

# OpenSplat3D: Open-Vocabulary 3D Instance Segmentation using Gaussian Splatting

Jens Piekenbrinck<sup>1</sup>, Christian Schmidt<sup>1</sup>, Alexander Hermans<sup>1</sup>, Narunas Vaskevicius<sup>2</sup>, Timm Linder<sup>2</sup>, Bastian Leibe<sup>1</sup>

<sup>1</sup>RWTH Aachen University      <sup>2</sup>Robert Bosch GmbH

## Abstract

*3D Gaussian Splatting (3DGS) has emerged as a powerful representation for neural scene reconstruction, offering high-quality novel view synthesis while maintaining computational efficiency. In this paper, we extend the capabilities of 3DGS beyond pure scene representation by introducing an approach for open-vocabulary 3D instance segmentation without requiring manual labeling, termed OpenSplat3D. Our method leverages feature-splatting techniques to associate semantic information with individual Gaussians, enabling fine-grained scene understanding. We incorporate Segment Anything Model instance masks with a contrastive loss formulation as guidance for the instance features to achieve accurate instance-level segmentation. Furthermore, we utilize language embeddings of a vision-language model, allowing for flexible, text-driven instance identification. This combination enables our system to identify and segment arbitrary objects in 3D scenes based on natural language descriptions. We show results on LERF-mask and LERF-OVS as well as the full ScanNet++ validation set, demonstrating the effectiveness of our approach.*

## 1. Introduction

Understanding and interpreting complex 3D scenes is a crucial challenge in computer vision, with applications spanning from robotics and augmented reality to autonomous systems. Traditional 3D instance segmentation methods rely heavily on manually labeled datasets, a process that is labor intensive and inherently limited to predefined object categories. This limitation has raised interest in open-vocabulary 3D scene understanding, where models are expected to segment and identify arbitrary objects without being restricted by a fixed set of labels. Additionally, most 3D instance segmentation methods rely on 3D point cloud datasets, which poses an additional constraint, as record-

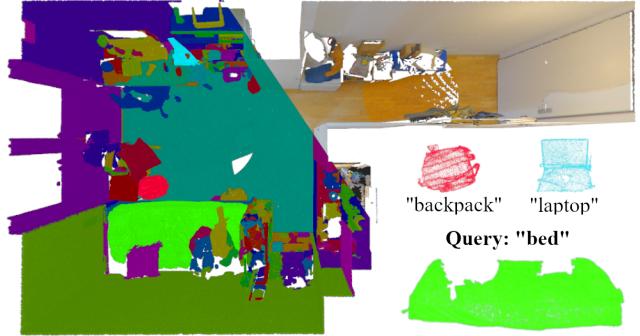


Figure 1. OpenSplat3D jointly optimizes 3D geometry and 3D instance segmentation of a scene based on a sequence of 2D input images using 3D Gaussian Splatting. We project the resulting 3D segmentations to the 2D views and extract per-instance language features. This allows us to perform open-vocabulary 3D instance segmentation, here demonstrated on a room scene of ScanNet++.

ing 3D data requires specialized recording setups [34]. This constraint limits dataset scalability, restricting the diversity and size of available 3D datasets.

In the domain of 2D image understanding, the development of powerful foundation models [2, 12, 18, 21, 35] has sparked a wave of research into open-vocabulary and language guided recognition. However, the underlying foundation models are only available for 2D image data, and the required amount of annotated 3D data to train equivalent native 3D foundation models currently does not exist. Previous work has therefore focused on projecting or embedding features from existing 2D foundation models into 3D point cloud representations [9, 27, 31]. While effective to some extent, 3D point clouds have notable limitations as their sparsity reduces feature coverage, they struggle with occlusion handling of surfaces and therefore a dense and differentiable 2D-to-3D mapping is not available.

In this work, we leverage multi-view data, and explore how 3D Gaussian Splatting (3DGS) [10], a state-of-the-art representation for neural scene reconstruction, can be ex-

tended beyond its original focus on novel view synthesis to support zero-shot open-vocabulary 3D instance segmentation. By combining differentiable rendering with 2D semantic cues, we introduce a framework capable of assigning semantic and instance-level labels to individual 3D Gaussians without requiring any manual 3D annotations.

Our method integrates SAM [12], a 2D foundational segmentation model for instance mask generation, as well as language embeddings from vision-language models [21, 30, 35] for text-driven object identification. These signals are projected into the 3D domain using differentiable rendering, allowing seamless alignment between 2D observations and the 3D Gaussian representation.

While earlier 3D Gaussian-based methods typically address *either* instance-level reasoning *or* language-based semantic retrieval in isolation [7, 20, 25, 33], OpenGaussian [29] very recently demonstrated that both tasks can be integrated into a joint approach. Similar to them, our concurrently developed method also unifies point-level instance segmentation and open-vocabulary semantics into a combined framework. This allows for an interesting quantitative comparison to recent *point cloud-based* open-vocabulary instance segmentation methods like Segment3D [9].

Additionally, we introduce a novel variance regularization loss and a pixel-wise contrastive loss inspired by [7, 29], to ensure robust and consistent segmentation of different instances independent of the viewpoint.

We evaluate our approach on the challenging ScanNet++(v1) [34] dataset following the class-agnostic approach of Segment3D [9], demonstrating that our method effectively segments instances in complex indoor scenes. We also showcase our open-vocabulary segmentation capabilities on the LERF-mask [33] and LERF-OVS [20] dataset. Our approach significantly outperforms the recently introduced OpenGaussian and Segment3D methods.

## 2. Related Work

**Foundation Models.** Foundation models [2, 12, 18, 21] have driven major advances in 2D image understanding. These models, specifically designed for image-based tasks, exhibit remarkable zero-shot capabilities, allowing a wide range of applications without the need for task-specific fine-tuning. Notably, CLIP [21], a large vision-language model, aligns visual and textual embeddings, facilitating zero-shot recognition of arbitrary object categories. SAM [12] achieves impressive zero-shot segmentation, providing high-quality instance masks for any object prompt. However, as these foundation models are predominantly image-based, it is essential to develop methods that enable transfer of the rich knowledge from 2D to the 3D domain.

**Radiance Fields for Scene Understanding.** Neural radiance fields (NeRFs) [16] have become a popular representa-

tion for novel view synthesis and 3D scene understanding. Methods such as LERF [11] and OpenNeRF [5] have introduced language embeddings, for example, CLIP features, to enable text-driven queries by distilling semantic information from the 2D domain into the continuous 3D volume provided by NeRF.

More recently, explicit representations such as 3D Gaussian Splatting (3DGS) [10] have gained prominence. 3DGS represents a scene as a set of Gaussian primitives, achieving NeRF-like view synthesis quality while preserving an interpretable point-based structure. Building upon this explicit representation, methods like LangSplat [20] have further extended semantic querying capabilities to 3DGS by integrating language embeddings into Gaussian representations. Approaches such as Gaussian Grouping [33], EgoLifter [7], and ClickGaussian [3] utilize SAM-generated 2D masks to segment and group Gaussians into distinct object instances, demonstrating the adaptability of 3D Gaussians for instance-level scene understanding.

In this work, we specifically leverage the explicit nature and rendering capability of 3DGS to ensure consistent multi-view feature transfer, which is crucial for effectively aligning rich 2D semantic observations with explicit 3D representations.

**Open-Vocabulary 3D Instance Segmentation.** Traditional 3D instance segmentation approaches, typically voxel or point cloud-based, often rely heavily on manually annotated datasets, limiting their ability to generalize beyond predefined categories [8, 13, 14, 23, 26].

To address these limitations, recent methods leverage vision-language models like CLIP [21] to enable open-vocabulary 3D scene understanding [9, 17, 19, 27]. Methods like OpenScene [19] focus on semantic understanding but lack instance-level granularity. OpenMask3D [27] addresses this by utilizing pretrained Mask3D [23] and SAM [12] models for instance labeling, and aggregates CLIP-based language information for each 3D instance. However, Mask3D was trained on ScanNet200 [4] and therefore is not a class-agnostic model. Segment3D [9] improves upon this by leveraging refined 2D SAM masks to extract confident masks for partial point clouds and performing self-supervised fine-tuning on full scenes, enhancing generalization to novel objects and categories.

Another line of work leveraging 3D Gaussian primitives as scene representation and exploiting the inherent 2D-to-3D connection is introduced by differentiable rendering. Besides semantic [20] and instance segmentation methods [7, 25, 33], the work most closely related to ours is OpenGaussian [29], which integrates instance-level features on 3D Gaussians with per-instance language embeddings to perform open-vocabulary 3D instance segmentation. Both OpenGaussian and our method employ contrastive learning with similar positive-pair objectives, pulling features from

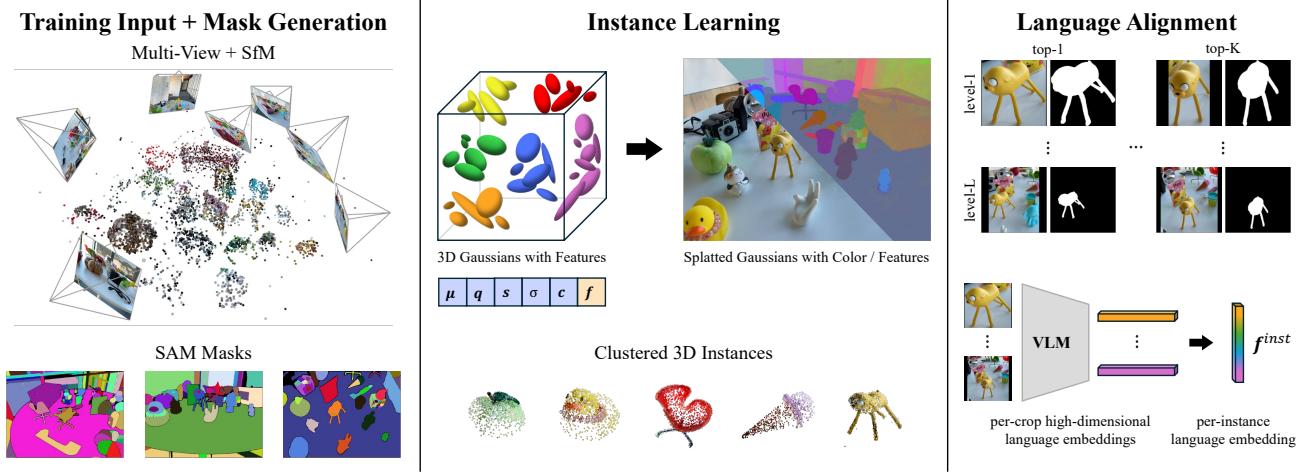


Figure 2. Overview of our proposed pipeline. On the left are the training inputs: posed RGB-images, a coarse SfM point cloud for initialization, and the extracted SAM masks. The middle section illustrates the instance learning with a Gaussian feature field optimization, as well as clustering to obtain coherent 3D instances. On the right, we demonstrate the language integration, where the top-k informative views are identified per instance, hierarchical crops are constructed and finally the language embedding per instance is computed.

the same instance together. However, while OpenGaussian penalizes negative pairs using inverse-distance weighting, our approach applies a margin-based loss to enforce a fixed minimum separation between distinct instances. Additionally, OpenGaussian explicitly enforces semantic consistency through a hierarchical discretized codebook, whereas our method implicitly encourages feature consistency via a novel variance-based regularization in continuous 3D space. Furthermore, our method converges in fewer than half the iterations required by OpenGaussian, resulting in substantially accelerated optimization.

### 3. Method

In this section, we introduce our approach for open-vocabulary 3D instance segmentation using 3D Gaussian representations. An overview of our pipeline is shown in Figure 2. In short, given a set of posed RGB images, we utilize the 3D Gaussian Splatting approach to optimize the 3D geometry of a scene and extend it to also optimize an instance embedding based on SAM masks. This allows us to obtain a class-agnostic segmentation for the 3D scene. These segments are in turn assigned a language feature based on a VLM output, which allows us to create a 3D open-vocabulary segmentation. First, we provide essential background on 3D Gaussian Splatting to contextualize our work. We then detail our method, and how we leverage the explicit and differentiable nature of 3DGS to effectively integrate semantic information from 2D observations into a coherent, multi-view consistent 3D representation.

#### 3.1. Background: 3D Gaussian Splatting

3D Gaussian Splatting (3DGS) represents a scene explicitly using oriented 3D Gaussian primitives. Each

Gaussian point  $n$  corresponds to the center of a local 3D Gaussian distribution and is parameterized by  $\theta_n = (\mu_n, \mathbf{R}_n, \mathbf{s}_n, \mathbf{c}_n, o_n)$ , where  $\mu_n$  is the 3D mean and  $\Sigma_n = (\mathbf{R}_n \mathbf{S}_n)^T \mathbf{S}_n \mathbf{R}_n$  is the covariance matrix parameterized via scale  $\mathbf{S}_n = \text{diag}(\mathbf{s}_n)$  and orientation  $\mathbf{R}_n$ . The view-dependent color is modeled via spherical harmonics  $\mathbf{c}_n = \text{SH}_i(\mathbf{d}_n)$ , with  $\mathbf{d}_n$  denoting the view direction. Additionally, each Gaussian has an associated scalar opacity  $o_n$ .

Rendering these Gaussians involves projecting them onto the 2D viewing plane via an approximation of the projective transformation, generating 2D Gaussians termed *splats* [10]. The 2D covariance matrix of each projected Gaussian is given by:

$$\hat{\Sigma}_n = [\mathbf{J}_n \mathbf{R}_c \Sigma_n \mathbf{R}_c^T \mathbf{J}_n^T]_{2 \times 2} \quad (1)$$

where  $\mathbf{J}_n$  is the Jacobian of the transformation, and  $\mathbf{R}_c$  is the camera's rotation matrix. These splats are depth-sorted relative to the camera viewpoint and composited via alpha blending. The contribution of a Gaussian at a pixel position  $\mathbf{p}$  is determined by:

$$\alpha_n = o_n \exp\left(-\frac{1}{2}(\mathbf{p} - \hat{\mu}_n)^\top \hat{\Sigma}_n^{-1}(\mathbf{p} - \hat{\mu}_n)\right), \quad (2)$$

where  $\hat{\mu}_n$  is the projected 2D mean. The final pixel color intensity  $\mathbf{I}(\mathbf{p})$  at pixel  $\mathbf{p}$  is computed as:

$$\mathbf{I}(\mathbf{p}) = \sum_{n=1}^N c_n \alpha_n \prod_{j=1}^{n-1} (1 - \alpha_j), \quad (3)$$

where  $c_n$  denotes the color contribution of Gaussian  $n$ ,  $\alpha_n$  the Gaussian's contribution and the product reflects the cumulative transmittance.

All rendering operations are implemented in an efficient CUDA-based rasterizer, allowing for rapid optimization and real-time rendering of complex scenes. The rendering is fully differentiable, allowing optimization using standard gradient-based methods. The Gaussians are optimized by minimizing image reconstruction errors with respect to a set of posed RGB images. Specifically, the RGB rendering loss is defined as:

$$\mathcal{L}_{\text{RGB}} = (1 - \beta)\mathcal{L}_{l_1} + \beta\mathcal{L}_{\text{SSIM}}, \quad (4)$$

where  $\mathcal{L}_{l_1}$  denotes the  $l_1$  loss for pixel-wise accuracy,  $\mathcal{L}_{\text{SSIM}}$  the structural similarity loss for perceptual quality [28], and  $\beta$  is a weighting factor balancing the two terms.

Once optimized, scenes can be rendered from arbitrary viewpoints, facilitating real-time novel view synthesis. Compared to conventional point clouds, Gaussian primitives can be densely rendered without occlusion artifacts, and the differentiable rendering pipeline inherently links 2D supervision signals to the 3D representation, enabling effective 2D-to-3D correspondence. These properties make 3D Gaussians Splatting highly suitable for integrating rich 2D information into detailed 3D representations.

### 3.2. OpenSplat3D

Given a set of posed RGB images  $\{\mathbf{I}_v | v = 1, \dots, V\}$  of a scene from multiple viewpoints  $v$ , we use the Segment Anything Model [12] to extract instance masks  $\{M_{v,i} | i = 1, \dots, N_v\}$  for each image where  $N_v$  represents the number of instances in viewpoint  $v$ . We write  $\mathbf{p} \in M_{v,i}$  to indicate that pixel  $\mathbf{p}$  is part of instance  $i$  in view  $v$ , and  $|M_{v,i}|$  to denote the number of pixels belonging to instance  $i$  in view  $v$ . Unlike in Gaussian Grouping [33], these instance masks are not required to be consistent across different viewpoints.

To encode instance-level information within the 3D Gaussians, we extend each Gaussian primitive with a view-independent instance feature embedding  $\mathbf{f}_n \in \mathbb{R}^d$ . These feature embeddings can be rendered from arbitrary viewpoints using the 3DGS splatting algorithm. To be precise, the rendered feature map  $\mathbf{F}(\mathbf{p})$  at pixel  $\mathbf{p}$  is computed as:

$$\mathbf{F}(\mathbf{p}) = \sum_{n=1}^N \mathbf{f}_n \alpha_n \prod_{j=1}^{n-1} (1 - \alpha_j), \quad (5)$$

where  $\mathbf{F} \in \mathbb{R}^{H \times W \times d}$ . This encoding provides two key benefits: First, the embeddings can be supervised with 2D data in 3D space via differentiable rendering. Second, the structured feature embeddings in 3D space facilitate scene understanding tasks such as 3D instance segmentation.

#### 3.2.1. Instance Learning

The per-Gaussian instance feature embeddings  $\mathbf{f}_n \in \mathbb{R}^d$  are optimized by rendering a feature map from a given viewpoint  $v$  and guiding the features using the 2D instance

masks  $M_{v,i}$  with a contrastive loss formulation, similar to prior work [3, 7, 25]. Given a rendered feature map, we compute a prototype feature  $\mathbf{z}_i$  per instance mask:

$$\mathbf{z}_i = \frac{1}{|M_{v,i}|} \sum_{\mathbf{p} \in M_{v,i}} \mathbf{F}(\mathbf{p}). \quad (6)$$

This is used in the positive part of the contrastive loss to pull all features of the corresponding mask to the prototype by:

$$\mathcal{L}_{\text{pos}} = \frac{1}{N_v} \sum_{i=1}^{N_v} \frac{1}{|M_{v,i}|} \sum_{\mathbf{p} \in M_{v,i}} \|\mathbf{F}(\mathbf{p}) - \mathbf{z}_i\|_2^2, \quad (7)$$

where  $|M_{v,i}|$  denotes the number of pixels in the mask for instance  $i$  and viewpoint  $v$ . The negative part of the contrastive loss pushes the prototypes away from each other using a margin-based loss:

$$\mathcal{L}_{\text{neg}} = \frac{2}{N_v(N_v - 1)} \sum_{i=1}^{N_v} \sum_{j > i} \text{ReLU}(\gamma - \|\mathbf{z}_i - \mathbf{z}_j\|_2^2), \quad (8)$$

where  $\gamma$  is the margin, a hyperparameter which defines the boundary where negative prototypes are too close and should be pushed away, preventing embeddings from collapsing into a single cluster.

The final contrastive instance-level loss is defined as:

$$\mathcal{L}_{\text{inst2D}} = w_p \cdot \mathcal{L}_{\text{pos}} + w_n \cdot \mathcal{L}_{\text{neg}}, \quad (9)$$

where  $w_p$  and  $w_n$  control the balance between instance compactness and separability, ultimately enhancing clusterability. While the SAM segmentation masks for the different views might give somewhat conflicting optimization targets, we empirically find that SAM instance masks across a scene typically tend to have a sufficiently consistent mode of segmenting objects. The joint optimization across the different views can be seen as a sort of majority voting during optimization and merely serves as a supervisory signal for the instance embeddings.

After optimization, the instance embeddings  $\mathbf{f}_n$ , and hence the Gaussians, are clustered into meaningful groups while also determining the number of clusters and identifying outliers. We leverage HDBSCAN [1], which is noise-aware and probabilistic, making it well-suited for the inherent noise and uncertainty in the instance labels and Gaussian representation.

#### 3.2.2. Language Alignment

Previous works without explicit instance information [11, 20] compute per-point language embeddings. Specifically within the context of 3DGS, it is infeasible to optimize high dimensional instance information for each Gaussian. For this reason, LangSplat projects language features to a low-dimension space with a scene specific auto-encoder [20].

To avoid this, we follow recent approaches and adopt a per-instance embedding strategy inspired by [9, 27, 29].

Rather than assigning and optimizing a language embedding for each Gaussian, we generate a single language embedding per instance after the optimization. To obtain this instance-level embedding, we render binary masks for each instance from multiple viewpoints. Gaussians belonging to a specific instance are rendered in white, while all others are set to black, resulting in an object silhouette respecting occlusion. Afterwards, the rendering is thresholded to produce a binary mask. For selecting the most informative views, we employ a visibility-based scoring method similar to OpenMask3D [27]. The visibility score for a given instance  $i$  and a viewpoint  $v$  is computed as:

$$s_{v,i} = \frac{|\mathbf{M}_{v,i}|}{|\mathbf{I}_v|} \cdot \frac{|G_{v,i}|}{|G_i|}, \quad (10)$$

where  $|\mathbf{I}_v|$  is the number of pixels of image  $\mathbf{I}_v$ ,  $|G_i|$  is the number of Gaussians associated with instance  $i$ , and  $|G_{v,i}|$  represents the number of Gaussians for instance  $i$  visible in the viewport of viewpoint  $v$ . Since the rasterizer provides information about all Gaussians within the viewport,  $|G_{v,i}|$  can be computed efficiently.

For each of the selected top- $K$  viewpoints,  $L$  image crops centered around the predicted instance mask are extracted at different zoom levels to capture both detailed and contextual information. These crops are processed by the image encoder of a vision-language model to produce language embeddings  $\mathbf{l}_{i,k,l}$ , where  $i$  indicates the instance,  $k$  represents the viewpoint, and  $l$  denotes the zoom level.

The final language embedding for the instance is obtained by averaging the embeddings across the selected viewpoints and zoom levels:

$$\mathbf{l}_i = \frac{1}{KL} \sum_{k=1}^K \sum_{l=1}^L \mathbf{l}_{i,k,l}. \quad (11)$$

This approach ensures that the language embedding captures diverse visual information about the instance, enhancing its semantic representation.

### 3.2.3. Instance Feature Regularization

A significant challenge in the differential rendering via alpha compositing lies in the discrepancy between 2D supervision and the underlying 3D representation. During rendering, overlapping Gaussians are blended, combining their features to produce the final 2D feature map. Although this blending ensures alignment with 2D supervision signals, it does not inherently enforce consistency among the individual Gaussian embeddings. Consequently, they can diverge, potentially leading to inconsistent instance representations or feature misalignment across views.

To address this, we propose a novel variance regularization loss that minimizes the variance of feature embeddings

along rendering rays. Using  $\text{Var}(X) = E(X^2) - E(X)^2$ , we compute the variance of the feature map  $\mathbf{F}$  at pixel  $\mathbf{p}$  as

$$\text{Var}(\mathbf{F}(\mathbf{p})) = \left( \sum_{n=1}^N \mathbf{f}_n^2 \alpha_n \prod_{j=1}^{n-1} (1 - \alpha_j) \right) - \mathbf{F}(\mathbf{p})^2, \quad (12)$$

which is efficiently calculated within the CUDA kernel during rendering, minimizing computational overhead. The variance is minimized by an  $l_2$  loss given as

$$\mathcal{L}_{\text{var}} = \frac{1}{|\mathbf{I}|} \sum_{\mathbf{p} \in \mathbf{I}} \|\text{Var}(\mathbf{F}(\mathbf{p}))\|_2^2. \quad (13)$$

Our final optimization objective combines the RGB reconstruction loss with the contrastive instance loss and the variance regularization term:

$$\mathcal{L} = \mathcal{L}_{\text{RGB}} + \lambda_{\text{inst2d}} \mathcal{L}_{\text{inst2d}} + \lambda_{\text{var}} \mathcal{L}_{\text{var}}, \quad (14)$$

where  $\lambda_{\text{inst2d}}$  and  $\lambda_{\text{var}}$  are weighting factors of each loss.

## 4. Experiments

**Implementation Details.** For the instance mask extraction, we utilize only the default output from SAM [12], sorting the masks by the combination of intersection-over-union (IoU) and stability scores and combining the binary masks into a single mask containing unique instance IDs.

We retain most hyperparameters of the original Gaussian Splatting formulation [10]. Since our additional loss terms lead to higher per-Gaussian gradients, we increase the gradient threshold for densification, preventing excessive growth in the number of Gaussians during optimization. Following 3DGS, each scene is optimized for 30000 iterations. For the instance feature embedding  $\mathbf{f}_n^i \in \mathbb{R}^d$  we set the dimension to  $d = 8$  in all our experiments. Loss weights are set to  $\lambda_{\text{inst2d}} = 0.1$ ,  $\lambda_{\text{var}} = 0.5$ ,  $w_{\text{pos}} = 1.0$ ,  $w_{\text{neg}} = 1.0$ , and the margin of the negative contrastive loss to  $\gamma = 1.0$ . Notably, we employ a consistent set of hyperparameters across all experiments, in contrast to Open-Gaussian [29] which uses adapted parameters per scene to achieve optimal results. The experiments are conducted on an RTX 3090 GPU. Optimizing the instances per scene typically takes 20-45 minutes. Clustering via the cuML [22] CUDA implementation of HDBSCAN takes between a few seconds and approximately 2 minutes depending on the number of Gaussians. For the language embeddings, we use MasQCLIP [30] as vision-language model due to its usage of masks to generate embeddings for specific objects in an image while retaining context. The computation of these embeddings using top  $k = 5$  views and  $l = 3$  zoom levels with an expansion ratio of 0.3 finishes in a few minutes, varying based on the number of instances in the scene.

Method	figurines		ramen		teatime		mean	
	mIoU	mBIoU	mIoU	mBIoU	mIoU	mBIoU	mIoU	mBIoU
LERF [11]*	33.5	30.6	28.3	14.7	49.7	42.6	37.2	29.3
LangSplat [20]*	52.8	50.5	50.4	44.7	69.5	65.6	57.6	53.6
Gaussian Grouping [33]*	69.7	67.9	<b>77.0</b>	<b>68.7</b>	71.7	66.1	72.8	67.6
CGC [25]	<u>91.6</u>	<u>88.8</u>	68.7	63.1	<u>80.5</u>	<b>78.9</b>	<u>80.3</u>	<u>76.9</u>
OpenSplat3D (Ours)	<b>92.3</b>	<b>89.4</b>	<u>75.9</u>	<u>68.2</u>	<b>83.7</b>	<u>78.8</u>	<b>84.0</b>	<b>78.8</b>

Table 1. Semantic segmentation results on the LERF-mask dataset. We report the mean IoU and mean BIoU for each scene and the overall average. Our method achieves the best overall performance across all metrics. Only for the ramen scene, Gaussian Grouping performs slightly better. \*: Results as reported in the Gaussian Grouping paper [33].

Method	figurines		ramen		teatime		waldo_kitchen		mean	
	mIoU	mAcc.	mIoU	mAcc.	mIoU	mAcc.	mIoU	mAcc.	mIoU	mAcc.
LangSplat [20]	10.16	8.93	7.92	11.27	11.38	20.34	9.18	9.09	9.66	12.41
LEGaussians [24]	17.99	23.21	15.79	26.76	19.27	27.12	11.78	18.18	16.21	23.82
OpenGaussian [29]	39.29	55.36	31.01	42.25	60.44	76.27	22.70	31.82	38.36	51.43
OpenSplat3D (Ours)	<b>60.71</b>	<b>85.71</b>	<b>49.20</b>	<b>76.06</b>	<b>73.27</b>	<b>88.14</b>	<b>55.63</b>	<b>77.27</b>	<b>59.70</b>	<b>81.79</b>

Table 2. LERF-OVS 3D object selection evaluation from textual query. Following OpenGaussian [29], only the Gaussians responding to the query are rendered, therefore the rendering does not respect occlusion by other objects in the scene. Accuracy is provided by mAcc@0.25. Note that OpenGaussian fine-tunes parameters per scene for best results.

**Datasets.** We evaluate our method on open-vocabulary segmentation on two datasets, LERF-mask [33] and LERF-OVS. Both datasets extend the original LERF dataset introduced by Kerr *et al.* [11], which provides multi-view images of real-world scenes with accurate camera poses and annotations for the task of object localization. LERF-mask and LERF-OVS augment a subset of these scenes with additional semantic segmentation masks and corresponding textual prompts for various objects.

Furthermore, to comprehensively assess the ability of 3D instance segmentation of our method, we perform evaluations on the ScanNet++ (v1) dataset [34], utilizing its validation split consisting of 50 annotated indoor scenes. Note that this dataset actually performs the evaluation in 3D, unlike the LERF-Mask/OVS datasets which evaluate the performance on 2D views. Additionally, ScanNet++ is significantly larger, is annotated much more densely, and evaluated across 84 instance classes.

#### 4.1. Open-Vocabulary Segmentation

**LERF-mask.** For a fair comparison with other approaches, we adopt the same observer-based protocol proposed by Gaussian Grouping [33] for the LERF-mask dataset. Given an observed reference frame and a textual prompt, the open-vocabulary detector Grounding DINO [15] is leveraged to compute object proposals in form of bounding boxes, which are subsequently used as prompts for SAM to extract semantic mask proposals. These masks

are used to determine the instances of interest by splatting a binary mask for each instance into the 2D view and selecting the instances over an intersection-over-area (*IoA*) threshold. In the subsequent evaluation frames, only the masks for the selected instances are rendered. This multi-view evaluation strategy assesses the quality of 3D localization via intermediate 2D segmentation consistency, requiring the model to render stable and accurate 2D segmentation masks across multiple views based on the information of the reference frame.

Table 1 reports results on LERF-mask using the official metrics, specifically the mean Intersection-over-Union (*mIoU*) for masks and bounding boxes (*mBIoU*). OpenSplat3D shows superior performance on most scenes, achieving significantly better scores on average. Gaussian Grouping performs slightly better on the *ramen* scene.

We present qualitative results for textual queries and object extraction on the LERF-mask dataset in Figure 3. To clearly illustrate the Gaussians belonging to each instance, we render extracted objects without occlusion. This visualization also highlights Gaussians incorrectly associated with the instances, typically found near object boundaries that merge with other objects, *e.g.*, the porcelain hand with the table, or due to specular reflections. These specularity-related Gaussians often have an offset from the surface due to the viewpoint-dependent nature of reflections.

**LERF-OVS.** The LERF-OVS dataset, introduced by [20], provides an open-vocabulary segmentation benchmark of

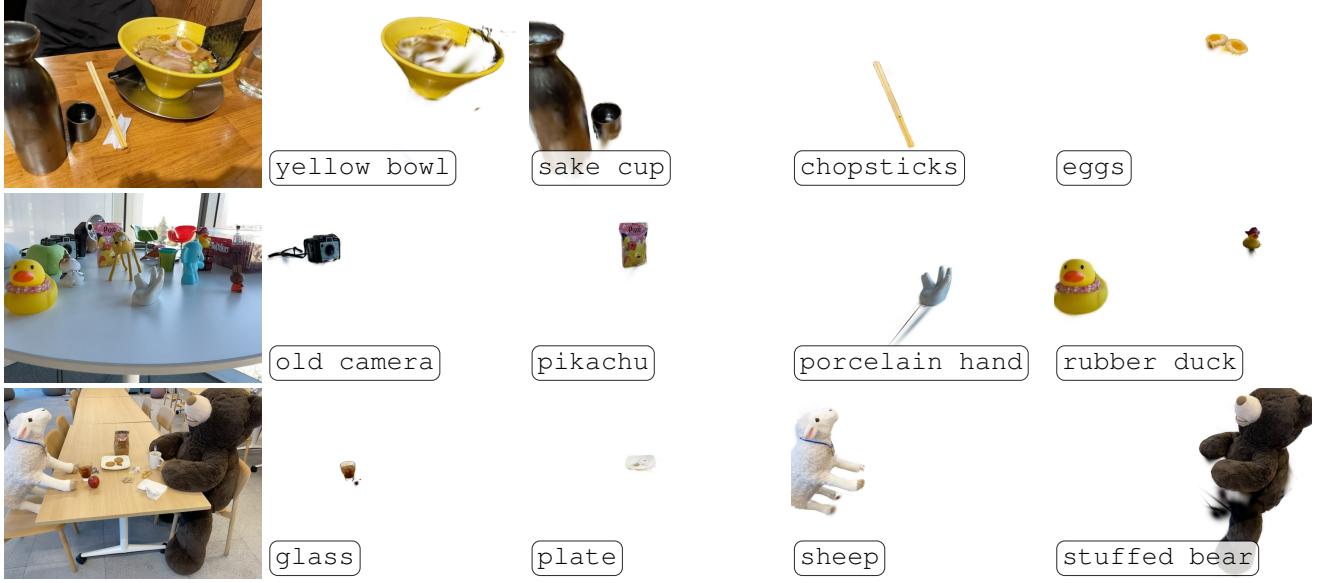


Figure 3. Language query results on the LERF-mask dataset. The left side shows example RGB training images for reference. The right side showcases rendered 3D instances extracted with OpenSplat3D using natural language queries. The query string is inset in each image.

the LERF dataset on four subscenes: *figurines*, *ramen*, *teatime*, *waldo\_kitchen*. OpenGaussian [29] introduced a protocol to evaluate 3D object selection on this dataset, differing from earlier observer-based evaluations that depended on 2D reference views. Instead, their protocol directly selects Gaussians based on language queries without preliminary rendering. Methods such as LangSplat and LEGaussian utilize *per-point* language embeddings which can be leveraged to select the Gaussians based on the similarity to the language. In contrast, both OpenGaussian and our method employ *per-instance* language embeddings, selecting entire instances based on embedding similarity. Selected Gaussians are rendered to binary masks from multiple viewpoints without occlusion handling. Although this rendering approach lowers performance metrics due to the occlusion-aware ground-truth masks, it effectively highlights Gaussians incorrectly assigned to instances in the 3D space. We evaluate our approach using the *mIoU* and *mAcc* metrics for the 3D object selection task in Table 2. Our method notably outperforms existing approaches—including the recent OpenGaussian—across all metrics, demonstrating significant improvements in 3D object selection accuracy. We attribute this to our novel variance loss.

#### 4.2. 3D Instance Segmentation

To assess the performance of our method in the 3D space, we evaluate on the validation split of the ScanNet++ (v1) dataset. Gaussians are initialized using a sampled scene point cloud containing up to 750k points. For optimization, we uniformly select 300 frames per scene. Due to the high resolution of the training images, optimization is performed

at half resolution. For inference on a scene, each point of the point cloud is assigned the instance ID of the nearest Gaussian mean. We further apply post-processing using a graph-based smoothing method [6] commonly employed in related approaches [9, 23, 27]. Unlike Segment3D [9], we omit the DBSCAN post-processing step, as it did not enhance results in our experiments. Each predicted instance is assigned to the semantic label with the highest similarity to the corresponding instance language embedding.

We report the official evaluation metrics, including the average precision *AP* metric, computed over IoU thresholds ranging from 0.5 to 0.95 with a step size of 0.05, and also the metrics *AP<sub>50</sub>* and *AP<sub>25</sub>* at IoU thresholds 0.5 and 0.25 respectively.

We first evaluate our method on the class-agnostic instance segmentation task following [9]. The results detailed in Table 3 demonstrate that our approach significantly improves over the baseline methods. Additionally, the metrics reveal that our approach exhibits reduced dependency on post-processing as indicated by a smaller gap between the with and without post-processing variants.

Results on the ScanNet++ instance segmentation task on the validation split are shown in Table 4. OpenSplat3D operates in *zero-shot open-vocabulary* setting and surpasses other open-vocabulary methods by a substantial margin (11.2 points over Segment3D on the *AP<sub>50</sub>* metric). Furthermore, OpenSplat3D significantly narrows the performance gap to the *fully-supervised closed-set* SGIFformer [32]. Our analysis indicates this mainly stems from the tendency of SAM to over-segment scenes. Therefore, addressing this issue might further reduce the gap.

Finally, we present qualitative results of the class-

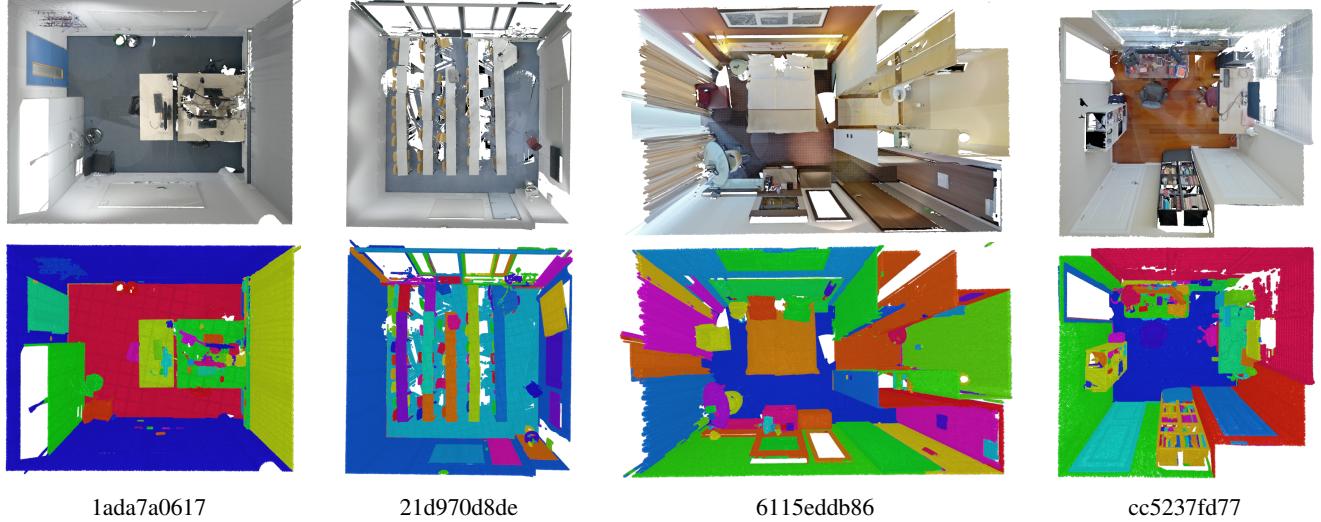


Figure 4. Qualitative Results on four scenes of the ScanNet++ validation split. Top row showing the mesh of the scan in a top-down view, the bottom row the respective point cloud with instance segmentation labels. The corresponding scene ID is shown below each segmentation.

Method	without post-processing			with post-processing		
	AP	AP50	AP25	AP	AP50	AP25
SAM3D [31]	3.9	9.3	22.1	8.4	16.1	30.0
Segment3D [9]	13.0	23.8	38.3	20.2	30.9	42.7
<i>Open3DIS</i> [17]*	-	-	-	20.7	38.6	47.1
OpenSplat3D (Ours)	<b>19.2</b>	<b>37.3</b>	<b>56.2</b>	<b>24.5</b>	<b>41.7</b>	<b>57.1</b>

Table 3. Class-agnostic instance segmentation on ScanNet++ val split. \*: Open3DIS uses superpoints produced by Felzenszwalb and Huttenlocher segmentation [6] directly in their pipeline.

Method	Setting	AP	AP50	AP25
SGIFormer [32]	fully-supervised	23.9	37.5	46.6
Mask3D [23] (+ OpenMask3D [27])	open-vocabulary	-	15.0	-
Segment3D [9] (+ OpenMask3D [27])	open-vocabulary	-	18.5	-
OpenSplat3D (Ours)	open-vocabulary	16.5	<b>29.7</b>	39.0

Table 4. Instance Segmentation on the ScanNet++ validation split. Our method not only outperforms the other open-vocabulary methods by a large margin, it also reduces the gap to the state-of-the-art fully-supervised SGIFormer [32] approach.

agnostic instance segmentation on four arbitrary ScanNet++ scenes in Figure 4, showing the colored mesh in the top row from which the point clouds are sampled and our instance segmentation of the point cloud in the bottom row.

**Variance Loss Ablation.** We conduct an ablation study on the LERF-mask dataset to analyze the effectiveness of our proposed variance regularization. Fig. 5 shows that disabling the variance loss leads to a substantial decrease in performance, and even a small weighting factor  $\lambda_{\text{var}}$  significantly improves segmentation accuracy and this improvement is robust to the exact value. These results strongly support our hypothesis that substantial feature blending occurs during the rendering process, and emphasize the im-

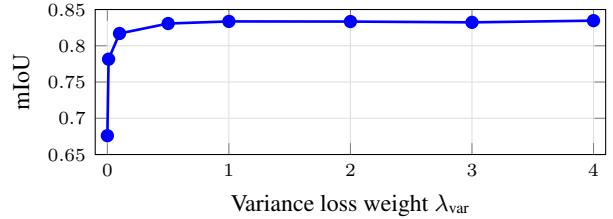


Figure 5. Influence of the variance loss, evaluated on the LERF-mask dataset. Even small  $\lambda_{\text{var}}$  improves results significantly.

portance of variance regularization for 3D instance learning by preserving the 2D to 3D consistency of the features of all Gaussians that affect a pixel.

## 5. Conclusion

We propose OpenSplat3D, a method for open-vocabulary 3D instance segmentation, extending 3DGS with instance features and per-instance language embeddings. Our novel variance regularization loss effectively improves feature consistency during alpha-composited rendering, significantly improving the segmentation performance, while being efficient to compute. Comprehensive evaluations across multiple open-vocabulary benchmarks demonstrate the effectiveness of OpenSplat3D, suggesting that 3D Gaussian representations in combination with 2D VFM are an effective setup. Future research directions may explore enhancements to instance learning, and different ways of incorporating language embeddings, but also extending the framework to dynamic environments is a promising direction.

**Acknowledgments.** This work was funded by Bosch Research as part of a Bosch-RWTH collaboration on “Context Understanding for Autonomous Systems”.

## References

- [1] Ricardo J. G. B. Campello, Davoud Moulavi, and Joerg Sander. Density-based clustering based on hierarchical density estimates. In *Advances in Knowledge Discovery and Data Mining*, 2013. 4
- [2] Mathilde Caron, Hugo Touvron, Ishan Misra, Hervé Jégou, Julien Mairal, Piotr Bojanowski, and Armand Joulin. Emerging properties in self-supervised vision transformers. In *ICCV*, 2021. 1, 2
- [3] Seokhun Choi, Hyeonseop Song, Jaechul Kim, Taehyeong Kim, and Hoseok Do. Click-Gaussian: Interactive segmentation to any 3D gaussians. In *ECCV*, 2024. 2, 4
- [4] Angela Dai, Angel X. Chang, Manolis Savva, Maciej Halber, Thomas Funkhouser, and Matthias Nießner. ScanNet: Richly-annotated 3D reconstructions of indoor scenes. In *CVPR*, 2017. 2
- [5] Francis Engelmann, Fabian Manhardt, Michael Niemeyer, Keisuke Tateno, Marc Pollefeys, and Federico Tombari. OpenNeRF: Open set 3D neural scene segmentation with pixel-wise features and rendered novel views. In *ICLR*, 2024. 2
- [6] Pedro F Felzenszwalb and Daniel P Huttenlocher. Efficient graph-based image segmentation. *IJCV*, 59, 2004. 7, 8
- [7] Qiao Gu, Zhaoyang Lv, Duncan Frost, Simon Green, Julian Straub, and Chris Sweeney. EgoLifter: Open-world 3D segmentation for egocentric perception. In *ECCV*, 2024. 2, 4
- [8] Lei Han, Tian Zheng, Lan Xu, and Lu Fang. OccupSeg: Occupancy-aware 3D instance segmentation. In *CVPR*, 2020. 2
- [9] Rui Huang, Songyou Peng, Ayca Takmaz, Federico Tombari, Marc Pollefeys, Shiji Song, Gao Huang, and Francis Engelmann. Segment3D: Learning fine-grained class-agnostic 3D segmentation without manual labels. In *ECCV*, 2025. 1, 2, 5, 7, 8
- [10] Bernhard Kerbl, Georgios Kopanas, Thomas Leimkühler, and George Drettakis. 3D Gaussian Splatting for Real-Time Radiance Field Rendering. *ACM TOG*, 2023. 1, 2, 3, 5
- [11] Justin Kerr, Chung Min Kim, Ken Goldberg, Angjoo Kanazawa, and Matthew Tancik. LERF: Language embedded radiance fields. In *CVPR*, 2023. 2, 4, 6
- [12] Alexander Kirillov, Eric Mintun, Nikhila Ravi, Hanzi Mao, Chloe Rolland, Laura Gustafson, Tete Xiao, Spencer Whitehead, Alexander C Berg, Wan-Yen Lo, et al. Segment anything. In *CVPR*, 2023. 1, 2, 4, 5
- [13] Maksim Kolodiaznyi, Anna Vorontsova, Anton Konushin, and Danila Rukhovich. Top-down beats bottom-up in 3D instance segmentation. In *WACV*, 2024. 2
- [14] Maxim Kolodiaznyi, Anna Vorontsova, Anton Konushin, and Danila Rukhovich. OneFormer3D: One transformer for unified point cloud segmentation. In *CVPR*, 2024. 2
- [15] Shilong Liu, Zhaoyang Zeng, Tianhe Ren, Feng Li, Hao Zhang, Jie Yang, Qing Jiang, Chunyuan Li, Jianwei Yang, Hang Su, et al. Grounding DINO: Marrying DINO with grounded pre-training for open-set object detection. In *ECCV*, 2024. 6
- [16] Ben Mildenhall, Pratul P. Srinivasan, Matthew Tancik, Jonathan T. Barron, Ravi Ramamoorthi, and Ren Ng. NeRF: Representing scenes as neural radiance fields for view synthesis. In *ECCV*, 2020. 2
- [17] Phuc Nguyen, Tuan Duc Ngo, Evangelos Kalogerakis, Chuang Gan, Anh Tran, Cuong Pham, and Khoi Nguyen. Open3DIS: Open-vocabulary 3D instance segmentation with 2D mask guidance. In *CVPR*, 2024. 2, 8
- [18] Maxime Oquab, Timothée Darret, Théo Moutakanni, Huy Vo, Marc Szafraniec, Vasil Khalidov, Pierre Fernandez, Daniel Haziza, Francisco Massa, Alaeldin El-Nouby, et al. DINoV2: Learning robust visual features without supervision. *TMLR*, 2024. 1, 2
- [19] Songyou Peng, Kyle Genova, Chiyu Jiang, Andrea Tagliasacchi, Marc Pollefeys, Thomas Funkhouser, et al. OpenScene: 3D scene understanding with open vocabularies. In *CVPR*, 2023. 2
- [20] Minghan Qin, Wanhua Li, Jiawei Zhou, Haoqian Wang, and Hanspeter Pfister. LangSplat: 3D language gaussian splatting. In *CVPR*, 2024. 2, 4, 6
- [21] 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 *ICML*, 2021. 1, 2
- [22] Sebastian Raschka, Joshua Patterson, and Corey Nolet. Machine learning in Python: Main developments and technology trends in data science, machine learning, and artificial intelligence. *Information*, 11(4):193, 2020. 5
- [23] Jonas Schult, Francis Engelmann, Alexander Hermans, Or Litany, Siyu Tang, and Bastian Leibe. Mask3D: Mask transformer for 3D semantic instance segmentation. In *ICRA*, 2023. 2, 7, 8
- [24] Jin-Chuan Shi, Miao Wang, Hao-Bin Duan, and Shao-Hua Guan. Language embedded 3D gaussians for open-vocabulary scene understanding. In *CVPR*, 2024. 6
- [25] Myrna C. Silva, Mahtab Dahaghin, Matteo Toso, and Alessio Del Bue. Contrastive gaussian clustering for weakly supervised 3D scene segmentation. In *ICPR*, 2024. 2, 4, 6
- [26] Jiahao Sun, Chunmei Qing, Junpeng Tan, and Xiangmin Xu. Superpoint transformer for 3D scene instance segmentation. *AAAI*, 2023. 2
- [27] Ayça Takmaz, Elisabetta Fedele, Robert W. Sumner, Marc Pollefeys, Federico Tombari, and Francis Engelmann. Open-Mask3D: Open-Vocabulary 3D Instance Segmentation. In *NeurIPS*, 2023. 1, 2, 5, 7, 8
- [28] Zhou Wang, A.C. Bovik, H.R. Sheikh, and E.P. Simoncelli. Image quality assessment: from error visibility to structural similarity. *TIP*, 13(4):600–612, 2004. 4
- [29] Yanmin Wu, Jiarui Meng, Haijie Li, Chenming Wu, Yahao Shi, Xinhua Cheng, Chen Zhao, Haocheng Feng, Errui Ding, Jingdong Wang, and Jian Zhang. OpenGaussian: Towards point-level 3D gaussian-based open vocabulary understanding. In *NeurIPS*, 2024. 2, 5, 6, 7
- [30] Xin Xu, Tianyi Xiong, Zheng Ding, and Zhuowen Tu. MasQCLIP for open-vocabulary universal image segmentation. In *ICCV*, 2023. 2, 5
- [31] Yunhan Yang, Xiaoyang Wu, Tong He, Hengshuang Zhao, and Xihui Liu. SAM3D: Segment anything in 3D scenes. In *ICCV Workshops*, 2023. 1, 8

- [32] Lei Yao, Yi Wang, Moyun Liu, and Lap-Pui Chau. SGI-Former: Semantic-guided and geometric-enhanced inter-leaving transformer for 3D instance segmentation. *TCSVT*, 2024. [7](#), [8](#)
- [33] Mingqiao Ye, Martin Danelljan, Fisher Yu, and Lei Ke. Gaussian Grouping: Segment and edit anything in 3D scenes. In *ECCV*, 2024. [2](#), [4](#), [6](#)
- [34] Chandan Yeshwanth, Yueh-Cheng Liu, Matthias Nießner, and Angela Dai. ScanNet++: A high-fidelity dataset of 3D indoor scenes. In *CVPR*, 2023. [1](#), [2](#), [6](#)
- [35] Xiaohua Zhai, Basil Mustafa, Alexander Kolesnikov, and Lucas Beyer. Sigmoid loss for language image pre-training. In *ICCV*, 2023. [1](#), [2](#)