcal{L}_{\text{focal}} + \mathcal{L}_{\text{dice}} + \mathcal{L}_{\text{aux}} + \mathcal{L}_{\text{score}} + \mathcal{L}_{\text{text}}. \quad (2)$$

See § 7.2 for additional details of each loss term.

Table 1. **In-distribution interactive point cloud segmentation.** SNAP-*dataset* refers to the model trained exclusively on the respective dataset. † denotes the method is evaluated by us.

	IoU@k ↑				
Method	@1	@2	@3	@5	@10
Trained and evaluated on SemanticKITTI					
AGILE3D [3]	53.1	63.7	70	76.7	83.3
Interactive4D [5]	67.5	73.9	78.3	83.4	88.2
SNAP - KITTI	68.1	75.9	80.1	84.5	88.7
SNAP - C	**71.5**	**78.1**	**81.0**	**86.0**	**90.1**
Trained and evaluated on ScanNet20					
InterObject3D [2]	40.8	55.9	63.9	72.4	79.9
AGILE3D [3]	63.3	70.9	75.4	79.9	83.7
Point-SAM† [4]	52.7	69.6	75.9	80.6	83.3
SNAP - SN	**68.6**	74.2	78.4	82.1	84.6
SNAP - C	67.7	**74.7**	**78.5**	**82.3**	**85.5**

4. Experiments

4.1. Experimental Setup

Datasets. We train SNAP on 7 diverse datasets with ground-truth instance segmentation labels, which span three domains: (i) *indoor* scenes from ScanNet [11] and HM3D [12, 37]; (ii) *outdoor* driving sequences from SemanticKITTI [8, 20], nuScenes [9], and Pandaset [10]; and (iii) *aerial* point clouds from STPLS3D [13] and DALES [14, 23]. We evaluate zero-shot performance on held-out datasets from each domain: S3DIS [19] (full and crops), ScanNet++ [17], and Matterport3D [18] for indoor; Waymo [22] and KITTI-360 [21] (full, crops, and single scan) for outdoor; and UrbanBIS [24] for aerial. See § 8.2.

Evaluation Metrics. Following conventions from [2–5], we evaluate spatial-prompted segmentation using IoU@k, the average intersection over union (IoU) achieved with k clicks per object across all objects. To assess object category prediction capabilities, we use the mean Average Precision (mAP) metric following [25–27] and for panoptic segmentation, we use panoptic quality (PQ), segmentation quality (SQ), and recognition quality (RQ) as done in [7].

Model Variants. SNAP operates on XYZ coordinates only, without relying on any additional per-point attributes (*e.g.*, color, normals, intensity), as such features are not consistently available across datasets. As shown in Sec. 4.4, using only XYZ coordinates achieves performance comparable to models that leverage all available modalities, while avoiding the dependency on dataset-specific properties. We evaluate 6 SNAP variants to provide comprehensive comparisons. First, we train dataset-specific models: SNAP-KITTI on SemanticKITTI [8] and SNAP-SN on ScanNet [11]. Second, we train domain-specific models that leverage all available in-domain datasets: SNAP-Indoor, SNAP-Outdoor, and SNAP-Aerial. Finally, SNAP-C represents our complete model trained across all datasets, serving as our most generalizable variant. In Sec. 4.3, SNAP-C@auto refers to the automatic prompt points generation setting, in which the model is provided solely with the point cloud to enable a fair comparison with baseline methods.

4.2. Spatial-prompted Interactive Segmentation

To fairly evaluate SNAP against other purely interactive segmentation baselines like [3–5], we first evaluate the predicted instance mask quality without taking the class-prediction into account (class-agnostic), as typically done in prior methods on interactive point cloud segmentation.

In-distribution Evaluation. In **Tab. 1** we compare SNAP's performance on the SemanticKITTI [8] and ScanNet [11] datasets, where SNAP outperforms all prior baselines on both datasets. Notably, while AGILE3D [3] needs to be trained separately for both datasets, SNAP is evaluated with the same set of parameters for both datasets. This versatility makes SNAP a more practical segmentation tool, which can thus be evaluated on a broader set of datasets.

Zero-Shot Evaluation. To test the generalization capabilities of the SNAP, we evaluate on 9 *unseen* benchmarks covering indoor, outdoor, and aerial domains. As shown in **Tab. 2**, SNAP-C outperforms the baseline AGILE3D [3], Point-SAM [4] and Interactive4D [5] on 8 out of 9 benchmarks. Particularly in the 1-Click experiments, SNAP-C provides a **20.6%** average improvement across all benchmarks with a single set of parameters. This positions SNAP-C as a practical, general-purpose tool for interactive segmentation, addressing the key limitation of existing domain-specific approaches.

4.3. Text-prompted Segmentation

For text-prompted segmentation, we evaluate SNAP's effectiveness for panoptic segmentation and open-vocabulary segmentation. Our primary evaluation targets SNAP-C@auto; we additionally report SNAP-C@1 Click as an upper-bound reference, with the click sampled from the *ground-truth* masks to simulate ideal user input. SNAP-auto takes only the point cloud as input and outputs panoptic segmentation masks like the baselines we compare against.

Panoptic Segmentation. SAL [7] pioneered automated panoptic segmentation on outdoor datasets, making it our primary baseline for outdoor scenes. As shown in **Tab. 3**, SNAP-C@auto, which has the equivalent setting to SAL with only point clouds as input, performs robustly against it and improves PQ score by 3.4 points on SemanticKITTI while remaining comparable on nuScenes (37.9 PQ vs 38.4 from SAL). With single-click guidance, SNAP-C@1 Click achieves even better results on both datasets.

Table 2. **Zero-shot interactive point cloud segmentation**. We compare SNAP with the current methods for interactive segmentation across different domains on several unseen datasets. † denotes that the methods are evaluated by us.

	Dataset	Method	IoU@k ↑		
			@1	**@3**	**@5**
Indoor	ScanNet++	Point-SAM† [4]	28.6	56.3	62.9
		SNAP - C	**52.0**	**67.3**	**73.2**
		Δ	*+23.4*	*+11.0*	*+10.3*
	Matterport3D	Point-SAM† [4]	41.1	67.2	73.7
		SNAP - C	**52.6**	**69.6**	**75.2**
		Δ	*+11.5*	*+2.4*	*+1.5*
	S3DIS (crops)	AGILE3D [3]	**58.7**	**77.4**	83.6
		Point-SAM† [4]	45.9	77.6	**84.6**
		SNAP - C	56.6	73.8	80.9
		Δ	*-2.1*	*-3.6*	*-3.7*
	S3DIS (full)	Point-SAM† [4]	35.6	68.0	76.3
		SNAP - C	**53.6**	**71.1**	**77.6**
		Δ	*+18.0*	*+3.1*	*+1.3*
Outdoor	Waymo	Point-SAM† [4]	12.8	43.0	53.1
		Interactive 4D† [5]	7.2	7.3	7.5
		SNAP - C	**69.8**	**82.3**	**86.6**
		Δ	*+57.0*	*+39.3*	*+33.5*
	KITTI-360 (full)	Point-SAM† [4]	6.8	22.7	28.1
		SNAP - C	**23.1**	**40.1**	**48.1**
		Δ	*+16.3*	*+17.4*	*+20.0*
	KITTI-360 Single Scan	AGILE3D [3]	36.3	47.3	53.5
		Interactive 4D [5]	47.7	59.4	64.1
		SNAP - C	**60.4**	**64.6**	**67.7**
		Δ	*+12.7*	*+5.2*	*+3.6*
	KITTI-360 (crops)	AGILE3D [3]	34.8	42.7	44.4
		Point-SAM [4]	49.4	74.4	**81.7**
		SNAP - C	**65.6**	**76.1**	80.0
		Δ	*+16.2*	*+1.7*	*-1.7*
Aerial	UrbanBIS	Point-SAM† [4]	39.3	79.1	89.4
		SNAP - C	**71.6**	**86.2**	**90.2**
		Δ	*+32.3*	*+7.1*	*+0.8*

Open-Vocabulary Segmentation. There exist limited number of approaches for direct alignment between 3D point clouds and CLIP [16] text embeddings without intermediate image representations, we therefore evaluate against three non-interactive instance segmentation baselines. First, OpenScene (3D Distill) [25] represents the most comparable approach to ours as it only uses the predicted per point CLIP embeddings during inference without requiring images. Second, OpenIns3D [26] generates class-agnostic instance masks and render multiview images of the input point cloud for CLIP [16] based segmentation. Third, OpenMask3D [27] and SAI3D [28] leverage the original 2D images from the dataset to perform open-vocabulary instance segmentation.

Table 3. **Panoptic segmentation on outdoor Lidar datasets.** We compare the SNAP-C @ auto variant with SAL [7] for panoptic segmentation on outdoor lidar datasets. We also show SNAP-C @ 1 Click results as a reference that represents the upper-bound for SNAP-C @ auto. Note that SNAP-C @ auto represents the equivalent setting to SAL [7] since the only input is raw point clouds.

Method	PQ	RQ	SQ	PQ_{Th}	PQ_{St}
Evaluation → SemanticKITTI					
SAL [7]	24.8	32.3	66.8	17.4	30.2
SNAP - C @ auto	**28.2**	**34.1**	**78.6**	**20.7**	**34.4**
SNAP - C @ 1 Click	40.1	47.3	81.6	30.7	46.9
Evaluation → nuScenes					
SAL [7]	**38.4**	**47.8**	77.2	**47.5**	29.2
SNAP - C @ auto	37.9	47.1	**82.2**	33.1	**52.4**
SNAP - C @ 1 Click	47.2	56.1	83.9	40.3	56.2

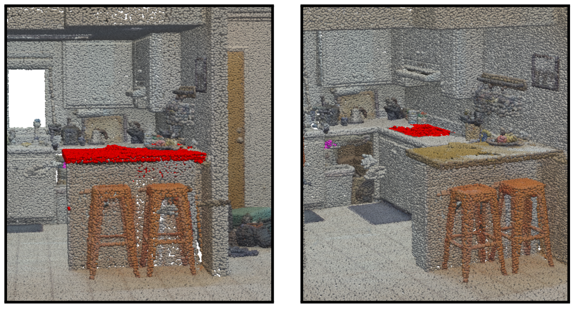

Figure 5. **Qualitative segmentation results of open-set scene understanding on the ScanNet++ Dataset.** Given a text prompt in the format of "*Segment* {*open-set vocabulary*}", our SNAP model finds the corresponding masks 🟥 in the scenes.

Results are summarized in **Tab. 4**, where baseline methods are trained on ScanNet200 [11]. SNAP-C demonstrates strong performance across 3 zero-shot datasets without using image embeddings. On Matterport3D [18], SNAP-C@auto achieves 16.5 AP, substantially outperforming baselines which rely on images and CLIP image embeddings. This is largely attributed to the large overlap between semantic classes of Matterport3D [18] and ScanNet200 [11]. On Replica [38] and S3DIS [19], SNAP-C@auto shows competitive results against baselines that have access to visual information and surpasses the image-free OpenScene(3D Distill) [25]. We also include qualitative results on the ScanNet++ [17] dataset in **Fig. 5**.

In summary, unlike methods that need RGB images or pre-computed CLIP embeddings, SNAP-C@auto performs open-vocabulary segmentation directly on point clouds and achieves competitive results against image-based methods. The image-free approach also offers practical advantages for handling LiDAR-only datasets, synthetic data, or legacy datasets where corresponding images were never collected.

Table 4. **Open-Vocabulary Segmentation**. We evaluate the SNAP-C @ auto variant against image-based and image-free open vocabulary segmentation models. We also show SNAP-C @ 1 Click results as a reference that represents the upper-bound for SNAP-C @ auto. Note that SNAP-C @ auto represents the equivalent setting to baseline methods since the only input is raw point clouds. [†]Uses RGB images, [‡]Uses CLIP image embeddings from rendered point cloud views.

Model	Uses Img[†]	CLIP IE.[‡]	AP	AP_{50}	AP_{25}
Zero Shot - Matterport3D					
OpenMask3D [27]	✓	✓	7.7	13.9	20.3
SAI3D [28]	✓	✓	8.9	15.3	20.9
SNAP - C @ auto	✗	✗	**16.5**	**25.2**	**30.7**
SNAP - C @ 1 click	✗	✗	18.3	28.2	34.7
Zero Shot - Replica					
OpenMask3D [27]	✓	✓	13.1	18.4	24.2
OpenIns3D [26]	✗	✓	13.6	18.0	19.7
OpenScene(3D Distill) [25]	✗	✗	8.2	10.5	12.6
SNAP - C @ auto	✗	✗	**10.1**	**11.7**	**13.8**
SNAP - C @ 1 click	✗	✗	11.7	14.5	15.4
Zero Shot - S3DIS					
OpenIns3D [26]	✗	✓	21.1	28.3	29.5
OpenScene(3D Distill) [25]	✗	✗	15.2	21.5	23.7
SNAP - C @ auto	✗	✗	**16.1**	**23.8**	**30.9**
SNAP - C @ 1 click	✗	✗	17.6	25.5	33.6

Table 5. **Input Properties Ablations.** X - XYZ coordinates, C - Color, N- Normals, S-Strength / Lidar intensity. - marks property data not available.

Dataset	X	C	N	S	IoU@1	IoU@5	IoU@10
SemanticKITTI	✓	-	-	✓	68.2	84.5	88.7
SemanticKITTI	✓	-	-	✗	68.1	84.5	88.7
ScanNet20	✓	✓	✓	-	69.6	83.0	85.5
ScanNet20	✓	✓	✗	-	68.8	82.1	84.6
ScanNet20	✓	✗	✓	-	69.3	83.0	85.5
ScanNet20	✓	✗	✗	-	68.6	82.1	84.6
STPLS3D	✓	✓	-	-	67.9	78.9	83.9
STPLS3D	✓	✗	-	-	67.2	78.9	83.9

4.4. Ablation Studies

Effect of Input Properties. We conducted an ablation study to evaluate the contribution of different input properties to our model's performance, with results summarized in **Tab. 5**. The experiments reveal that SNAP relies predominantly on geometric features. On the ScanNet20 dataset, the inclusion of surface normals (N) consistently improves performance across all metrics, boosting IoU@1 from 67.4 to 68.3. In contrast, color information (C) appears to be largely redundant; its inclusion provides no discernible benefit when normals are present and only a marginal gain otherwise. This trend is consistent across other datasets, where Lidar intensity (S) on SemanticKITTI and color on STPLS3D offer only minor improvements to IoU@1. While surface normals improve performance, computing accurate normals is challenging for real-world point clouds. Therefore, SNAP operates solely on XYZ coordinates to ensure applicability across diverse data sources.

Table 6. **Loss Function Ablations.** We report IoU@k with different combinations of Focal, Dice, Auxiliary, Weighted, and Text losses on the ScanNet20 dataset.

Component					IoU@k ↑		
Dice	Focal	Aux	Weighted	Text	@1	@5	@10
✓	✓				66.7	80.5	82.4
✓	✓	✓			67.3	81.0	83.2
✓	✓	✓	✓		67.5	81.7	83.7
✓	✓	✓	✓	✓	**68.6**	**82.1**	**84.6**

Effect of Loss functions on Mask Accuracy. SNAP uses a combination of loss functions to guide the network. To check the effectiveness of each module, we run ablations by adding each component sequentially. The results are summarized in **Tab. 6**. Results indicate that auxiliary loss and weighted loss improve performance at a higher number of clicks. Including text classification loss leads to the highest improvements and indicates that semantic understanding is important for segmentation tasks.

Effect of Normalization Strategies. We evaluate 3 normalization strategies to validate our domain-based design choice: (1) a single model with standard batch normalization across all data, (2) dataset-specific normalization with individual layers for each training dataset, and (3) our proposed domain normalization with separate layers for each domain. As shown in **Tab. 7**, batch normalization shows reasonable in-distribution performance (64.3 IoU@1 on SemanticKITTI) but tends to underperform on zero-shot datasets. Dataset normalization can achieve slightly higher scores on in-distribution sets , but this comes at the cost of limited generalization. Domain-specific normalization performs the best in zero-shot generalization, suggesting potential benefits to domain-level grouping over dataset-specific parameters. More importantly, this simple domain-norm translates into a simple domain-type checkbox selection, making it highly user-friendly.

Effect of Progressively Adding Datasets. To verify the effectiveness of scaling up training data, we evaluate various SNAP variants on in-distribution and unseen datasets. As shown in **Tab. 8**, scaling up the training set leads to consistent improvements in performance over both in-distribution as well as unseen datasets. SNAP-C consistently matches or surpasses the performances of single dataset or single do-

Table 7. **Domain Normalization Ablation.** We compare 3 normalization strategies for training a multi-dataset model: Batch Norm, Dataset Norm, and Domain Norm. The applied normalization for Dataset Norm and Domain Norm is provided in *teal* beneath each result.

Model	IoU@k ↑											
	In-Distribution						Zero-Shot					
	SemanticKITTI		ScanNet20		STPLS3D		Waymo		ScanNet++		UrbanBIS	
	@1	@5	@1	@5	@1	@5	@1	@5	@1	@5	@1	@5
Batch Norm	64.3	83.9	63.5	81.2	54.2	68.4	52.1	64.3	38.1	56.2	31.3	61.1
Dataset Norm	**72.0**	**86.1**	**69.5**	**83.8**	**67.9**	79.0	56.2	76.3	48.5	67.3	61.8	77.8
Norm used	*KITTI*		*ScanNet*		*STPLS3D*		*KITTI*		*ScanNet*		*STPLS3D*	
Domain Norm	71.5	86.0	67.7	82.3	67.8	**80.4**	**69.8**	**86.6**	**52**	**73.2**	**71.6**	**90.2**
Norm used	*Outdoor*		*Indoor*		*Aerial*		*Outdoor*		*Indoor*		*Aerial*	

Table 8. **Effect of Adding Datasets.** We progressively evaluate baseline models trained on single datasets (*SNAP-SN* or *SNAP-KITTI*) and baseline models trained on specific domains *SNAP-Indoor*, *SNAP-Outdoor*, and *SNAP-Aerial* against *SNAP-C*, the complete model trained on all data. The gray-shaded cells denote evaluations performed on out-of-distribution datasets.

Model	IoU@k ↑															
	In-Distribution						Zero-Shot									
	SemanticKITTI		ScanNet20		STPLS3D		Matterport3D		S3DIS Full		KITTI-360 Full		Waymo		UrbanBIS	
	@1	@5	@1	@5	@1	@5	@1	@5	@1	@5	@1	@5	@1	@5	@1	@5
SNAP - SN	3.1	5.9	**68.6**	82.1	2.5	4.1	**53.4**	71.3	51.4	70.0	0.2	0.5	0.7	3.1	6.3	16.8
SNAP - KITTI	68.1	84.5	13.8	33.6	24.6	52.9	16	33.3	11.6	29.1	6.7	29.7	48.2	66.3	43.1	71.3
SNAP - Indoor	3.1	15.3	66.0	81.3	5.9	18.6	49.9	74.2	51.9	76.9	0.2	2.1	1.1	6.5	3.4	33.6
SNAP - Outdoor	71.3	85.7	14.9	43.5	36.4	57.9	15.1	39.2	14.9	43.6	18.3	44.6	68.5	86.0	45.9	78
SNAP - Aerial	27.1	57.9	6.3	19.1	65.8	79.1	9.4	21.2	7.3	20.6	5.6	19.4	25.1	60.4	**74.2**	86.9
SNAP - C	**71.5**	**86.0**	67.7	**82.3**	**67.8**	**80.4**	52.6	**75.2**	**53.6**	**77.6**	**23.1**	**48.1**	**69.8**	**86.6**	71.6	**90.2**

Table 9. **Comparison of Automatic Prompt Points Generation Strategies.** We evaluate our iterative prompting approach against four baseline methods on ScanNet200. Best results are highlighted in **bold**, and second best results are underlined.

Approach	AP	AP_{50}	AP_{25}	Head mAP	Common mAP	Tail mAP	Time (s)
Uniform grid	**39.1**	**58.5**	68.8	**39.8**	**40.6**	36.3	1421
FPS	36.2	54.5	65.0	38.6	38.4	30.4	702
HDBSCAN	4.7	6.5	7.2	6.5	5.4	1.7	714
Ours	38.7	58.4	**69.2**	38.2	39.7	**38.1**	**461**

main models, suggesting that the proposed domain normalization is largely able to reduce the effects of any negative transfer from cross-domain datasets.

Design Choices for Automatic Prompt Points Generation. To generate prompt points automatically, we initially considered uniform grid sampling following SAM [1]. However, this approach faces a fundamental trade-off: large voxel sizes miss small objects while small voxels yield computationally prohibitive numbers of points. Alternative approaches like Farthest Point Sampling and HDBSCAN [39] require scene-specific tuning and generally undersample dense regions while oversampling sparse regions. Our iterative approach addresses these limitations and maintains uniform sampling independent of local point density. As shown in **Tab. 9**, this strategy achieves comparable coverage to naive grid sampling (39.1 AP vs 38.7 AP) with significantly fewer points, reducing computation time by **68%**. Notably, our approach excels on tail classes (38.1 mAP), outperforming all baselines.

5. Conclusion

In this paper, we introduced SNAP, a unified model for flexible promptable point cloud segmentation that is compatible with both spatial and textual prompts. By training on heterogeneous datasets with cross-domain normalization SNAP demonstrates state-of-the-art performance across a wide range of datasets. We believe that this unified model will come across as a handy tool for users, moving away from the need for dataset or domain specific models. For future work, following the promising trend found in this paper, investigating the effect of scaling up further via self-supervised or weakly-supervised learning on unlabeled data is an appealing direction.

6. Acknowledgements

This work was partially supported by the National Science Foundation under Award IIS-2310254.

References

[1] A. Kirillov, E. Mintun, N. Ravi, H. Mao, C. Rolland, L. Gustafson, T. Xiao, S. Whitehead, A. C. Berg, W.-Y. Lo *et al.*, “Segment anything,” in *Proceedings of the IEEE/CVF International Conference on Computer Vision*, 2023. 1, 2, 3, 4, 8

[2] T. Kontogianni, E. Celikkan, S. Tang, and K. Schindler, “Interactive object segmentation in 3d point clouds,” in *2023 IEEE International Conference on Robotics and Automation (ICRA)*. IEEE, 2023, pp. 2891–2897. 1, 2, 5, 12

[3] Y. Yue, S. Mahadevan, J. Schult, F. Engelmann, B. Leibe, K. Schindler, and T. Kontogianni, “Agile3d: Attention guided interactive multi-object 3d segmentation,” *arXiv:2306.00977*, 2024. [Online]. Available: https://arxiv.org/abs/2306.00977 2, 4, 5, 6, 3, 12

[4] Y. Zhou, J. Gu, T. Y. Chiang, F. Xiang, and H. Su, “Point-sam: Promptable 3d segmentation model for point clouds,” *arXiv:2406.17741*, 2024. [Online]. Available: https://arxiv.org/abs/2406.17741 1, 2, 5, 6, 3, 12

[5] I. Fradlin, I. E. Zulfikar, K. Yilmaz, T. Kontogianni, and B. Leibe, “Interactive4d: Interactive 4d lidar segmentation,” *arXiv:2410.08206*, 2024. [Online]. Available: https://arxiv.org/abs/2410.08206 1, 2, 4, 5, 6, 12

[6] X. Wu, Z. Tian, and et al., “Towards large-scale 3d representation learning with multi-dataset point prompt training,” in *CVPR*, 2024. 1, 3

[7] A. Ošep, T. Meinhardt, F. Ferroni, N. Peri, D. Ramanan, and L. Leal-Taixé, “Better call sal: Towards learning to segment anything in lidar,” *arXiv:2403.13129*, 2024. [Online]. Available: https://arxiv.org/abs/2403.13129 2, 4, 5, 6

[8] J. Behley, M. Garbade, A. Milioto, J. Quenzel, S. Behnke, C. Stachniss, and J. Gall, “SemanticKITTI: A Dataset for Semantic Scene Understanding of LiDAR Sequences,” in *Proc. of the IEEE/CVF International Conf. on Computer Vision (ICCV)*, 2019. 2, 5, 3, 6

[9] W. K. Fong, R. Mohan, J. V. Hurtado, L. Zhou, H. Caesar, O. Beijbom, and A. Valada, “Panoptic nuscenes: A large-scale benchmark for lidar panoptic segmentation and tracking,” *arXiv preprint arXiv:2109.03805*, 2021. 5, 3

[10] P. Xiao, Z. Shao, S. Hao, Z. Zhang, X. Chai, J. Jiao, Z. Li, J. Wu, K. Sun, K. Jiang, Y. Wang, and D. Yang, “Pandaset: Advanced sensor suite dataset for autonomous driving,” 2021. [Online]. Available: https://arxiv.org/abs/2112.12610 2, 5, 3

[11] A. Dai, A. X. Chang, M. Savva, M. Halber, T. Funkhouser, and M. Nießner, “Scannet: Richly-annotated 3d reconstructions of indoor scenes,” in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2017, pp. 5828–5839. 2, 5, 6, 3

[12] S. K. Ramakrishnan, A. Gokaslan, E. Wijmans, O. Maksymets, A. Clegg, J. M. Turner, E. Undersander, W. Galuba, A. Westbury, A. X. Chang, M. Savva, Y. Zhao, and D. Batra, “Habitat-matterport 3d dataset (HM3d): 1000 large-scale 3d environments for embodied AI,” in *Thirty-fifth Conference on Neural Information Processing Systems Datasets and Benchmarks Track*, 2021. [Online]. Available: https://arxiv.org/abs/2109.08238 5, 3

[13] M. Chen, Q. Hu, Z. Yu, H. THOMAS, A. Feng, Y. Hou, K. McCullough, F. Ren, and L. Soibelman, “Stpls3d: A large-scale synthetic and real aerial photogrammetry 3d point cloud dataset,” in *33rd British Machine Vision Conference 2022, BMVC 2022, London, UK, November 21-24, 2022*. BMVA Press, 2022. [Online]. Available: https://bmvc2022.mpi-inf.mpg.de/0429.pdf 2, 5, 4

[14] N. M. Singer and V. K. Asari, “Dales objects: A large scale benchmark dataset for instance segmentation in aerial lidar,” *IEEE Access*, pp. 1–1, 2021. 2, 5, 4

[15] X. Wu, Z. Tian, X. Wen, B. Peng, X. Liu, K. Yu, and H. Zhao, “Towards large-scale 3d representation learning with multi-dataset point prompt training,” in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2024, pp. 19 551–19 562. 2, 4

[16] A. Radford, J. W. Kim, C. Hallacy, A. Ramesh, G. Goh, S. Agarwal, G. Sastry, A. Askell, P. Mishkin, J. Clark, G. Krueger, and I. Sutskever, “Learning transferable visual models from natural language supervision,” 2021. [Online]. Available: https://arxiv.org/abs/2103.00020 2, 4, 6

[17] C. Yeshwanth, Y.-C. Liu, M. Nießner, and A. Dai, “Scannet++: A high-fidelity dataset of 3d indoor scenes,” in *Proceedings of the International Conference on Computer Vision (ICCV)*, 2023. 2, 5, 6, 3

[18] A. Chang, A. Dai, T. Funkhouser, M. Halber, M. Niessner, M. Savva, S. Song, A. Zeng, and Y. Zhang, “Matterport3d: Learning from rgb-d data in indoor environments,” *International Conference on 3D Vision (3DV)*, 2017. 5, 6, 3

[19] Stanford Doerr School of Sustainability, “Stanford Large-Scale 3D Indoor Spaces Dataset (S3DIS),” Jun. 2024. [Online]. Available: https://redivis.com/datasets/9q3m-9w5pa1a2h?v=1.0 2, 5, 6, 3

[20] A. Geiger, P. Lenz, and R. Urtasun, “Are we ready for Autonomous Driving? The KITTI Vision Benchmark Suite,” in *Proc. of the IEEE Conf. on Computer Vision and Pattern Recognition (CVPR)*, 2012, pp. 3354–3361. 2, 5, 3

[21] Y. Liao, J. Xie, and A. Geiger, “KITTI-360: A novel dataset and benchmarks for urban scene understanding in 2d and 3d,” *arXiv preprint arXiv:2109.13410*, 2021. 5, 3, 4

[22] P. Sun, H. Kretzschmar, X. Dotiwalla, A. Chouard, V. Patnaik, P. Tsui, J. Guo, Y. Zhou, Y. Chai, B. Caine, V. Vasudevan, W. Han, J. Ngiam, H. Zhao, A. Timofeev, S. Ettinger, M. Krivokon, A. Gao, A. Joshi, Y. Zhang, J. Shlens, Z. Chen, and D. Anguelov, “Scalability in perception for autonomous driving: Waymo open dataset,” in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2020. 2, 5, 4

[23] N. Varney, V. K. Asari, and Q. Graehling, “Dales: A large-scale aerial lidar data set for semantic segmentation,” in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition Workshops*, 2020, pp. 186–187. 2, 5

[24] G. Yang, F. Xue, Q. Zhang, K. Xie, C.-W. Fu, and H. Huang, “Urbanbis: a large-scale benchmark for fine-grained urban building instance segmentation,” in *ACM SIGGRAPH*, 2023, pp. 16:1–16:11. 2, 5, 4, 6, 11

[25] S. Peng, K. Genova, C. M. Jiang, A. Tagliasacchi, M. Pollefeys, and T. Funkhouser, "Openscene: 3d scene understanding with open vocabularies," in *CVPR*, 2023. 2, 5, 6, 7

[26] Z. Huang, X. Wu, X. Chen, H. Zhao, L. Zhu, and J. Lasenby, "Openins3d: Snap and lookup for 3d open-vocabulary instance segmentation," 2024. [Online]. Available: https://arxiv.org/abs/2309.00616 2, 6, 7

[27] A. Takmaz, E. Fedele, R. W. Sumner, M. Pollefeys, F. Tombari, and F. Engelmann, "Openmask3d: Open-vocabulary 3d instance segmentation," *arXiv preprint arXiv:2306.13631*, 2023. 5, 6, 7

[28] Y. Yin, Y. Liu, Y. Xiao, D. Cohen-Or, J. Huang, and B. Chen, "Sai3d: Segment any instance in 3d scenes," 2024. [Online]. Available: https://arxiv.org/abs/2312.11557 2, 6, 7

[29] W. Roh, H. Jung, G. Nam, J. Yeom, H. Park, S. H. Yoon, and S. Kim, "Edge-aware 3d instance segmentation network with intelligent semantic prior," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2024, pp. 20 644–20 653. 2, 3, 5, 11

[30] Y. Yang, X. Wu, T. He, H. Zhao, and X. Liu, "Sam3d: Segment anything in 3d scenes," *arXiv:2306.03908*, 2023. [Online]. Available: https://arxiv.org/abs/2306.03908 2

[31] M. Xu, X. Yin, L. Qiu, Y. Liu, X. Tong, and X. Han, "Sampro3d: Locating sam prompts in 3d for zero-shot instance segmentation," *International Conference on 3D Vision (3DV)*, 2015. 2

[32] R. Huang, S. Peng, A. Takmaz, F. Tombari, M. Pollefeys, S. Song, G. Huang, and F. Engelmann, "Segment3d: Learning fine-grained class-agnostic 3d segmentation without manual labels," *arXiv:2312.17232*, 2023. [Online]. Available: https://arxiv.org/abs/2312.17232 2

[33] X. Wu, L. Jiang, P.-S. Wang, Z. Liu, X. Liu, Y. Qiao, W. Ouyang, T. He, and H. Zhao, "Point transformer v3: Simpler, faster, stronger," in *CVPR*, 2024. 2, 5

[34] M. Tancik, P. Srinivasan, B. Mildenhall, S. Fridovich-Keil, N. Raghavan, U. Singhal, R. Ramamoorthi, J. Barron, and R. Ng, "Fourier features let networks learn high frequency functions in low dimensional domains," *Advances in neural information processing systems*, vol. 33, pp. 7537–7547, 2020. 3

[35] T.-Y. Lin, P. Goyal, R. Girshick, K. He, and P. Dollár, "Focal loss for dense object detection," 2018. [Online]. Available: https://arxiv.org/abs/1708.02002 4

[36] C. H. Sudre, W. Li, T. Vercauteren, S. Ourselin, and M. Jorge Cardoso, *Generalised Dice Overlap as a Deep Learning Loss Function for Highly Unbalanced Segmentations*. Springer International Publishing, 2017, p. 240–248. [Online]. Available: http://dx.doi.org/10.1007/978-3-319-67558-9_28 4

[37] K. Yadav, R. Ramrakhya, S. K. Ramakrishnan, T. Gervet, J. Turner, A. Gokaslan, N. Maestre, A. X. Chang, D. Batra, M. Savva *et al.*, "Habitat-matterport 3d semantics dataset," *arXiv preprint arXiv:2210.05633*, 2022. [Online]. Available: https://arxiv.org/abs/2210.05633 5

[38] J. Straub, T. Whelan, L. Ma, Y. Chen, E. Wijmans, S. Green, J. J. Engel, R. Mur-Artal, C. Ren, S. Verma, A. Clarkson, M. Yan, B. Budge, Y. Yan, X. Pan, J. Yon, Y. Zou, K. Leon, N. Carter, J. Briales, T. Gillingham, E. Mueggler, L. Pesqueira, M. Savva, D. Batra, H. M. Strasdat, R. D. Nardi, M. Goesele, S. Lovegrove, and R. Newcombe, "The replica dataset: A digital replica of indoor spaces," 2019. [Online]. Available: https://arxiv.org/abs/1906.05797 6

[39] L. McInnes, J. Healy, S. Astels *et al.*, "hdbscan: Hierarchical density based clustering." *J. Open Source Softw.*, vol. 2, no. 11, p. 205, 2017. 8, 3, 4

[40] A. Kirillov, K. He, R. Girshick, C. Rother, and P. Dollár, "Panoptic segmentation," 2019. [Online]. Available: https://arxiv.org/abs/1801.00868 2

[41] C. Choy, J. Gwak, and S. Savarese, "4d spatio-temporal convnets: Minkowski convolutional neural networks," in *2019 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2019, pp. 3070–3079. 5

[42] J. Schult, F. Engelmann, A. Hermans, O. Litany, S. Tang, and B. Leibe, "Mask3D: Mask Transformer for 3D Semantic Instance Segmentation," 2023. 11

[43] K. Yilmaz, J. Schult, A. Nekrasov, and B. Leibe, "Mask4former: Mask transformer for 4d panoptic segmentation," 2024. [Online]. Available: https://arxiv.org/abs/2309.16133 6, 11

[44] L. Yao, Y. Wang, C. Yawen, M. Liu, and L.-P. Chau, "Lassm: Efficient semantic-spatial query decoding via local aggregation and state space models for 3d instance segmentation," *xxx*, 2025. 6, 11

[45] A. Jain, P. Katara, N. Gkanatsios, A. W. Harley, G. Sarch, K. Aggarwal, V. Chaudhary, and K. Fragkiadaki, "Odin: a single model for 2d and 3d segmentation," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2024, pp. 3564–3574. 6, 11