Multi-Modality Task Cascade for 3D Object Detection

Jinhyung Park
jinhyun1@andrew.cmu.edu
Xinshuo Weng
xinshuow@cs.cmu.edu
Yunze Man
yman@cs.cmu.edu
Kris Kitani
kkitani@cs.cmu.edu

Robotics Institute
Carnegie Mellon University
Pittsburgh, PA

Abstract

Point clouds and RGB images are naturally complementary modalities for 3D visual understanding - the former provides sparse but accurate locations of points on objects, while the latter contains dense color and texture information. Despite this potential for close sensor fusion, many methods train two models in isolation and use simple feature concatenation to represent 3D sensor data. This separated training scheme results in potentially sub-optimal performance and prevents 3D tasks from being used to benefit 2D tasks that are often useful on their own. To provide a more integrated approach, we propose a novel Multi-Modality Task Cascade network (MTC-RCNN) that leverages 3D box proposals to improve 2D segmentation predictions, which are then used to further refine the 3D boxes. We show that including a 2D network between two stages of 3D modules significantly improves both 2D and 3D task performance. Moreover, to prevent the 3D module from over-relying on the overfitted 2D predictions, we propose a dual-head 2D segmentation training and inference scheme, allowing the second 3D module to learn to interpret imperfect 2D segmentation predictions. Evaluating our model on the challenging SUN RGB-D dataset, we improve upon state-of-the-art results of both single modality and fusion networks by a large margin (**+3.8** mAP@0.5). Code will be released at https://github.com/Divadi/MTC_RCNN.

1 Introduction

3D detection requires precise localization of objects in three-dimensional space, which is made difficult by the inherent sparsity and noise present in point clouds when using LiDAR as the sole input source. Many objects have only a few points captured on them due to either distance, occlusion, or reflectivity, causing structurally similar objects to be indistinguishable in the point cloud. On the other hand, RGB images have a far higher resolution than point clouds, containing a dense array of color and texture cues. Objects with only a few points in 3D can occupy a magnitude more pixels in 2D, allowing richer semantic features to be extracted for these objects than can be extracted from 3D. Conversely, features with 3D structural information obtained from spatially reasoning on point clouds can in turn complement RGB images as well, which lack explicit depth information. These observations support our intuition that point clouds and RGB images are *mutually* beneficial modalities.

Recent work proposes different methods of fusing image and point cloud semantics for 3D detection. Some works use a pre-trained 2D detector to generate initial frustum-based region proposals [28, 39]. Alternatively, instead of constraining the 3D search space, other methods propose to use 2D features to enrich point features. Some methods fuse semantics obtained at the end of a 2D network, either using 2D task predictions [25, 30, 37, 41] or using 2D backbone features [34, 44]. Another line of work [11, 18] fuse features from every layer of the 2D network. Despite demonstrating improvements over 3D-only methods, the image and point cloud fusion methods for 3D detection in prior work exploit only half of the mutually beneficial relationship between these two modalities. These methods only extract 2D semantics to benefit 3D tasks and do not consider using 3D semantics for better 2D feature extraction. We argue that by additionally including 3D task predictions as input to a 2D image network, the resulting improved 2D semantics can better benefit the 3D task.

In this work, to leverage the cyclic, mutually complementary relationship between images and point clouds, we propose a new **M**ulti-modality **T**ask **C**ascade network for 3D object detection (**MTC-RCNN**). The key idea of our approach is to include a 2D segmentation network [2] *between* the first and second stages of a 3D detection network allowing it to both *benefit from* and *improve* the 3D detection modules. This 2D network takes as input both the 2D RGB-D image as well as the 3D-to-2D projected point-level semantic and geometric features. Given a point, we obtain its features from the 3D proposal box it is in, using both the 3D box's class prediction and its box parameters. From our experiments, we observe that fusing these 3D features into the 2D network improves both 2D segmentation performance as well as the quality of the final 3D boxes. Further, the 2D network is supervised by ground truth generated from 3D box annotations, requiring no additional labels.

After obtaining the 2D segmentation predictions, we refine the 3D proposal boxes via a simple second stage 3D network inspired by [17]. For each 3D proposal box, the points within it are projected onto the 2D segmentation predictions, and the corresponding channel-wise probability distribution is concatenated with the point's 3D box-based geometric features. A PointNet [26] processes the resulting point features and refines the 3D proposal.

Different from some other multi-modality methods [30, 37, 41], we jointly optimize the 2D and 3D networks, allowing them to learn better shared feature representations. However, we find that the 2D network tends to overfit faster than the 3D network, quickly converging to perfect segmentations. This causes the 2D segmentation predictions to dominate the 3D features in the 2^{nd} stage 3D network during training. To alleviate this issue, we propose to instead fuse the segmentation predictions of a weaker, auxiliary head during training to allow the 2^{nd} stage 3D network to learn to interpret imperfect 2D segmentation predictions. During testing, an ensemble of the auxiliary head and the DeepLabV3+ head [2] is used.

Finally, we find that our framework is particularly well-suited for PointPainting [37]. The addition of a 2D segmentation network before the first 3D network generates better proposals that not only directly benefit the 3D detection task but also improves 2D segmentation predictions which further improves 3D box refinement.

We evaluate our approach on the difficult SUN RGB-D dataset [35]. Our models offer significant gains over our two-stage 3D-only baseline (+5.1 AP@0.25, +3.0 AP@0.50), validating the importance of adding a 2D segmentation network between the 1^{st} and 2^{nd} stage 3D modules. We also provide ablations, investigating the difficulties of multi-modality training and justifying our design choices. MTC-RCNN improves upon state-of-the-art results of both single modality and fusion networks (**+1.2 AP@0.25**, **+3.8 AP@0.5**).

The contributions of this paper can be summarized as follows:

- Our novel image and point cloud fusion network fully leverages both directions of the

mutually beneficial relationship between the two modalities.

- We investigate the difficulties of multi-modality training and propose a new direction of limiting the performance of one modality during training.
- Our method not only achieves new SOTA performance in 3D object detection on the SUN RGB-D dataset but also yields 2D segmentation predictions significantly better than a 2D-only baseline.

2 Related Work

3D Object Detection on Point Clouds. To address the irregularity and sparsity of point clouds, one direction of research proposes to organize the points into either 2D or 3D grids to be further processed by CNNs. The early work MV3D [3] processes multiple 2D projections and fuses them at a proposal level. VoxelNet [46] instead works with 3D voxels, using a 3D CNN to extract features. Followup methods [5, 9, 32, 42] exploit the sparsity of point clouds, only performing convolution operations on non-empty voxels. Following the works PointNet [26] and PointNet++ [27], another line of research directly processes point clouds. PointRCNN [31] generates 3D proposal boxes from predicted foreground points, while [29, 43] use a "voting" mechanism to shift candidate points closer to object centers. To take advantage of both the efficiency of voxel-based methods and precision of point-based methods, some recent works use both [16, 22, 24, 36]. These 3D detection methods achieve great success *without using RGB images*. To further improve performance over geometry-only methods, our work proposes a new method of fusing 2D images and point clouds.

Point Cloud and Image Fusion-based 3D Object Detection. Early methods seeking to leverage RGB images for 3D detection were 2D-driven, using mature 2D detectors to constrain the 3D search space before regressing 3D boxes [15, 28, 39]. Observing that the performance of this paradigm is upper bounded by the 2D detector, other methods instead use image semantics to enhance 3D features. MV3D [3] and AVOD [14] extract image features for 3D proposals, while [25] proposes to flexibly fuse 2D and 3D proposals. Methods can further be divided by whether they fuse features from multiple intermediate layers [11, 18], features from the end of a 2D model [3, 14, 34, 44], or 2D task-level predictions [30, 37, 41]. UberATG-MMF [19] uses all of the above. Our method is most closely related to works that fuse 2D task-level predictions, but we differ from prior methods in critical areas. First, most methods choose to freeze the 2D model [28, 30, 37, 41] when training the 3D detector, but we train end-to-end. Second, our 2D network takes 3D proposals as input to improve the 2D task predictions to further improve 3D box refinement.

Multi-Modality Fusion Training. Outside of detection, multi-modal fusion has been investigated in domains of visual question answering [1], action recognition [13], and acoustic event detection [8]. A recent work [38] analyzes some of the difficulties of multi-modal training, finding that naively incorporating multiple modalities can result in a drop in performance compared to single-modality networks, due to the fact that different modalities can hinder each other from properly training. We observe a similar but different challenge in our work, as [38] deals with late-stage concatenation of features of different modalities for a single task, while our method has a cascade of modalities for multiple tasks. In our work, we propose to resolve this issue by *inhibiting* the performance of one modality on the train set so later modules in the cascade can learn to work with imperfect intermediate task predictions.

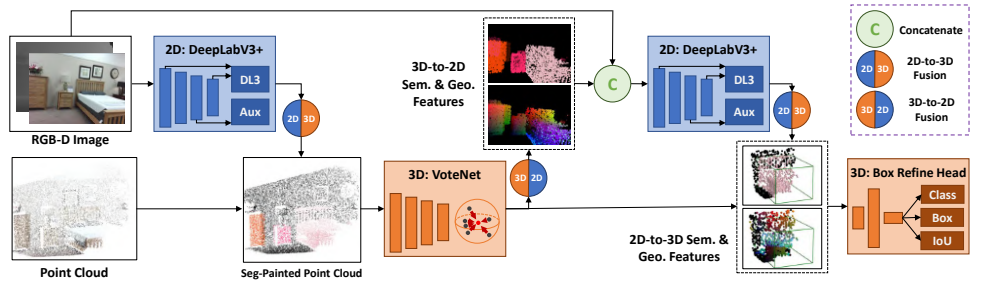

Figure 1: Overview of our proposed MTC-RCNN.

3 Method

3.1 Overview

MTC-RCNN is illustrated in Figure 1. At a high level, our multi-modality cascade can be described as (2D→)3D→2D→3D, with the first 2D being the addition of PointPainting [37]. In Sec. 3.2, we briefly outline VoteNet [29], which we use for 3D proposal generation. In Sec. 3.3, we describe the 3D-to-2D fusion, the 2D segmentation architecture, and the method of generating 2D segmentation ground truth. Then, in Sec. 3.4, we present the method of 2D-to-3D fusion and the 3D box refinement network. Sec. 3.5 describes how we apply PointPainting to our pipeline. Finally, Sec. 3.6 outlines our training losses.

3.2 3D Proposal Generation: Deep Hough Voting (VoteNet)

For this work, we choose VoteNet to generate 3D proposals because it is a representative, 3D-only baseline. Given a set of points, VoteNet uses a PointNet++ [27] backbone to extract features for a reduced set of sampled "seed" points. The seed features are then used to generate a set of "votes," with each vote consisting of a new 3D location closer to object centers as well as a new feature vector to be used for the final detection task. Finally, votes are clustered based on location, and each cluster predicts a 3D bounding box as well as its classification score. These 3D proposals are passed on to the later stages of our pipeline.

3.3 3D-to-2D: Using 3D Proposals for 2D Semantic Segmentation

3D-to-2D Fusion. In order to use 3D task-level information to benefit the 2D segmentation task, given a set of 3D proposals, our method extracts point-level *semantic* and *geometric* features from them to include as input to the 2D semantic segmentation network. The semantic features help the 2D network reason about which class each point/pixel belongs to, while the geometric features helps encode information about objects' 3D structure. Given a 3D box proposal, we randomly sample a set of points contained within the box $\{p_i\}_{i=1}^n$, where each $p_i \in \mathbb{R}^3$ are 3D coordinates. For each point p_i, we obtain its semantic features $s_i \in \mathbb{R}^C$ by giving it the object class probability distribution of the 3D proposal, where C is the number of classes. Then, to obtain geometric features, we transform each point to the box's canonical coordinates centered at the proposal's center and aligned to the proposal's heading direction. The geometric feature $g_i \in \mathbb{R}^9$ consists of two parts - the point's canonical coordinates and the point's offset distances to the box's 6 surfaces. In all, for each proposal, we have a set of points sampled within the box as well as their semantic and geometric features:

$$\{(p_i, f_i) \mid p_i \in \mathbb{R}^3, f_i = [s_i^\top, g_i^\top]^\top \in \mathbb{R}^{C+9}\}_{i=1}^n \quad (1)$$

These points are then projected to the 2D image using the camera intrinsics, and their corresponding features are assigned to the projected 2D location, generating a 3D-to-2D feature

map $\mathbf{F}^{3D\text{-}to\text{-}2D}$ of shape $H \times W \times (C+9)$ (H and W are height and width of RGB-D image). We note that some points can be in multiple overlapping proposals, and we resolve this conflict by assigning each point to the highest confidence proposal it is in. 2D locations without corresponding 3D points are given a zero vector of features. Finally, $\mathbf{F}^{3D\text{-}to\text{-}2D}$ is concatenated channel-wise with the RGB-D image, resulting in a feature map $\mathbf{F}^{2D\text{-}input}$ of shape $H \times W \times (C+13)$ to be used as the input to our 2D segmentation network.

2D Semantic Segmentation Network. For our 2D network, we use DeepLabV3+ [2] with a lightweight ResNet18 [10] backbone. We remove subsampling in the final two stages (C4 and C5) of the backbone and replace them with dilated convolutions. The DeepLabV3+ architecture includes a simple auxiliary head attached to the C4 stage, used for better supervision of intermediate layers. However, this head plays an important role in our work - we use predictions of this auxiliary head during training for 2D-to-3D fusion instead of the main head. We explain further in Sec. 3.4 after we introduce our 3D proposal refinement module.

2D Ground Truth Generation. Here, we briefly describe the generation of 2D segmentation ground truth and provide more details in the supplementary. To generate 2D labels, we expand the 2D depth map to a 3D point cloud and assign points (pixels) within a 3D box the box's class label. We ignore pixels within multiple boxes and pixels that lack a depth value.

3.4 2D-to-3D: Using 2D Segmentation for 3D Proposal Refinement

3D Proposal Refinement Network. From the first stage 3D network, we have a set of 3D proposal boxes. To refine these proposals, we extend a second-stage 3D refinement introduced by LiDAR-RCNN [17]. Given a proposal box, we enlarge it to capture more context. Then, we sample points within this enlarged box and extract each point's geometric features as described in Sec. 3.3 - this yields a 9-dimensional feature vector for each point.

In our 3D-only method, for each proposal, this set of geometric point features is run through a simple PointNet which predicts residuals for the box parameters as well the box's class type. Different from LiDAR-RCNN, this module is trained jointly, with gradients propagated through the per-point geometric features back to the first stage 3D network. Further, we add an IoU estimation branch to the PointNet to better the quality of the 3D box.

2D-to-3D Fusion. For 2D-to-3D fusion, we seek to use the 2D segmentation predictions from the 2D network as richer semantic features with which to guide the 3D box refinement. Extending the second stage 3D framework described above, for each proposal, the sampled points are projected onto the 2D segmentation predictions using the camera intrinsics. The corresponding channel-wise class distribution at the 2D projected point is appended to the 9-dimensional geometric features, resulting in each point having $9+C$ dimensional features. These concatenated features are then passed through the PointNet as described previously.

During training, we fuse the 2D predictions of the auxiliary head because the 2D performance gap between training and testing is much smaller for the auxiliary head than it is for the main head. If we fuse the main head during training, the points around proposals are perfectly segmented, trivializing the refinement task. Fusing the auxiliary head during training and using both heads during testing, the model learns to adapt to imperfect 2D predictions.

3.5 Early 2D-to-3D Fusion: PointPainting

So far, we have presented a 3D-to-2D-to-3D pipeline consisting of the initial 3D proposal generation stage, the fusion 2D segmentation network, and the 3D refinement module. However, this cascade of modalities does not necessarily need to start with 3D detection. We can add a 2D segmentation network before the first 3D proposal generation following PointPainting [37]. Specifically, before the point cloud is input into VoteNet, each point is projected

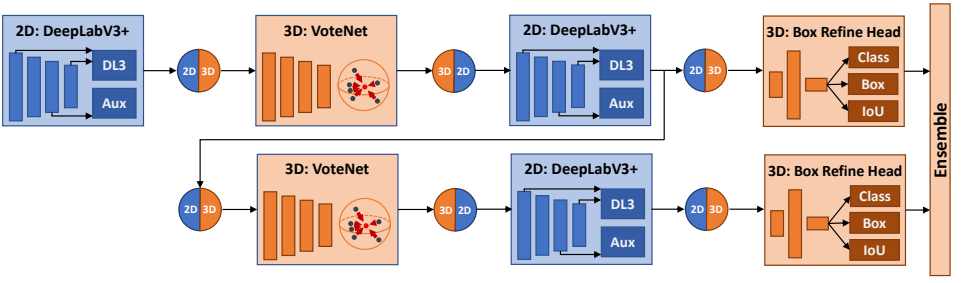

Figure 2: Illustration of recursively applying our pipeline twice.

onto initial 2D segmentation predictions and assigned the probability distribution of the corresponding 2D projection. This *painted* point cloud, each point of dimension $(3+C)$, is passed into the initial VoteNet. We then obtain 3D proposals and continue as before.

Compared to applying PointPainting to 3D-only approaches, PointPainting is especially effective on our framework for three reasons. First, improving the initial proposal generation stage directly raises the quality of 3D detections even after refinement - this is the usual benefit PointPainting has on 3D-only methods. Second, in our pipeline, better initial proposals mean better 2D segmentation predictions, which in turn further improves the second stage 3D refinement. Finally, we can recursively apply our entire pipeline as shown in Figure 2. Starting with 2D segmentations from any 2D baseline, we fuse them into our first stage 3D proposal generation stage following PointPainting. Then, using these proposals, our fusion 2D network outputs even better 2D predictions, which can be used to refine these 3D proposals. In the next iteration, we then take our improved 2D predictions and fuse them into the first stage 3D proposal generation stage (previously, we had used a weaker 2D-only baseline). We continue this process and ensemble the refined 3D boxes from every iteration. We find that recursively applying our pipeline results in better 3D box predictions.

3.6 Training Losses

3D Proposal Generation. For the first stage 3D network, we simply inherit the training losses from VoteNet [29], which can be summarized as:

$$\mathcal{L}_{rpn} = \mathcal{L}_{vote\text{-}reg} + \lambda_{obj\text{-}cls}\mathcal{L}_{obj\text{-}cls} + \lambda_{box}\mathcal{L}_{box} + \lambda_{sem\text{-}cls}\mathcal{L}_{sem\text{-}cls} \tag{2}$$

The loss terms correspond to the voting loss, objectness classification loss, box estimation loss, and multi-class semantic classification loss. We use the same loss weights λ as VoteNet.

2D Semantic Segmentation. Our 2D segmentation model has two prediction heads - the DeepLabV3+ head and the auxiliary head. Given the multi-modality input 2D feature map $\mathbf{F}^{2D\text{-}input}$, let **DL3** and **Aux** be the per-pixel segmentation predictions of the two heads and let $\mathbf{Y}$ be the ground truth 2D segmentation map. Then, our 2D segmentation loss is:

$$\mathcal{L}_{2d\text{-}seg} = \mathcal{L}_{ce}(\mathbf{DL3}, \mathbf{Y}) + \lambda_{aux}\mathcal{L}_{ce}(\mathbf{Aux}, \mathbf{Y}) \tag{3}$$

where $\mathcal{L}_{ce}$ is the multi-class cross entropy loss and λ_{aux} is set to be 0.4.

3D Box Refinement. The second stage 3D refinement loss consists of three parts - the box refinement loss, the multi-class semantic classification loss, and the IoU prediction loss:

$$\mathcal{L}_{rcnn} = \mathcal{L}_{box\text{-}refine} + \mathcal{L}_{sem\text{-}cls} + \mathcal{L}_{iou} \tag{4}$$

The box refinement loss is as follows:

$$\mathcal{L}_{box\text{-}refine} = \sum_{r\in\{x,y,z,l,h,w,\theta\}} \mathcal{L}_{smooth\text{-}L1}(\widehat{\Delta r}, \Delta r) \tag{5}$$

where Δr is the residual between the 3D proposal box and the matched ground truth box for box parameter r, and $\widehat{\Delta r}$ is the prediction for this residual. $\mathcal{L}_{sem\text{-}cls}$ is multi-class cross entropy loss between the predicted class of the proposal and the ground truth box. Finally, $\mathcal{L}_{iou}$ is the confidence loss used to better predict the quality of the bounding box. The network targets the normalized 3D IoU y between each proposal and its matched ground truth:

$$y = \min(1, \max(0, 2\text{IoU} - 0.3)) \tag{6}$$

and is trained via binary cross entropy loss.

Overall Loss. The total loss term is then:

$$\mathcal{L} = \mathcal{L}_{rpn} + \mathcal{L}_{2d\text{-}seg} + \mathcal{L}_{rcnn} \tag{7}$$

4 Experiments

4.1 Experimental Setup

Dataset. We evaluate on the SUN RGB-D benchmark [12, 33, 35, 40], which is a single-view, indoor dataset for scene understanding[1]. The dataset consists of 10,335 RGB-D images, of which 5,285 images are used for training and 5,050 are used for testing. We train and report results on the 10 most prevalent object classes following prior work [30, 45]. Point clouds are generated from 2D depth maps using camera intrinsics.

Network Architecture. We use the DeepLabV3+ model with an ImageNet pre-trained ResNet18 backbone for both 2D segmentation networks. Our 3D box refinement network has three linear layers of size [128, 128, 1024] before max pooling, and two shared layers $[512, 512]$ after pooling. Then, each of the 3D heads have a separate linear layer of size 512.

Training and Inference Details. Our core 3D→2D→3D framework is trained end-to-end for 240 epochs with the ADAM optimizer with a batch size of 8. The initial learning rate is 5e-4 and is decayed 10x at 160 and 210 epochs. We follow the data augmentation strategy in [30], sampling 20k points per view. More training details are provided in the supplementary.

Evaluation Protocol. For 3D detection, Average Precision (AP) over 10 classes is reported at the standard 0.25, 0.50, and 0.75 3D IoU thresholds. However, we find that AP at specific IoU thresholds fluctuate between evaluation runs. So, we adapt the robust challenge metric used in COCO object detection [20] and average the AP at 3D IoU thresholds from 0.25 to 0.95 with step size 0.05 and denote this as mAP. We prefer this metric as it fairly balances many IoU thresholds and is more stable. For 2D segmentation, we report the mIoU based on our generated 2D ground truth, ignoring overlapped regions and pixels without depth value.

4.2 Comparison with State-of-the-art Methods

In this section, we compare with state-of-the-art methods. Previous works [4, 29] usually train multiple times on different seeds and report the best results on the testing set. For fair comparison, we follow [21] in *training every setting 5 times* and *evaluating each setting 5 times*. The average performance over the 25 evaluations is in parentheses, and the best average evaluation result of the 5 training runs is presented on its left as the main comparison.

[1] We focus on the SUN RGB-D dataset like most image & point cloud fusion methods [10, 30] instead of the similar ScanNet [2] dataset as ScanNet point clouds are constructed from *many* RGB-D images, requiring extra processing to reconcile multiple point-to-image location correspondences.

Methods	Input	AP@0.25	AP@0.50	AP@0.75	mAP
F-PointNet [28]	point+RGB	54.0	-	-	-
VoteNet [29]	point	57.7	32.9	-	-
VoteNet [29]*	point	58.7	35.1	1.5	23.8
ImVoteNet [30]*	point+RGB	64.1	38.7	2.1	25.8
H3DNet [45]†	point	61.1	39.0	3.5	-
EPNet [11]	point+RGB	59.8	-	-	-
BRNet [4]	point	61.1	43.7	5.3	-
SparsePoint [23]	point	61.5	44.2	-	-
Group-Free [21]	point	63.0 (62.6)	45.2 (44.4)	-	-
Ours (3D→3D)	point	60.2 (59.5)	46.0 (45.5)	6.4 (6.5)	29.8 (29.4)
Ours (3D→2D→3D)	point+RGB	64.6 (64.1)	**49.0** (48.0)	7.7 (7.9)	31.8 (31.5)
Ours (2D→3D→2D→3D)	point+RGB	65.0 (64.5)	48.4 (48.0)	8.2 (7.9)	32.0 (31.6)
Ours (2D→3D) ×3	point+RGB	**65.3** (64.7)	48.6 (48.2)	**8.4** (8.1)	**32.2** (31.8)

Table 1: Performance comparison on SUN RGB-D with SOTA methods *We report 5-times evaluation results on the checkpoint from MMdetection3D[6] which has higher results than

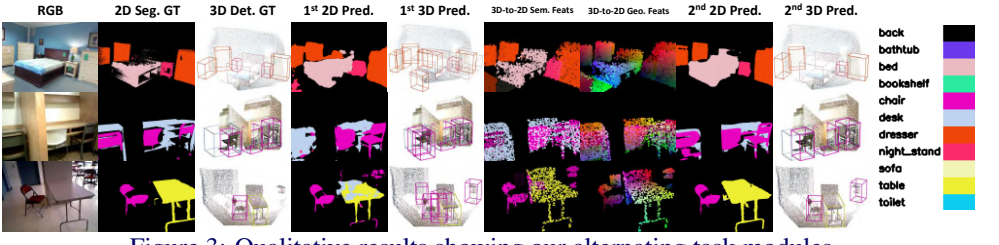

Figure 3: Qualitative results showing our alternating task modules.

Quantitative Results. We show results in Table 1. Simply by adding the 3D refinement module to VoteNet, we have a very strong 3D-only baseline, out-performing all previous methods on the AP@0.50 metric. Adding the fusion 2D segmentation model between the two 3D modules further boosts performance beyond our 3D-only baseline, improving AP@0.25 by 4.4 points, AP@0.50 by 3.0 points, and mAP by 2 points. This 3D→2D→3D model out-performs previous methods on all metrics. Adding another 2D segmentation model before the first 3D model improves the mAP by a small but significant margin, and recursively applying our pipeline once more as in Sec. 3.5 further improves performance.

Qualitative Results. In Fig. 3, we show step-by-step task predictions of our 2D→3D→2D→3D pipeline. We observe that predictions from the initial 2D-only model (1st 2D Pred.) are incomplete and messy. For example, in the middle row, the 2D model almost completely misses the left chair, mixing it into the desk. This leads to poor 3D boxes in the first stage 3D predictions (1st 3D Pred.) - that same chair has two boxes associated with it. However, after getting more information from the 3D detections (via 3D-to-2D features), the 2nd 2D predictions correctly segment the chair. These better 2D segmentations in turn lead to better 3D localization - we see that the same chair now only has a single, high-quality detection (2nd 3D Pred.). Through our visualizations, we confirm that MTC-RCNN effectively leverages the mutually beneficial relationship between the 2D image and the 3D point cloud.

4.3 Ablation Studies and Discussion

In this section, we present ablation studies to justify our training protocol and design choices. We use the stable mAP metric, averaged over the 25 evaluation runs as explained in Sec. 4.2.

2D-to-3D: Different Methods of Fusion. We ablate different methods of 2D-to-3D fusion in

(Train) 2D→3D	(Test) 2D→3D	(Train) 2D mIoU	(Test) 2D mIoU	(Test) 3D mAP
-	-	-	-	29.31
DL3 Features	DL3 Features	-	-	27.21
DL3 Seg. Preds	DL3 Seg. Preds	89.80	49.36	29.73
Aux Seg. Preds	DL3 Seg. Preds	65.37	50.36	**30.00**
DL3 Seg. Preds	DL3 + Aux Seg. Preds	89.80	50.33	30.81
Aux Seg. Preds	DL3 + Aux Seg. Preds	65.37	**50.84**	**31.09**

Table 2: Ablation on different methods of 2D-to-3D fusion. On the left, (Train) 2D→3D denotes the fusion method during training, and (Test) the method for inference. On the right, (Train) denotes performance on train set, while (Test) denotes performance on test set.

Method	Longer Training	2nd Stage 3D	3D mAP
VoteNet			23.81
VoteNet	✓		24.55
Ours (3D→3D)	✓	✓	**29.31**

Table 3: Effects of longer training and adding 2nd stage 3D module.

Method	# Points per RoI	2D mIoU	3D mAP
Ours (3D→3D)	512	-	29.31
Ours (3D→3D)	1024	-	**29.46**
Ours (3D→3D)	2048	-	29.43
Ours (3D→3D)	4096	-	29.33
Ours (3D→2D→3D)	512	50.84	31.09
Ours (3D→2D→3D)	1024	**51.35**	31.35
Ours (3D→2D→3D)	2048	51.03	**31.46**
Ours (3D→2D→3D)	4096	50.06	31.43

Table 4: Ablation on the number of points sampled in each proposal during inference.

Fuse 3D-to-2D	2D mIoU	3D mAP
	46.05	31.05
✓	51.03	**31.46**

Table 5: Fusion of 3D proposals into 2D segmentation network.

Method	2D mIoU	3D mAP
Ours (3D→2D→3D)	51.03	31.46
Ours (2D→3D→2D→3D)	52.65	31.62
Ours (2D→3D→2D→3D →2D→3D)	**52.93**	31.79
Ours (2D→3D→2D→3D →2D→3D→2D→3D)	52.91	**31.80**

Table 6: Effects of incorporating PointPainting into our framework.

Table 2. Simply fusing the 128-dim features from the DeepLabV3+ head (DL3) without 2D supervision (row 2) decreases performance compared to the two-stage 3D-only baseline (row 1) - likely due to overfitting of the 2D network. Regularizing this 2D-to-3D fusion by adding 2D supervision and fusing segmentation predictions instead (row 3) is able to out-perform the 3D-only baseline. However, there is a huge gap in mIoU between the train and test sets, causing the 3D refinement to perform poorly with imperfect test-time 2D segmentations. In the fourth row, we find that instead training on the weaker auxiliary predictions can remedy this issue, pulling train & test mIoU closer and further boosting mAP. In the final two rows, we show that ensembling the predictions of the two heads during inference can further boost performance and that training with the auxiliary head still has better mAP.

Two-stage 3D-only Baseline. In Table 3, we analyze our 3D-only baseline. We first find that training VoteNet for 240 epochs instead of the 180 epochs in the original paper can improve performance. Then, adding the 2nd stage 3D refinement module significantly boosts mAP.

Number of Points per Proposal. In Table 4, we ablate the number of points sampled within each proposal during inference. (512 points are sampled during training). We find that the boost in 3D mAP is larger for 3D→2D→3D than the 3D-only model, likely due to a corresponding increase in 2D performance. We do notice, however, that from 1024 to 2048, 2D mIoU drops while 3D mAP increases, suggesting that despite the positive relationship between 2D mIoU and 3D mAP, the two many not be perfectly correlated.

3D-to-2D: Fusing 3D Proposals for 2D Segmentation. Table 5 shows that the 3D-to-2D fusion significantly improves both 2D mIoU and 3D mAP. This ablation verifies our intuition that 3D predictions can benefit 2D predictions, which can further improve 3D predictions.

Additional Early 2D-to-3D Fusion: PointPainting In Table 6, we see that adding a 2D

network (2D-only ResNet18+DeepLabV3+ achieves 46.37 mIoU) before the initial 3D proposal generation boosts both 2D mIoU and 3D mAP. Then, recursively re-using the improved 2D segmentation to generate new proposals further improves metrics. Experiments with a larger initial 2D network are in the supplementary.

5 Conclusion

In this work, we have presented a new framework that recursively uses 3D detections and 2D segmentation predictions to improve each other in a cascade fashion. Fusing semantic and geometric features extracted from 3D box proposals into a 2D segmentation network, our model generates greatly improved 2D segmentation predictions that can then be used to refine the 3D proposals. Further, by training the initial 3D proposal generator to also take as input 2D segmentation results, the entire pipeline can be repeated recursively. Our experiments demonstrate that 2D images and 3D point clouds are *mutually* complementary modalities. Our network, MTC-RCNN, achieves new state-of-the-art 3D detection performance on the SUN RGB-D dataset without any 2D annotations.

References

[1] Aishwarya Agrawal, Jiasen Lu, Stanislaw Antol, Margaret Mitchell, C. L. Zitnick, Devi Parikh, and Dhruv Batra. Vqa: Visual question answering. *International Journal of Computer Vision*, 123:4–31, 2015.

[2] Liang-Chieh Chen, Yukun Zhu, G. Papandreou, Florian Schroff, and Hartwig Adam. Encoder-decoder with atrous separable convolution for semantic image segmentation. *ArXiv*, abs/1802.02611, 2018.

[3] Xiaozhi Chen, Huimin Ma, Jixiang Wan, B. Li, and Tian Xia. Multi-view 3d object detection network for autonomous driving. *2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 6526–6534, 2017.

[4] Bowen Cheng, Lu Sheng, Shaoshuai Shi, Ming Yang, and Dong Xu. Back-tracing representative points for voting-based 3d object detection in point clouds. *ArXiv*, abs/2104.06114, 2021.

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

[6] MMDetection3D Contributors. MMDetection3D: OpenMMLab next-generation platform for general 3D object detection. https://github.com/open-mmlab/mmdetection3d, 2020.

[7] Angela Dai, Angel X. Chang, M. Savva, Maciej Halber, T. Funkhouser, and Matthias Nießner. Scannet: Richly-annotated 3d reconstructions of indoor scenes. *2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 2432–2443, 2017.

[8] J. Gemmeke, D. Ellis, Dylan Freedman, A. Jansen, W. Lawrence, R. C. Moore, Manoj Plakal, and M. Ritter. Audio set: An ontology and human-labeled dataset for audio

events. *2017 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 776–780, 2017.

[9] Benjamin Graham, Martin Engelcke, and L. V. D. Maaten. 3d semantic segmentation with submanifold sparse convolutional networks. *2018 IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 9224–9232, 2018.

[10] Kaiming He, X. Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. *2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 770–778, 2016.

[11] Tengteng Huang, Zhe Liu, Xiwu Chen, and X. Bai. Epnet: Enhancing point features with image semantics for 3d object detection. In *ECCV*, 2020.

[12] Allison Janoch, S. Karayev, Y. Jia, Jonathan T. Barron, Mario Fritz, Kate Saenko, and Trevor Darrell. A category-level 3-d object dataset: Putting the kinect to work. In *ICCV Workshops*, 2011.

[13] Will Kay, João Carreira, K. Simonyan, Brian Zhang, Chloe Hillier, Sudheendra Vijayanarasimhan, Fabio Viola, Tim Green, T. Back, A. Natsev, Mustafa Suleyman, and Andrew Zisserman. The kinetics human action video dataset. *ArXiv*, abs/1705.06950, 2017.

[14] Jason Ku, Melissa Mozifian, Jungwook Lee, Ali Harakeh, and Steven L. Waslander. Joint 3d proposal generation and object detection from view aggregation. *2018 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pages 1–8, 2018.

[15] Jean Lahoud and Bernard Ghanem. 2d-driven 3d object detection in rgb-d images. *2017 IEEE International Conference on Computer Vision (ICCV)*, pages 4632–4640, 2017.

[16] Alex H. Lang, Sourabh Vora, Holger Caesar, Lubing Zhou, Jiong Yang, and Oscar Beijbom. Pointpillars: Fast encoders for object detection from point clouds. *2019 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 12689–12697, 2019.

[17] Zhichao Li, Feng Wang, and Naiyan Wang. Lidar r-cnn: An efficient and universal 3d object detector. *CVPR*, 2021.

[18] Ming Liang, B. Yang, Shenlong Wang, and R. Urtasun. Deep continuous fusion for multi-sensor 3d object detection. In *ECCV*, 2018.

[19] Ming Liang, B. Yang, Yun Chen, Rui Hu, and R. Urtasun. Multi-task multi-sensor fusion for 3d object detection. *2019 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 7337–7345, 2019.

[20] Tsung-Yi Lin, M. Maire, Serge J. Belongie, James Hays, P. Perona, D. Ramanan, Piotr Dollár, and C. L. Zitnick. Microsoft coco: Common objects in context. In *ECCV*, 2014.

[21] Ze Liu, Zheng Zhang, Yue Cao, Han Hu, and Xin Tong. Group-free 3d object detection via transformers. *ArXiv*, abs/2104.00678, 2021.

[22] Zhijian Liu, Haotian Tang, Yujun Lin, and Song Han. Point-voxel cnn for efficient 3d deep learning. In *NeurIPS*, 2019.

[23] Zili Liu, Guodong Xu, Honghui Yang, Haifeng Liu, and Deng Cai. Sparsepoint: Fully end-to-end sparse 3d object detector. *ArXiv*, abs/2103.10042, 2021.

[24] Jongyoun Noh, Sanghoon Lee, and Bumsub Ham. Hvpr: Hybrid voxel-point representation for single-stage 3d object detection. *ArXiv*, abs/2104.00902, 2021.

[25] Su Pang, D. Morris, and H. Radha. Clocs: Camera-lidar object candidates fusion for 3d object detection. *2020 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pages 10386–10393, 2020.

[26] C. Qi, Hao Su, Kaichun Mo, and L. Guibas. Pointnet: Deep learning on point sets for 3d classification and segmentation. *2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 77–85, 2017.

[27] C. Qi, L. Yi, Hao Su, and L. Guibas. Pointnet++: Deep hierarchical feature learning on point sets in a metric space. In *NIPS*, 2017.

[28] C. Qi, W. Liu, Chenxia Wu, Hao Su, and L. Guibas. Frustum pointnets for 3d object detection from rgb-d data. *2018 IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 918–927, 2018.

[29] C. Qi, O. Litany, Kaiming He, and L. Guibas. Deep hough voting for 3d object detection in point clouds. *2019 IEEE/CVF International Conference on Computer Vision (ICCV)*, pages 9276–9285, 2019.

[30] C. Qi, Xinlei Chen, O. Litany, and L. Guibas. Imvotenet: Boosting 3d object detection in point clouds with image votes. *2020 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 4403–4412, 2020.

[31] Shaoshuai Shi, Xiaogang Wang, and Hongsheng Li. Pointrcnn: 3d object proposal generation and detection from point cloud. *2019 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 770–779, 2019.

[32] Shaoshuai Shi, Zhe Wang, Jianping Shi, Xiaogang Wang, and Hongsheng Li. From points to parts: 3d object detection from point cloud with part-aware and part-aggregation network. *IEEE transactions on pattern analysis and machine intelligence*, 2020.

[33] N. Silberman, Derek Hoiem, P. Kohli, and R. Fergus. Indoor segmentation and support inference from rgbd images. In *ECCV*, 2012.

[34] V. Sindagi, Yin Zhou, and Oncel Tuzel. Mvx-net: Multimodal voxelnet for 3d object detection. *2019 International Conference on Robotics and Automation (ICRA)*, pages 7276–7282, 2019.

[35] Shuran Song, Samuel P. Lichtenberg, and J. Xiao. Sun rgb-d: A rgb-d scene understanding benchmark suite. *2015 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 567–576, 2015.

[36] Haotian Tang, Zhijian Liu, Shengyu Zhao, Yujun Lin, Ji Lin, Hanrui Wang, and Song Han. Searching efficient 3d architectures with sparse point-voxel convolution. *ArXiv*, abs/2007.16100, 2020.

[37] Sourabh Vora, Alex H. Lang, Bassam Helou, and Oscar Beijbom. Pointpainting: Sequential fusion for 3d object detection. *2020 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 4603–4611, 2020.

[38] Weiyao Wang, Du Tran, and Matt Feiszli. What makes training multi-modal classification networks hard? *2020 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 12692–12702, 2020.

[39] Zhixin Wang and Kui Jia. Frustum convnet: Sliding frustums to aggregate local point-wise features for amodal. *2019 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pages 1742–1749, 2019.

[40] J. Xiao, Andrew Owens, and A. Torralba. Sun3d: A database of big spaces reconstructed using sfm and object labels. *2013 IEEE International Conference on Computer Vision*, pages 1625–1632, 2013.

[41] Liang Xie, Chao Xiang, Zhengxu Yu, Guodong Xu, Zheng Yang, Deng Cai, and Xiaofei He. Pi-rcnn: An efficient multi-sensor 3d object detector with point-based attentive cont-conv fusion module. *ArXiv*, abs/1911.06084, 2020.

[42] Yan Yan, Yuxing Mao, and B. Li. Second: Sparsely embedded convolutional detection. *Sensors (Basel, Switzerland)*, 18, 2018.

[43] Zetong Yang, Y. Sun, Shu Liu, and Jiaya Jia. 3dssd: Point-based 3d single stage object detector. *2020 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 11037–11045, 2020.

[44] Jin Hyeok Yoo, Yeocheol Kim, Ji Song Kim, and J. W. Choi. 3d-cvf: Generating joint camera and lidar features using cross-view spatial feature fusion for 3d object detection. In *ECCV*, 2020.

[45] Zaiwei Zhang, Bo Sun, Haitao Yang, and Qi-Xing Huang. H3dnet: 3d object detection using hybrid geometric primitives. In *ECCV*, 2020.

[46] Yin Zhou and Oncel Tuzel. Voxelnet: End-to-end learning for point cloud based 3d object detection. *2018 IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 4490–4499, 2018.