Revisiting Point Cloud Classification: A New Benchmark Dataset and Classification Model on Real-World Data

Mikaela Angelina Uy[1] Quang-Hieu Pham[2] Binh-Son Hua[3] Duc Thanh Nguyen[4] Sai-Kit Yeung[1]

[1]Hong Kong University of Science and Technology [2]Singapore University of Technology and Design
[3]The University of Tokyo [4]Deakin University

Abstract

Deep learning techniques for point cloud data have demonstrated great potentials in solving classical problems in 3D computer vision such as 3D object classification and segmentation. Several recent 3D object classification methods have reported state-of-the-art performance on CAD model datasets such as ModelNet40 with high accuracy (∼92%). Despite such impressive results, in this paper, we argue that object classification is still a challenging task when objects are framed with real-world settings. To prove this, we introduce ScanObjectNN, a new real-world point cloud object dataset based on scanned indoor scene data. From our comprehensive benchmark, we show that our dataset poses great challenges to existing point cloud classification techniques as objects from real-world scans are often cluttered with background and/or are partial due to occlusions. We identify three key open problems for point cloud object classification, and propose new point cloud classification neural networks that achieve state-of-the-art performance on classifying objects with cluttered background. Our dataset and code are publicly available in our project page [1].

1. Introduction

The task of understanding our real world has achieved a great leap in recent years. The rise of powerful computational resources such as GPUs and the availability of 3D data from depth sensors have accelerated the fast-growing field of 3D deep learning. Among various 3D data representations, point clouds are widely used in computer graphics and computer vision thanks to their simplicity. Recent works have shown great promises in solving classical scene understanding problems with point clouds such as 3D object classification and segmentation.

However, the current progress on classification with 3D point clouds has witnessed a trend of performance saturation. For example, many recent object classification methods have reported very high accuracies in 2018, and the trend of bringing the accuracy towards perfection is still ongoing. This phenomenon inspires us to raise a question on whether problems such as 3D object classification have been totally solved, and to think about how to move forward.

To answer this question, we perform a benchmark of existing point cloud object classification techniques with both synthetic and real-world data. For synthetic objects, we use ModelNet40 [43], the most popular dataset in point cloud object classification that contains about 10,000 CAD models. To support the investigation of object classification methods on real-world data, we introduce ScanObjectNN, a new point cloud object dataset from the state-of-the-art scene mesh datasets SceneNN [19] and ScanNet [9]. Based on the initial instance segmentation from the scene datasets, we manually filter and select objects for 15 common categories, and further enrich the dataset by considering additional object perturbations.

Our study shows that while the accuracy with CAD data is reaching perfection, learning to classify a real-world object dataset is still a very challenging task. By analyzing the benchmark results, we identify three open issues that are worth to further explore for future researches. First, classification models trained on synthetic data often do not generalize well to real-world data such as point clouds reconstructed from RGB-D scans [19, 9], and vice versa. Second, challenging in-context and partial observations of real-world objects are common due to occlusions and reconstruction errors; for example, they can be found in window-based object detectors [38] in many robotics or autonomous vehicle applications. Finally, how to handle background effectively when they appear together with objects due to clutter in the real-world scenes.

As our dataset opens up opportunities to tackle such open problems in real-world object classification, we also present a new method for point cloud object classification that can improve upon the state-of-the-art results on our dataset by jointly learning the classification and segmentation tasks in a single neural network.

In summary, we make the following contributions:

- A new object dataset from meshes of scanned real-world scene for training and testing point cloud classification,

[1] https://hkust-vgd.github.io/scanobjectnn/

- A comprehensive benchmark of existing object classification techniques on synthetic and real-world point cloud data,
- A new network architecture that is able to classify objects observed in a real-world setting by a joint learning of classification and segmentation.

2. Related Works

In this paper, we focus on object classification with point cloud data, which has advanced greatly in the past few years. We briefly discuss the related works and their datasets, below.

Object Classification on Point Clouds. Early attempts to classifying point clouds were developed by adapting ideas from deep learning on images, *e.g.*, using multiple view images [39, 48, 46, 22], or applying convolutions on 3D voxel grids [27, 43]. While it seems natural to extend the convolution operations from 2D to 3D, it is shown that performing convolutions on a point cloud is not a trivial task [30, 49]. The difficulty stems from the fact that a point cloud has no well-defined order of points on which convolutions can be performed. Qi et al. [30] addressed this problem by learning global features of point clouds using a symmetric function that is invariant to the order of points. Alternatively, some other methods proposed to learn local features from convolutions, *e.g.*, [32, 25, 20, 42, 44, 18, 2, 24, 33, 11] or from autoencoders [45]. There are also methods jointly learning features from point clouds and multi-view projections [47]. It is also possible to treat point clouds and views as sequences [26, 17, 15], or to use unsupervised learning [16].

Recent works demonstrate very competitive and compelling performances on standard datasets. For example, the gap between state-of-the-art methods such as SpecGCN [41], SpiderCNN [44], DGCNN [42], PointCNN [25] is less than 1% on ModelNet40 dataset [43]. In the online leaderboard maintained by the authors of ModelNet40, the accuracy of the object classification task is reaching perfection, with 92% for point cloud methods [25, 42, 44, 26].

Object Datasets. There are a limited number of datasets that can be used to train and test 3D object classification methods. ModelNet40 was originally developed by Wu et al. [43] for learning a convolutional deep-belief network to model 3D shapes represented in voxel grids. Objects in ModelNet40 are CAD models of 40 common categories such as airplane, motorbike, chair and table, to name a few. This dataset has been a common benchmark for point cloud object classification [30]. ShapeNet [7] is an alternative large-scale dataset of 3D CAD shapes with approximately $51,000$ objects in 55 categories. However, this set is usually used for benchmarking part segmentation.

So far, object classification on ModelNet40 is done with the assumption that objects are clean, complete, and free from any background noise. Unfortunately, this assumption is not often held in practice. It is common to see incomplete (partial) objects due to the imperfection of 3D reconstruction. In addition, objects in real-world settings are often scanned when being placed in a scene, which makes them appear in a clutter, and thus may be attached with background elements. A potential treatment is to remove such background using human annotators [28]. However, this solution is tedious, prone to errors, and subjective to the experience of annotators. Other works synthesize challenges on CAD data by introducing noise simulated by Gaussians [4, 12] or created with a parametic model [6] to mimic real world scenarios. Recently, the trend of sim2real [3] also aims to bridge the gap between synthetic and real data.

Prior to our work, there are also a few datasets of real-world object scans [10, 8, 5] but most are small in scale and are not suitable for training object classification networks, which often have thousands of parameters. For example, in robotics, Sydney urban objects dataset [10] contains only 631 objects of 26 categories captured by a LiDAR camera, which is mainly used for evaluation [27, 2] but not for training. Some datasets [36, 5] are captured in controlled environment which might greatly differ from real-world scenes. Choi et al. [8] proposed a dataset of more than 10,000 object scans in the real world. However, not all of their scans can be successfully reconstructed; the online repository by the authors also provided only about 400 reconstructed objects. RGB-D and 3D scene meshes datasets [19, 9, 1, 37, 34] have more objects that are reconstructed along with the scenes, but such objects are often considered in a scene segmentation or object detection task, and not under an object classification setup. RGBD-to-CAD object classification challenge [21, 29] provides an object dataset that mixes CAD models and real-world scans. Its goal is to classify RGB-D objects such that a retrieval can be done to find similar CAD models. However, several categories are ambiguous, and objects are supposed to be well segmented before classification. ScanNet [9] has a benchmark on 3D object classification with partially scanned objects. However, this dataset is designed for volume-based object classification [31], and there are quite few techniques that report their results with this data.

3. Benchmark Data

Our goal is to quantitatively analyze the performances of existing object classification methods on point clouds. We split our task into two parts: benchmarking with synthetic data and with real-world data.

3.1. Synthetic Data - ModelNet40

For synthetic data, we experiment with the well-known ModelNet40 dataset [43]. This set is a collection of CAD models with 40 object categories. The dataset includes 9,840 objects for training and 2,468 objects for testing. The objects in ModelNet40 are synthetic, and thus are complete,

Figure 1. Sample objects from our dataset.

well-segmented, and noise-free. In this experiment, we use the uniformly dense point cloud variant as preprocessed by Qi et al. [30]. Each point cloud is randomly sampled to 1024 points as input to the networks unless otherwise stated. The point clouds are centered at zero, and we use local coordinates (x, y, z) normalized to $[-1, 1]$ as point attributes. We follow the default train/test split, and use the default parameters as in the original implementations of the methods. Our benchmark is performed with a NVIDIA Tesla P100 GPU. We re-trained PointNet [30], PointNet++ [32], PointCNN [25], Dynamic Graph CNN (DGCNN) [42], 3D modified Fisher Vector (3DmFV) [2], and SpiderCNN [44]. For remaining methods, we provided the results reported in the original papers. We additionally report each method's best performance when provided with additional information such as point normals. The results are shown in Table 1. It can be observed that the performance of recent methods is becoming incremental, and fluctuates around 92%. This saturating score inspires us to revisit the object classification problem: Can classification methods trained on ModelNet40 perform well on real-world data? Or is there still room for more research problems to be explored?

Method	Avg. Class Accuracy	Overall Accuracy
ECC [35]	83.2	87.4
PointNet [30]	86.2	89.2
DeepSets [49]	-	90.0
Flex-Convolution [14]	-	90.2
Kd-Net [23]	88.5	90.6 (91.8 *)
PointNet++ [32]	87.8	90.7 (91.9 w/ normal)
SO-Net [24]	87.3	90.9 (93.4 w/ normal)
KCNet [33]	-	91
3DmFV [2]	86.3	91.4
SpecGCN [41]	-	91.5 (92.1 w/ normal)
SpiderCNN [44]	86.8	90.0 (92.4 w/ normal)
DGCNN [42]	90.2	92.2
PointCNN [25]	88.8	92.5

Table 1. Baseline results on ModelNet40 dataset for point cloud classification. Inputs are point coordinates, unless otherwise stated; * denotes the use of more input points (32K).

3.2. Real-World Data - ScanObjectNN

Objects obtained from real-world 3D scans are significantly different from CAD models due to the presence of background noise and the non-uniform density due to holes from incomplete scans/reconstructions and occlusions. This situation is often seen in sliding window-based object detection [38] in which a window may enclose an object of interest partially and also include background elements within the window. Due to these properties, applying existing point cloud classification methods to real-world data may not produce the same good results as CAD models.

3.2.1 Data Collection

To study this potential issue, we build a real-world object dataset based on two popular scene meshes datasets: SceneNN [19] and ScanNet [9]. SceneNN has 100 annotated

Class	Bag	Bed	Bin	Box	Cabinet	Chair	Desk	Display	Door	Pillow	Shelf	Sink	Sofa	Table	Toilet
#Objects	78	135	201	127	347	395	149	181	221	105	267	118	254	242	82

Table 2. Classes and objects in our dataset.

scenes with highly cluttered objects while ScanNet has a larger collection of 1513 indoor scenes. From a total of more than 1600 scenes from SceneNN and ScanNet, we selected 700 unique scenes. We then manually examined each object, fixed inconsistent labels, and discard objects that are ambiguous, have low reconstruction quality, have unknown labels, are too sparse, and have too few instances to form a category for training. During categorization, we also took into account inter-class balancing to avoid any bias potentially coming from classes with more samples.

The results are 2902 objects that are categorized into 15 categories. The raw objects are represented by a list of points with global and local coordinates, normals, colors attributes and semantic labelsOther works synthesize challenges on CAD data by introducing noise simulated by Gaussians [4, 12] or created with a parametic model [6]. Recently, the trend of sim2real [3] also aims to bridge the gap between synthetic and real data. As in the experiment with synthetic data, we sample all raw objects to 1024 points as input to the networks and all methods were trained using only the local (x, y, z) coordinates. We will make our dataset publicly available for future research. Table 2 summarizes classes and objects in our dataset.

3.2.2 Data Enrichment

Based on the selected objects, we construct several variants that represent different levels of difficulty of our dataset. This allows us to explore the robustness of existing classification methods in more extreme real-world scenarios.

Vanilla. The first variant is referred to as **OBJ_ONLY** which includes only ground truth segmented objects extracted from the scene meshes datasets. This variant has the closest form analogous to its CAD counterpart, and is used to investigate the robustness of classification methods to noisy objects with deformed geometric shape and non-uniform surface density. Sample objects of this variant are shown in Figure 2(a).

Background. The previous variant assumes that an object can be accurately segmented before being classified. However, in real-world scans, objects are often presented in under-segmentation situations, i.e., background elements or parts of nearby objects are included, and accurate annotations for such under-segmentations are also not always available. Those background elements may provide the context where objects belong to, and thus would become a good hint for object classification, *e.g.*, laptops often sit on desks. However, they may also introduce distractions which corrupt the classification, *e.g.*, a pen may be under-segmented with a table where it sits on and thus could be considered as a part of the table rather than a separate object. To study these factors, we introduce a variant of our dataset where objects are attached with background data (**OBJ_BG**). We determine such background by using the ground truth axis-aligned object bounding boxes. Specifically, given a bounding box, all points in the box are extracted to form an object. Sample objects with background are shown in Figure 2(b).

(a) Objects only. (b) Objects with background.

Figure 2. Example objects from our dataset.

Perturbed. The given bounding boxes from the ground-truth tightly enclose the objects. However, in real-world scenarios bounding boxes may over- or under-cover, or even split objects. For example, in object detection techniques such as R-CNN [13], object category has to be predicted from a rough bounding box that localizes a candidate object. To simulate this challenge, we extend our dataset by translating, rotating (about the gravity axis), and scaling the ground truth bounding boxes before extracting the geometry in the box. We name the variants of these perturbations with a common prefix **PB**.

The perturbations introduce various degrees of background and partiality to objects. In this work, we use four perturbation variants in the increasing order of difficulty: **PB_T25**, **PB_T25_R**, **PB_T50_R**, and **PB_T50_RS**. Suffix **_T25** and **_T50** denote translation that randomly shifts the bounding box up to 25% and 50% of its size from the box centroid along each world axis. Suffix **_R** and **_S** denotes rotation and scaling. Each perturbation variant contains five random samples for each original object, resulting in up to $14,510$ perturbed objects in total. Since perturbation might introduce invalid objects, e.g., objects that are almost completely out of the bounding box of interest, we perform an additional check after perturbation by ensuring that at least 50% of the original object points remain in the bounding box. Objects that do not satisfy this condition are discarded. Sample point clouds of these variants are shown in Figure 3. More details about perturbing objects can be found in our supplementary material.

(a) OBJ_BG (b) PB_T25 (c) PB_T25_R (d) PB_T50_R (e) PB_T50_RS

Figure 3. An object in different perturbation variants.

4. Benchmark on ScanObjectNN

For a clearer picture of the maturity of point cloud-based object classification, we benchmark several representative methods on our dataset. We aim to identify the limitations of current works on real-world data. We choose 3DmFV [2], PointNet [30], SpiderCNN [44], PointNet++ [32], DGCNN [42] and PointCNN[25] as our representative works.

4.1. Training on ModelNet40

We first study the case when training is done on ModelNet40 and testing is done on ScanObjectNN. Since objects in ModelNet40 are standalone with no background objects, we also removed background in all our variants for fair evaluations. Furthermore, we only evaluated the current methods on 11 (out of 15) common classes between ModelNet40 and our dataset. Please refer to the supplementary material for the details on these common classes.

Evaluation results are reported in Table 3. These results show that the current techniques trained on CAD models are not able to generalize to real-world data; all techniques achieved less than 50% of accuracy. This is expected and is because of the fact that real-world objects and CAD objects are significantly different in their geometry. Real-world objects are often incomplete and partial due to construction errors and occlusions; their surfaces have low-frequency noise; object boundaries are inaccurate. These are in contrast to CAD objects, which are often clean and noise-free. We also found that the harder the data is (*i.e.* more noise and partiality), the lower the performance is, and this is consistent for all techniques. In other words, knowledge learned from synthetic objects in ModelNet40 is not well transferable and/or applicable to real-world data.

4.2. Training on ScanObjectNN

In this experiment, we train and test the techniques on ScanObjectNN to demonstrate training on datasets with real-world properties should improve the performance in classifying real-world objects. We also analyze how different perturbations can affect the classification performance. We randomly split our dataset into two subsets: training (80%) and test (20%) set. We ensure that the training and test sets contain objects from different scenes so that similar objects do not occur in the same set, *e.g.* same types of chairs can be found in the same room. We report the performance of all the techniques on the hardest split in Table 4. Full performances on all splits are provided in our supplementary material.

	OBJ_ONLY	PB_T25	PB_T25_R	PB_T50_R	PB_T50_RS
3DmFV [2]	30.9	28.4	27.2	24.5	24.9
PointNet [30]	42.3	37.6	35.3	32.1	31.1
SpiderCNN [44]	44.2	37.7	34.5	31.7	30.9
PointNet++ [32]	43.6	37.8	37.2	33.3	32.0
DGCNN [42]	49.3	42.4	40.3	36.6	36.8
PointCNN [25]	32.2	28.7	28.1	26.4	24.6

Table 3. Overall accuracy in % on our dataset when training was done on ModelNet40. Note that for a fair comparison, background has been removed in all variants. The results show that training on CAD models and testing on real-world data is challenging. Most methods do not generalize well in this test.

	OBJ_ONLY	OBJ_BG	PB_T25	PB_T25_R	PB_T50_R	PB_T50_RS
3DmFV [2]	73.8	68.2	67.1	67.4	63.5	63.0
PointNet [30]	79.2	73.3	73.5	72.7	68.2	68.2
SpiderCNN [44]	79.5	77.1	78.1	77.7	73.8	73.7
PointNet++ [32]	84.3	82.3	82.7	81.4	79.1	77.9
DGCNN [42]	86.2	82.8	83.3	81.5	80.0	78.1
PointCNN [25]	85.5	86.1	83.6	82.5	78.5	78.5

Table 4. Overall accuracy in % when training and testing were done on ScanObjectNN. The training and testing are done on the same variant. With real-world data, the more background and partiality are introduced, the more challenging the classification task is.

For fair comparisons, we kept the same data augmentation process in all the methods (*e.g.*, random rotation and per-point jitter). We trained the methods to convergence rather than selecting the best performance on the test set.

Vanilla. The 2nd column in Table 4 shows the overall performance of existing methods when trained on the simplest variant of our dataset (OBJ_ONLY). This clearly shows that the classification accuracy increased significantly when training and testing are both done using ScanObjectNN versus when training is done using ModelNet40 (Table 3 Column 2). However, we also notice an observable performance drop comparing to the pure synthetic setting in Table 1. This gives an important message: point cloud classification on real-world data is still open, a dataset with real-world properties can help, but further research is necessary to regain the high performance as in synthetic setting. In the following, we investigate the performance change in different types of perturbations in our dataset.

Background. As shown in Table 4 Columns 3-7, background makes strong impact to the classification performance of all methods. Specifically, except PointCNN [25], all methods performed worse on OBJ_BG compared with OBJ_ONLY. It can be explained by the fact that

	Ours		ModelNet40	
	w/o BG	w/ BG	w/o BG	w/ BG
3DmFV [2]	69.8	63.0	54.1	51.5
PointNet [30]	74.4	68.2	60.4	50.9
SpiderCNN [44]	76.9	73.7	52.7	46.6
PointNet++ [32]	80.2	77.9	55.0	47.4
DGCNN [42]	81.5	78.1	58.7	54.7
PointCNN [25]	80.8	78.5	38.1	49.2

Table 5. Overall accuracy in % when training on our hardest variant PB_T50_RS, with and without background (BG) points. Testing is done on the same variant of our dataset, and on ModelNet40. The second header indicates the results corresponding to the training set. The results show that (1) background impacts negatively to the classification performance, and (2) training on our real-world objects generalizes to CAD evaluation better than the opposite case.

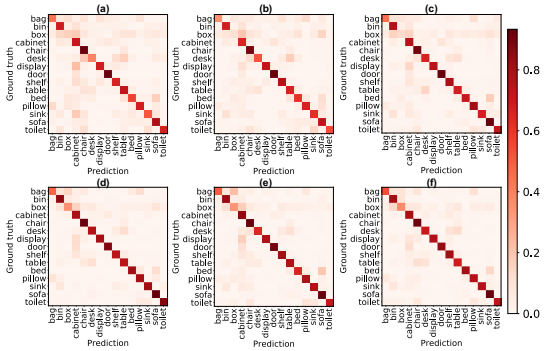

Figure 4. Confusion matrices of (a) 3DmFV [2], (b) PointNet [30], (c) SpiderCNN [44], (d) PointNet++ [32], (e) DGCNN [42] and (f) PointCNN [25] on our hardest PB_T50_RS. This shows that there are no major ambiguity issues among object classes in our dataset.

background elements could distract the learning in existing methods by confusing between foreground and background points. To further confirm the negative effect of having background objects, we conduct a control experiment using the hardest perturbation variant, *i.e.*, PB_T50_RS. Table 5 shows the overall accuracy of all existing models decrease when trained and tested *with* the presence of background.

Perturbation. Table 4 also shows the impact of perturbations to the classification performance (compared with Column 2). In this result, we observe that translation and rotation both make the classification performance decrease significantly, especially with larger perturbations that introduce more background and partiality. Scale further degrades the performance by a small gap. Figure 4 illustrates the confusion matrices of all methods on our hardest variant PB_T50_RS. It can be seen that there are no major ambiguity issues in our categories, and our dataset is challenging due to the high variations in real-world data.

Generalization to CAD Data. While it is shown that networks trained on synthetic data generalizes poorly to our dataset (Table 3), the reverse is not true. Here we tested the generalization capability of existing methods when trained

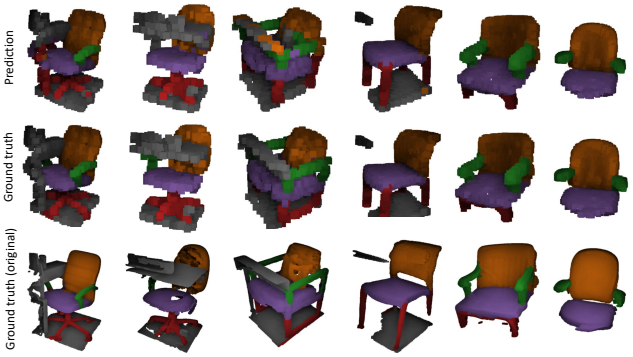

Figure 5. Part segmentation on the chair category. From top to bottom: part prediction, ground truth in 2048 points, and high-resolution ground truth from original point clouds.

on ScanObjectNN. In this experiment, all methods were trained on our PB_T50_RS (with and without background) and tested on ModelNet40. The results in the last two columns in Table 5 clearly show that existing methods could generalize better when they were trained on real-world data (compared with the results in Table 3). Performance on individual classes are presented in Table 6. As shown in Table 6, lower accuracies are achieved on classes such as bed, cabinet, and desk, where complete structures are never observed in real scans because these objects are often situated adjacent to walls or near corners of rooms. Therefore, we advocate using real-world data in training object classification because the generalization is shown to be much better.

4.3. Part Annotation on Real-World Data

We further support part-based annotation in our dataset. So far, point cloud classification methods only evaluate part segmentation task on ShapeNet [40]. However, there has been no publicly available dataset for part segmentation on real-world data despite the availability of scene meshes datasets [19, 9]. We close this gap with our dataset, which will be released for future research. Figure 5 shows a visualization of part segmentation on our data. Table 7 and Table 8 provide a baseline part segmentation evaluation on our data. Using these part annotations may also improve partial object classification in the future.

4.4. Discussion

Our quantitative evaluations show that performing object classification on real-world data is challenging. The state-of-the-art methods in our benchmark have up to 78.5% accuracy on our hardest variant (PB_T50_RS). The benchmark also helps us recognize the following open problems:

Background is expected to provide context information but also introduce noise. It is desirable to have an approach that can distinguish foreground from background to effectively exploit context information in the classification task.

Object partiality, caused by low reconstruction quality or

	cabinet	chair	desk	display	door	shelf	table	bed	sink	sofa	toilet
3DmFV [2]	20.8	67.1	**8.1**	75.0	75.0	86.0	97.0	10.0	50.0	21.0	64.0
PointNet [30]	**2.8**	72.1	43.0	83.0	100.0	98.0	93.0	**4.0**	35.0	23.0	26.0
SpiderCNN [44]	17.9	54.3	17.4	86.0	90.0	90.0	88.0	**7.0**	40.0	32.0	14.0
PointNet++ [32]	18.9	71.4	12.8	94.0	45.0	79.0	88.0	**2.0**	45.0	14.0	35.0
DGCNN [42]	47.2	75.7	11.6	94.0	85.0	83.0	100.0	**9.0**	45.0	42.0	12.0
PointCNN [25]	42.5	77.9	24.4	76.0	20.0	92.0	76.0	**4.0**	35.0	24.0	19.0

Table 6. Per class average accuracy in % on ModelNet40 when training was done on our PB_T50_RS. Low accuracies are highlighted.

	OBJ_BG	PB_T25	PB_T25_R	PB_T50_R	PB_T50_RS
PointNet [30]	81.3	83.1	82.2	79.9	78.8
PointNet++ [32]	80.3	85.4	84.1	81.3	82.8

Table 7. Overall accuracy in % of part segmentation of chairs in the different variants of ScanObjectNN.

	background	seat	back	base	arm
PointNet [30]	81.4	81.8	86.7	52.5	40.5
PointNet++ [32]	81.9	87.7	89.2	62.3	64.6

Table 8. Per part average accuracy in % of chairs in our hardest variant PB_T50_RS.

inaccurate object proposals, also needs to be addressed. Part segmentation techniques [30, 25] could help to describe partial objects.

Generalization between CAD models and real-world scans needs more investigations. In general, we found that training on real-world data and testing on CADs can generalize better than the opposite case. It could be explained that real-world data have more variations including background and partiality as discussed above. However, CAD models are still important because real-world scans are seldom complete and noise free. Bridging this domain gap could be an important research direction.

To facilitate future work, in the next sections, we propose ideas and baseline solutions.

5. Background-aware Classification Network

We propose here a simple deep network to handle the occurrence of background in point clouds obtained from real scans; this is one of the open problems we raised in the previous section. An issue with existing point cloud classification networks is the lack of capability to distinguish between foreground and background points. In other words, existing methods take point clouds as a whole and directly calculate features for classification. This issue stems from the design of these networks and also from the simplicity of available training datasets, *e.g.*, ModelNet40.

To tackle this issue, our idea is to make the network aware of the presence of background by adding a segmentation-guided branch to the classification network. The segmentation branch predicts an object mask that separates the foreground from the background. Note that the mask can be easily obtained from our training data since our objects are originally from scene instance segmentation datasets [19, 9].

5.1. Network Architecture

Our background-aware (BGA) model is built on top of PointNet++ [32] (**BGA-PN++**). Our network is depicted in Figure 6. In particular, we use three levels of set abstractions from the PointNet++ to extract point cloud global features. Global features are then passed through three fully connected layers to produce object classification score. Dropout is also used in a similar manner with the original PointNet++ architecture. Three PointNet feature propagation modules are then employed to compute object masks in segmentation. The feature vector just before the last fully connected layer for the classification score is used as the input to the first PointNet feature propagation modules, making the predicted object mask driven by the classification output. We trained both branches jointly. The loss function is the sum of the classification and segmentation loss, which can be written as $\mathcal{L}_{\text{total}} = \mathcal{L}_{\text{class}} + \lambda\mathcal{L}_{\text{seg}}$ where $\mathcal{L}_{\text{class}}$ and $\mathcal{L}_{\text{seg}}$ are both cross entropy losses between the predicted and ground-truth class labels and object masks, respectively. We set $\lambda = 0.5$ in our experiments.

Joint learning for both classification and segmentation with the use of object masks allows the network to be aware of relevant points (*i.e.*, acknowledge the presence of background points). In addition, using classification prediction as a prior to segmentation guides the network to learn object masks that are consistent with the true shape of desired object classes. As to be detailed in our experiments, jointly learning classification and mask prediction results in better classification accuracy in noisy scenarios.

Furthermore, we also introduce **BGA-DGCNN**, which is a background-aware network based on DGCNN [42]. We apply the same concept as BGA-PN++ that jointly predicts both classification and segmentation, where the last fully connected layer of the classification branch is used as input to the segmentation branch. Our experimental results show that our bga model is adaptive to different network architectures.

5.2. Evaluation

We evaluate our network on both our dataset and ModelNet40. Table 9 shows a comparison between our network

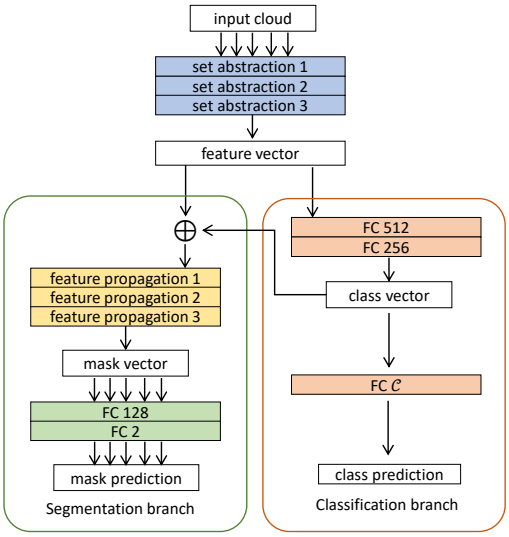

Figure 6. Our proposed network.

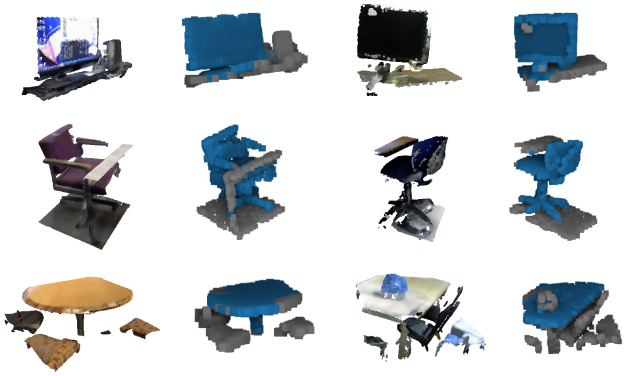

Figure 7. Sample objects and their corresponding predicted masks from the test set of PB_T50_RS by our BGA-PN++. Note that color on point clouds is for visualization purposes, but the input to the networks are (x, y, z) coordinates only.

and existing ones on our hardest variant PB_T50_RS and ModelNet40 respectively. Our BGA models, BGA-PN++ and BGA-DGCNN, both outperform their vanilla counterparts with BGA-PN++ achieving the best performance on our PB_T50_RS. On ModelNet40, our BGA-PN++ improves upon PointNet++ by almost 5% (with 52.6% of accuracy), while our BGA-DGCNN achieves the top performance of 56.5%. Note that, in this evaluation all methods were trained on our *i.e.* PB_T50_RS. As shown, our BGA models gains improvements in both ModelNet40 and our dataset.

In addition, we also evaluated the segmentation performance of our network. Experimental results showed that our BGA-PN++ performed at 77.6% and 71.0%, while our BGA-DGCNN achieved 78.5% and 74.3% of segmentation accuracy on our PB_T50_RS and ModelNet40, respectively. We visualize some of the object masks predicted by our BGA-PN++ in Figure 7. It can be seen that our proposed network is able to mask out the background fairly accurately.

	Ours		ModelNet40	
	OA	mAcc	OA	mAcc
3DmFV [2]	63.0	58.1	51.5	52.2
PointNet [30]	68.2	63.4	50.9	52.7
SpiderCNN [44]	73.7	69.8	46.6	48.8
PointNet++ [32]	77.9	75.4	47.4	45.9
DGCNN [42]	78.1	73.6	54.7	54.9
PointCNN [25]	78.5	75.1	49.2	44.6
BGA-PN++ (ours)	**80.2**	**77.5**	52.6	50.6
BGA-DGCNN (ours)	79.9	75.7	**56.5**	**57.6**

Table 9. Overall and average class accuracy in % on our PB_T50_RS and on ModelNet40. Training is done on our PB_T50_RS.

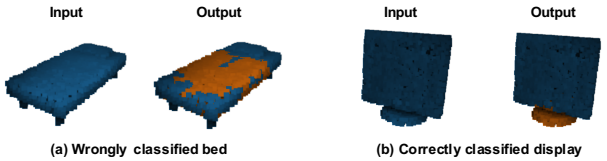

Figure 8. Sample segmentation results of our BGA-PN++ on ModelNet40. Background and foreground are marked in orange and blue, respectively.

5.3. Discussion and Limitation

While both BGA models demonstrate good performance, we found that DGCNN-based networks generalizes well between real and CAD data, e.g., when being trained on real and tested on CAD data (Table 9) and vice versa (Table 3). Moreover, Table 3 also show that the same is true for DGCNN-based models on the synthetic to real case. More investigations on the DGCNN architecture could lead to models that generalize better and bridge the gap between synthetic and real data.

Our proposed BGA is not without limitation. In general, it requires object masks and background to be included in the data. Fig. 8-(a) shows a fail case of our method when evaluating on a background-free ModelNet40 object.

6. Conclusion

This paper revisits state-of-the-art object classification methods on point cloud data. We found that existing methods were successful with synthetic data but failed on realistic data. To prove this, we built a new real-world object dataset containing $\sim 15,000$ objects in 15 categories. Compared with current datasets, our dataset offers more practical challenges including background occurrence, object partiality, and different deformation variants. We benchmarked existing methods on our new dataset, discussed issues, identified open problems, and suggested possible solutions. We also proposed a new point cloud network to classify objects with background. Experimental results showed the advance of our method on both synthetic and real-world object datasets.

Acknowledgment This research project is partially supported by an internal grant from HKUST (R9429).

References

[1] Iro Armeni, Ozan Sener, Amir R. Zamir, Helen Jiang, Ioannis Brilakis, Martin Fischer, and Silvio Savarese. 3d semantic parsing of large-scale indoor spaces. In *CVPR*, 2016. 2

[2] Yizhak Ben-Shabat, Michael Lindenbaum, and Anath Fischer. 3dmfv: Three-dimensional point cloud classification in real-time using convolutional neural networks. *IEEE Robotics and Automation Letters*, 2018. 2, 3, 5, 6, 7, 8

[3] Alex Bewley, Jessica Rigley, Yuxuan Liu, Jeffrey Hawke, Richard Shen, Vinh-Dieu Lam, and Alex Kendall. Learning to drive from simulation without real world labels. In *International Conference on Robotics and Automation (ICRA)*, 2019. 2, 4

[4] Dmytro Bobkov, Sili Chen, Ruiqing Jian, Muhammad Z. Iqbal, and Eckehard Steinbach. Noise-resistant deep learning for object classification in three-dimensional point clouds using a point pair descriptor. *IEEE Robotics and Automation Letters*, 2018. 2, 4

[5] Berk Calli, Arjun Singh, James Bruce, Aaron Walsman, Kurt Konolige, Siddhartha Srinivasa, Pieter Abbeel, and Aaron M Dollar. Yale-cmu-berkeley dataset for robotic manipulation research. *International Journal of Robotics Research*, 2017. 2

[6] Ben Chandler and Ennio Mingolla. Mitigation of effects of occlusion on object recognition with deep neural networks through low-level image completion. In *Comp. Int. and Neurosc.*, 2016. 2, 4

[7] Angel X. Chang, Thomas Funkhouser, Leonidas Guibas, Pat Hanrahan, Qixing Huang, Zimo Li, Silvio Savarese, Manolis Savva, Shuran Song, Hao Su, Jianxiong Xiao, Li Yi, and Fisher Yu. ShapeNet: An Information-Rich 3D Model Repository. Technical Report arXiv:1512.03012, Stanford University — Princeton University — Toyota Technological Institute at Chicago, 2015. 2

[8] Sungjoon Choi, Qian-Yi Zhou, Stephen Miller, and Vladlen Koltun. A large dataset of object scans. *arXiv:1602.02481*, 2016. 2

[9] Angela Dai, Angel X Chang, Manolis Savva, Maciej Halber, Thomas Funkhouser, and Matthias Niessner. Scannet: Richly-annotated 3d reconstructions of indoor scenes. In *CVPR*, 2017. 1, 2, 3, 6, 7

[10] Mark De Deuge, Alastair Quadros, Calvin Hung, and Bertrand Douillard. Unsupervised feature learning for classification of outdoor 3d scans. In *Australasian Conference on Robotics and Automation*, 2013. 2

[11] M. Dominguez, R. Dhamdhere, A. Petkar, S. Jain, S. Sah, and R. Ptucha. General-purpose deep point cloud feature extractor. In *WACV*, 2018. 2

[12] Alberto Garcia-Garcia, Jose Rodriguez, Sergio Orts, Sergiu Oprea, Francisco Gomez-Donoso, and Miguel Cazorla. A study of the effect of noise and occlusion on the accuracy of convolutional neural networks applied to 3d object recognition. *Computer Vision and Image Understanding*, 2017. 2, 4

[13] Ross Girshick, Jeff Donahue, Trevor Darrell, and Jitendra Malik. Rich feature hierarchies for accurate object detection and semantic segmentation. In *CVPR*, 2014. 4

[14] Fabian Groh, Patrick Wieschollek, and Hendrik P. A. Lensch. Flex-convolution (million-scale point-cloud learning beyond grid-worlds). In *ACCV*, 2018. 3

[15] Z. Han, H. Lu, Z. Liu, C. Vong, Y. Liua, M. Zwicker, J. Han, and C. L. P. Chen. 3d2seqviews: Aggregating sequential views for 3d global feature learning by cnn with hierarchical attention aggregation. *IEEE Transactions on Image Processing*, 2019. 2

[16] Zhizhong Han, Mingyang Shang, Yu-Shen Liu, and Matthias Zwicker. View inter-prediction GAN: unsupervised representation learning for 3d shapes by learning global shape memories to support local view predictions. In *AAAI*, 2018. 2

[17] Zhizhong Han, Mingyang Shang, Xiyang Wang, Yu-Shen Liu, and Matthias Zwicker. Y^2seq2seq: Cross-modal representation learning for 3d shape and text by joint reconstruction and prediction of view and word sequences. In *AAAI*, 2019. 2

[18] P. Hermosilla, T. Ritschel, P-P Vazquez, A. Vinacua, and T. Ropinski. Monte carlo convolution for learning on non-uniformly sampled point clouds. *ACM Transactions on Graphics (Proceedings of SIGGRAPH Asia)*, 2018. 2

[19] Binh-Son Hua, Quang-Hieu Pham, Duc Thanh Nguyen, Minh-Khoi Tran, Lap-Fai Yu, and Sai-Kit Yeung. Scenenn: A scene meshes dataset with annotations. In *International Conference on 3D Vision (3DV)*, 2016. http://www.scenenn.net. 1, 2, 3, 6, 7

[20] Binh-Son Hua, Minh-Khoi Tran, and Sai-Kit Yeung. Pointwise convolutional neural networks. In *CVPR*, 2018. 2

[21] Binh-Son Hua, Quang-Trung Truong, Minh-Khoi Tran, Quang-Hieu Pham, Asako Kanezaki, Tang Lee, HungYueh Chiang, Winston Hsu, Bo Li, Yijuan Lu, Henry Johan, Shoki Tashiro, Masaki Aono, Minh-Triet Tran, Viet-Khoi Pham, Hai-Dang Nguyen, Vinh-Tiep Nguyen, Quang-Thang Tran, Thuyen V. Phan, Bao Truong, Minh N. Do, Anh-Duc Duong, Lap-Fai Yu, Duc Thanh Nguyen, and Sai-Kit Yeung. RGB-D to CAD Retrieval with ObjectNN Dataset. In *Eurographics Workshop on 3D Object Retrieval*, 2017. 2

[22] Asako Kanezaki, Yasuyuki Matsushita, and Yoshifumi Nishida. Rotationnet: Joint object categorization and pose estimation using multiviews from unsupervised viewpoints. In *CVPR*, 2018. 2

[23] Roman Klokov and Victor S. Lempitsky. Escape from cells: Deep kd-networks for the recognition of 3d point cloud models. 2017. 3

[24] Jiaxin Li, Ben M Chen, and Gim Hee Lee. So-net: Self-organizing network for point cloud analysis. In *CVPR*, 2018. 2, 3

[25] Yangyan Li, Rui Bu, Mingchao Sun, and Baoquan Chen. Pointcnn: Convolution on x-transformed points. *Advances in Neural Information Processing Systems*, 2018. 2, 3, 5, 6, 7, 8

[26] Xinhai Liu, Zhizhong Han, Yu-Shen Liu, and Matthias Zwicker. Point2sequence: Learning the shape representation of 3d point clouds with an attention-based sequence to sequence network. *arXiv:1811.02565*, 2018. 2

[27] D. Maturana and S. Scherer. VoxNet: A 3D Convolutional Neural Network for Real-Time Object Recognition. In *IROS*, 2015. 2

[28] Duc Thanh Nguyen, Binh-Son Hua, Lap-Fai Yu, and Sai-Kit Yeung. A robust 3d-2d interactive tool for scene segmentation

and annotation. *IEEE Transactions on Visualization and Computer Graphics (TVCG)*, 2017. 2

[29] Quang-Hieu Pham, Minh-Khoi Tran, Wenhui Li, Shu Xiang, Heyu Zhou, Weizhi Nie, Anan Liu, Yuting Su, Minh-Triet Tran, Ngoc-Minh Bui, Trong-Le Do, Tu V. Ninh, Tu-Khiem Le, Anh-Vu Dao, Vinh-Tiep Nguyen, Minh N. Do, Anh-Duc Duong, Binh-Son Hua, Lap-Fai Yu, Duc Thanh Nguyen, and Sai-Kit Yeung. RGB-D Object-to-CAD Retrieval. In *Eurographics Workshop on 3D Object Retrieval*, 2018. 2

[30] Charles R Qi, Hao Su, Kaichun Mo, and Leonidas J Guibas. Pointnet: Deep learning on point sets for 3d classification and segmentation. *CVPR*, 2017. 2, 3, 5, 6, 7, 8

[31] Charles R. Qi, Hao Su, Matthias Niessner, Angela Dai, Mengyuan Yan, and Leonidas J. Guibas. Volumetric and multi-view cnns for object classification on 3d data. In *CVPR*, 2016. 2

[32] Charles R Qi, Li Yi, Hao Su, and Leonidas J Guibas. Pointnet++: Deep hierarchical feature learning on point sets in a metric space. *Advances in Neural Information Processing Systems*, 2017. 2, 3, 5, 6, 7, 8

[33] Yiru Shen, Chen Feng, Yaoqing Yang, and Dong Tian. Mining point cloud local structures by kernel correlation and graph pooling. In *CVPR*, 2018. 2, 3

[34] Nathan Silberman, Derek Hoiem, Pushmeet Kohli, and Rob Fergus. Indoor segmentation and support inference from rgbd images. In *ECCV*, 2012. 2

[35] Martin Simonovsky and Nikos Komodakis. Dynamic edge-conditioned filters in convolutional neural networks on graphs. In *CVPR*, 2017. 3

[36] A. Singh, J. Sha, K. S. Narayan, T. Achim, and P. Abbeel. Bigbird: A large-scale 3d database of object instances. In *International Conference on Robotics and Automation (ICRA)*, 2014. 2

[37] Shuran Song, Samuel P. Lichtenberg, and Jianxiong Xiao. Sun rgb-d: A rgb-d scene understanding benchmark suite. In *CVPR*, 2015. 2

[38] Shuran Song and Jianxiong Xiao. Deep Sliding Shapes for amodal 3D object detection in RGB-D images. In *CVPR*, 2016. 1, 3

[39] Hang Su, Subhransu Maji, Evangelos Kalogerakis, and Erik G. Learned-Miller. Multi-view convolutional neural networks for 3d shape recognition. In *ICCV*, 2015. 2

[40] Julien Valentin, Vibhav Vineet, Ming-Ming Cheng, David Kim, Jamie Shotton, Pushmeet Kohli, Matthias Nießner, Antonio Criminisi, Shahram Izadi, and Philip Torr. Semanticpaint: Interactive 3d labeling and learning at your fingertips. *ACM Transactions on Graphics*, 2015. 6

[41] Chu Wang, Babak Samari, and Kaleem Siddiqi. Local spectral graph convolution for point set feature learning. *ECCV*, 2018. 2, 3

[42] Yue Wang, Yongbin Sun, Ziwei Liu, Sanjay E. Sarma, Michael M. Bronstein, and Justin M. Solomon. Dynamic graph cnn for learning on point clouds. *arXiv preprint arXiv:1801.07829*, 2018. 2, 3, 5, 6, 7, 8

[43] Zhirong Wu, Shuran Song, Aditya Khosla, Xiaoou Tang, and Jianxiong Xiao. 3d shapenets: A deep representation for volumetric shapes. In *CVPR*, 2015. 1, 2

[44] Yifan Xu, Tianqi Fan, Mingye Xu, Long Zeng, and Yu Qiao. Spidercnn: Deep learning on point sets with parameterized convolutional filters. In *ECCV*, 2018. 2, 3, 5, 6, 7, 8

[45] Yaoqing Yang, Chen Feng, Yiru Shen, and Dong Tian. Foldingnet: Point cloud auto-encoder via deep grid deformation. In *CVPR*, 2018. 2

[46] Mohsen Yavartanoo and Euyoung Kim. Spnet: Deep 3d object classification and retrieval using stereographic projection. In *ACCV*, 2018. 2

[47] Haoxuan You, Yifan Feng, Rongrong Ji, and Yue Gao. Pvnet: A joint convolutional network of point cloud and multi-view for 3d shape recognition. In *Proceedings of the ACM International Conference on Multimedia*, 2018. 2

[48] Tan Yu, Jingjing Meng, and Junsong Yuan. Multi-view harmonized bilinear network for 3d object recognition. In *CVPR*, 2018. 2

[49] Manzil Zaheer, Satwik Kottur, Siamak Ravanbakhsh, Barnabas Poczos, Ruslan R Salakhutdinov, and Alexander J Smola. Deep sets. In *Advances in Neural Information Processing Systems*, 2017. 2, 3