Deep Hough Voting for 3D Object Detection in Point Clouds Supplementary Material

Charles R. Qi[1] Or Litany[1] Kaiming He[1] Leonidas J. Guibas[1,2]
[1]Facebook AI Research [2]Stanford University

A. Appendix

This appendix provides additional details on the network architectures and loss functions (Sec. A.1), more analysis experiment results (Sec. A.2), per-category results on ScanNet (Sec. A.3), and finally more visualizations (Sec. A.4).

A.1. Details on Architectures and Loss Functions

VoteNet architecture details. As mentioned in the main paper, the VoteNet architecture composes of a backbone point feature learning network, a voting module and a proposal module.

The backbone network, based on the PointNet++ architecture [4], has four set abstraction layers and two feature up-sampling layers. The detailed layer parameters are shown in Table 1. Each set abstraction (SA) layer has a receptive field specified by a ball-region radius r, a MLP network for point feature transform $MLP[c_1, ..., c_k]$ where c_i is output channel number of the i-th layer in the MLP. The SA layer also subsamples the input point cloud with farthest point sampling to n points. Each SA layer is specified by $(n, r, [c_1, ..., c_k])$ as shown in the Table 1. Compared to [4], we also normalize the XYZ scale of points in each local region by the region radius.

Each set feature propagation (FP) layer upsamples the point features by interpolating the features on input points to output points (each output point's feature is weighted average of its three nearest input points' features). It also combines the skip-linked features through a MLP (interpolated features and skip-linked features are concatenated before fed into the MLP). Each FP layer is specified by $[c_1, ..., c_k]$ where c_i is the output of the i-th layer in the MLP.

The voting module as mentioned in the main paper is a MLP that transforms seeds' features to votes including a XYZ offset and a feature offset. The seed points are outputs of the fp2 layer. The voting module MLP has output sizes of $256, 256, 259$ for its fully connected layers. The last fully connected layer does not have ReLU or BatchNorm.

The proposal module as mentioned in the main paper is a SA layer followed by another MLP after the max-pooling in each local region. We follow [3] on how to parameterize the oriented 3D bounding boxes. The layer's output has $5+2NH+4NS+NC$ channels where NH is the number of heading bins (we predict a classification score for each heading bin and a regression offset for each bin – relative to the bin center and normalized by the bin size), NS is the number of size templates (we predict a classification score for each size template and 3 scale regression offsets for height, width and length) and NC is the number of semantic classes. In SUN RGB-D: $NH = 12, NS = NC = 10$, in ScanNet: $NH = 12, NS = NC = 18$. In the first 5 channels, the first two are for objectness classification and the rest three are for center regression (relative to the vote cluster center).

VoteNet loss function details. The network is trained end-to-end with a multi-task loss including a voting loss, an objectness loss, a 3D bounding box estimation loss and a semantic classification loss. We weight the losses such that they are in similar scales with $\lambda_1 = 0.5$, $\lambda_2 = 1$ and $\lambda_3 = 0.1$.

$$L_{\text{VoteNet}} = L_{\text{vote-reg}} + \lambda_1 L_{\text{obj-cls}} + \lambda_2 L_{\text{box}} + \lambda_3 L_{\text{sem-cls}} \quad (1)$$

Among the losses, the vote regression loss is as defined in the main paper (with L1 distance). For ScanNet we compute the ground truth votes as offset from the mesh vertices of an instances to the centers of the axis-aligned tight bounding boxes of the instances. Note that since the bounding box is not amodal, they can vary in sizes due to scan completeness (e.g. a chair may have a floating bounding box if its leg is not recovered from the reconstruction). For SUN RGB-D since the dataset does not provide instance segmentation annotations but only amodal bounding boxes, we cannot compute a ground truth vote directly (as we don't know which points are on objects). Instead, we consider any point inside an annotated bounding box as an object point (required to vote) and compute its offset to the box center as the ground truth. In cases that a point is in multiple ground truth boxes, we keep a set of up to three ground truth votes, and consider the minimum distance between the predicted

layer name	input layer	type	output size	layer params
sa1	raw point cloud	SA	(2048,3+128)	(2048,0.2,[64,64,128])
sa2	sa1	SA	(1024,3+256)	(1024,0.4,[128,128,256])
sa3	sa2	SA	(512,3+256)	(512,0.8,[128,128,256])
sa4	sa3	SA	(256,3+256)	(256,1.2,[128,128,256])
fp1	sa3, sa4	FP	(512,3+256)	[256,256]
fp2	sa2, sa3	FP	(1024,3+256)	[256,256]

Table 1. Backbone network architecture: layer specifications.

vote and any ground truth vote in the set during vote regression on this point.

The objectness loss is just a cross-entropy loss for two classes and the semantic classification loss is also a cross-entropy loss of NC classes.

The box loss follows [3] (but without the corner loss regularization for simplicity) and is composed of center regression, heading estimation and size estimation sub-losses. In all regression in the box loss we use the robust $L1$-smooth loss. Both the box and semantic losses are only computed on positive vote clusters and normalized by the number of positive clusters. We refer readers to [3] for more details.

$$L_{\text{box}} = L_{\text{center-reg}} + 0.1L_{\text{angle-cls}} + L_{\text{angle-reg}} + 0.1L_{\text{size-cls}} + L_{\text{size-reg}} \quad (2)$$

One difference though is that, instead of a naive regression loss, we use a *Chamfer loss* [1] for $L_{\text{center-reg}}$ (between regressed centers and ground truth box centers). It requires that each positive proposal is close to a ground truth object and each ground truth object center has a proposal near it. The latter part also influences the voting in the sense that it encourages non-object seed points near the object to also vote for the center of the object, which helps further increase contexts in detection.

BoxNet architecture details. Our baseline network without voting, BoxNet, shares most parts with the VoteNet. They share the same backbone architecture. But instead of voting from seeds, the BoxNet directly proposes bounding boxes and classifies object classes from seed points' features. To make the BoxNet and VoteNet have similar capacity we also include a SA layer for the proposal in BoxNet. However this SA layer takes *seed clusters* instead of *vote clusters* i.e. it samples seed points and then combines neighboring seeds with MLP and max-pooling. This SA layer has exactly the same layer parameters with that in the VoteNet, followed by the same MLP_2.

BoxNet loss function details. BoxNet has the same loss function as VoteNet, except it is not supervised by vote regression. There is also a slight difference in how objectness labels (used to supervise objectness classification) are computed. As seed points (on object surfaces) are often far from object centroids, it no longer works well to use the distances between seed points and object centroids to compute the objectness labels. In BoxNet, we assign positive objectness labels to seed points that are on objects (those belonging to the semantic categories we consider) and negative labels to all the other seed points on clutter (e.g. walls, floors).

$$L_{\text{BoxNet}} = \lambda_1 L_{\text{obj-cls}} + \lambda_2 L_{\text{box}} + \lambda_3 L_{\text{sem-cls}} \quad (3)$$

A.2. More Analysis Experiments

Average precision and recall plots Fig. 1 shows how average precision (AP) and average recall (AR) change as we increase the number of proposals. The AP and AR are both averaged across 10 categories on SUN RGB-D. We report two ways of using the proposals: joint and per-class. For the joint proposal we propose K objects' bounding boxes for all the 10 categories, where we consider each proposal as the semantic class it has the largest confidence in, and use their objectness scores to rank them. For the per-class proposal we duplicate the K proposal 10 times thus have K proposals per class where we use the multiplication of semantic probability for that class and the objectness probability to rank them. The latter way of using proposals gives us a slight improvement on AP and a big boost on AR.

We see that with as few as 10 proposals our VoteNet can achieve a decent AP of around 45% while having 100 proposals already pushes the AP to above 57%. With a thousand proposals, our network can achieve around 78.7% recall with joint proposal and around 87.7% recall with per-class proposal.

Context of voting One difference of a deep Hough voting scheme with the traditional Hough voting is that we can take advantage of deep features, which can provide more context knowledge for voting. In Table 4 we show how features from different levels of the PointNet++ affects detection performance (from SA2 to FP3, the network has increasing contexts for voting). FP3 layer is extended from the FP2 with a MLP of output sizes 256 and 256 with 2048 output points (the same set of XYZ as that output by SA1).

	cab	bed	chair	sofa	tabl	door	wind	bkshf	pic	cntr	desk	curt	fridg	showr	toil	sink	bath	ofurn	mAP
3DSIS 5views [2]	19.76	69.71	66.15	71.81	36.06	30.64	10.88	27.34	0.00	10.00	46.93	14.06	53.76	35.96	87.60	42.98	84.30	16.20	40.23
3DSIS Geo [2]	12.75	63.14	65.98	46.33	26.91	7.95	2.79	2.30	0.00	6.92	33.34	2.47	10.42	12.17	74.51	22.87	58.66	7.05	25.36
VoteNet ours	36.27	87.92	88.71	89.62	58.77	47.32	38.10	44.62	7.83	56.13	71.69	47.23	45.37	57.13	94.94	54.70	92.11	37.20	58.65

Table 2. 3D object detection scores per category on the ScanNetV2 dataset, evaluated with mAP@0.25 IoU.

	cab	bed	chair	sofa	tabl	door	wind	bkshf	pic	cntr	desk	curt	fridg	showr	toil	sink	bath	ofurn	mAP
3DSIS 5views [2]	5.73	50.28	52.59	55.43	21.96	10.88	0.00	13.18	0.00	0.00	23.62	2.61	24.54	0.82	71.79	8.94	56.40	6.87	22.53
3DSIS Geo [2]	5.06	42.19	50.11	31.75	15.12	1.38	0.00	1.44	0.00	0.00	13.66	0.00	2.63	3.00	56.75	8.68	28.52	2.55	14.60
VoteNet (ours)	8.07	76.06	67.23	68.82	42.36	15.34	6.43	28.00	1.25	9.52	37.52	11.55	27.80	9.96	86.53	16.76	78.87	11.69	33.54

Table 3. 3D object detection scores per category on the ScanNetV2 dataset, evaluated with mAP@0.5 IoU.

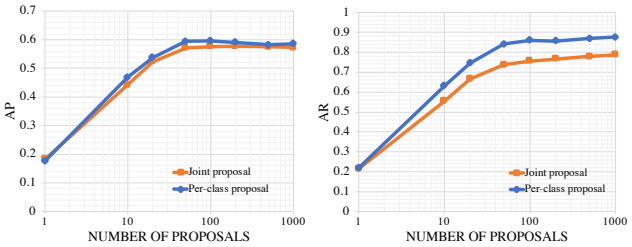

Figure 1. **Number of proposals per scene v.s. Average Precision (AP) and Average Recall (AR) on SUN RGB-D.** The AP and AR are averaged across the 10 classes. The recall is maximum recall given a fixed number of detection per scene. The "joint proposal" means that we assign each proposal to a single class (the class with the highest classification score); The "per-class proposal" means that we assign each proposal to all the 10 classes (the objectness score is multipled by the semantic classification probability).

It is surprising to find that voting from even SA2 can achieve reasonable detection results (mAP 51.2%) while voting from FP2 achieves the best performance. Having larger context (e.g. FP3) than FP2 does not show further improvements on the performance.

Seed layer	SA2	SA3	SA4	FP1	FP2	FP3
mAP	51.2	56.3	55.1	56.6	**57.7**	57.1

Table 4. **Effects of seed context for 3D detection.** Evaluation metric is mAP@0.25 on SUN RGB-D.

Multiple votes per seed In default we just generate one vote per seed since we find that with large enough context there is little need to generate more than one vote to resolve ambiguous cases. However, it is still possible to generate more than one vote with our network architecture. Yet to break the symmetry in multiple vote generation, one has to introduce some bias to different votes to prevent then from pointing to the same place.

In experiments, we find that one vote per seed achieves the best results, as shown in Table 5. We ablate by using a vote factor of 3, where the voting module generates 3 votes per seed with a MLP layer spec: $[256, 256, 259 * 3]$). In computing the vote regression loss on a seed point, we consider the minimum distance between any predicted votes to the ground truth vote (in case of SUN RGB-D where we may have a set of ground truth votes for a seed, we compute the minimum distance among any pair of predicted vote and ground truth vote).

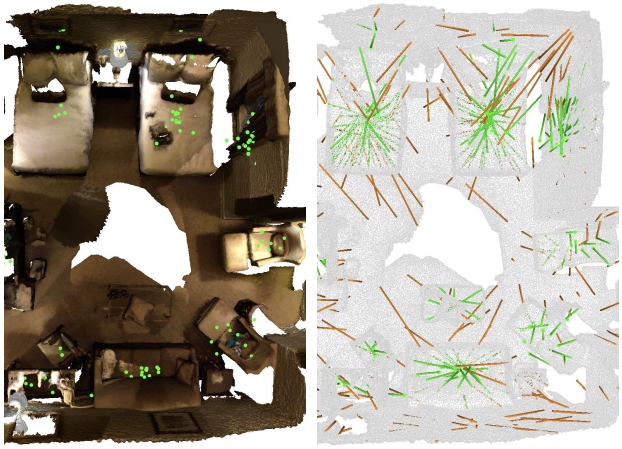

Figure 2. **Vote meeting point.** *Left:* ScanNet scene with votes coming from object points. *Right:* vote offsets from source seed-points to target-votes. Object votes are colored green, and non-object ones are colored red. See how object points from all-parts of the object vote to form a cluster near the center. Non-object points, however, either vote "nowhere" and therefore lack structure, or are near object and have gathered enough context to also vote properly.

To break symmetry, we generate 3 random numbers and inject them to the second last features from the MLP layer. We show results both with and without this procedure which shows no observable difference.

Vote factor	1	3	3
Random number	N	N	Y
mAP	**57.7**	55.8	55.8

Table 5. **Effects of number of votes per seed.** Evaluation metric is mAP@0.25 on SUN RGB-D. If random number is on, we concatenate a random number to the seed feature before voting, which helps break symmetry in the case of multiple votes per seed.

Proposal sampling	mAP
Random sampling	57.5
Farthest point sampling on votes	57.2
Farthest point sampling on seeds	**57.7**

Table 6. **Effects of proposal sampling.** Evaluation metric is mAP@0.25 on SUN RGB-D. 256 proposals are used for all evaluations. Our method is not sensitive to how we choose centers for vote groups/clusters.

On proposal sampling In the proposal step, to generate K proposals from the votes, we need to select K vote clusters. How to select those clusters is a design choice we study here (each cluster is simply a group of votes near a center vote). In Table 6, we report mAP results on SUN RGB-D with 256 proposals (joint proposal) using cluster sampling strategies of vote FPS, seed FPS and random sampling, where FPS means farthest point sampling. From 1024 vote clusters, vote FPS samples K clusters based on votes' XYZ. Seed FPS firstly samples on seed XYZ and then finds the votes corresponding to the sampled seeds – it enables a direct comparison with BoxNet as it uses the same sampling scheme, making the two techniques similar up to the space in which the points are grouped: VoteNet groups votes according to vote XYZ, while BoxNet groups seeds according to seed XYZ. Random sampling simply selects a random set of K votes and take their neighborhoods for proposal generation. Note that the results from Table 6 are from the same model trained with vote FPS to select proposals.

We can see that while seed FPS gets the best number in mAP, the difference caused by different sampling strategies is small, showing the robustness of our method.

Effects of the height feature In point clouds from indoor scans, point height is a useful feature in recognition. As mentioned in the main paper, we can use 1% of the Z values (Z-axis is up-right) of all points from a scan as an approximate as the floor height z_{floor}, and then compute the a point (x, y, z)'s height as $z - z_{\text{floor}}$. In Table 7 we show how this extra height feature affect detection performance. We see that adding the height feature consistently improves performance in both SUN RGB-D and ScanNet.

Dataset	with height	without height
SUN RGB-D	57.7	57.0
ScanNet	58.6	58.1

Table 7. **Effects of the height feature.** Evaluation metric is mAP@0.25 on both datasets.

A.3. ScanNet Per-class Evaluation

Table 2 and Table 3 report per-class average precision on 18 classes of ScanNetV2 with 0.25 and 0.5 box IoU thresholds respectively. Relying on purely geonetric data, our method excels (esp. with mAP@0.25) in detecting objects like bed, chair, table, desk etc. where geometry is a strong cue for recognition; and struggles with objects best recognized by texture and color like pictures.

A.4. Visualization of Votes

Fig. 2 shows (a subset of) votes predicted from our VoteNet in a typical ScanNet scene. We clearly see that seed points on objects (bed, sofa etc.) vote to object centers while clutter points vote either to object center as well (if the clutter point is close to the object) or to nowhere due to lack of structure in the clutter area (e.g. a wall).

References

[1] H. Fan, H. Su, and L. J. Guibas. A point set generation network for 3d object reconstruction from a single image. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 605–613, 2017. 2

[2] J. Hou, A. Dai, and M. Nießner. 3D-SIS: 3d semantic instance segmentation of rgb-d scans. *arXiv preprint arXiv:1812.07003*, 2018. 3

[3] C. R. Qi, W. Liu, C. Wu, H. Su, and L. J. Guibas. Frustum pointnets for 3d object detection from rgb-d data. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 918–927, 2018. 1, 2

[4] C. R. Qi, L. Yi, H. Su, and L. J. Guibas. Pointnet++: Deep hierarchical feature learning on point sets in a metric space. *arXiv preprint arXiv:1706.02413*, 2017. 1