Survey and Evaluation of Neural 3D Shape Classification Approaches Supplementary Material

Martin Mirbauer, Miroslav Krabec, Jaroslav Křivánek, Elena Šikudová

1 Our ShapeNetCore Split

Due to uneven distribution of training and testing subsets within categories, duplicates and some models being present in multiple categories in the official ShapeNetCore split, we generated our split where each category has 70% of objects used for training, 10% for validation and 20% for testing. Table 1 shows the category sizes and train/test/val subset sizes. The final split can be downloaded from the project webpage.[1]

TABLE 1
List of ShapeNetCore categories and the number of training and test models in each category in our split

Category	Train/Test/Val	Category	Train/Test/Val
table	5862/1676/838	pistol	185/53/27
chair	4616/1319/660	telephone	180/52/26
airplane	2831/809/404	piano	167/48/24
car	2460/703/351	bed	153/44/22
sofa	2103/601/300	stove	152/44/22
rifle	1631/467/233	mug	150/43/21
lamp	1621/464/232	bowl	121/35/17
vessel	1356/388/194	washer	115/34/17
bench	1213/347/173	printer	115/33/17
loudspeaker	1114/319/159	helmet	113/33/16
cabinet	1082/310/155	skateboard	106/31/15
display	755/216/108	microwave	105/31/15
bus	657/188/94	tower	86/25/12
bathtub	598/171/85	camera	79/23/11
cellular telephone	580/166/83	can	73/21/11
		basket	71/21/10
guitar	557/160/80	pillow	66/20/10
faucet	519/149/74	mailbox	65/19/9
clock	450/129/64	dishwasher	63/19/9
pot	384/110/55	rocket	59/17/9
jar	383/110/55	bag	57/17/8
bottle	333/96/48	birdhouse	51/15/7
laptop	315/91/45	earphone	51/15/7
bookshelf	308/89/44	microphone	46/14/7
knife	297/85/42	remote control	52/14/7
train	272/78/39		
motorcycle	235/68/34	computer keyboard	45/13/6
ashcan	225/65/32		
file	199/58/29	cap	38/12/6
(continues in the next column)		**Total**	**35513/10178/5078**

1. https://cgg.mff.cuni.cz/~martinm/papers/2021-survey-eval

2 Comparison of Different Data Conversion Methods

Here we show outputs of different conversion methods for each representation. The model *airplane_0627* from ModelNet40 is used as an example 3D shape. Some original conversion tools were not included in our framework because they use of commercial software ([27] and [44]) or they were not publicly available ([61]).

Figure 1 shows slight differences in the voxelization result of the OpenVDB library, which we used in our conversion, compared to the original voxelization provided by Brock et al. [27].

(a) Original voxel representation provided by authors of VRN

(b) Our voxelization using OpenVDB

Fig. 1. Illustration of voxel representation

For image-based multi-view approaches, we render three kinds of images: one using a physically based renderer *PBRT* [**?**] using a perspective (not orthographic) camera projection, and *shaded* and *depth* images using code by Su et al. [13]. Outputs of these rendering methods are shown in Figure 2, compared to the pre-rendered subset of ModelNet40 provided by Su et al. [44]. Across all three dataset variants used in our experiments, the *shaded* method yields the best results: on average 0.59 pp better than *depth*, which is on average better by 0.68 pp than *pbrt*.

The methods we used for point cloud generation by sampling a mesh surface also affect the achieved performance: the *uniform* sampling reached the highest maximum accuracy, followed by *lloyd* and *sobol*, but did not reach the

(a) Phong-shaded image from [44]

(b) Our *PBRT* rendering

(c) *depth* image rendered using code from [13]

(d) *shaded* image rendered using code from [13]

Fig. 2. Illustration of differently rendered airplane_0627 model

(a) Original point cloud provided by the authors of PointNet

(b) Uniform sampling

(c) Lloyd sampling

(d) Sobol sequence sampling

Fig. 3. Illustration of point cloud representations, each sampling contains 2048 points

performance of the authors-provided point clouds created using farthest point sampling. Figure 3 shows a visualization of the different point cloud sampling methods.

3 Training Parameters

The most important parameters we used to train the neural networks were set as follows:

- *vrn*
  training epochs: 20
  batch size: 24
  learning rate: 0.002 for 10 epochs and then 0.0002
  number of rotations: 12
- *octree*
  training epochs: 50
  batch size: 64
  learning rate: 0.1, divided by ten every ten epochs
  depth: 6
  number of rotations: 12
- *octree-adaptive*
  training epochs: 50
  batch size: 64
  learning rate: 0.1, divided by ten every ten epochs
  depth: 5
  number of rotations: 12
- *vgg*
  training epochs: 20
  batch size: 60
  learning rate: 0.0001, multiplied by 0.75 every three epochs
  number of views: 12
- *mvcnn2*
  training epochs: 30+30
  batch size: 64
  learning rate: 0.00005
  number of views: 12
- *rotnet*
  training epochs: 200
  batch size: 40
  learning rate: 0.0001 divided by ten every fifty epochs
  number of views: 12
- *seq2seq*
  training epochs: 200
  batch size: 32
  learning rate: 0.0002
  number of views: 12
- *pointnet*
  training epochs: 200
  batch size: 64
  number of points: 2048
  learning rate: 0.0001 multiplied by 0.8 every 20 epochs
  number of rotations: 12
- *pointnet2*
  training epochs: 200
  batch size: 32
  number of points: 2048
  learning rate: 0.0001 multiplied by 0.7 every 20 epochs
  number of rotations: 12
- *sonet*
  training epochs: 400
  batch size: 8
  number of points: 5000
  learning rate: 0.001 divided by two every 40 epochs
  number of rotations: 1

- *kdnet*
  training epochs: 200
  batch size: 16
  number of points: 2048
  learning rate: 0.001
  network depth: 10
  number of rotations: 12

4 Results

The results of our experiments are shown in Figure 6. In Table 3 we show the source data of the plot – the highest accuracies on test and validation sets achieved during our experiments and the test accuracy at the epoch where the highest validation accuracy was achieved.

5 Additional Experiments

In addition to evaluating the methods on the ModelNet40 and ShapeNetCore datasets split into train/test/val subsets, we also ran experiments on a train/test split and include a manually-aligned version of the ModelNet40 dataset[29]. The official train/test split was used in case of ModelNet40 and its aligned version. For ShapeNetCore, we used our own split. In all cases the shapes assigned to the test subset are the same as in the experiments in the main paper body. Using the two-way split allows us to use all the available data, not excluding the validation subset from training. Because the validation subset is not available for determining the training epoch count and possibly other hyperparameters, and because most of the reported results use the ModelNet40 train/test split and the reported accuracies appear to be the achieved maxima, we follow the same evaluation approach. In addition, we report the distribution of test subset accuracies *around the best epoch*: three epochs before and three epochs after the epoch with the highest test set accuracy. This is done to reduce the impact of the bias caused by taking just the maximum of noisy test set accuracies.

There are following changes to the dataset conversion, compared to the experiments in the body of the article. Instead of Poisson-sampled point clouds, we use a low-discrepancy Sobol' sequence for sampling the faces. However, the points within the chosen face is sampled uniformly, leading to small-scale point clusters. For rendering images for multi-view image-based networks, we additionally use a physically-based renderer PBRT, which could capture global illumination effects such as occlusion, which are not present in the images created by commonly-used rasterization or simple ray casting. Neither of these changes provide a significant improvement to the accuracy, therefore we don't include these methods in the main paper body.

The results of our experiments are shown in Figure 7. In Table 4 we present details about the distribution of the achieved accuracies during our experiments.

5.1 The Impact of Rotational Alignment

Overall, we do not observe a significant change in average or maximum accuracy on the aligned ModelNet40 dataset compared to the original ModelNet40; however, networks in each representation behave differently.

Volumetric grid-based networks *vrn* and *octree* tend to achieve slightly lower accuracy (by about 0.11 pp on average) on the aligned dataset. A possible explanation could be "cheating": the network may be exploiting the fact that some categories are more likely to be already axis aligned, which the network might be able to learn[2]. This distinction is lost in the ModelNet40-aligned dataset. The *octree-adaptive* network uses a plane in each leaf octree node to approximate the original shape – the planes' normal vectors can represent any direction making the representation less reliant on the shape alignment relative to the voxel grid.

Multi-view image-based approaches are unimpacted by the alignment, likely due to using 12 turntable views of the shape for both training and inference, making rotational alignment of the input irrelevant.

All point cloud-based methods in our experiments appear to benefit from the alignment, reaching on average 2.02 pp higher accuracy on the aligned dataset version. The *pointnet* network uses an initial transformation network (T-Net) to normalize the input object's rotation. Using a pre-aligned input simplifies T-Net's work – reducing possible rotational error should produce more consistent point sets, which are easier to detect in the following layers. Similarly, in the case of *pointnet2* and *sonet*, which detect and then aggregate local shape features, learned weights re-use is possible: thanks to the alignment, fewer fundamental shapes need to be detected. For *kdnet*, the alignment also helps since the kd-tree data structure uses axis-aligned splits.

5.2 The Impact of Dataset Size

Comparing the training performance on ShapeNetCore to the ModelNet40 dataset, most networks' accuracy improves (on average by 1.60 pp) with two exceptions: *octree-adaptive* and *kdnet* networks achieve a lower accuracy, possibly due to limited capacity. Both networks have the smallest model size of the evaluated networks within their representation. Another possible reason is their hyperparameters may have been optimized for a smaller ModelNet40 dataset. All other networks seem to benefit from a larger dataset, image-based networks showing the most consistent increase in accuracy (1.82 pp). These networks have the capacity to internally learn more features as the image-based models are largest of all the representations.

5.3 Relationship of achieved accuracy and computational cost

In Table 2 we show run times and sizes of the stored model for each network. Figures 4 and 5 show the relationships of accuracy and time or model size graphically.

2. E.g. some early convolutional kernels could be axis-aligned edge detectors, which would give lower activation if the object is unaligned.

TABLE 2
Table of approximate training times on ModelNet40 (time spent training one epoch) and approximate sizes of the saved models, roughly corresponding to the number of trainable parameters of the model

Network	Epoch time [min]	Model size [MB]
Volumetric grid		
vrn	407	52
octree	5.5	2.5
octree-adaptive	1.5	2.5
Multi-view		
mvcnn2	12.5	510
rotnet	6	230
vgg	29	550
seq2seq	0.6	30
Point cloud		
pointnet	4.5	40
pointnet2	2	17
sonet	1.5	10
kdnet	3	8

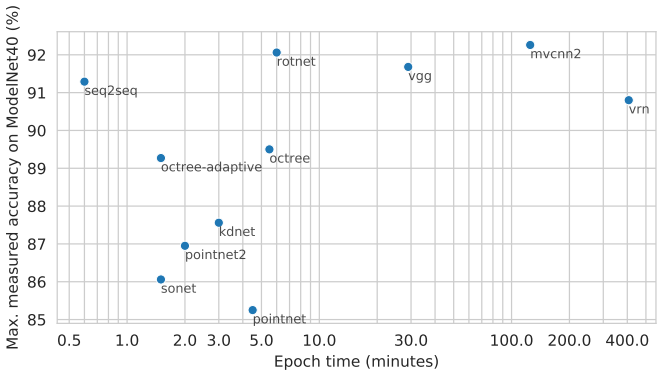

Fig. 4. Relationship of the achieved accuracy and training time.

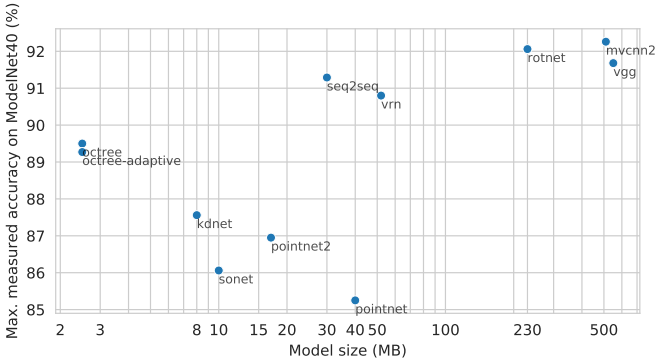

Fig. 5. Relationship of the achieved accuracy and model size.

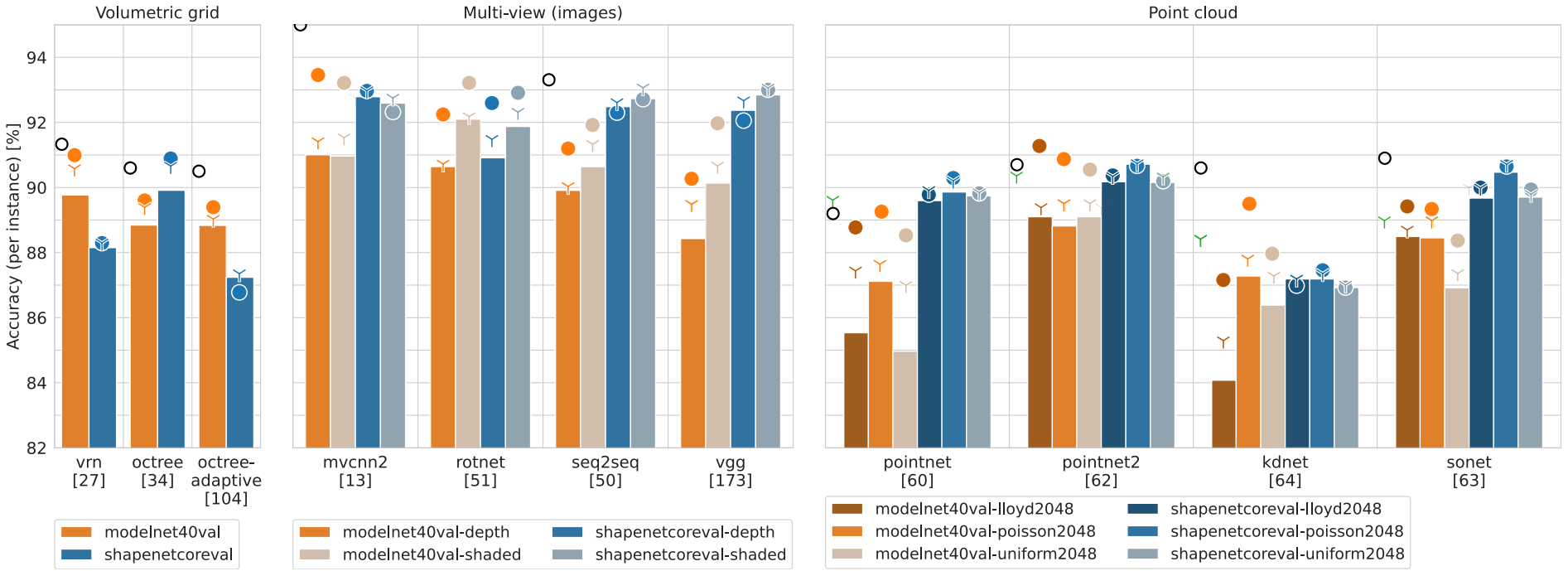

Fig. 6. The measured accuracies on different datasets. The bars show the test set accuracy at the epoch with the best validation accuracy and same-colored ϒ and ● mark the highest achieved accuracy on the test and validation subsets on the same dataset, respectively. ○ marks accuracies reported on ModelNet40 and ϒ marks the replicated test set accuracy on the point clouds provided by Qi et al. [61] using 1024 points (*qi1024*).

TABLE 3: Accuracies (%) measured at the epoch with the highest test (@best_test) or highest validation (@best_val) accuracy. In this table, *MN40val* is short for "modelnet40val", *SNCval* is short for "shapenetcoreval".

representation	net	dataset	test_acc@best_test	test_acc@best_val	val_acc@best_val
Multi-view (images)	mvcnn2	MN40val-depth	91.41	91.00	93.46
		MN40val-shaded	91.53	90.96	93.21
		SNCval-depth	92.98	92.79	92.97
		SNCval-shaded	92.75	92.59	92.32
	rotnet	MN40val-depth	90.72	90.64	92.25
		MN40val-shaded	92.18	92.10	93.21
		SNCval-depth	91.48	90.91	92.60
		SNCval-shaded	92.32	91.87	92.91
	seq2seq	MN40val-depth	90.03	89.91	91.20
		MN40val-shaded	91.33	90.64	91.92
		SNCval-depth	92.61	92.48	92.30
		SNCval-shaded	93.06	92.73	92.71
	vgg	MN40val-depth	89.48	88.43	90.27
		MN40val-shaded	90.66	90.13	91.97
		SNCval-depth	92.67	92.37	92.06
		SNCval-shaded	93.06	92.84	93.00
Point cloud	kdnet	MN40val-lloyd2048	85.29	84.08	87.16
		MN40val-poisson2048	87.80	87.28	89.50
		MN40val-uniform2048	87.28	86.39	87.96
		SNCval-lloyd2048	87.19	87.19	86.98
		SNCval-poisson2048	87.36	87.19	87.46
		SNCval-uniform2048	87.00	86.92	86.92
		modelnet40-qi1024	88.41	88.25	88.65
	pointnet	MN40val-lloyd2048	87.44	85.53	88.77
		MN40val-poisson2048	87.64	87.12	89.26
		MN40val-uniform2048	86.99	84.97	88.53
		SNCval-lloyd2048	89.91	89.60	89.78
		SNCval-poisson2048	90.20	89.86	90.29
		SNCval-uniform2048	89.84	89.74	89.82
	pointnet2	MN40val-lloyd2048	89.38	89.10	91.28
		MN40val-poisson2048	89.51	88.82	90.87
		MN40val-uniform2048	89.51	89.10	90.55
		SNCval-lloyd2048	90.34	90.17	90.37
		SNCval-poisson2048	90.73	90.72	90.67
		SNCval-uniform2048	90.32	90.16	90.19
	sonet	MN40val-lloyd2048	88.70	88.49	89.42
		MN40val-poisson2048	88.98	88.45	89.34
		MN40val-uniform2048	87.36	86.91	88.37
		SNCval-lloyd2048	89.99	89.67	90.00
		SNCval-poisson2048	90.72	90.47	90.65
		SNCval-uniform2048	89.79	89.70	89.94
Volumetric grid	octree	MN40val	89.39	88.84	89.60
		SNCval	90.66	89.91	90.89
	octree-adaptive	MN40val	89.03	88.83	89.40
		SNCval	87.35	87.24	86.78
	vrn	MN40val	90.58	89.77	90.99
		SNCval	88.30	88.15	88.29

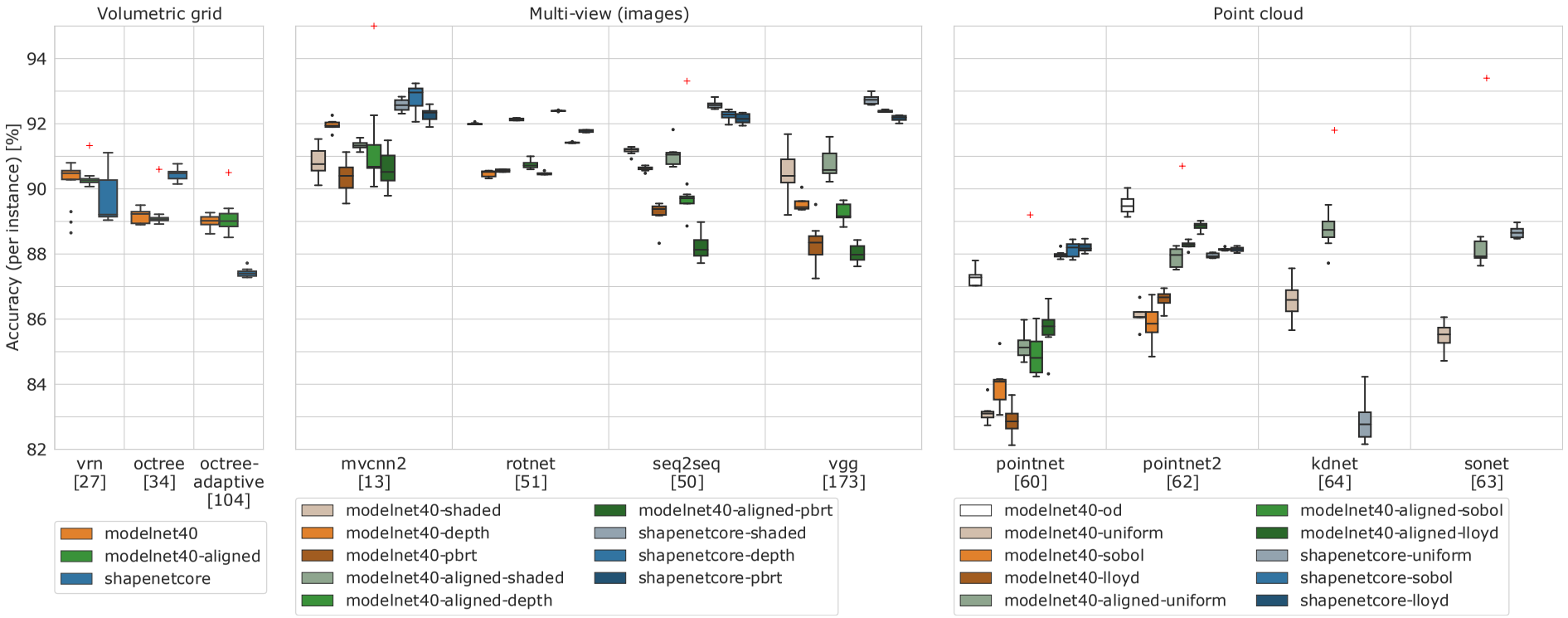

Fig. 7. The measured accuracies. For each network x dataset (x dataset variant), a distribution (boxplot) of accuracies on the test set in the best epoch and 3 before/after. • marks outliers and + marks the reported accuracy on ModelNet40 (official train/test split).

TABLE 4: Minimum, maximum and mean test set accuracies (%) and standard deviation (in percentage points) of the measured accuracies around the epoch with the highest test accuracy. The suffix *od* marks "original data" (provided by respective authors)

			accuracy			
			min	max	mean	std
representation	net	dataset				
Multi-view (images)	mvcnn2	modelnet40-aligned-depth	90.07	92.26	91.00	0.72
		modelnet40-aligned-pbrt	89.79	91.49	90.62	0.61
		modelnet40-aligned-shaded	91.13	91.57	91.34	0.14
		modelnet40-depth	91.65	92.26	91.96	0.20
		modelnet40-pbrt	89.55	91.13	90.35	0.53
		modelnet40-shaded	90.11	91.53	90.84	0.50
		shapenetcore-depth	92.06	93.24	92.79	0.44
		shapenetcore-pbrt	91.90	92.60	92.27	0.25
		shapenetcore-shaded	92.31	92.83	92.57	0.21
	rotnet	modelnet40-aligned-depth	90.60	91.00	90.75	0.14
		modelnet40-aligned-pbrt	90.44	90.56	90.47	0.04
		modelnet40-aligned-shaded	92.10	92.18	92.13	0.04
		modelnet40-depth	90.32	90.56	90.46	0.11
		modelnet40-pbrt	90.52	90.60	90.56	0.04
		modelnet40-shaded	91.98	92.06	92.00	0.03
		shapenetcore-depth	91.40	91.45	91.42	0.02
		shapenetcore-pbrt	91.73	91.81	91.78	0.04
		shapenetcore-shaded	92.37	92.42	92.39	0.02
	seq2seq	modelnet40-aligned-depth	88.86	90.15	89.62	0.39
		modelnet40-aligned-pbrt	87.72	88.98	88.23	0.44
		modelnet40-aligned-shaded	90.68	91.82	91.04	0.39
		modelnet40-depth	90.48	90.72	90.62	0.08
		modelnet40-pbrt	88.33	89.55	89.23	0.42
		modelnet40-shaded	90.92	91.29	91.17	0.13
		shapenetcore-depth	91.97	92.44	92.26	0.16
		shapenetcore-pbrt	91.94	92.34	92.16	0.16
		shapenetcore-shaded	92.45	92.82	92.58	0.13
	vgg	modelnet40-aligned-depth	88.83	89.65	89.28	0.30
		modelnet40-aligned-pbrt	87.62	88.43	88.02	0.31
		modelnet40-aligned-shaded	90.22	91.60	90.79	0.49
		modelnet40-depth	89.36	90.05	89.57	0.32
		modelnet40-pbrt	87.25	89.52	88.31	0.70
		modelnet40-shaded	89.20	91.68	90.39	0.70
		shapenetcore-depth	92.36	92.44	92.39	0.04
		shapenetcore-pbrt	92.01	92.26	92.17	0.10
		shapenetcore-shaded	92.58	93.00	92.74	0.16
Point cloud	kdnet	modelnet40-aligned-uniform	87.72	89.51	88.71	0.57
		modelnet40-uniform	85.66	87.56	86.58	0.61
		shapenetcore-uniform	82.16	84.23	82.89	0.72
	pointnet	modelnet40-aligned-lloyd	84.32	86.63	85.67	0.71
		modelnet40-aligned-sobol	84.24	86.02	84.92	0.68
		modelnet40-aligned-uniform	84.68	85.98	85.18	0.44
		modelnet40-lloyd	82.13	83.67	82.88	0.49
		modelnet40-od	87.03	87.80	87.27	0.28
		modelnet40-sobol	83.06	85.25	83.96	0.71
		modelnet40-uniform	82.74	83.83	83.14	0.34
		shapenetcore-lloyd	88.01	88.47	88.21	0.16
		shapenetcore-sobol	87.82	88.45	88.13	0.25
		shapenetcore-uniform	87.84	88.24	87.98	0.13
	pointnet2	modelnet40-aligned-lloyd	88.61	89.02	88.85	0.17
		modelnet40-aligned-sobol	88.05	88.45	88.27	0.16
		modelnet40-aligned-uniform	87.52	88.25	87.89	0.31

		modelnet40-lloyd	86.10	86.95	86.60	0.36
		modelnet40-od	89.14	90.03	89.52	0.33
		modelnet40-sobol	84.85	86.75	85.87	0.63
		modelnet40-uniform	85.53	86.67	86.11	0.41
		shapenetcore-lloyd	88.03	88.25	88.14	0.08
		shapenetcore-sobol	88.10	88.23	88.15	0.06
		shapenetcore-uniform	87.87	88.05	87.95	0.08
	sonet	modelnet40-aligned-uniform	87.64	88.53	88.09	0.34
		modelnet40-uniform	84.72	86.06	85.48	0.44
		shapenetcore-uniform	88.47	88.97	88.67	0.19
Volumetric grid	octree	modelnet40	88.90	89.50	89.16	0.24
		modelnet40-aligned	88.92	89.22	89.07	0.10
		shapenetcore	90.15	90.77	90.44	0.20
	octree-adaptive	modelnet40	88.62	89.27	89.00	0.22
		modelnet40-aligned	88.51	89.40	89.01	0.31
		shapenetcore	87.28	87.72	87.43	0.15
	vrn	modelnet40	88.65	90.80	90.19	0.68
		modelnet40-aligned	90.07	90.40	90.25	0.14
		shapenetcore	89.04	91.11	89.74	0.94