3D Instance Segmentation via Multi-Task Metric Learning

Jean Lahoud
KAUST

Bernard Ghanem
KAUST

Marc Pollefeys
ETH Zurich

Martin R. Oswald
ETH Zurich

Abstract

We propose a novel method for instance label segmentation of dense 3D voxel grids[1]. We target volumetric scene representations, which have been acquired with depth sensors or multi-view stereo methods and which have been processed with semantic 3D reconstruction or scene completion methods. The main task is to learn shape information about individual object instances in order to accurately separate them, including connected and incompletely scanned objects. We solve the 3D instance-labeling problem with a multi-task learning strategy. The first goal is to learn an abstract feature embedding, which groups voxels with the same instance label close to each other while separating clusters with different instance labels from each other. The second goal is to learn instance information by densely estimating directional information of the instance's center of mass for each voxel. This is particularly useful to find instance boundaries in the clustering post-processing step, as well as, for scoring the segmentation quality for the first goal. Both synthetic and real-world experiments demonstrate the viability and merits of our approach. In fact, it achieves state-of-the-art performance on the ScanNet 3D instance segmentation benchmark [5].

1. Introduction

A central goal of computer vision research is high-level scene understanding. Recent methodological progress for 2D images makes reliable results possible for a variety of computer vision problems, including image classification [24, 44, 48], image segmentation [1, 32, 42], object detection [30, 39, 41] and instance segmentation in 2D images [9, 18, 37]. Furthermore, it is now possible to recover highly-detailed 3D geometry with low-cost depth sensors [20, 35, 47, 55] or with image-based 3D reconstruction algorithms [12, 22, 43]. Combining both these concepts, many algorithms have been developed for 3D scene and object classification [33, 45, 51], 3D object detection [26, 52], and joint 3D reconstruction and semantic labeling [4,6,7,25,49].

[1] https://sites.google.com/view/3d-instance-mtml
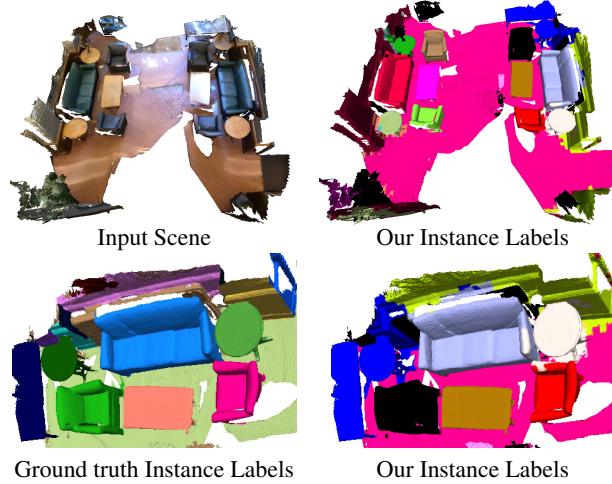
Figure 1. **Sample results of our method.** Our proposed method takes as input a 3D point cloud, and outputs instance labels unique to each object within the scene. The labels are generated by learning a metric that groups parts of the same object instance and estimates the direction towards the instance's center of mass.

Advances in 2D instance segmentation were mainly fueled by the large number of datasets and challenges available in the 2D realm. When compared to the plethora of powerful methods for instance segmentation of 2D images, the 3D counterpart problem has been less explored in the literature. In addition to the lack of datasets, the majority of 2D methods are not applicable to the 3D setting or their extension is by no means straightforward.

With the emergence of labeled datasets and benchmarks for the task of 3D instance segmentation (e.g. ScanNet [5]), numerous works have surfaced to tackle this task. In many cases, the work in 3D benefits from pioneering work in 2D, with modifications that allow processing of 3D input data. As such, this 3D processing tends to be similar to other 3D understanding techniques, mainly semantic segmentation.

In this paper, we address the problem of 3D instance segmentation. Given the 3D geometry of a scene, we want to label all the geometry that belongs to the same object with a unique label. Unlike previous methods that entangle instance labeling with semantic labeling, we propose a technique that mainly focuses on instance labeling through

grouping/clustering of information pertaining to a single object. Our method still benefits from semantic information as a local cue, but adds to it information related to 3D dimensions and 3D connectivity, whose usefulness is unique to the 3D setting.

In particular, we propose a learning algorithm that processes a 3D voxel grid and learns two main characteristics: (1) a feature descriptor unique to every instance, and (2) a direction that would point towards the instance center. Our method aims to provide a grouping force that is independent of the size of the scene and the number of instances within.

Contributions. Our contributions are two fold. *(i)* We propose a multi-task neural network architecture for 3D instance segmentation of voxel-based scene representations. In addition to a metric learning task, we task our network to predict directional information to the object's center. We demonstrate that the multi-task learning improves the results for both tasks. Our approach is robust and scalable, therefore suitable for processing large amounts of 3D data. *(ii)* Our experiments demonstrate state-of-the-art performance for 3D instance segmentation. At the time of submission, our method ranks first in terms of average AP50 score on the ScanNet 3D instance segmentation benchmark [5].

2. Related Work

This section gives a brief overview of related 2D and 3D approaches. It is worthwhile to note that a large amount of related work exists for 2D deep learning-based semantic segmentation and instance label segmentation. Recent surveys can be found in [13, 16].

2D Instance Segmentation via Object Proposals or Detection. Girshick [14] proposed a network architecture that creates region proposals as candidate object segments. In a series of followup work, this idea has been extended to be faster [41] and to additionally output pixel-accurate masks for instance segmentation [18]. The authors of YOLO [39] and its follow-up work [40] apply a grid-based approach, in which each grid cell generates an object proposal. Deep-Mask [37] learns to jointly estimate an object proposal and an object score. Lin *et al.* [30] propose a multi-resolution approach for object detection, which they call feature pyramid networks. In [17], the region proposals are refined with a network that predicts the distance to the boundary which is then transformed into a binary object mask. Khoreva *et al.* [21] jointly perform instance and semantic segmentation. A similar path follows [27], which combines fully convolutional networks for semantic segmentation with instance mask proposals. Dai *et al.* [9] use fully convolutional networks (FCNs) and split the problem into bounding box estimation, mask estimation, and object categorization and propose a multi-task cascaded network architecture. In a follow-up work [8], they combine FCNs with windowed

instance-sensitive score maps.

While all these approaches have been very successful in the 2D domain, many of them require large amounts of resources and their extension to the 3D domain is non-trivial and challenging.

2D Instance Segmentation via Metric Learning. Liang *et al.* [28] propose a method without object proposals as they directly estimate bounding box coordinates and confidences in combination with clustering as a post-processing step. Fathi *et al.* [10] compute likelihoods of pixels to belong to the same object by grouping similar pixels together within an embedding space. Bai and Urtasun [2] learn an energy map of the image in which object instances can be easily predicted. Novotny *et al.* [36] learn a position sensitive metric (semi-convolution embedding) to better distinguish between identical copies of the same object. Kong and Fowlkes [23] train a network that assigns all pixels to a spherical embedding, in which points of the same object instance are within a close vicinity and non-instance related points are placed apart from each other. The instances are then extracted via a variant of mean-shift clustering [11] that is implemented as a recurrent network. The approach by DeBrabandere *et al.* [3] follows the same idea, but the authors do not impose constraints on the shape of the embedding space. Likewise, they compute the final segmentation via mean-shift clustering in the feature space.

None of these approaches has been applied to a 3D setting. Our approach builds upon the work of DeBrabandere *et al.* [3]. We extend this method with a multi-task approach for 3D instance segmentation on dense voxel grids.

3D Instance Segmentation. Wang *et al.* [50] propose SGPN, an instance segmentation for 3D point clouds. In the first step, they extract features with PointNet [38] and subsequently build a similarity matrix, in which each element classifies whether two points belong to the same object instance. The approach is not very scalable and limited to small point cloud sizes, since the size of the similarity matrix is squared the number of points in the point cloud. Moreover, there is a number of recent concurrent or unpublished works that address 3D instance segmentation. The GSPN method [54] proposes a generative shape proposal network, which relies on object proposals to identify instances in 3D point clouds. The 3D-SIS approach [19] combines 2D and 3D features aggregated from multiple RGB-D input views. MASC [31] relies on the superior performance of the SparseConvNet [15] architecture and combines it with an instance affinity score that is estimated across multiple scales. PanopticFusion [34] predicts pixel-wise labels for RGB frames and carries them over into a 3D grid, where a fully connected CRF is used for final inference.

Apart from these recent concurrent works, there has generally been sparse research on 3D instance segmentation.
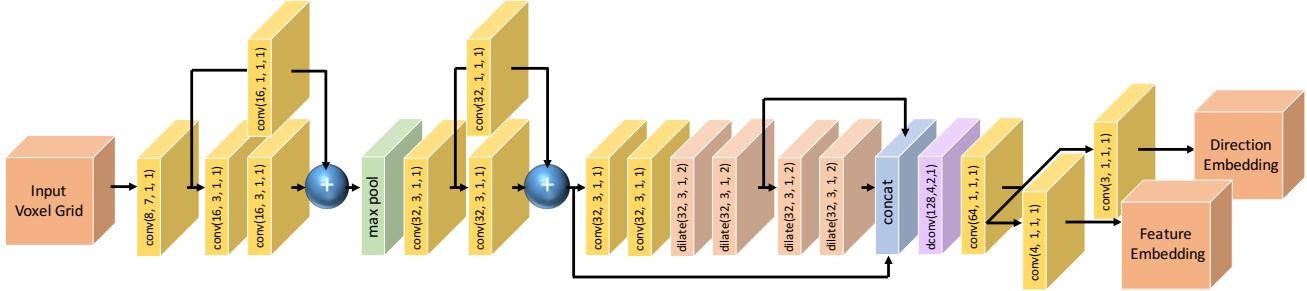
Figure 2. **Overview of our network architecture.** We cast 3D instance segmentation as a multi-task learning problem. The input to our method is a voxel grid and the output are two latent spaces: 1) a feature vector embedding that groups voxels with similar instance label close in the latent space; 2) a 3D latent space that encodes directional predictions for each voxel. The inputs and outputs of our network are visualized and explained in Fig. 3. The parameters in the figure correspond to (number of filters, kernel size, stride, dilation).

3. Method Overview

In this work, we aim at segmenting 3D instances within a given 3D scene. To fully locate a 3D instance, one would require both a semantic label and an instance label. Rather than solving the complex task of scene completion, semantic labeling and instance segmentation at once, we model our 3D instance segmentation process as a post-processing step for semantic segmentation labeling. We focus on the grouping and splitting of semantic labels, relying on inter-instance and intra-instance relations. We benefit from the real distances in 3D scenes, where sizes and distances between objects are key to the final instance segmentation.

We split our task into a label segmentation then instance segmentation problem, as we believe that features learned in each step possess task-specific information. Semantic segmentation on one hand can rely on local information to predict the class label. Learning to semantically label a volumetric representation inherently encodes features from neighboring volumes but does not require knowledge of the whole environment. On the other hand, instance segmentation requires a holistic understanding of the scene in order to join or separate semantically labeled volumes.

Problem Setting. Our method's input is a voxelized 3D space with each voxel encoding either a semantic label or a local feature vector learned through semantic labeling. In this paper, we use the semantic labeling network in [15]. We fix the voxel size to preserve 3D distances among all voxels within a scene. In problem settings where point clouds or meshes are available, one could generate a 3D voxelization by grouping information from points within every voxel. Our method then processes the voxelized 3D space and outputs instance label masks, each corresponding to a single object in the scene, along with its semantic label. The output mask can also be reprojected back into a point cloud by assigning the voxel label to all points within it.

3.1. Network Architecture

In order to process the 3D input, we utilize a 3D convolution network, which is based on the SSCNet architecture [46]. We apply some changes to the original SSCNet network to better suit our task. As shown in Figure 2, the network input and output are equally sized. Since the pooling layer scales down the scene size, we use a transpose of convolution (also referred to as deconvolution [56]) to upsample back into the original size. We also use larger dilations for diluted 3D convolution layers to increase the receptive field. We make the receptive field large enough to access all the voxels of usual indoor rooms. With a voxel size of 10cm, our receptive field is as large as 14.2m. With larger scenes, our 3D convolution network would still be applicable to the whole scene, while preserving the filter and voxel sizes, and thus preserving the real distances. Objects lying at distances larger than the receptive field are separated by default.

3.2. Multi-task Loss Function

In order to group voxels of the same instance, we aim to learn two types of feature embedding. The first type maps every voxel into a feature space, where voxels of the same instance are closer to each other than voxels belonging to different instances. This is similar to the work of DeBrabandere *et al.* [3], but applied in a 3D setting. The second type of feature embedding assigns a 3D vector to every voxel, where the vector would point towards the physical center of the object it belongs to. This enables the learning of shape containment and removes ambiguities among similar shapes.

In order to learn both feature embeddings, we introduce a multi-task loss function that is minimized during training. The first part of the loss encourages discrimination in the feature space among multiple instances, while the second part penalizes angular deviations of vectors from the desired direction.
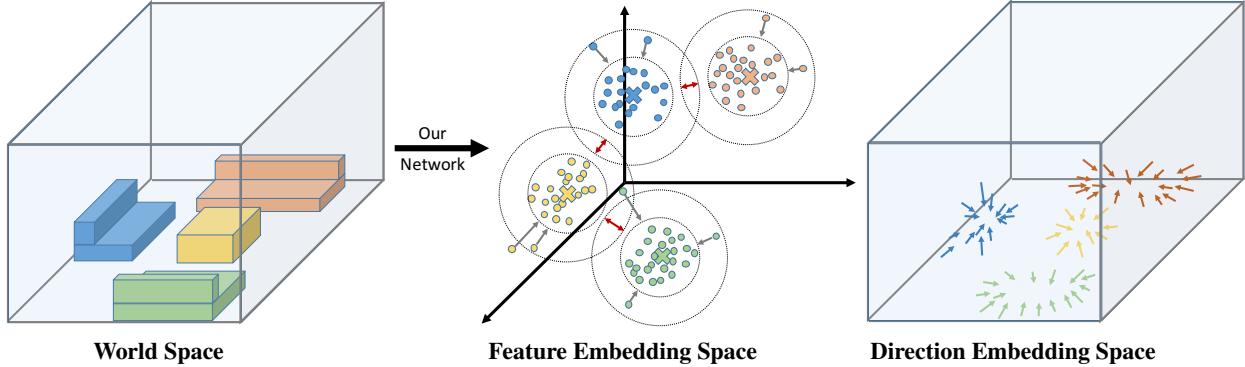
Figure 3. **Embedding space visualization.** Voxels with similar instance labels in the world space **(left)** are mapped: (1) to similar locations in the feature embedding space such that the instances form clusters **(middle)** and (2) to directional vectors pointing to the object center **(right)**. The red arrows depict inter-class push forces among cluster centers, while the grey arrows indicate intra-class pull forces of between points and cluster centers. The other colors differentiate voxels or features of different object instances.

Feature Embedding Loss. We follow the work of DeBrabandere *et al.* [3], which learns a feature embedding that can be subsequently clustered. Thus, we define the feature embedding loss as a weighted sum of three terms: **(1)** an intra-cluster variance term $\mathcal{L}_{\text{var}}$ that pulls features that should belong to the same instance towards the mean feature, **(2)** an inter-cluster distance term $\mathcal{L}_{\text{dist}}$ that encourages clusters with different instance labels to be pushed apart, and **(3)** a regularization term $\mathcal{L}_{\text{reg}}$ that pulls all features towards the origin in order to bound the activations.

$$\mathcal{L}_{\text{FE}} = \gamma_{\text{var}}\mathcal{L}_{\text{var}} + \gamma_{\text{dist}}\mathcal{L}_{\text{dist}} + \gamma_{\text{reg}}\mathcal{L}_{\text{reg}} \quad (1)$$

The individual loss functions are weighted by $\gamma_{\text{var}} = \gamma_{\text{dist}} = 1$, $\gamma_{\text{reg}} = 0.001$ and are defined similar to [3] as follows:

$$\mathcal{L}_{\text{var}} = \frac{1}{C}\sum_{c=1}^{C}\frac{1}{N_c}\sum_{i=1}^{N_c}[\|\boldsymbol{\mu}_c - \mathbf{x}_i\| - \delta_{\text{var}}]_+^2 \quad (2)$$

$$\mathcal{L}_{\text{dist}} = \frac{1}{C(C-1)}\sum_{\substack{c_A=1}}^{C}\sum_{\substack{c_B=1 \\ c_B \neq c_A}}^{C}[2\delta_{\text{dist}} - \|\boldsymbol{\mu}_{c_A} - \boldsymbol{\mu}_{c_B}\|]_+^2 \quad (3)$$

$$\mathcal{L}_{\text{reg}} = \frac{1}{C}\sum_{c=1}^{C}\|\boldsymbol{\mu}_c\| \quad (4)$$

Here C is the number of ground truth clusters, N_c denotes the number of elements in cluster c, $\boldsymbol{\mu}_c$ is the cluster center, *i.e.* the mean of the elements in cluster c, and $\mathbf{x}_i$ is a feature vector. Further, the norm $\|\cdot\|$ denotes the ℓ_2-norm and $[x]_+ = \max(0, x)$ the hinge. The parameter δ_{var} describes the maximum allowed distance between a feature vector $\mathbf{x}_i$ and the cluster center $\boldsymbol{\mu}_c$ in order to belong to cluster c. Likewise, $2\delta_{\text{dist}}$ is the minimum distance that different cluster centers should have in order to avoid overlap. A visualization of the forces and the embedding spaces can be found

in Figure 3. Feature embeddings of different clusters exert forces on each other, *i.e.* each feature embedding is affected by the number and location of other cluster centers. This connection might be disadvantageous in some cases, especially when a large number of instances exist in a single scene. Therefore, we propose next an additional loss that provides local information essential for instance separation without being affected by other instances.

Directional Loss. We here aim to generate a vector feature that would locally describe the intra-cluster relationship without being affected by other clusters. We choose the vector to be the one pointing towards the ground truth center of the object. To learn this vector feature, we attend to the following directional loss:

$$\mathcal{L}_{\text{dir}} = -\frac{1}{C}\sum_{c=1}^{C}\frac{1}{N_c}\sum_{i=1}^{N_c}\mathbf{v}_i^\top\mathbf{v}_i^{GT} \quad \text{with} \quad \mathbf{v}_i^{GT} = \frac{\mathbf{z}_i - \mathbf{z}_c}{\|\mathbf{z}_i - \mathbf{z}_c\|} \quad (5)$$

Here, $\mathbf{v}_i$ denotes the normalized directional vector feature, $\mathbf{v}_i^{GT}$ is the desired direction which points towards the object center, $\mathbf{z}_i$ is the voxel center location, and $\mathbf{z}_c$ is the object center location.

Joint Loss. We jointly minimize both the feature embedding loss and the directional loss during training. Our final joint loss reads as:

$$\mathcal{L}_{\text{joint}} = \alpha_{\text{FE}}\mathcal{L}_{\text{FE}} + \alpha_{\text{dir}}\mathcal{L}_{\text{dir}} \quad (6)$$

We use $\alpha_{\text{FE}} = 0.5$ and $\alpha_{\text{dir}} = 1$.

Post-processing. We apply mean-shift clustering [11] on the feature embedding. Similar to object detection algorithms, instance segmentation does not restrict the labeling to one coherent set, and thus allows overlap between multiple objects. We use the mean-shift clustering output with multiple thresholds as proposals that are scored according to

their direction feature consistency. We also use connected components for suggested splitting that would further be scored by the coherency of its feature embeddings. The coherency of the feature embedding is described by the number of feature embeddings that lie within a given threshold from the feature cluster center. The directional feature coherency score is simply $\mathcal{L}_{\text{dir}}$, which is the average cosine similarity between the normalized vector pointing from the voxel to the center of the object and the predicted normalized direction feature. We then sort all object proposals and perform non-maximum suppression (NMS) to remove objects that overlap by more than a threshold. The final score is obtained by appending both feature embedding scores with a score that encourages objects of regular sizes over extremely large or small objects. As for the semantic label, it is chosen to be the most occurring label among all points within the clustered voxels.

3.3. Network Training

Training Data. During training, we append flips of voxelized scenes as well as multiple orientations around the vertical axis to our training data. We pretrain our network using ground truth segmentation labels as input, with labels one-hot encoded to maintain the same sized input as training using the semantic segmentation output.

4. Results and Evaluation

Setup. Our network was implemented in Tensorflow and run with an Nvidia GTX1080Ti GPU. For the network training, we use the ADAM optimizer and a learning rate of $5e-4$ and batch size of 2. The training converged after about 100 epochs and took about 2 days. The inference time for our network is about 1s for scene sizes of 1.6M voxels.

Datasets. For experimental evaluation, we trained and tested our method on the following datasets that include real and synthetic data.
- **Synthetic Toy Dataset:** In order to validate our approach, we create a synthetic dataset with objects of different sizes and aspect ratios placed on a planar surface. We introduce 5 object shapes, where each shape is analogous to an object class in the real data. The shapes of the objects considered are shown in Figure 4. We then randomly orient and position objects on the surface plane, and randomly choose whether an object is in contact with another object. We generate 1000 scene, and split our dataset into 900 training scenes, and 100 testing scenes.
- **ScanNet [5]:** We conduct experiments on the ScanNet v2 dataset, which contains 1513 scans with 3D instance annotations. The training set contains 1201 scans, and the remaining 312 scans are used for validation. An additional 100 unlabeled scans form an evaluation test set.
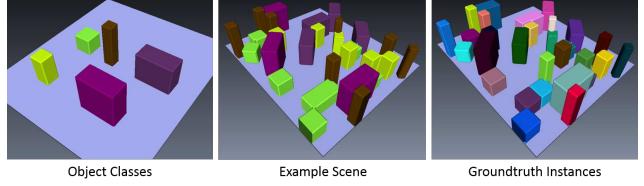

Figure 4. **Overview of the synthetic toy dataset. Left:** We consider 5 different object classes represented by cubes with various edge lengths. **Middle:** Example scene with object colors showing the class labels. **Right:** Corresponding ground truth instance labeling (randomly chosen color per instance).

Evaluation metrics. Following the evaluation procedure adopted in most instance segmentation methods as well as the ScanNet evaluation benchmark, we use the average precision metric (AP) score to evaluate our proposed algorithm. We use the AP25 and AP50 metrics, which denote the AP score with a minimum intersection-over-union (IoU) threshold of 25% and 50%, respectively. The AP score averages scores obtained with IoU thresholds ranging from 50% to 95% with a step of 5%.

Baselines. To assess the performance of our method, we consider the following baseline methods:
- **Input Segmentation:** In this case, we assume that the segmentation label, which is input to our method, to be the desired instance segmentation label. If every scene contains a single instance of every semantic label, this baseline would be ideal. In reality, these scenes barely occur, but such a metric would still serve as an inception to whether splitting and/or grouping voxels is reasonable.
- **Connected Components:** Given the ground truth segmentation labels, a connected components algorithm tends to correctly label all instances that are not touching. Since this happens seldom in a 3D setting, this is usually a high-scoring and challenging baseline.
- We further compare against submissions to the ScanNet benchmark, specifically **MaskRCNN proj** [18], **SGPN** [38], **GSPN** [54], **3D-SIS** [19], **Occipital-SCS**, **MASC** [31], **PanopticFusion** [34], and **3D-BoNet** [53].

4.1. Evaluation on Synthetic 3D Data

We evaluate our method on the simple toy dataset, and report AP50 score for all objects in Table 1. In this part, we allow only one coherent labeling. Note that the directional loss alone is not discriminative enough for subsequent clustering and is thus not considered in the ablation study. Generating object proposals from directional information only is tedious, since it is noisy and the clustering problem is much more difficult and less efficient. Therefore, we do not evaluate the directional prediction alone, but instead, we resort to using object proposals from mean shift clustering and using the directional information for scoring them.

Method	Obj1	Obj2	Obj3	Obj4	Obj5
Connected comp.	92.5	85.1	86.9	93.5	79.9
Ours (FE only)	97.3	92.7	95.0	96.4	95.2
Ours (Multi-task)	**98.0**	**93.5**	**96.1**	**96.6**	**95.3**

Table 1. **AP50 results on synthetic toy dataset.** On this dataset with 5 objects, our approach with multi-task learning as well as the baseline with only feature embedding (FE) outperform the connected components baseline, even though it uses the ground truth semantic labels. The difference between *FE only* and *Multi-task* is small in a noise-free setting.

Input Scenes with Semantic Labels   Output Scenes with Instance Labels
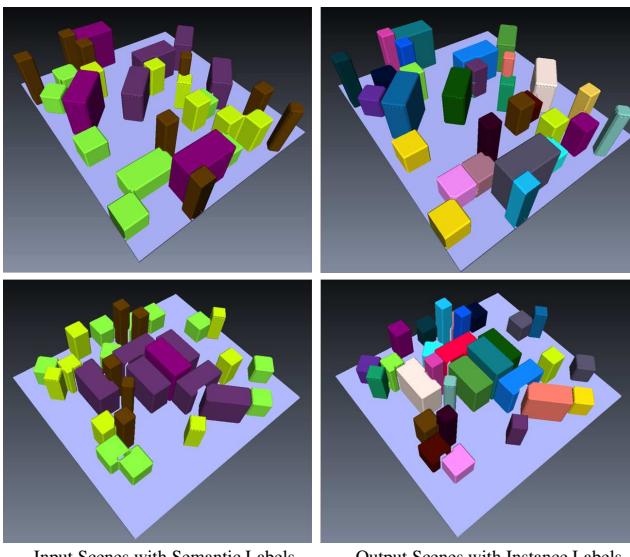
Figure 5. **Experiment on synthetic toy dataset.** Two examples of random scenes for which our network generated instance labels.

The goal of the simple toy problem in Figure 5 is to study whether the network can abstract and differentiate various object sizes although their shapes are rather similar. Furthermore, it is interesting to see how our method performs when object instances are spatially touching, especially when they belong to the same semantic class. Although the input features are very similar (due to the same object class and the spatial proximity), our network is able to successfully place the corresponding feature vectors in different locations in the feature space.

4.2. Evaluation on Real 3D Data

Feature Space Study. Minimizing the feature loss in Eq. (1) works toward two tasks: pulling points belonging to the same instance together and pushing clusters of different instances apart. Since real data contains noise, outliers, and missing data, the mapping of individual points in the feature space might be less discriminative and clusters might be overlapping. In Figure 6, we visualize the 3D feature

space in order to study these effects and observe that feature points of the same instance do indeed spread towards neighboring clusters. But for this example, the feature clustering results are not influenced and still achieve high accuracy. Note that we exclude ground and wall labels since their instance segmentation and splitting is less meaningful and is also ignored in the benchmark.

Evaluation on ScanNet Output. In Figure 7, we present qualitative results on the ScanNet dataset [5]. The results of our method on the voxel grid are simply projected onto the mesh which is then used for evaluation on the benchmark. As can be seen in the rightmost column, our method sometimes splits objects like 'desk' or the labels of 'furniture' bleed into neighboring geometry. Due to our mostly geometric approach, our method needs structural changes to recognize object boundaries and to potentially relabel a new instance. Nonetheless, our proposed method was able to group single object instances together in most cases.

In Table 2, we provide an ablation study and include comparisons against simple baselines. The first baseline uses the input segmentation labels (SparseConvNet [15]) as instance labels. Furthermore, we evaluate a simple connected component labeling method on the segmentation labeling, because in the 3D setting in general, and considering the given datasets, very few object instances are touching each other. Hence, this connected component baseline is already a challenging one especially for a rather noise free geometry and labeling. It is clear that this method tends to substantially improve the instance labeling results. With increasing amounts of noise connected component labeling rapidly performs worse. In rare cases, the results of this method get worse, which is due to the fact that the scenes are not completely scanned and a single object instance might be disconnected due to missing scene parts.

Ablation Study: Single-task vs. Multi-task. We compare our network with single-task learning to that of multi-task learning. The six rightmost columns in Table 2 show the results of single-task learning and multi-task learning. With very few exceptions, the network trained with a multi-task loss consistently outperforms the single-task one. This is in line with the results on the synthetic dataset and supports our hypothesis that the directional loss adds more discriminative features, which are helpful to group the features according to object instances in the feature space. For objects that rarely have multiple instances within a scene, such as the 'counter' class, the segmentation as instance outperforms our method. Since this occurrence is uncommon, its effect on the overall average evaluation is negligible.

Table 3 provides an overview of our benchmark results on the ScanNet test dataset (with held out ground truth). One can see that our method outperforms the others in AP50 score. Other methods include those that process all RGB-D

| Input (RGB) | Feature Label GT | Feature Label Ours | GT Label | Clustering Label |
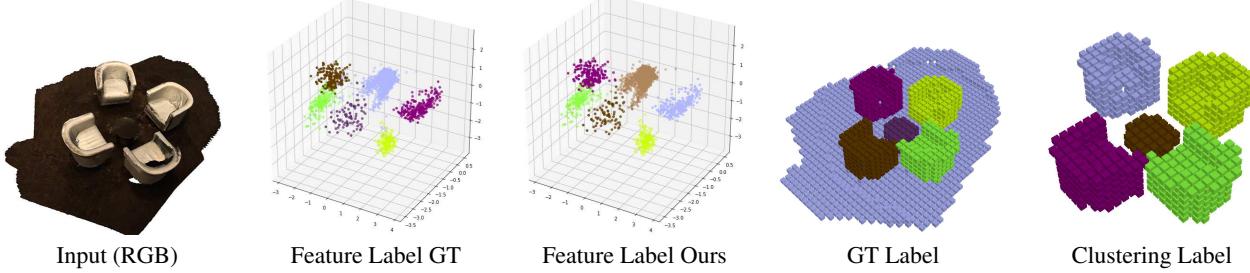
Figure 6. **Visualization of the feature embedding and labeling.** This figure shows (from left to right) the colored 3D scene input, its generated 3D feature embeddings, along with the ground truth (GT) labels and our instance labeling result after mean-shift clustering (colors of the instances in the final results are chosen randomly and do not correspond to GT label colors).

Class	Segment. [15] as Instance			Connect. Comp. on [15]			Ours (FE only)			Ours (Multi-task)		
	AP	AP50	AP25	AP	AP50	AP25	AP	AP50	AP25	AP	AP50	AP25
cabinet	0.002	0.008	0.039	0.024	0.081	0.153	0.036	0.118	**0.396**	**0.042**	**0.145**	0.346
bed	0.105	0.197	0.540	**0.200**	0.467	0.651	0.154	0.446	0.696	0.197	**0.540**	**0.806**
chair	0.000	0.001	0.027	0.138	0.239	0.434	0.475	0.689	0.814	**0.567**	**0.792**	**0.877**
sofa	0.066	0.240	0.462	0.157	0.398	0.533	0.172	0.369	0.684	0.226	**0.488**	**0.803**
table	0.027	0.061	0.160	0.154	0.324	0.428	0.207	0.361	0.593	**0.242**	**0.427**	**0.674**
door	0.019	0.037	0.070	0.041	0.073	0.108	0.142	0.304	0.429	**0.152**	**0.324**	**0.458**
window	0.015	0.023	0.023	0.020	0.031	0.037	0.113	0.258	0.423	**0.152**	**0.327**	**0.472**
bookshelf	0.013	0.024	0.187	0.077	0.198	**0.453**	0.075	0.175	0.423	**0.080**	**0.219**	**0.453**
picture	0.001	0.005	0.005	0.001	0.005	0.008	0.028	0.067	0.169	**0.044**	**0.109**	**0.198**
counter	0.007	0.032	0.216	**0.008**	**0.034**	**0.266**	0.001	0.004	0.094	0.001	0.008	0.097
desk	0.012	0.057	0.211	0.022	0.109	0.364	0.011	0.053	0.327	**0.031**	**0.142**	**0.499**
curtain	0.034	0.085	0.185	0.081	0.173	0.225	0.114	0.285	0.450	**0.174**	**0.399**	**0.542**
refrigerator	0.059	0.112	0.211	0.105	0.162	0.225	0.124	0.302	0.317	**0.185**	**0.421**	**0.441**
shower curtain	0.119	0.231	0.231	0.128	0.227	0.284	0.392	0.593	0.710	**0.402**	**0.643**	**0.749**
toilet	0.326	0.676	0.701	0.575	0.801	0.801	**0.636**	0.962	0.977	0.625	**0.965**	**0.980**
sink	0.048	0.130	0.328	0.054	0.135	0.307	0.094	0.294	0.397	**0.120**	**0.364**	**0.445**
bathtub	0.357	0.677	0.677	**0.319**	0.631	0.700	0.235	0.553	0.674	0.311	**0.708**	**0.794**
otherfurniture	0.004	0.010	0.039	0.021	0.052	0.107	0.061	0.154	0.283	**0.097**	**0.215**	**0.335**
average	0.068	0.145	0.239	0.118	0.230	0.338	0.171	0.333	0.492	**0.203**	**0.402**	**0.554**

Table 2. **Ablation study on the ScanNet dataset [5] validation set.** We show the instance labeling performance of the segmentation method in [15], connected components labeling on the [15] segmentation, our method with feature embedding (FE) only and our method with multi-task learning.

Method	Avg AP	bathtub	bed	bookshelf	cabinet	chair	counter	curtain	desk	door	otherfurniture	picture	refrigerator	shower curtain	sink	sofa	table	toilet	window
MTML (Ours)	**0.55**	**1.00**	**0.81**	0.59	0.33	**0.65**	0.00	**0.82**	0.18	**0.42**	0.36	0.18	0.45	**1.00**	0.44	0.69	0.57	**1.00**	0.40
Occipital-SCS	0.51	**1.00**	0.72	0.51	**0.51**	0.61	0.09	0.60	0.18	0.35	0.38	0.17	0.44	0.85	0.39	0.62	0.54	0.89	0.39
3D-BoNet	0.49	**1.00**	0.67	0.59	0.30	0.48	0.10	0.62	**0.31**	0.34	0.26	0.13	0.43	0.80	0.40	0.50	0.51	0.91	**0.44**
PanopticFusion [34]	0.48	0.67	0.71	**0.60**	0.26	0.55	0.00	0.61	0.18	0.25	**0.43**	**0.44**	0.41	0.86	**0.49**	0.59	0.27	0.94	0.36
ResNet-backbone [29]	0.46	**1.00**	0.74	0.16	0.26	0.59	**0.14**	0.48	0.22	0.42	0.41	0.13	0.32	0.71	0.41	0.54	**0.59**	0.87	0.30
MASC [31]	0.45	0.53	0.56	0.38	0.38	0.63	0.00	0.51	0.26	0.36	0.43	0.33	**0.45**	0.57	0.37	0.64	0.39	0.98	0.28
3D-SIS [19]	0.38	**1.00**	0.43	0.25	0.19	0.58	0.01	0.26	0.03	0.32	0.24	0.08	0.42	0.86	0.12	**0.70**	0.27	0.88	0.24
Unet-backbone [29]	0.32	0.67	0.72	0.23	0.19	0.48	0.01	0.22	0.07	0.20	0.17	0.11	0.12	0.44	0.2	0.62	0.36	0.92	0.09
R-PointNet [54]	0.31	0.50	0.41	0.31	0.35	0.59	0.05	0.07	0.13	0.28	0.29	0.03	0.22	0.21	0.33	0.40	0.28	0.82	0.25
3D-BEVIS	0.25	0.67	0.57	0.08	0.04	0.39	0.03	0.04	0.10	0.10	0.03	0.03	0.10	0.38	0.13	0.60	0.18	0.85	0.17
Seg-Cluster	0.22	0.37	0.34	0.29	0.11	0.33	0.03	0.28	0.09	0.11	0.11	0.01	0.08	0.32	0.11	0.31	0.30	0.59	0.12
SGPN [50]	0.14	0.21	0.39	0.17	0.07	0.28	0.03	0.07	0.00	0.09	0.04	0.02	0.03	0.00	0.11	0.35	0.17	0.44	0.14
MaskRCNN proj	0.06	0.33	0.00	0.00	0.05	0.00	0.00	0.02	0.00	0.05	0.02	0.24	0.07	0.00	0.01	0.11	0.02	0.11	0.01

Table 3. **State-of-the-art comparison on the ScanNet 3D instance segmentation dataset [5].** The table shows the AP50 score of individual semantic categories and the average score (sorted by avg AP50 score in descending order). We achieve the best average score.

images that were used to reconstruct the scenes of ScanNet. Instance labels of single RGB-D frames in these methods are propagated throughout the whole scene and concatenated based on the location estimation. On the other hand, our method directly operates in the 3D setting, without the need to use the 2D information. This leads to much faster processing on the 3D scenes, and requires substantially less information to extract the 3D object instance segmentations.

| Input (RGB) | Semantic GT | SPC [15] | CC | SGPN [50] | Instance GT | Ours |
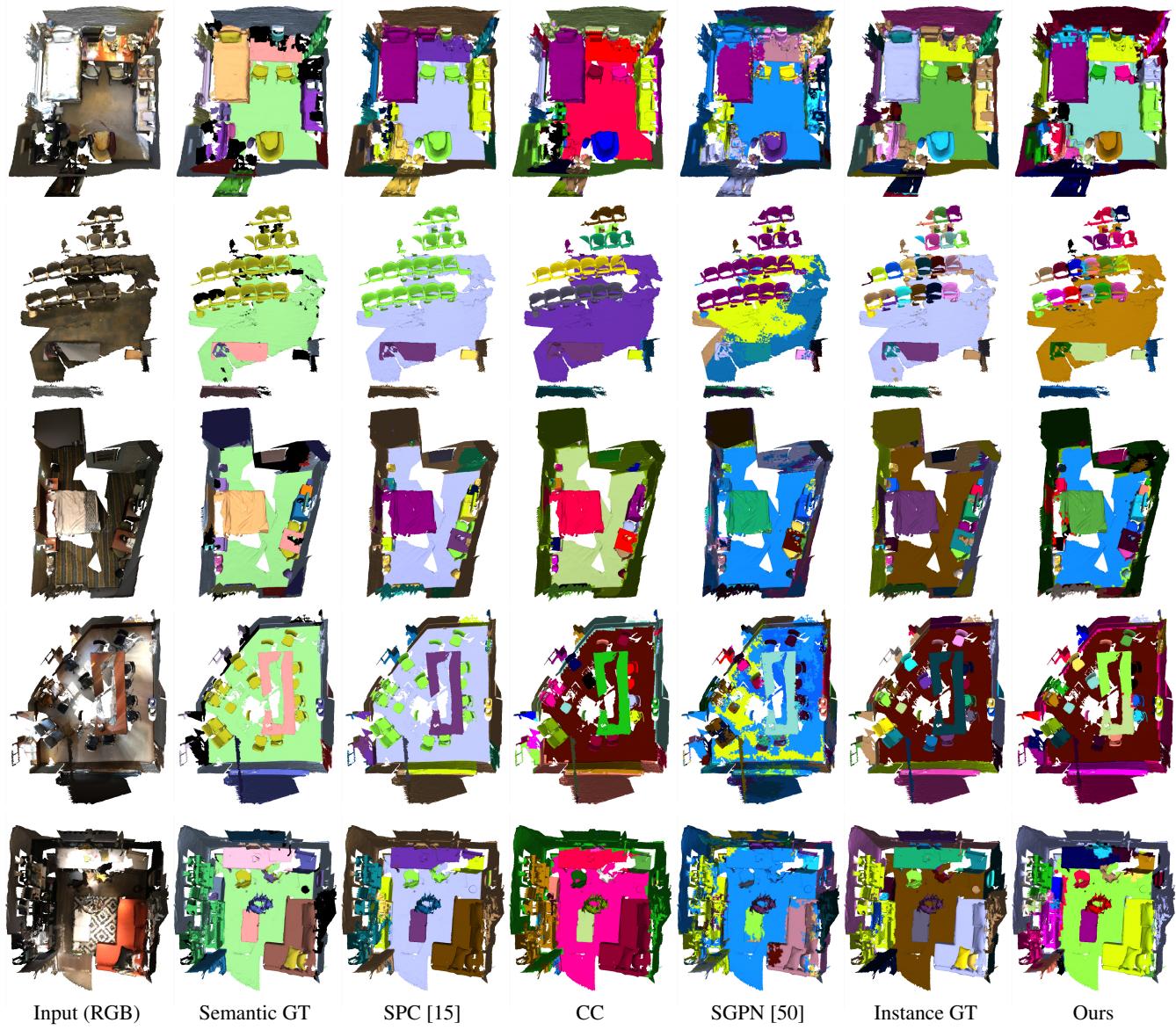
Figure 7. **Qualitative results of our method on the ScanNet validation dataset [5].** This figure shows the original input scene as a textured mesh, the semantic labeling results of SparseConvNet (SPC) [15] which we use as input and our instance labeling results as well as the semantic groundtruth (GT). We further show multiple 3D instance segmentation baselines: connected component (CC) labeling on the SPC semantic labeling, SGPN [50], and the groundtruth instance labels next to our labeling results.

5. Conclusion

We proposed a method for 3D instance segmentation of voxel-based scenes. Our approach is based on metric learning and the first part assigns all voxels belonging to the same object instance feature vectors that are in close vicinity. Conversely, voxels belonging to different object instances are assigned features that are further apart from each other in the feature space. The second part estimates directional information of object centers, which is used to score the segmentation results generated by the first part.

References

[1] Vijay Badrinarayanan, Alex Kendall, and Roberto Cipolla. Segnet: A deep convolutional encoder-decoder architecture for image segmentation. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 39(12):2481–2495, 2017.

[2] Min Bai and Raquel Urtasun. Deep watershed transform for instance segmentation. In *Proc. International Conference on Computer Vision and Pattern Recognition (CVPR)*, 2017.

[3] Bert De Brabandere, Davy Neven, and Luc Van Gool. Semantic instance segmentation with a discriminative loss function. *CoRR*, abs/1708.02551, 2017.

[4] Ian Cherabier, Johannes L. Schönberger, Martin R. Oswald, Marc Pollefeys, and Andreas Geiger. Learning priors for semantic 3d reconstruction. In *Proc. European Conference on Computer Vision (ECCV)*, September 2018.

[5] Angela Dai, Angel X. Chang, Manolis Savva, Maciej Halber, Thomas Funkhouser, and Matthias Nießner. Scannet: Richly-annotated 3d reconstructions of indoor scenes. In *Proc. International Conference on Computer Vision and Pattern Recognition (CVPR)*, 2017.

[6] Angela Dai and Matthias Nießner. 3dmv: Joint 3d-multi-view prediction for 3d semantic scene segmentation. In *Proc. European Conference on Computer Vision (ECCV)*, pages 458–474, 2018.

[7] Angela Dai, Daniel Ritchie, Martin Bokeloh, Scott Reed, Jrgen Sturm, and Matthias Niener. Scancomplete: Large-scale scene completion and semantic segmentation for 3d scans. In *Proc. International Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2018.

[8] Jifeng Dai, Kaiming He, Yi Li, Shaoqing Ren, and Jian Sun. Instance-sensitive fully convolutional networks. In Bastian Leibe, Jiri Matas, Nicu Sebe, and Max Welling, editors, *Proc. European Conference on Computer Vision (ECCV)*, pages 534–549, Cham, 2016. Springer International Publishing.

[9] Jifeng Dai, Kaiming He, and Jian Sun. Instance-aware semantic segmentation via multi-task network cascades. In *Proc. International Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 3150–3158, 2016.

[10] Alireza Fathi, Zbigniew Wojna, Vivek Rathod, Peng Wang, Hyun Oh Song, Sergio Guadarrama, and Kevin P. Murphy. Semantic instance segmentation via deep metric learning. *CoRR*, abs/1703.10277, 2017.

[11] K. Fukunaga and L. Hostetler. The estimation of the gradient of a density function, with applications in pattern recognition. *IEEE Transactions on Information Theory*, 21(1):32–40, January 1975.

[12] Yasutaka Furukawa and Jean Ponce. Accurate, dense, and robust multiview stereopsis. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 32(8):1362–1376, 2010.

[13] Alberto Garcia-Garcia, Sergio Orts-Escolano, Sergiu Oprea, Victor Villena-Martinez, and Jose Garcia-Rodriguez. A Review on Deep Learning Techniques Applied to Semantic Segmentation. *ArXiv e-prints*, April 2017.

[14] Ross B. Girshick. Fast R-CNN. In *Proc. International Conference on Computer Vision (ICCV)*, pages 1440–1448, 2015.

[15] Benjamin Graham, Martin Engelcke, and Laurens van der Maaten. 3d semantic segmentation with submanifold sparse convolutional networks. *Proc. International Conference on Computer Vision and Pattern Recognition (CVPR)*, 2018.

[16] Yanming Guo, Yu Liu, Theodoros Georgiou, and Michael S. Lew. A review of semantic segmentation using deep neural networks. *International Journal of Multimedia Information Retrieval*, Nov 2017.

[17] Zeeshan Hayder, Xuming He, and Mathieu Salzmann. Boundary-aware instance segmentation. In *Proc. International Conference on Computer Vision and Pattern Recognition (CVPR)*.

[18] Kaiming He, Georgia Gkioxari, Piotr Dollár, and Ross B. Girshick. Mask R-CNN. In *Proc. International Conference on Computer Vision (ICCV)*, pages 2980–2988, 2017.

[19] Ji Hou, Angela Dai, and Matthias Nießner. 3d-sis: 3d semantic instance segmentation of rgb-d scans. *Proc. International Conference on Computer Vision and Pattern Recognition (CVPR)*, 2019.

[20] Shahram Izadi, Richard A. Newcombe, David Kim, Otmar Hilliges, David Molyneaux, Steve Hodges, Pushmeet Kohli, Jamie Shotton, Andrew J. Davison, and Andrew W. Fitzgibbon. Kinectfusion: real-time dynamic 3d surface reconstruction and interaction. In *International Conference on Computer Graphics and Interactive Techniques, SIGGRAPH 2011, Vancouver, BC, Canada, August 7-11, 2011, Talks Proceedings*, page 23, 2011.

[21] Anna Khoreva, Rodrigo Benenson, Jan Hendrik Hosang, Matthias Hein, and Bernt Schiele. Simple does it: Weakly supervised instance and semantic segmentation. In *Proc. International Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 1665–1674, 2017.

[22] Kalin Kolev, Maria Klodt, Thomas Brox, and Daniel Cremers. Continuous global optimization in multiview 3d reconstruction. *International Journal of Computer Vision*, 84(1):80–96, 2009.

[23] Shu Kong and Charless C. Fowlkes. Recurrent pixel embedding for instance grouping. In *Proc. International Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 9018–9028, 2018.

[24] Alex Krizhevsky, Ilya Sutskever, and Geoffrey E. Hinton. Imagenet classification with deep convolutional neural networks. In *Advances in Neural Information Processing Systems 25: 26th Annual Conference on Neural Information Processing Systems 2012.*, pages 1106–1114, 2012.

[25] Abhijit Kundu, Yin Li, Frank Dellaert, Fuxin Li, and James M. Rehg. Joint semantic segmentation and 3d reconstruction from monocular video. In *Proc. European Conference on Computer Vision (ECCV)*, pages 703–718. Springer, 2014.

[26] Jean Lahoud and Bernard Ghanem. 2d-driven 3d object detection in rgb-d images. In *Proc. International Conference on Computer Vision (ICCV)*, pages 4622–4630, 2017.

[27] Yi Li, Haozhi Qi, Jifeng Dai, Xiangyang Ji, and Yichen Wei. Fully convolutional instance-aware semantic segmentation. In *Proc. International Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 4438–4446, 2017.

[28] Xiaodan Liang, Liang Lin, Yunchao Wei, Xiaohui Shen, Jianchao Yang, and Shuicheng Yan. Proposal-free network for instance-level semantic object segmentation. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, PP(99):1–1, 2017.

[29] Zhidong Liang, Ming Yang, and Chunxiang Wang. 3d graph embedding learning with a structure-aware loss function for point cloud semantic instance segmentation. *arXiv preprint arXiv:1902.05247*, 2019.

[30] Tsung-Yi Lin, Piotr Dollár, Ross B. Girshick, Kaiming He, Bharath Hariharan, and Serge J. Belongie. Feature pyramid networks for object detection. In *Proc. International Conference on Computer Vision and Pattern Recognition (CVPR)*.

[31] Chen Liu and Yasutaka Furukawa. Masc: Multi-scale affinity with sparse convolution for 3d instance segmentation. *arXiv preprint arXiv:1902.04478*, 2019.

[32] Jonathan Long, Evan Shelhamer, and Trevor Darrell. Fully convolutional networks for semantic segmentation. In *Proc. International Conference on Computer Vision and Pattern Recognition (CVPR)*.

[33] Daniel Maturana and Sebastian Scherer. Voxnet: A 3d convolutional neural network for real-time object recognition. In *IEEE/RSJ International Conference on Intelligent Robots and Systems*, Pittsburgh, PA, September 2015.

[34] Gaku Narita, Takashi Seno, Tomoya Ishikawa, and Yohsuke Kaji. Panopticfusion: Online volumetric semantic mapping at the level of stuff and things. *arXiv preprint arXiv:1903.01177*, 2019.

[35] Matthias Nießner, Michael Zollhöfer, Shahram Izadi, and Marc Stamminger. Real-time 3d reconstruction at scale using voxel hashing. *ACM Trans. Graph.*, 32(6):169:1–169:11, 2013.

[36] David Novotný, Samuel Albanie, Diane Larlus, and Andrea Vedaldi. Semi-convolutional operators for instance segmentation. In *Proc. European Conference on Computer Vision (ECCV)*, pages 89–105, 2018.

[37] Pedro H. O. Pinheiro, Ronan Collobert, and Piotr Dollár. Learning to segment object candidates. In *Advances in Neural Information Processing Systems 28: Annual Conference on Neural Information Processing Systems 2015, December 7-12, 2015, Montreal, Quebec, Canada*, pages 1990–1998, 2015.

[38] Charles Ruizhongtai Qi, Hao Su, Kaichun Mo, and Leonidas J. Guibas. Pointnet: Deep learning on point sets for 3d classification and segmentation. In *Proc. International Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 77–85, 2017.

[39] Joseph Redmon, Santosh Kumar Divvala, Ross B. Girshick, and Ali Farhadi. You only look once: Unified, real-time object detection. In *Proc. International Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 779–788, 2016.

[40] Joseph Redmon and Ali Farhadi. YOLO9000: better, faster, stronger. In *Proc. International Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 6517–6525, 2017.

[41] Shaoqing Ren, Kaiming He, Ross B. Girshick, and Jian Sun. Faster R-CNN: towards real-time object detection with region proposal networks. In *Advances in Neural Information Processing Systems 28: Annual Conference on Neural Information Processing Systems 2015, December 7-12, 2015, Montreal, Quebec, Canada*, pages 91–99, 2015.

[42] Olaf Ronneberger, Philipp Fischer, and Thomas Brox. U-net: Convolutional networks for biomedical image segmentation. In *Medical Image Computing and Computer-Assisted Intervention - MICCAI 2015 - 18th International Conference Munich, Germany, October 5 - 9, 2015, Proceedings, Part III*, pages 234–241, 2015.

[43] Johannes Lutz Schönberger, Enliang Zheng, Marc Pollefeys, and Jan-Michael Frahm. Pixelwise view selection for unstructured multi-view stereo. In *Proc. European Conference on Computer Vision (ECCV)*, 2016.

[44] Karen Simonyan and Andrew. Zisserman. Very deep convolutional networks for large-scale image recognition. In *International Conference on Learning Representations*, 2015.

[45] Richard Socher, Brody Huval, Bharath Putta Bath, Christopher D. Manning, and Andrew Y. Ng. Convolutional-recursive deep learning for 3d object classification. In *Advances in Neural Information Processing Systems 25: 26th Annual Conference on Neural Information Processing Systems 2012.*, pages 665–673, 2012.

[46] Shuran Song, Fisher Yu, Andy Zeng, Angel X. Chang, Manolis Savva, and Thomas A. Funkhouser. Semantic scene completion from a single depth image. In *Proc. International Conference on Computer Vision and Pattern Recognition (CVPR)*, 2017.

[47] Frank Steinbrücker, Christian Kerl, and Daniel Cremers. Large-scale multi-resolution surface reconstruction from RGB-D sequences. In *Proc. International Conference on Computer Vision (ICCV)*, pages 3264–3271, 2013.

[48] Christian Szegedy, Wei Liu, Yangqing Jia, Pierre Sermanet, Scott E. Reed, Dragomir Anguelov, Dumitru Erhan, Vincent Vanhoucke, and Andrew Rabinovich. Going deeper with convolutions. In *Proc. International Conference on Computer Vision and Pattern Recognition (CVPR)*.

[49] Keisuke Tateno, Federico Tombari, Iro Laina, and Nassir Navab. CNN-SLAM: real-time dense monocular SLAM with learned depth prediction. In *Proc. International Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 6565–6574, 2017.

[50] Weiyue Wang, Ronald Yu, Qiangui Huang, and Ulrich Neumann. Sgpn: Similarity group proposal network for 3d point cloud instance segmentation. In *Proc. International Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2018.

[51] Zhirong Wu, Shuran Song, Aditya Khosla, Fisher Yu, Linguang Zhang, Xiaoou Tang, and Jianxiong Xiao. 3d shapenets: A deep representation for volumetric shapes. In *Proc. International Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 1912–1920, 2015.

[52] Bin Yang, Wenjie Luo, and Raquel Urtasun. Pixor: Real-time 3d object detection from point clouds. In *Proc. International Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2018.

[53] Bo Yang, Jianan Wang, Ronald Clark, Qingyong Hu, Sen Wang, Andrew Markham, and Niki Trigoni. Learning object bounding boxes for 3d instance segmentation on point clouds. *arXiv preprint arXiv:1906.01140*, 2019.

[54] Li Yi, Wang Zhao, He Wang, Minhyuk Sung, and Leonidas Guibas. Gspn: Generative shape proposal network for 3d instance segmentation in point cloud. *Proc. International Conference on Computer Vision and Pattern Recognition (CVPR)*, 2019.

[55] Christopher Zach, Thomas Pock, and Horst Bischof. A globally optimal algorithm for robust tv-l1 range image integration. In *Proc. International Conference on Computer Vision (ICCV)*, pages 1–8, 2007.

[56] Matthew D Zeiler, Dilip Krishnan, Graham W Taylor, and Robert Fergus. Deconvolutional networks. *Proc. International Conference on Computer Vision and Pattern Recognition (CVPR)*, 2010.