Architect3D: Improving the Mask Quality for Open-Vocabulary Pipelines

Hans Baumann-Ortiz, Parth Shandilya, Dogukan Bagci, Pal Aggarwal
Computer Vision and Geometry Group | ETH Zurich
habaumann@ethz.ch, pshandilya@ethz.ch, dogukan.bagci@uzh.ch, aggarwalp@ethz.ch

Abstract

Although closed-vocabulary instance segmentation has demonstrated strong performance with a fixed set of object categories, its applicability to real-world environments is limited due to its reliance on a closed set of labels. Recent advances in vision-language models, such as CLIP, offer a promising approach to open-vocabulary instance segmentation. However, existing open-vocabulary methods have difficulty with architectural elements. Heavy occlusions, texture-poor surfaces, and attached structures commonly cause failure. This issue is further complicated by the fact that 3D instance segmentation methods are typically trained using datasets that disregard architectural elements. These datasets prioritize discrete, countable objects with well-defined shapes and identifiable parts (things) over amorphous background regions without distinct parts (stuff) in common labeling schemes.

*In this work, we present **Architect3D**, a model that aims to improve upon Mask3D by incorporating a key modification. To train the model, we replaced ScanNet200 with ScanNet++, an enhanced dataset that corrects prior labeling inaccuracies and provides finer-grained annotations of object instances and background classes such as architectural elements. We hypothesized that this replacement would generate higher-quality mask proposals and enable better overall instance segmentation of **things** (e.g. tables, cups, and chairs) **and stuff** (e.g., walls, ceilings, and floors). However, this approach did not yield the expected improvements; Architect3D is ineffective at producing reliable results on ScanNet++. We provide a detailed analysis of the failures, identifying potential causes, including increased inter-class ambiguity in ScanNet++, training constraints, and architectural limitations. These findings offer important insights into the challenges of fine-grained 3D instance segmentation.*

1. Introduction

In recent years, advancements in 2D foundation models have unlocked new possibilities for interpreting 3D environments. Techniques like OpenMask3D [16] enable open-vocabulary querying of 3D object instances. The querying of object instances through text makes the model more applicable to open-world problems such as autonomous driving or human-robot interaction. Nevertheless, a significant limitation remains: current approaches prioritize generic objects such as chairs and tables (also known as *things*) while neglecting structural architectural elements like walls, floors, and columns (also known as *stuff*). Current 3D instance segmentation methods, such as Mask3D [13] and SAM3D [19] fail to address this issue leaving a critical gap in applications such as robotic construction inspection and automated architectural planning. Current approaches lack the ability to **reliably identify architectural elements** which may be *not fully visible*, *largely obscured by other scene objects* (e.g., a painting on a wall) or a not recognized due to *lack of texture or distinctiveness*.

This work presents **Architect3D**, a novel model that advances *open-vocabulary* 3D instance segmentation by expanding the set of training classes of *Mask3D* [13]–a widely adopted backbone network [4, 6, 16, 17]. The proposed method enhances mask quality for both *things* (discrete, countable objects) and *stuff* (amorphous background regions). It also addresses **key challenges** in 3D architectural scene understanding, including:

- **Structural instance identification**: Identifying architectural elements is hard due to a variety of reasons. Due to the lack of texture of these elements. Isolating 3D structural elements (walls, floors, ceilings) from attached objects (e.g., paintings on walls) or objects occluding them (e.g. a couch infront of a wall).
- **Feature representation**: Computing open-vocabulary semantic features that reflect only the properties of the structural elements.
- **Datasets with coarse-level annotations**: Common datasets such as ScanNet200 [12] do not have fine-grained feature masks that clearly outline the architectural elements.

Our work targeted these challenges by leveraging ScanNet++, which offers higher-quality annotations and more detailed class granularity. However, as our results show,

this approach encountered significant difficulties that we analyze in detail.

2. Related Work

3D instance segmentation. The goal of 3D semantic segmentation is to categorize each point in a given 3D scene as a specific semantic class from a predefined set of classes [23]. 3D instance segmentation takes it a step further, distinguishing between objects with the same semantic category [7, 13, 15]. The ScanNet200 benchmark [12] is often used as a standard when evaluating 3D instance segmentation models. While these models perform well, closed-vocabulary tasks are limited by their closed set of classes and cannot adapt to changing open-world environments.

Open-vocabulary segmentation. The emergence of vision-language models such as CLIP [10] or SigLIP [22] are the main drivers of open-vocabulary 2D segmentation because they align image and text feature spaces. . Several new methods [5, 7–9, 16, 24] use these models for open-vocabulary or zero-shot semantic segmentation.

Despite these advances, a key limitation persists: these methods have been optimized for recognizing *things* and not *stuff*, which presents unique challenges due to their scale, texture properties, and complex interaction with other scene elements (e.g. painting attached on a wall). ScanNet++ [20] has emerged as an opportunity to address this issue. It is a large-scale dataset that was recently released. It includes over 1000 scenes, higher-resolution scans, and finer-grained annotations for all instances (things and stuff), improving upon ScanNet200[12].

3. Method

Overview. We present Architect3D, a new version of the Mask3D model [13] designed to improve mask quality, particularly for structural elements in 3D scenes. Our approach introduces a key modification to the original system: training the Mask3D model [13] on the larger and more detailed ScanNet++ dataset [20] which includes background classes such as walls and floors that are not typically covered in instance segmentation datasets. This modification is intended to result in a boost in performance over the original model, especially for architectural elements.

Training Configuration. We used the official Mask3D training code without making any architectural modifications to the model. The hyperparameters such as the learning rate, batch size, optimizer, and training schedule are summarized in Table4. We initialize the model with the official checkpoint that was trained on the ScanNet200 dataset. Due to the differences in the number of classes, we still had to randomly initialize the model head.

Model	$\text{mAP}_{50:95}$	AP@25	AP@50
Architect3D	0.7	1.4	2.4

Table 1. Quantitative results on all classes of ScanNet++.

4. Experiment

We now evaluate the effectiveness of our proposed method through a series of experiments.

4.1. Quantitative Results

Datasets. Following [4], we evaluated the performance of Architect3D on the ScanNet++ dataset [20]. However, in contrast to [4], we trained and evaluated on all 2753 classes of ScanNet++. Since we are also interested in architectural elements, we evaluated on a subset of categories corresponding to structural elements such as walls, floors, ceilings, and beams. We define architectural elements as *elements that define the structural layout of a room.* A full list of the architectural elements can be found in the appendix section A.1. This list was created by manually reviewing all 2,753 ScanNet++ instance classes and selecting those that met the criteria.

Evaluation Metrics. We adopted the average precision (AP) metric, which is widely used in 3D instance segmentation tasks. Consistent with prior work [16, 21, 24], we report AP at IoU thresholds of 25% (AP@25) and 50% (AP@50), as well as the mean AP over thresholds ranging from 50% to 95% in 5% increments ($\text{mAP}_{50:95}$). The metrics are reported on both the full set of annotated instances and the architectural subset specifically.

Evaluation Protocol. We evaluated Architect3D using the described the evaluation metrics for all classes and the architectural classes.

Results. Our results are below our expectations. As shown in Table 1, Architect3D—despite being trained on the more richly annotated ScanNet++ dataset—fails to generalize effectively, yielding near-zero scores across all metrics. This trend persists when restricting the evaluation to architectural elements(Table 2). While Mask3D achieves reasonable performance when trained on ScanNet200, our approach underperforms severely, suggesting that simply swapping in a higher-quality dataset with a finer set of classes does not automatically translate to improve instance segmentation.

These results motivate a deeper analysis of why the anticipated benefits of ScanNet++ did not materialize in our framework. We explore this in section 4.2 and 4.3.

4.2. Qualitative Results

To evaluate the quality of the predicted 3D instance masks, we visualize model predictions alongside RGB inputs and

Model	$\text{mAP}_{50:95}$	AP@25	AP@50
Architect3D (architect)	1.4	2.7	5.2
Architect3D (rest)	0.6	1.3	2.3

Table 2. Quantitative results on only architectural classes and only the rest of the classes ScanNet++.

ground truth annotations for three scenes from ScanNet++ (Figure 2) using the `pyviz3d` visualization toolkit. Each row presents one scene: the RGB image is shown on the left, the ground truth instance masks in the middle, and the predictions from our model are shown on the right.

Architectural element segmentation. Despite the poor quantitative metrics in Table 2, visual inspection reveals that large structural elements such as floors and walls can be detected as coherent surfaces across scenes, though with significant issues at times. These masks often cover large portions of architectural elements but fail to maintain consistent instance boundaries. Finer architectural details such as doors and windows are either under-segmented or merged with adjacent wall regions. For instance, in the top row of Figure 2, the door on the far side of the room is not cleanly isolated from the surrounding wall despite being annotated in the ground truth.

Overlapping instances. The model has difficulty distinguishing overlapping or adjacent objects, especially in cluttered environments. In the kitchen scene (middle row), multiple small items (e.g., mugs, containers) are fused into a single coarse segment, in contrast to the more granular ground truth. Similarly, in the bedroom scene (bottom row), the bed and surrounding furniture are not cleanly separated, and occluded objects such as hanging garments are often missed or overextended. These errors suggest limited spatial precision in proposal boundaries and insufficient object-aware discrimination.

Mask granularity. Compared to the ground truth, the predicted masks have coarser geometry, particularly for objects with fine structures, such as chairs or shelves. Fine details like chair legs or cabinet edges are missing or blurred. In highly cluttered regions, both mask fragmentation and merging are observed. This issue is especially apparent in the kitchen scene, where the model is unable to accurately define the edges of densely grouped items.

More failure cases. Challenging conditions include textureless surfaces (e.g., white walls), strong occlusions, and semantically similar objects in close proximity. The model often fails to distinguish between visually and spatially adjacent elements (e.g., tables and walls, or beds and surrounding furniture), likely due to both visual ambiguity and limitations in the mask proposal quality.

Summary. While the model occasionally demonstrates some coverage for large-scale architectural components

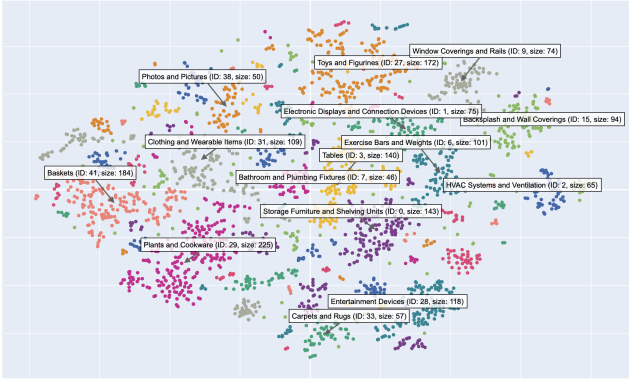

Figure 1. t-SNE [18] visualization of MPNetV2 embeddings [14] for all 2753 ScanNet++ classes. Each color represents a different cluster. We identified 109 clusters using DBSCAN [1]

when trained on ScanNet++, it severely underperforms on fine-grained segmentation and in cluttered scenes. These limitations appear to stem not just from our training procedure, but from the challenges posed by the dataset: high object density, fine-grained structural details, and subtle class differences. Such conditions likely exceed the capacity of standard instance segmentation pipelines. Addressing these challenges may require more specialized training strategies, finer masks, or architectures designed specifically for complex 3D environments.

4.3. Limitations

Despite the potential of using ScanNet++ for improving structural instance segmentation, our current approach faces several limitations:

- **Coarse voxel resolution:** Due to memory constraints and the computational budget, we used a relatively large voxel size of 0.08 cm, as opposed to the 0.02 cm of [13], which may obscure finer geometric details such as thin structural components.
- **Limited training schedule:** Due to limited computational resources, we were only able train for 250 epochs which is significantly fewer than the 600 epochs of the original Mask3D. This, combined with slow convergence, likely led to underfitting of the fine-grained classes of ScanNet++.
- **High class granularity:** The ScanNet++ dataset contains masks for a fine-grained set of 2753 instance classes. This increases class ambiguity. As previously observed in image classification tasks [11], a finer set of classes introduces problems such as punishing the model when it can't differentiate between types of the same object (e.g. *office chair* vs *chair*) as well as an imbalanced representation of classes. This problem becomes apparent when evaluating on a single object class (see Tab. 3). In this experiment

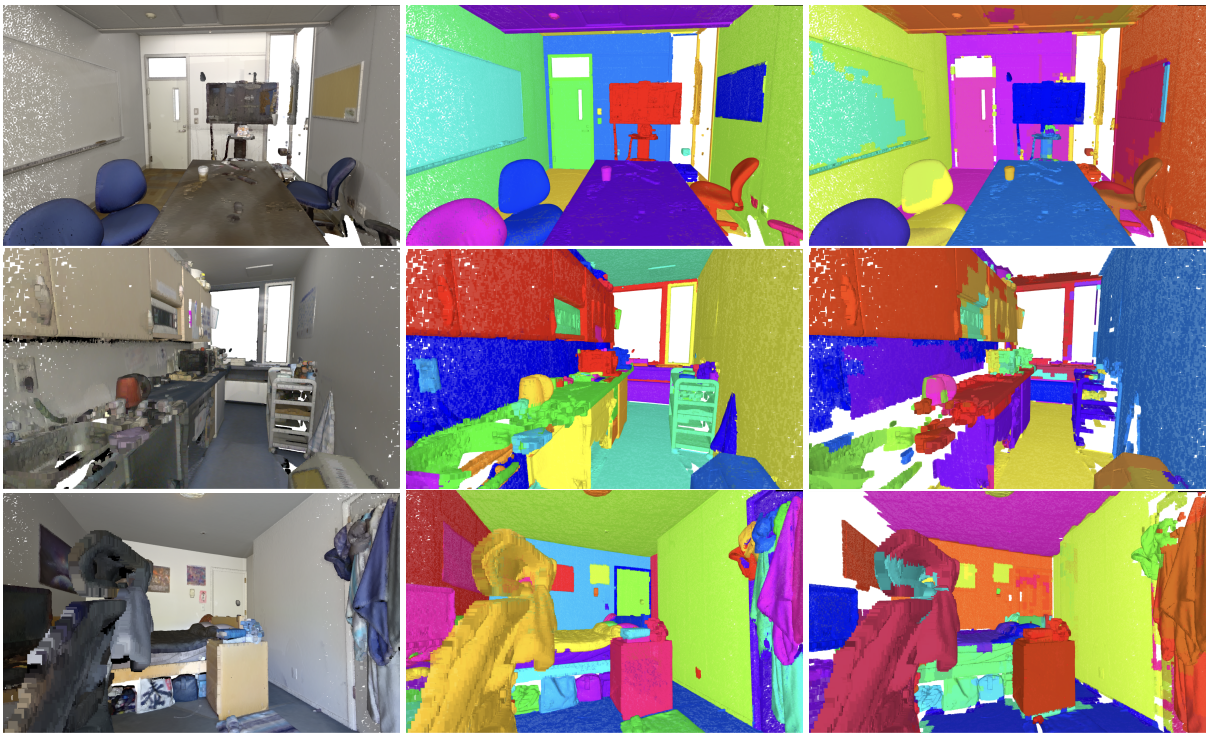

Figure 2. Visualizations for three scenes from the ScanNet++ validation set. RGB scene is on the left, the ground truth masks in the middle and the prediction of our model on the right. The predictions are not limited to the architectural elements in ScanNet++.The masks are from the following scenes: 5748ce6f01 (Top), 3864514494 (Middle), c49a8c6cff (Bottom).

we map each every predicted & ground truth instance label to the *object* instance label. This way we can measure the quality of the masks as the final score isn't influenced by the label matching. Architect3D's improves significantly, suggesting that matching the right labels to the right masks is among the issues. The model likely struggles with the increased number of similar classes that must be distinguished. In an additional experiment, we visualize the MPNetV2 [14] embeddings of all 2753 classes using t-SNE [18]. To explore the semantic structure of the embedding space, we apply DBSCAN clustering [1], which identified 109 distinct clusters. This finding suggests that many class embeddings form semantic groups, indicating redundancy or high similarity between classes. The resulting visualization is shown in Figure **??**.

- **Resource constraints:** Time and compute limitations restricted our ability to perform comprehensive hyperparameter tuning, ablations, or architectural modifications, limiting the potential of the current implementation. The original Mask3D implementation was optimized for ScanNet200, and adapting it properly to ScanNet++ requires substantial retuning.
- **Checkpoint initialization:** The Mask3D model was initialized from checkpoints trained on ScanNet200, which may have introduced bias or instability when fine-tuning on the more detailed ScanNet++ labels.

Model	**mAP**$_{50:95}$	**AP@25**	**AP@50**
Mask3D (all)	0.7	1.4	2.4
Mask3D (object)	6.7	11.8	20.1

Table 3. Assessing the mask quality of Mask3D. The first row shows the results for evaluating on all classes and the second one when mapping every class to the *object* class.

These challenges help explain the performance gap between our method and the baseline, and point to important directions for future work.

5. Conclusion

In this paper, we presented **Architect3D**, an approach to improve 3D instance segmentation on architectural elements. By replacing ScanNet200 with the more detailed ScanNet++ dataset, we hypothesized that we could improve mask proposal generation and thereby enhance segmentation performance for structural elements.

Contrary to our expectations, our approach yielded significantly worse performance than the baseline. Quantitative evaluation showed near-zero AP scores across vari-

ous metrics, while qualitative analysis revealed issues with mask granularity, instance separation, and overall segmentation quality. These negative results highlight the difficulty of adapting existing 3D segmentation approaches to a finer set of classes and background classes.

Our failure analysis identified several possible causes, including training constraints, dataset complexity, and architectural limitations. Despite these negative results, we believe our work contributes to the field by highlighting the specific challenges in extending open-vocabulary 3D segmentation to architectural elements:

- **Dataset complexity**: Although ScanNet++ provides more detailed annotations with over 2,000 instance classes, our results indicate that increasing annotation granularity without improving the model architecture can result in decreased performance.
- **Architectural priors**: Unlike general objects, architectural elements adhere to specific structural rules and relationships (e.g., walls are perpendicular to floors). Incorporating these domain-specific prior knowledge could improve the performance of segmenting these elements.
- **Multi-scale approaches**: Architectural elements span a wide range of scales, from large walls to thin columns. Future work could focus on multi-scale representations that capture both coarse structures and fine details.
- **Training efficiency**: Our limited training schedule likely contributed significantly to the poor performance. Developing more advanced, tailored training strategies for ScanNet++ could improve results [4].
- **Architecture adaptations**: The standard Mask3D architecture, originally designed for the smaller and less complex ScanNet200 dataset, may be suboptimal for the higher-resolution, more semantically rich scenes found in ScanNet++. The increased geometric and semantic complexity of ScanNet++ likely requires models with greater capacity and finer feature extraction capabilities. As suggested in [2, 3], scaling up the model enhances its ability to learn more detailed representations, potentially leading to improved performance on this more demanding dataset.

Despite the negative quantitative results, our qualitative analysis suggests that the goal of achieving robust open-vocabulary understanding of architectural elements remains valuable and achievable with appropriate architectural modifications and training strategies. The challenges we encountered underscore the necessity of domain-specific approaches that address the unique properties of architectural elements rather than applying methods optimized for general object segmentation.

References

[1] Martin Ester, Hans-Peter Kriegel, Jörg Sander, and Xiaowei Xu. A density-based algorithm for discovering clusters in large spatial databases with noise. In *Proceedings of the Second International Conference on Knowledge Discovery and Data Mining (KDD-96), Portland, Oregon, USA*, pages 226–231, 1996. 3, 4

[2] Micah Goldblum, Hossein Souri, Renkun Ni, Manli Shu, Viraj Prabhu, Gowthami Somepalli, Prithvijit Chattopadhyay, Mark Ibrahim, Adrien Bardes, Judy Hoffman, Rama Chellappa, Andrew Gordon Wilson, and Tom Goldstein. Battle of the backbones: A large-scale comparison of pretrained models across computer vision tasks. In *NeurIPS*, 2023. 5

[3] Robin Hesse, Dogukan Bagci, Bernt Schiele, Simone Schaub-Meyer, and Stefan Roth. Beyond accuracy: What matters in designing well-behaved models? *CoRR*, abs/2503.17110, 2025. 5

[4] Rui Huang, Songyou Peng, Ayça 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*, pages 278–295, 2024. 1, 2, 5

[5] Justin Kerr, Chung Min Kim, Ken Goldberg, Angjoo Kanazawa, and Matthew Tancik. LERF: language embedded radiance fields. In *IEEE/CVF International Conference on Computer Vision, ICCV 2023, Paris, France, October 1-6, 2023*, pages 19672–19682, 2023. 2

[6] Junha Lee, Chunghyun Park, Jaesung Choe, Yu-Chiang Frank Wang, Jan Kautz, Minsu Cho, and Christopher B. Choy. Mosaic3d: Foundation dataset and model for open-vocabulary 3d segmentation. *CoRR*, abs/2502.02548, 2025. 1

[7] Seungjun Lee, Yuyang Zhao, and Gim Hee Lee. Segment any 3d object with language. *arXiv preprint arXiv:2404.02157*, 2024. 2

[8] Phuc D. A. 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.

[9] Songyou Peng, Kyle Genova, Chiyu Max Jiang, Andrea Tagliasacchi, Marc Pollefeys, and Thomas A. Funkhouser. Openscene: 3d scene understanding with open vocabularies. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition, CVPR 2023, Vancouver, BC, Canada, June 17-24, 2023*, pages 815–824, 2023. 2

[10] Alec Radford, Jong Wook Kim, Chris Hallacy, Aditya Ramesh, Gabriel Goh, Sandhini Agarwal, Girish Sastry, Amanda Askell, Pamela Mishkin, Jack Clark, Gretchen Krueger, and Ilya Sutskever. Learning transferable visual models from natural language supervision. In *Proceedings of the 38th International Conference on Machine Learning, ICML 2021, 18-24 July 2021, Virtual Event*, pages 8748–8763, 2021. 2

[11] Tal Ridnik, Emanuel Ben Baruch, Asaf Noy, and Lihi Zelnik. Imagenet-21k pretraining for the masses. In *NeurIPS*, 2021. 3

[12] Dávid Rozenberszki, Or Litany, and Angela Dai. Language-grounded indoor 3d semantic segmentation in the wild. In *Computer Vision - ECCV 2022 - 17th European Conference, Tel Aviv, Israel, October 23-27, 2022, Proceedings, Part XXXIII*, pages 125–141, 2022. 1, 2

[13] Jonas Schult, Francis Engelmann, Alexander Hermans, Or Litany, Siyu Tang, and Bastian Leibe. Mask3d: Mask transformer for 3d semantic instance segmentation. In *IEEE International Conference on Robotics and Automation, ICRA 2023, London, UK, May 29 - June 2, 2023*, pages 8216–8223. IEEE, 2023. 1, 2, 3

[14] Kaitao Song, Xu Tan, Tao Qin, Jianfeng Lu, and Tie-Yan Liu. Mpnet: Masked and permuted pre-training for language understanding. In *NeurIPS 2020*, 2020. 3, 4

[15] Jiahao Sun, Chunmei Qing, Junpeng Tan, and Xiangmin Xu. Superpoint transformer for 3d scene instance segmentation. In *AAAI Conference on Artificial Intelligence*, 2022. 2

[16] Ayça Takmaz, Elisabetta Fedele, Robert W. Sumner, Marc Pollefeys, Federico Tombari, and Francis Engelmann. Openmask3d: Open-vocabulary 3d instance segmentation. In *NeurIPS 2023*, 2023. 1, 2

[17] Ayça Takmaz, Alexandros Delitzas, Robert W. Sumner, Francis Engelmann, Johanna Wald, and Federico Tombari. Search3d: Hierarchical open-vocabulary 3d segmentation. *IEEE Robotics Autom. Lett.*, 10(3):2558–2565, 2025. 1

[18] Laurens Van der Maaten and Geoffrey Hinton. Visualizing data using t-sne. *Journal of machine learning research*, 2008. 3, 4

[19] Yunhan Yang, Xiaoyang Wu, Tong He, Hengshuang Zhao, and Xihui Liu. Sam3d: Segment anything in 3d scenes, 2023. 1

[20] Chandan Yeshwanth, Yueh-Cheng Liu, Matthias Nießner, and Angela Dai. Scannet++: A high-fidelity dataset of 3d indoor scenes. In *ICCV*, pages 12–22. IEEE, 2023. 2

[21] Yingda Yin, Yuzheng Liu, Yang Xiao, Daniel Cohen-Or, Jingwei Huang, and Baoquan Chen. SAI3D: segment any instance in 3d scenes. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition, CVPR 2024, Seattle, WA, USA, June 16-22, 2024*, pages 3292–3302. IEEE, 2024. 2

[22] Xiaohua Zhai, Basil Mustafa, Alexander Kolesnikov, and Lucas Beyer. Sigmoid loss for language image pre-training. In *IEEE/CVF International Conference on Computer Vision, ICCV 2023, Paris, France, October 1-6, 2023*, pages 11941–11952, 2023. 2

[23] Zihui Zhang, Bo Yang, Bing Wang, and Bo Li. Growsp: Unsupervised semantic segmentation of 3d point clouds. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition, CVPR 2023, Vancouver, BC, Canada, June 17-24, 2023*, pages 17619–17629, 2023. 2

[24] Xiaoyu Zhu, Hao Zhou, Pengfei Xing, Long Zhao, Hao Xu, Junwei Liang, Alexander Hauptmann, Ting Liu, and Andrew Gallagher. Open-vocabulary 3d semantic segmentation with text-to-image diffusion models. In *ECCV*, 2024. 2

A. Appendix

A.1. List of architectural elements

In the following, we present a list of all 116 architectural elements:

arched ceiling, balcony door, bathroom door, bathroom floor, bathroom tiled wall, bathroom wall, brick wall, car door, ceiling camp, ceiling lapm, ceiling pipe, ceiling sprinkler, ceiling vent, ceiling ventilation panel, ceiling ventilation valve, ceiling ventilator, chimney, climbing wall, column, cubicle door, decorative pillar, decorative wall, door, door blocker, door intercom, door vent, door window, doorbell chime, doorframw, doorway, double door, dropped ceiling, duct, electric panel door, fireplace, floor cushion, floor mat, floor mounted air conditioner, floor panel, floor scrubber, floor vent, folding door, frame window screen, french door, french window, garage door, glass door, glass wall, grated ceiling, hall door, indoor crane, kitchen ceiling, kitchen floor, kitchen tiled floor, kitchen tiled wall, kitchen wall, opaque window panel, panel door, partition wall, pillar, railing, raised floor, roof, separation wall, sill, sliding door, sliding door frame, sliding door rail, sliding glass door, sliding window, stair, stair railing, stair railling, staircase, staircase railing, stairs, stall door, steel beam, structural column, structural pillar, suspended ceiling, tile wall, tiled floor, tiled wall, utility room door, vent, wall beam, wall board, wall cord cover, wall flush, wall frame, wall hanging, wall intercom, wall mounted thermostat, wall outlet, wall strip, wall telephone sockets, wall thermostat, wall unit, wall/other room, wicket door, window, window blind, window casing, window counter, window cover panel, window door, window head, window lintel, window pane, window sill, window wiper, windowframe, windows, wooden beam, wooden panel window

A.2. Work Distribution

The workload was distributed equally among group members, taking into account individual availability during different project phases (meaning that if someone was unavailable during a period, they received less work during that period, but more work during a later period). The primary tasks in this group project included, but not limited to:

- Paper presentation
- Preparing and presenting initial, intermediate, and final project updates
- Preparing meetings with supervisors
- Writing the research proposal and the final project report
- Writing code for training, evaluation and visualization

A.3. Source Code

Access to the source code is provided via Polybox. The download link is available through the ETH Moodle course for 3D Vision. The repository includes comprehensive documentation detailing the functionality of the code, usage instructions, and visualizations, along with an overview of the development process. All relevant information can be found in the `README_3DVision.md` file included in the repository. The `README_3DVision.md` file gives also a clear statement which parts of the code we coded ourselves and which part we used external code for.

A.4. Hyperparameters

In table 4 we list the important hyperparameters that were used to train our Mask3D model.

Hyperparameter	Value
General Configuration	
Voxel Size	0.08
Checkpoint	scannet200_val.ckpt
Model Architecture	
Model	Mask3D
Backbone	Res16UNet34C
Hidden Dimension	128
Feedforward Dimension	1024
Number of Queries	100
Number of Heads	8
Number of Decoders	3
Positional Encoding	Fourier
Training Parameters	
Batch Size	1
Optimizer	AdamW
Learning Rate	0.0001
Scheduler	OneCycleLR
Epochs	250
Dropout	0.0
Loss Function	
Matcher	HungarianMatcher
Class Cost	2.0
Mask Cost	5.0
Dice Cost	2.0
EOS Coefficient	0.1
Oversample Ratio	3.0
Importance Sample Ratio	0.75

Table 4. Hyperparameters for training Mask3D Instance Segmentation