Dynamic sampling pointnet notes

xyz

Feb 2018

Contents

1 Deep 3D Learning Notes

1.1 potential solutions for over fitting

- combine auxiliary tasks:

  1. use nxnynz as auxiliary loss 2. boundary

  3. planet

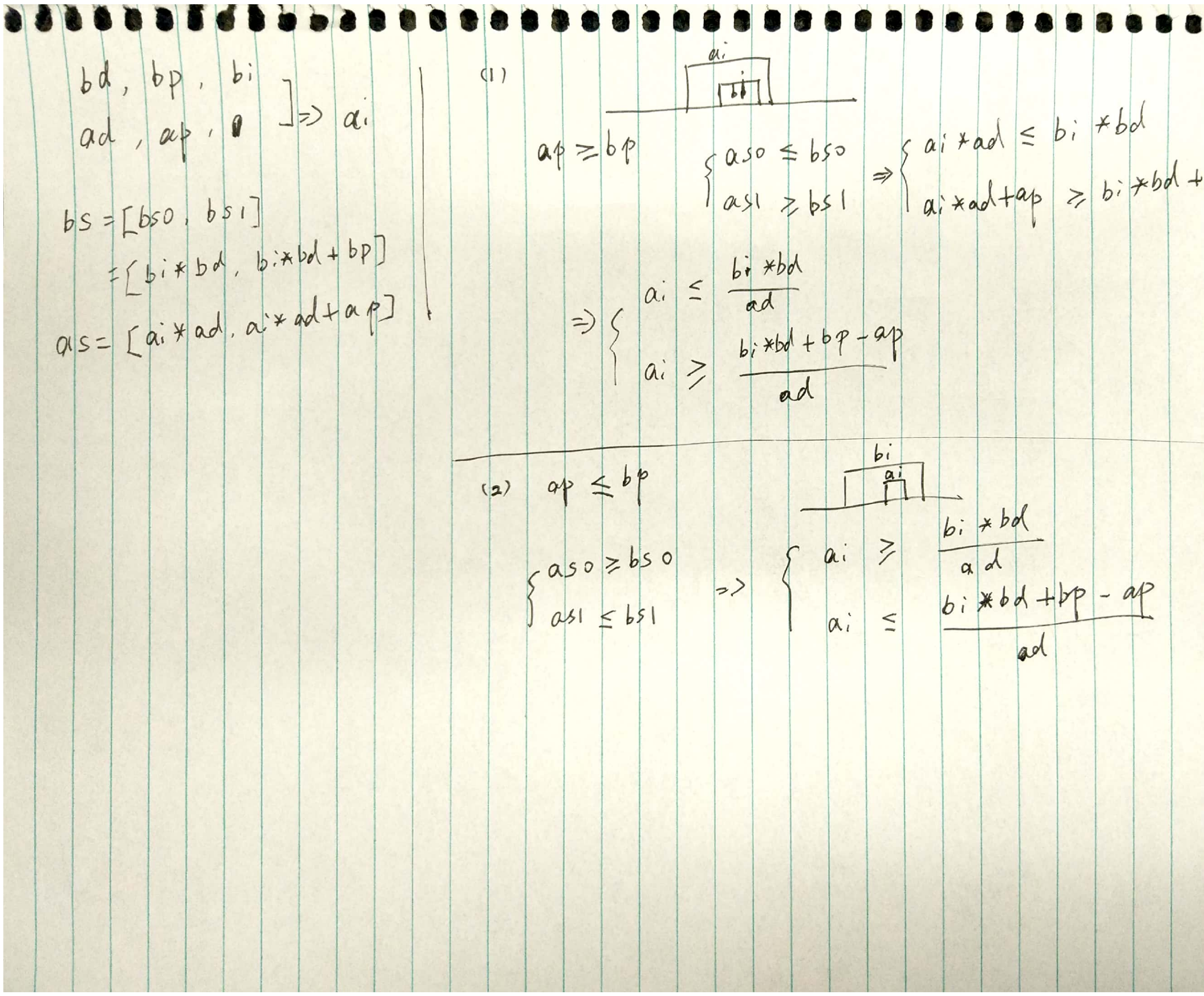
- data augmentation: rotation, different size scale
- data regulation: group norm
- dropout
- combine ShapeNet data
- more powerful network to learn more systematic information:
  1. use large global block size
  2. dynamic sampling
- smaller net

1.2 Important improvements

- Generate bxmh5 online. So the randomly missing part in each epoch is different. This maybe solve the info missing problem for sparse voxel 3d cnn, especially considering that block merging cannot be applied for voxel cnn. However, on line sampling can only solve missing problom of training, test missing still need some tricks to perform block merging.
- Check this: my usage of tf.gather_nd should cost a lot of memory, maybe too much!

1.3 Theory

1.3.1 bidxmap



1.3.2 group sampling configuration

$$steps = [0.1, 0.3, 0.9, 2.7] + [-6.3]$$

$$strides = [0.1, 0.2, 0.6, 1.8] + [-3.6]$$

$$voxel\ size = [, 3, 4, 4, 3]$$

principles:

(1)Alignment between differert scales:

$$steps[i] = steps[i-1] + strides[i-1] * (k-1) \ (k = voxel\ size)$$

(2)Alignment between voxels on one scale:

$$strides[i] \% steps[i-1] == 0$$

Examples:

$$0.3 = 0.1 + 0.1 * 2 \Rightarrow voxel\ size = 3$$
$$0.2 = 0.1 * 2$$
$$0.9 = 0.3 + 0.2 * 3 \Rightarrow voxel\ size = 4$$
$$0.6 = 0.3 * 2$$

$$6.3 = 2.7 + 1.8 * 2$$
$$3.6 = 1.8 * 2$$

1.3.3 Sparse voxel 3DCNN

1.3.4 multi-scale classification

1.3.5 Data Augmentation

- (1.1) Rotate corrdinate reference: Rotate both point and voxel box

  Performed by rotating points after sampling and grouping.

  This should only be applied to point position (cascade 0). What if also to features (upper cascades).

- (1.2) Rotae point only, or rotate voxel box only.

  a) It can be performed by rotating points before sampling and grouping.

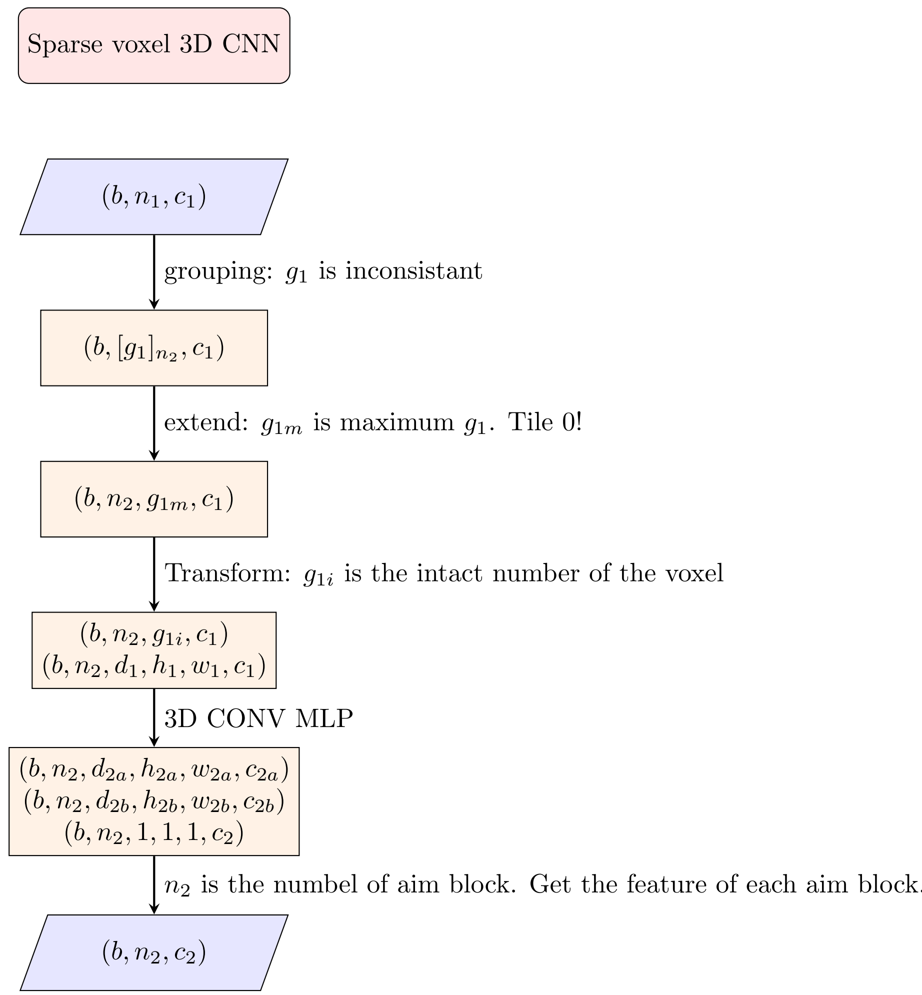
  b) If rotate angle is integral times of pi/2, it can be performed by rotating point indices inside the voxel.

  Rotate voxel can be applied to all cascades.

- (2.1) Rotate the global block by the same angle

- (2.2) Rotate each voxel by seperate angle in each scale.

  Since the features are calculated independently in each voxel, it should be fine to apply different rotatio angle for each voxel. It doesn't matter that the rotation center is voxel center or global block center. It alos doesn't matter that it rotates refference or only rotates voxel.



Two main obstacles for performing 3D convolution on point cloud are: (1) there are too many vacant points, (2) the position of points are not aligned. The key idea of sparse voxel is to perform 3DCONV on cascades from the second. Because the positions are actually almost aligned. At the same time, the vacant rate within a small block is acceptably large. Above all, it may be possible to do apply 3D-CONV within a small block.

Centres of blocks in cascades other than first one are actually aligned to the grid. So it is possible to perform 3d convolution directly. However, the average position of points inside these blocks are not aligned. Thus it is also maybe beneficial to utilise a transform net to align them.

On the other hand, there are many vacant points in the block. I am wondering if it is beneficial to set the features of vacant points by a T-net from around existing points.

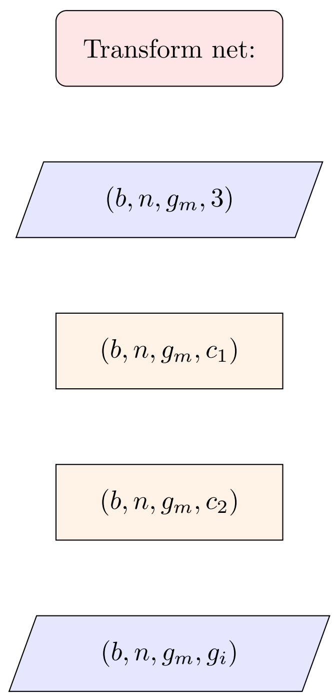
Purpose of T-Net: fix number + align + till

There are some interesting problems for Transform net:

- Only depend on position or feature.
- Should be resolution invariant.
- If it should be constant for all channels.
- If it should be constant for all local aim blocks.

Reasons that we do not need the T-Net:

- 3d-conv can till features of the vacant points.
- If the base-points are not strictly aligned, add the position to feature map. Or get a special feature of positions within the block and then add ot the main feature map.



1.4 batch size

1.4.1 bs=27 vs bs=81

batch size: 9,27,81
data: xyz-color_1norm
model: 1AG
sampling & grouping: stride_0d1_step_0d1_bmap_nh5_2048_0d5_1_fmn1-160_32-32_12-0d2_0d6-0d2_0d6

Figure 1: bs=9

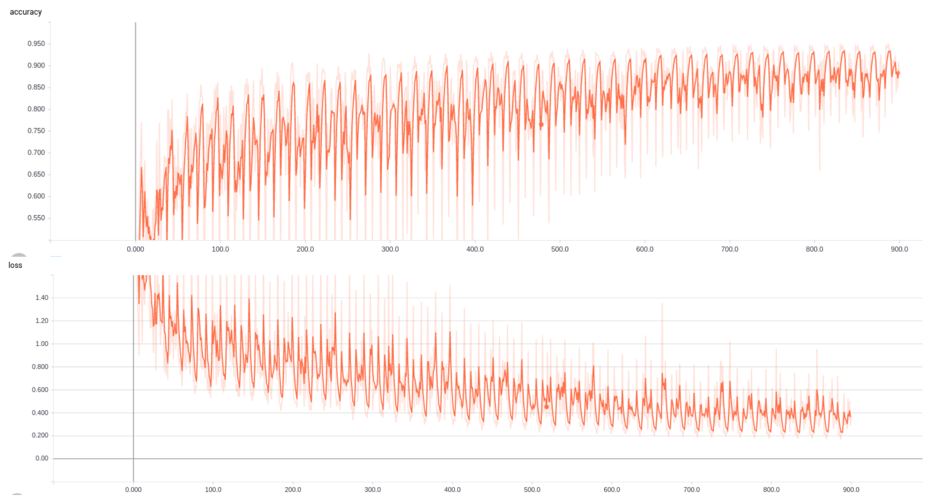


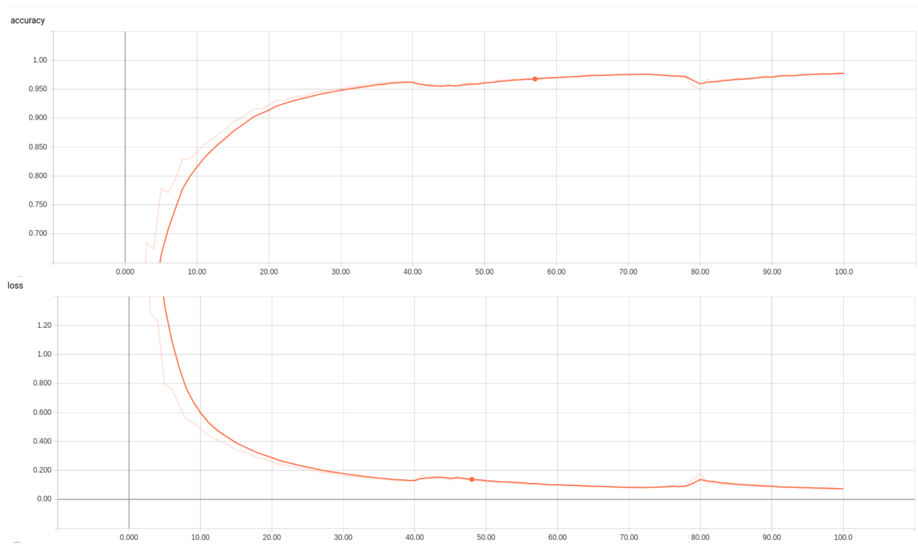
Figure 2: bs=27

1.5 feed elements

epoch num = 100

stride_0d1_step_0d1_bmap_nh5_2048_0d5_1_fmn1-160_32-32_12-0d2_0d6-0d2_0d6

Figure 3: bs=81



model	batch size	data elements	acc	loss
1AG	9	xyz color	0.890	0.356
1AG	27	xyz color	0.920	0.240
3AG	27	xyz color	0.912	0.273
2A	27	xyz color	0.908	0.294
2AG	27	xyz color	0.902	0.293
1A	27	xyz color	0.883	0.351
1AG	81	xyz color	0.978	0.072
1AG	9	xyz	0.861	0.427
1AG	27	xyz	0.907	0.257
1AG	81	xyz	0.975	0.078
1A	27	xyzmid color	0.889	0.357
3AG	27	xyzmid color	0.933	0.193
2A	27	xyzmid color	0.939	0.177
2AG	27	xyzmid color	0.929	0.208
3AG	27	xyz xyzmid color	0.924	0.230
2A	27	xyz xyzmid color	0.898	0.317
2AG	27	xyz xyzmid color	0.908	0.280
1A	27	xyz xyzmid color	0.910	0.281
1AG	27	xyz xyzmid color	0.944	0.163
1AG	81	xyz xyzmid color	0.976	0.078
2A	81	xyz xyzmid color	0.942	0.173
3AG	81	xyz xyzmid color	0.949	0.147

1. large batch size is better
2. $1AG(0.92) > 3AG(0.912) > 2A(0.908) > 2AG(0.902) > 1A(883)$
1AG is much better than 1A
1AG is a bit better than 3AG ???
3. xyz-color is only a bit better than xyz
4. xyzmid-color is much better than xyz-color
5. **xyzmid-color is normally much better than xyz-xyzmid-color ???**

1.6 model

batch size: 50
data: xyz_midnorm_block-color_1norm
epoch_num = 600
sampling & grouping: stride_0d1_step_0d1_bmap_nh5_12800_1d6_2_fmn3-600_64_24-60_16_12-0d2_0d6_1d2-0d2_0d6_1d2

model	acc	loss
3A	0.909	0.248
3AG	0.913	0.231
4AG	0.912	0.232

batch size: 32
data: xyz_midnorm_block-color_1norm
sampling & grouping: stride_0d1_step_0d1_bmap_nh5_12800_1d6_2_fmn6-2048_256_64-32_32_16-0d2_0d6_1d2-0d1_0d3_0d6
matterport3d
feed_data_elements:['xyz_midnorm_block', 'color_1norm']
feed_label_elements:['label_category', 'label_instance']
train data shape: [362 12800 6]
test data shape: [384 12800 6]
max epoch = 500

model	acc	loss
1AG	0.944/0.431	0.161/4.633
4AG	0.835/0.401	0.520/3.644

1.7 integration: matterport3d

stride_0d1_step_0d1_bmap_nh5_12800_1d6_2_fmn3-512_64_24-48_16_12-0d2_0d6_1d2-0d2_0d6_1d2 17D_1LX_1pX_29h_2az				
model	batch size batch num shuffle	lr ds	data elements	epoch-acc mean-std train/eval

1aG	30/60	0.005	'xyz_midnorm_block', 'color_1norm', 'nxnynz'	250-0.981
1DSaG	30/60	0.001-40	'xyz_midnorm_block', 'color_1norm', 'nxnynz'	300-0.914-0.775
1DSaG	30/60	0.001-40	'xyz_midnorm_block', 'color_1norm', 'nxnynz'	300-0.914-0.775
1DSaG kp0.5	30/60	0.001-80 300-3e-4	'xyz_midnorm_block', 'color_1norm', 'nxnynz'	300-0.942-0.842
1DSaG kp0.2	30/60	0.001-80 300-3e-4	'xyz_midnorm_block', 'color_1norm', 'nxnynz'	300-0.928-0.797
1DSaG kp0.5	30/60	0.005-80 300-1.7e-3	'xyz_midnorm_block', 'color_1norm', 'nxnynz'	300-0.970-0.916
1DSaG kp0.2	30/60	0.005-80 300-1.7e-3	'xyz_midnorm_block', 'color_1norm', 'nxnynz'	300-0.966-0.924
1DSaG kp0.8	30/60	0.005-80 300-1.7e-3	'xyz_midnorm_block', 'color_1norm', 'nxnynz'	300-0.976-0.933 500-0.984-0.954
1aG	30/1083	0.003	'xyz_midnorm_block', 'color_1norm', 'nxnynz'	200-0.947
1aG	30/1083	0.01	'xyz_midnorm_block', 'color_1norm'	200-0.783 500-0.791
1aG	30/1083	0.003/30 300-0.00012	'xyz_midnorm_block', 'color_1norm'	200-0.903 300-0.921
1bG	25/1083	0.001-30 100-3e-4 300-4e-5	'xyz_midnorm_block'	100-0.854 200-0.918 300-0.936
1bG	25/1083	0.001-30 100-3e-4 300-4e-5	'xyz_midnorm_block', 'color_1norm', 'nxnynz'	100-0.914 200-0.957 300-0.966
1bG	25/1083	0.02	'xyz_midnorm_block', 'color_1norm'	200-0.655 300-0.718
1bG	25/1083	0.02	'xyz_midnorm_block', 'color_1norm', 'nxnynz'	200-0.772 300-0.823
1bG	25/1083	0.001	'xyz'	200-0.772 90-0.553-0.210

4bG	25/1083	0.001-30 100-3e-4 200-1e-4 300-4e-5	'xyz_midnorm_block', 'color_1norm', 'nxnynz'	100-0.752 200-0.816 300-0.832
2 1DSaG	30/1083	0.002-80	'xyz_midnorm_block', 'color_1norm', 'nxnynz'	200-0.930-0.830/0.450 460-0.952-0.881/0.471
1aG	30/19755	0.001-30 50-7e-4 100-3e-4	'xyz_midnorm_block', 'color_1norm','nxnynz'	50-0.752/0.580 100-0.843/0.574 (NoShuf) 102-0.806/0.570 (Shufle)
1bG	25/19755	0.001-30	'xyz_midnorm_block', 'color_1norm','nxnynz'	38-0.719/0.587 80-0.823/0.583 (NoShuf) 81-0.782/0.587 (Shufle)
1aG	30/19755	0.02	'xyz_midnorm_block', 'color_1norm'	56-0.562
1aG	30/19755	0.02 127-0.00483	'xyz_midnorm_block', 'color_1norm', 'nxnynz'	87-0.616 127-0.686
1bG	25/18737	0.001 N	'xyz_midnorm_block', 'color_1norm', 'nxnynz'	24-0.682/0.509 70-0.858/0.509
1bG	25/18737	0.001 Y	'xyz_midnorm_block', 'color_1norm', 'nxnynz'	24-0.738/0.573 70-0.876/0.563 90-0.897 /0.561
4bG	25/18737	0.001 Y	'xyz_midnorm_block', 'nxnynz'	24-0.576/0.545
4bG	25/18737	0.001 Y	'xyz_midnorm_block', 'color_1norm', 'nxnynz'	24-0.594/0.569
1DSaG	30/18737	0.002-80 Y	'xyz_midnorm_block', 'color_1norm', 'nxnynz'	20-0.688-0.394/0.428-0.224 36-0.742/0.395
1DSaG	30/18737	0.007-80 Y	'xyz_midnorm_block', 'color_1norm', 'nxnynz'	20-0.725-0.453/0.435-0.206 38-0.783/0.396
Conclusion: 1: nxnynz helps a lot 2: 1bG is much deeper than 1aG, why worse than 1aG 3: learning rate is important, cannot be too large				

1.8 multi scales & mat 1083

nh5: stride_0d1_step_0d1_pl_nh5-1d6_2/17D_1LX_1pX_29h_2az bxmh5: stride_0d1_step_0d1_bxmh5-12800_1d6_2_fmn4-480_80_24-80_20_10-0d2_0d6_1d2-0d2_0d6_1d2-3A1					
model	bs/bn	lr-decay	elements	loss weight in drop	epoch-pacc-cacc train/eval
4bG_111	20/1083	2-40	xyz_midnorm_block-color_1norm-nxnynz	E	110-0.898-0.763 160-0.931-0.827 300-0.967-0.915
4bG_444	15/1083	3-40	xyz_midnorm_block-color_1norm-nxnynz	E	60-0.729-0.614 100-0.857-0.721 160-0.920-0.834 260-0.952-0.890 300-0.958-0.913
4bG_444	15/1083	2-40	xyz_midnorm_block-color_1norm-nxnynz	E	60-0.778-0.608 100-0.878-0.758 160-0.930-0.838 260-0.957-0.901 300-0.964-0.912
4bG_144	18/1083	2-40	xyz_midnorm_block-color_1norm-nxnynz	E	60-0.786-0.637 100-0.876-0.767 160-0.926-0.820 260-0.959-0.885 300-0.962-0.906
4bG_114	20/1083	2-40	xyz_midnorm_block-color_1norm-nxnynz	E	60-0.772-0.611 100-0.874-0.764 160-0.926-0.851 260-0.958-0.893 /par 300-0.963-0.904
3aG_444	45/1083	2-40	xyz_midnorm_block-color_1norm-nxnynz	E	60-0.893-0.737 100-0.908-0.786 /par 160-0.934-0.833 260-0.950-0.868 /par 300-0.952-0.882
2aG_144	30/1083	2-40	xyz_midnorm_block-color_1norm-nxnynz	E	60-0.890-0.754 100-0.922-0.820 /par 160-0.942-0.858: 260-0.957-0.897 /par 300-0.960-0.911

1.9 multi scales & mat 21826

nh5: stride_0d1_step_0d1_pl_nh5-1d6_2/ bxmh5: stride_0d1_step_0d1_bxmh5-12800_1d6_2_fmn4-480_80_24-80_20_10-0d2_0d6_1d2-0d2_0d6_1d2-3A1 eval: 17D_1LX_1pX_29h_2az					
model	bs/bn	lr-decay	elements	loss weight in drop	epoch-pacc-cacc train/eval
4bG_114	20/1080	1-30	xyz midnorm color nxnynz	E N	40-0.784-0.545/0.579-0.451 80-0.883-0.699/0.584-0.439 140-0.925-0.795/0.575-0.429
4bG_111	20/1080	2-30	xyz midnorm color nxnynz	E N	40-0.737-0.489/0.587-0.412 80-0.836-0.614/0.582-0.411 95-0.867/0.588
4bG_144	20/1200	2-30	xyz midnorm color nxnynz	E N	40-0.761-0.543/0.601-0.416 80-0.864-0.693/0.602-0.426 95-0.888/0.597
Conclusion: (1) Nein 114 is better than 111					

1.10 multi scales & scannet 12887

nh5: stride_0d1_step_0d1_pl_nh5-1d6_2/
bxmh5: stride_0d1_step_0d1_bxmh5-12800_1d6_2_fmn4-480_80_24-80_20_10-0d2_0d6_1d2-0d2_0d6_1d2-3A1
eval: test

model	bs/bn	lr-decay	elements	loss weight in drop	epoch-pacc-cacc train/eval
2aG_144	30/420	2-30	xyz midnorm	E N	40-0.833-0.546/0.686-0.89 100-0.926-0.727/0.683-0.326
3aG_144	48/260	2-30	xyz midnorm	E N	40-0.841-0.530/0.668-0.346 100-0.924-0.709/0.673-0.327 200-0.949-0.782/0.673-0.332 **300-0.955-0.802/0.671-0.330**
4bG_111	22-580	2-30	xyz midnorm	E N	60-0.738-0.434/0.706-0.344 100-0.796-0.506/0.699-0.315 180-0.863-0.589/0.695-0.308
4bG_111	22-580	7-30	xyz midnorm	E N	60-0.705-0.378/0.684-0.362
4bG_144	18-700	2-30	xyz midnorm	E N	40-0.714-0.470/0.6910.433 100-0.794-0.481/0.682-0.393 160-0.849-0.582/0.676-0.362
4aG_1a4	55-220	2-30	xyz midnorm	CN N	40-0.775-0.482/0.654-0.304 100-0.877-0.637/0.661-0.298 160-0.901-0.690/0.660-0.311 220-0.908-0.707/0.655-0.334
4aG_1a4	55-220	2-30	xyz midnorm	E N	40-0.819-0.527/0.684-0.333 100-0.923-0.706/0.681-0.304

Conclusion:
(1) 3aG is much better than 4bG. Potential reasons:(a) 4bG is too wide and deep, so that needs more time to train. (b) The batch size of 4bG is too small
(2) nein 144 seems is not better than 111
(3)Learning rate 0.002 is better than 0.007
(4)Loss weight CN does not help

1.11 integration: scannet

stride_0d1_step_0d1_bmap_nh5_12800_1d6_2_fmn3-256_48_16-56_8_8-0d2_0d6_1d2-0d2_0d6_1d2 scannet train					
model	loss: E,N,C input drop (No)	batch size batch num shuffle	lr ds	data elements	epoch-point ac-class ac train/eval
1bG	E	25/12887 test Y	0.001 40	xyzmid	23-0.732-0.326/0.664-0.260 25-0.746-0.340/0.669-0.273
1bG	N	25/12887 Y	0.001 40	xyzmid	25-0.733-0.390/0.666-0.252
1bG	C	25/12887 Y	0.001 40	xyzmid	25-0.703-0.356/0.655-0.252
1bG	CN	25/12887 Y	0.001 40	xyzmid	25-0.681-0.366/0.611-0.237
1DSaG	E idp9	30/12887 Y	0.003 80	xyzmid	40-0.738-0.376/0.513-0.228 90-0.832/0.496
1bG	E	25/13091 train_300 Y	0.002 80	xyzmid	60-0.765-0.389/0.700-0.252
1bG	E	25/13091 Y	0.003 80	xyzmid	10-0.646/0.689 60-0.753-0.349/0.691-0.234 100-0.833-0.480/0.672-0.261
1bG	CN	25/13091 Y	0.002 80	xyzmid	60-0.738-0.409/0.670-0.237
1bG	E idp9	25/13091 Y	0.003 80	xyzmid	10-0.641/0.585 16-0.646/0.633
1DSaG	E	30/13091 Y	0.003 80	xyzmid	40-0.794-0.456/0.420-0.154 100-0.872-0.602/0.417-0.153
Conclusion:					
4bG	CN	25/2998-3521 Y	0.001 40	xyzmid	142-0.726-0.445/0.625-0.242
4bG	E	25/2998-3521 Y	0.001 40	xyzmid	145-0.792-0.506/0.656-0.257

1.12 Semantic segmentation expamples

1.12.1 good: 1083, train, 0.946

log: log-model_1bG-gsbb_3B1-bs25-lr1-ds_30-xyz_midnorm_block-color_1norm-nxnynz-12800-mat_1083

model: 1bG

sampling & grouping:

stride_0d1_step_0d1_bmap_nh5_12800_1d6_2_fmn3-512_64_24-48_16_12-0d2_0d6_1d2-0d2_0d6_1d2

batch size: 25

learning rate: 0.001000

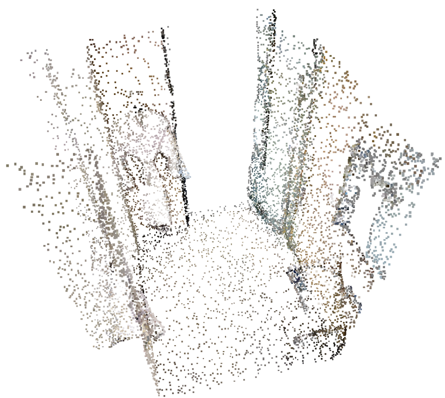
decay_epoch_step: 30

matterport3d

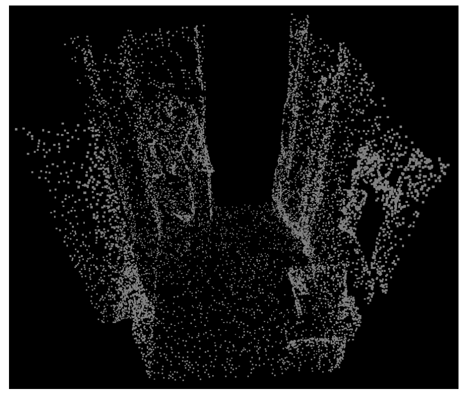
feed_data_elements:['xyz_midnorm_block', 'color_1norm', 'nxnynz']

feed_label_elements:['label_category', 'label_instance']

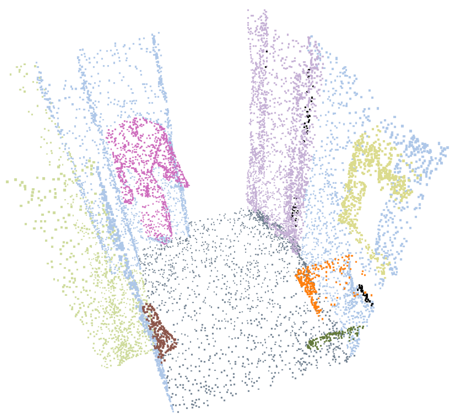
train data shape: [1083 12800 9]



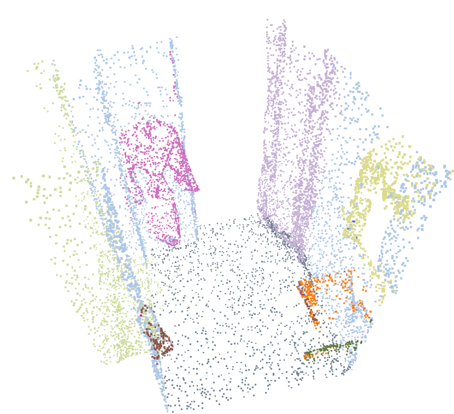
(a) colorized point cloud



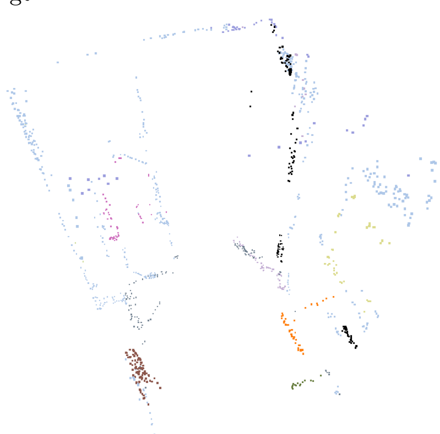
(b) raw point cloud



(c) gt

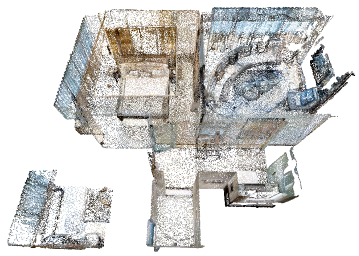


(d) pred

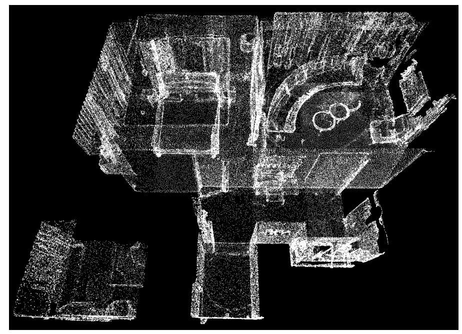


(e) err

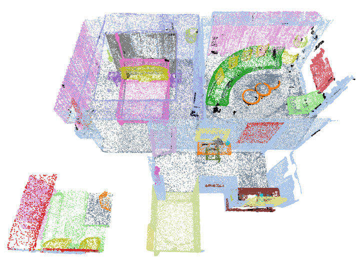
Figure 4: 17DRP5sb8fy_1_2_a946



(a) colorized point cloud

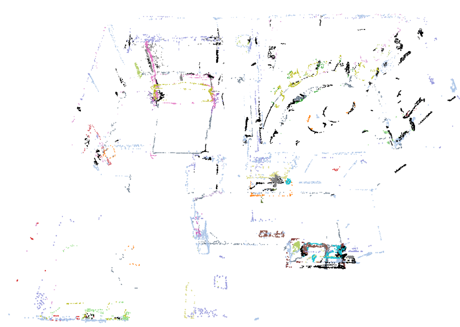


(b) raw point cloud



(c) gt

(d) pred



(e) err

Figure 5: 17DRP5sb8fy_0_25_a946

1.12.2 bad: 18737,eval 0.071

model: 1bG
sampling & grouping: stride_0d1_step_0d1_bmap_nh5_12800_1d6_2_fmn3-512_64_24-48_16_12-0d2_0d6_1d2-0d2_0d6_1d2
batch size: 25
learning rate: 0.001000
decay_epoch_step: 50
epoch 0 train IsShuffleIdx: True

epoch 0 train IsShuffleIdx: True
matterport3d
feed_data_elements:['xyz_midnorm_block', 'color_1norm', 'nxnynz']
feed_label_elements:['label_category', 'label_instance']
train data shape: [18737 12800 9]
test data shape: [4172 12800 9]

Figure 6: qoi_r1Q_r47_rPc_rqf_2_3_a0d071 (raw,gt,pred,err,crt)

1.13 point++

1.13.1 scannet seg

each room as a block, total 40 block			
batch size batch num	lr ds	data elements	epoch-point ac-class ac train/eval/eval whole scene
30/40	0.001	xyzmid	200-0.675/0.757-0.54/0.799-0.52
25	0.001	xyzmid	200-0.689/0.787-0.556/0.815-0.517i

1.14 whole room global block & multi scales & scan 305

nh5: stride_0d1_step_0d1_pl_nh5-1d6_2/ bxmh5: stride_0d1_step_0d1_bxmh5-12800_1d6_2_fmn4-480_80_24-80_20_10-0d2_0d6_1d2-0d2_0d6_1d2-3A1 eval: 17D_1LX_1pX_29h_2az					
model	bs/bn	lr-decay	elements	loss weight in drop	epoch-pacc-cacc train/eval
5bG_114	6/40	2-30	xyz midnorm color	E N	100-0.805-0.645 200-0.865-0.708 300-0.880-0.773
5aG_114	2/140	2-30	xyz midnorm color	E N	100-0.802-0.619 200-0.873-0.694 300-0.895-0.784
5aG_114	2/140	2-30	xyz midnorm color	E idp5	100-0.807-0.687 200-0.877-0.729 300-0.895-0.778
Conclusion: (1) Nein 114 is better than 111					

1.15 Sparse voxel net

1.15.1 90000

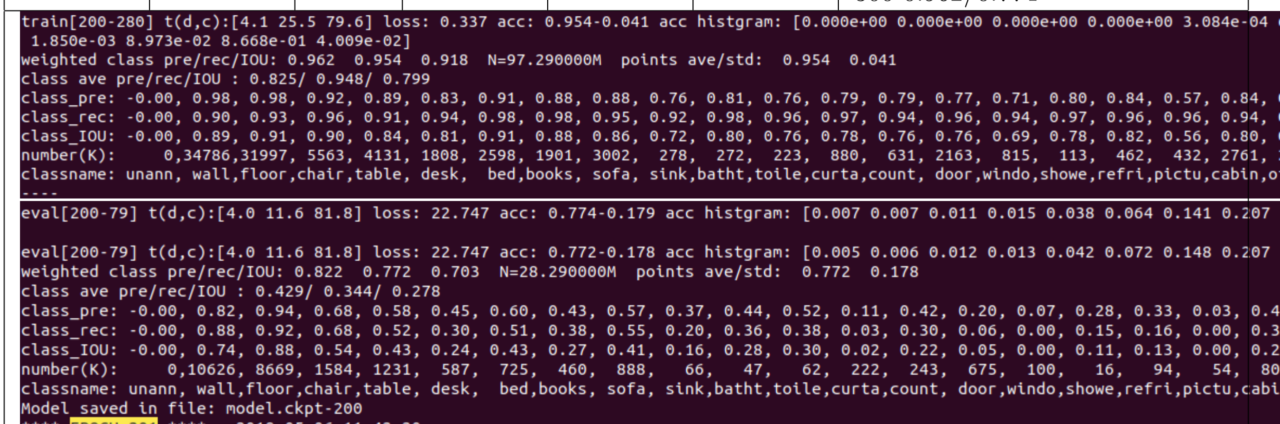
nh5: 90000_gs-3d6_-6d3/
bxmh5: 90000_gs-3d6_-6d3_fmn1444-6400_2400_320_32-32_16_32_48-0d1_0d3_0d9_2d7-0d1_0d2_0d6_1d8-pd3-mbf-4A1
eval: test

model	bs/bn	lr-decay	elements	norm in-net aug	loss weight in drop	epoch-pacc-cacc train/eval
5VaG_114	16/113	1-50	xyz mid	No	Num lw dp:3N5 N shuffle	20-0.764-0.556/0.645-0.362 40-0.864-0.695/0.675-0.351 100-0.935-0.842/0.682-0.380 200-0.961-0.897/0.671-0.374 300-0.969-0.920/0.676-0.360
5VaG_114	30/40	2-40	xyz mid color	No	Num lw dp:466 Y shuffle	20-0.605-0.443/0.577-0.381 40-0.668-0.490/0.544-0.372 100-0.795-0.581/0.653-0.384 120-0.805-0.594/0.676-0.377
5VaG_114	30/40	2-40	xyz mid color	No	Num lw dp:4N6 Y shuffle	20-0.677-0.520/0.607-0.373 40-0.801-0.624/0.659-0.373 100-0.906-0.754/0.687-0.368 120-0.911-0.776/0.692-0.421
5VaG_114	30/40	2-40	xyz mid color	No	Num lw dp:N66 Y shuffle	20-0.614-0.457/0.566-0.344 40-0.685-0.500/0.552-0.356 60-0.741-0.552/0.650-0.357 80-0.770-0.564/0.649-0.392 98-0.797-/0.675
5VaG_114	30/40	2-40	xyz mid color	Mid	Num lw dp:555 Y shuffle	100-0.746/0.649 300-0.823-0.608/0.682-0.377
5VaG_114	39/40	2-40	xyz mid color	Mid	Num lw dp:5N5 Y shuffle	40-0.839-0.649/0.656-0.353 100-0.918-0.771/0.681-0.381
5VaG_114	36/40	1-40	xyz mid color	Mid	Num lw dp:NN5 Y shuffle	40-0.833-0.655/0.628-0.329 100-0.916-0.772/0.682-0.339 178-0.941/0.686
5VaG_114	7/240	2-40	xyz mid color	Mid Group Norm	Num lw dp:NN5 Y shuffle	40-0.664/0.577 100-0.816-0.586 150-0.869-0.592
5VaG_114	9	2-40	xyz mid color	Rotate Ref	Num lw dp:NN5 Y shuffle	40-0.787-0.620/0.662-0.401 100-0.907-0.755/0.699-0.410 200-0.939-0.816/0.6990.430 300-0.950-0.845/0.691-0.430

Conclusion:
Mid norm in sub block seems worse than no.
The infuence of input drop seems not obvious.
Dropout of cnn (0.5) makes the net really hard to train. Seems no good for overfitting.
Group norm is poor.

1.15.2 30000

bxmh5: 30000_gs-2d4_-3d4_fmn1444-2048_1024_128_24-48_32_48_27-0d1_0d4_1_2d2-0d1_0d2_0d6_1d2-pd3-mbf-4B1 eval: test Void point id deleted						
model	bs	lr-decay	elements	norm in-net aug	loss weight in drop	epoch-pacc-cacc train/eval
5VaG_114	9	2-40	xyz mid color	Rotate Ref	Num lw dp:5N5 Y shuffle	20-0.749/0.645 30-0.810/0.705 40-0.870/0.744 80-0.917-0.737/0.7540.422 120-0.929-0.765 200-0.953-0.774 300-0.962/0.774



```
train[200-280] t(d,c):[4.1 25.5 79.6] loss: 0.337 acc: 0.954-0.041 acc histgram: [0.000e+00 0.000e+00 0.000e+00 0.000e+00 3.084e-04
 1.850e-03 8.973e-02 8.668e-01 4.009e-02]
weighted class pre/rec/IOU: 0.962  0.954  0.918  N=97.290000M  points ave/std:  0.954  0.041
class ave pre/rec/IOU : 0.825/ 0.948/ 0.799
class_pre: -0.00, 0.98, 0.98, 0.92, 0.89, 0.83, 0.91, 0.88, 0.88, 0.76, 0.81, 0.76, 0.79, 0.79, 0.77, 0.71, 0.80, 0.84, 0.57, 0.84,
class_rec: -0.00, 0.90, 0.93, 0.96, 0.91, 0.94, 0.98, 0.98, 0.95, 0.92, 0.98, 0.96, 0.97, 0.94, 0.96, 0.94, 0.97, 0.96, 0.96, 0.94,
class_IOU: -0.00, 0.89, 0.91, 0.90, 0.84, 0.81, 0.91, 0.88, 0.86, 0.72, 0.80, 0.76, 0.78, 0.76, 0.76, 0.69, 0.78, 0.82, 0.56, 0.80,
number(K):     0,34786,31997, 5563, 4131, 1808, 2598, 1901, 3002,  278,  272,  223,  880,  631, 2163,  815,  113,  462,  432, 2761,
classname: unann, wall,floor,chair,table, desk,  bed,books, sofa, sink,batht,toile,curta,count, door,windo,showe,refri,pictu,cabin,o
----
eval[200-79] t(d,c):[4.0 11.6 81.8] loss: 22.747 acc: 0.774-0.179 acc histgram: [0.007 0.007 0.011 0.015 0.038 0.064 0.141 0.207

eval[200-79] t(d,c):[4.0 11.6 81.8] loss: 22.747 acc: 0.772-0.178 acc histgram: [0.005 0.006 0.012 0.013 0.042 0.072 0.148 0.207
weighted class pre/rec/IOU: 0.822  0.772  0.703  N=28.290000M  points ave/std:  0.772  0.178
class ave pre/rec/IOU : 0.429/ 0.344/ 0.278
class_pre: -0.00, 0.82, 0.94, 0.68, 0.58, 0.45, 0.60, 0.43, 0.57, 0.37, 0.44, 0.52, 0.11, 0.42, 0.20, 0.07, 0.28, 0.33, 0.03, 0.4
class_rec: -0.00, 0.88, 0.92, 0.68, 0.52, 0.30, 0.51, 0.38, 0.55, 0.20, 0.36, 0.38, 0.03, 0.30, 0.06, 0.00, 0.15, 0.16, 0.00, 0.3
class_IOU: -0.00, 0.74, 0.88, 0.54, 0.43, 0.24, 0.43, 0.27, 0.41, 0.16, 0.28, 0.30, 0.02, 0.22, 0.05, 0.00, 0.11, 0.13, 0.00, 0.2
number(K):     0,10626, 8669, 1584, 1231,  587,  725,  460,  888,   66,   47,   62,  222,  243,  675,  100,   16,   94,   54,  80
classname: unann, wall,floor,chair,table, desk,  bed,books, sofa, sink,batht,toile,curta,count, door,windo,showe,refri,pictu,cabi
Model saved in file: model.ckpt-200
**** EPOCH 201 ****   2018-05-06 11:42:20
```

Conclusion:

1.16 Charles Point++, fast distance sampling

1.16.1 MODELNET40

config	epoch-train acc/eval acc-eval cls acc
batch_size=32, decay_rate=0.7, decay_step=200000, learning_rate=0.001, log_dir='log', max_epoch=251, model='pointnet2_cls_ssg', momentum=0.9, normal=False, num_gpus=2, num_point=1024, optimizer='a dam	4-0.746/0.819-0.748 10-0.802/0.848-0.788 40-0.886/0.875-0.858 60-0.916/0.892-0.859
aug=True, batch_size=32, decay_rate=0.7, decay_step=200000, gpu=1, indrop=True, learning_rate=0.001, log_dir='log', max_epoch=251, model='pointnet2_cls_ssg', momentum=0.9, normal=True, num_point=8 192, optimizer='adam', shuffle=True	10-0.806/0.853-0.817 60-0.939/0.8946/0.868 100-0.971/0.9036-0.883

1.17 MODELNET40, My point++

After fix shuffle problem

1.17.1 3m

bxmh5:1024_gs3_3_fmn1444-1024_320-24_32-0d2_0d4-0d1_0d2-pd3-2M1 No block merging. Replicate redundant.						
model	bs	lr bn decay	elements group pos	norm in-net aug	loss weight in drop	epoch-pacc-cacc train/eval
3m	36	1-30 7-7	xyzg bc	Rotate Ref	E, NN5	1-0.707/0.775 2-0.744/0.804 4-0.785/0.817 10-0.831/0.826 30-0.906/0.857 60-0.967/0.852
3m	36	1-30 7-7	xyzg mean	Rotate Ref	E, NN5	1-0.704/0.754 2-0.747/0.781 4-0.783/0.810 10-0.839/0.835 60-0.969/0.865
3m	36	1-30 7-7	xyzrsg mean	Rotate Ref	E, NN5	10-0.844/0.830 60-0.978/0.868
3m	36	1-30 7-7	xyzrsg, nxnynz mean	Rotate Ref	E, NN5	10-0.887/0.881 60-0.985/0.890
bxmh5:4096_gs3_3_fmn1444-1024_320-48_32-0d2_0d4-0d1_0d2-pd3-2M2 No block merging. Replicate redundant.						
3m	28	1-30 7-7	xyzg mean	Rotate Ref	E, NN5	1-0.698/0.795 10-0.834/0.853 60-0.962/0.874
3m	28	1-30 7-7	xyzg bc	Rotate Ref	E, NN5	1-0.703/0.786 10-0.832/0.847 60-0.957/0.867
3m	28	1-30 7-7	xyzrsg mean	Rotate Ref	E, NN5	1-0.695/0.764 10-0.840/0.847 60-0.976/0.880
3m	28	1-30 7-7	xyzrsg, nxnynz mean	Rotate Ref	E, NN5	1-0.747/0.814 10-0.882/0.879 60-0.985/0.897 160-0.998/0.905
bxmh5:4096_mgs1_gs2_2_fmn14_mvp1-1024_240_1-48_27_160-0d2_0d4-0d1_0d2-pd3-mbf-neg-2M2p						
3Vm	58	1-30 5-5	xyzrsg nxnynz mean	Rotate Ref	E, NN5	1-0.549/0.635 10-0.852/0.812 30-0.961/0.840 60-0.984/0.837 100-0.994/0.828
3Vm	58	1-30 5-5	xyzrsg nxnynz mean	Rotate Ref	E, 3N5	1-0.543/0.614 10-0.852/0.798 20-0.926/0.825 27-0.955/0.800
3m	52	1-30 5-5	xyzrsg nxnynz mean	RotateRef	E, NN5	10-9.763/0.803 60-0.879-0.876 100-0.921/0.872
Conclusion: (0) The performance of pointnet++ based on farest distance sampling is better. The reason may be on line sampling, the lost part is each epoch is different.						

1.17.2 4m

bxmh5:4096_mgs1_gs2_2d2_fmn1444_mvp1-3200_1024_48_1-18_24_56_56-0d1_0d2_0d6-0d0_0d1_0d4-pd3-mbf-neg-3M1						
model	bs	lr bn decay	elements group pos	aug	loss weight in drop	epoch-pacc-cacc train/eval
4m	16	1-30	xyzrsg-nxnynz	RotateRef	E NN5	10-0.688/0.755 60-0.845/0.859 100-0.873/0.875 200-0.914/0.880
4m	16	1-30	xyzg	RotateRef	E NN5	10-0.635/0.680 60-0.784/0.807 120-0.837/0.830 241-0.878/0.846
4m	16	1-30	xyzrsg-nxnynz	RotateRef	E 3N5	10-0.623/0.671 60-0.766/0.793 120-0.821/0.810
4m	16	1-30	xyzg-nxnynz	RotateRef	E NN5	10-0.691/0.726 60-0.840/0.853 1100-0.874/0.862 120-0.886/0.861
4Vm	56	1-30	xyzg-nxnynz	RotateRef	E NN5	10-0.894/0.817 60-0.993/0.834 79-0.997/0.842

bxmh5:4096_mgs1_gs2_2d2_fmn1444_mvp1-3200_1024_48_1-18_24_56_56-0d1_0d2_0d6-0d0_0d1_0d4-pd3-neg-3M1 No Block Merging						
4m	16	1-30	xyzg- nxnynz	RotateRef	E NN5	10-0.711/0.705 60-0.878/0.850 100-0.912/0.869
4m	16	1-30	xyzg	RotateRef	E NN5	10-0.644/0.682 60-0.778/0.802 100-0.816/0.827
4m	16	1-30	xyzs	RotateRef	E NN5	10-0.637/0.676 60-0.784/0.792 100-0.821/0.828
4m	16	1-30	xyzr	RotateRef	E NN5	10-0.658/0.698 60-0.801/0.821 70-0.815/0.819
4Vm	30	1-30	xyzg- nxnynz	RotateRef	E NN5	10-0.890/0.830 51-0.992/0.841 100-0.999/0.847 113-0.999/0.853 129-0.999/0.847
4Vm	30	1-30	xyzg- nxnynz	RotateRef	E 575	10-0.846/0.796 49-0.982/0.841
4Vm	30	1-30	xyzg- nxnynz	RotateRef	E N75	10-0.848/0.824 60-0.989/0.831
4Vm	28	1-30	xyzg	RotateRef	E NN5	10-0.884/0.821 60-0.994/0.838 80-0.996/0.842
4Vm	30	1-30	xyzg nxnynz	RotateRef RotateVox	E NN5	10-0.752/0.663 60-0.949/0.828 79-0.969/0.814

Conclusion:
(1) Input drop out increase overfitting here. This is not reasonable!
(2) Learns much slower than 3m?
(3) The variance is greater, maybe because of small bacth size.
(4) Block merge is a little bit helpful for pointnet++ (5) xyzs is a little bit better than xyzg for pointnet++

bxmh5:4096_mgs1_gs2_2d2_fmn1444_mvp1-3200_1024_48_1-18_24_56_56-0d1_0d2_0d6-0d0_0d1_0d4-pd3-neg-3M1 No Block Merging						
4Vm-S2	30	3-30	xyzs	RotateRef RotateVox	E NN5	10-0.885/0.815 30-0.979/0.847 40-0.982/0.855
4Vm-S2	30	3-30	xyzs nxnynz	RotateRef RotateVox	E NN5	10-0.889/0.843 30-0.980/0.843 60-0.995/0.848
4Vm-S3	30	3-30	xyzs nxnynz	RotateRef RotateVox	E NN5	10-0.894/0.838 30-0.981/0.851 60-0.996/0.853
4Vm-S4	30	3-30	xyzs nxnynz	RotateRef RotateVox	E NN5	10-0.888/0.862 30-0.976/0.865 48-0.989/0.850
4Vm-S3 normal label	50	1-20	xyzg	RotateRef RotateVox	E NN5	10-0.930/0.829 30-0.983/0.841 50-0.991/0.847 69-0.998/0.861 80-0.998/0.855
4Vm-S3 normal label	50	1-20	xyzrsg	RotateRef RotateVox	E NN5	10-0.923/0.847 30-0.981/0.837 52-0.996/0.865 53-0.993/0.856
4Vm- S2L2	29	1-30	xyzs nxnynz	RotateRef RotateVox	E NN5	10-0.900/0.841 30-0.981/0.838 60-0.996/0.851
4Vm- S3L3	29	1-30	xyzs nxnynz	RotateRef RotateVox	E NN5	10-0.899/0.841 30-0.992/0.847 60-0.996-0.854
Conclusion:						

bxmh5:4096_mgs1_gs2_2d2_fmn1444_mvp1-3200_1024_48_1-18_24_56_56-0d1_0d2_0d6-0d0_0d1_0d4-pd3-neg-3M1 No Block Merging						
4Vm1	24	1-10	xyzs nxnynz	RotateIn	E	10-0.862/0.859
4Vm1-S3	24	1-10	xyzs nxnynz	RotateIn	E	10-0.874/0.867 11-0.883/0.873 20-0.952/0.868 40-0.990/0.862

1.17.3 5m

bxmh5:10000_gs3_3d5_fmn1444_mvp1-2560_1024_80_16_1-24_32_48_27_48-0d0_0d2_0d5_1d1-0d0_0d1_0d3_0d6-pd3-mbf-neg-4M1						
model	bs	lr bn decay	elements group pos	aug	loss weight in drop	epoch-pacc-cacc train/eval
5m1	32	1-30	xyzg	RotateRef	E	60-0.956/0.843 200-0.998/0.845
5m	48	1-30	xyzrsg-nxnyznz	RotateRef	E	60-0.852/0.837 100-0.894/0.866 119-0.923-0.878 150-0.938-0.867
5Vm	32	1-30	xyzr	N	E	10-0.853/0.802 60-0.985/0.829 160-0.998/0.831
Conclusion: (0)						

2 res3d

2Mpp				
50-3Vm	32	xyzg-normal	all	12-0.905/0.782 96-0.994/0.832
50-3Vm	32	xyzg-normal	none	12-0.936/0.826 26-0.988/0.851
50-3Vm	48	xyzg-normal	none	12-0.843/0.779 20-0.920/0.800
34-3Vm	64	xyzg-normal	none	12-0.888/0.697 30-0.905/0.803
Conclusion: (0)Something wrong with aug				