A Snapshot-based Approach for Self-supervised Feature Learning and Weakly-supervised Classification on Point Cloud Data

Xingye Li [1], Zhigang Zhu[1,2][a]
[1]*The City College of New York - CUNY, New York, USA*
[2]*The Graduate Center - CUNY, New York, USA*
xli020@citymail.cuny.edu, zzhu@ccny.cuny.edu

Keywords: self-supervised learning, weakly-supervised learning, 3D point cloud

Abstract: Manually annotating complex scene point cloud datasets is both costly and error-prone. To reduce the reliance on labeled data, we propose a snapshot-based self-supervised method to enable direct feature learning on the unlabeled point cloud of a complex 3D scene. A snapshot is defined as a collection of points sampled from the point cloud scene. It could be a real view of a local 3D scan directly captured from the real scene, or a virtual view of such from a large 3D point cloud dataset. First the snapshots go through a self-supervised pipeline including both part contrasting and snapshot clustering for feature learning. Then a weakly-supervised approach is implemented by training a standard SVM classifier on the learned features with a small fraction of labeled data. We evaluate the weakly-supervised approach for point cloud classification by using varying numbers of labeled data and study the minimal numbers of labeled data for a successful classification. Experiments are conducted on three public point cloud datasets, and the results have shown that our method is capable of learning effective features from the complex scene data without any labels.

1 INTRODUCTION

Deep neural networks such as PointNet (Qi et al., 2017a), DGCNN (Wang et al., 2019) have been proposed for better performances on point cloud related tasks. Datasets on individual objects such as ModelNet (Wu et al., 2015a), or on complex scene such as the Oakland dataset (Munoz et al., 2009) have been developed alongside the deep networks. The collective effort between deep neural networks and dedicated datasets continues to push up the state-of-the-art performance on the point cloud understanding tasks. To keep the deep networks from overfitting, larger datasets are often involved in the model training process. This requirement leads to the development of larger and more sophisticated labeled datasets. For example, the ModelNet datasets increased from 4K object samples / 10 classes to 12K samples / 40 classes (Wu et al., 2015a), the later ShapeNet Core55 dataset has 51K object samples for 55 categories (Chang et al., 2015). However, annotating large scale datasets comes to be very expensive both in time and human labors. This issue is becoming more prominent in applications such as hazard assessment where drive-by and fly-by LiDAR mapping systems have been used to collect massive windstorm damage data sets in recent hurricane events (Bhargava et al., 2019; Gong et al., 2012; Gong, 2013; Hu and Gong, 2018), and LiDAR is starting to be integrated into smaller mobile devices (Apple Inc., 2020), which could lead to a boom in the scale of real life complex point cloud data.

To alleviate the dependence on the labels of large datasets, unsupervised learning methods have been gaining momentum. The self-supervised approach has found success in designing "pretext" tasks, such as jigsaw puzzle reassembly (Noroozi and Favaro, 2016), image clustering (Caron et al., 2018) and image rotation prediction (Jing et al., 2018) etc, by training deep learning models for feature extraction without labels being involved. Based on the idea of solving pretext tasks, Zhang and Zhu (Zhang and Zhu, 2019) have recently developed the model of Contrast-ClusterNet, which works on unlabeled point cloud datasets by part contrasting and object clustering. While this work achieved comparable results to its supervised counterparts, it inherits the problem of other pretext-driven models on image data: In the context of point cloud understanding, both part contrasting and object clustering assume the input data

[a] https://orcid.org/0000-0002-9990-1137

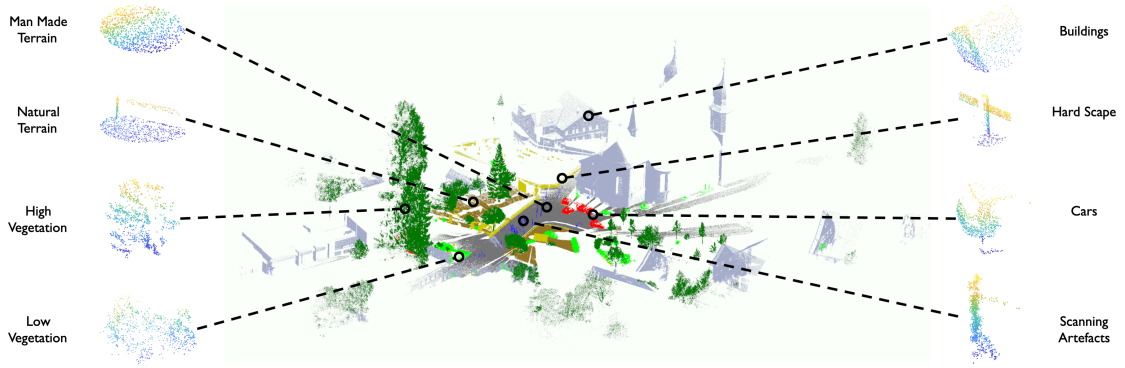

Figure 1: Visualizing snapshots sampled from the 'Bildstein3 ' scene of the Semantic3D dataset.

are well separated individual objects. This assumption limits the model's power on real life scene data or where 3D data of single objects cannot be easily obtained. Furthermore, even though the Contrast-ClusterNet approach is capable of extracting features in a self-supervised manner, a classifier still needs to be trained separately using labeled data for the classification task. Therefore an approach that uses minimal labeled data for 3D point cloud classification is of great interest.

To address the aforementioned problems, we propose a snapshot-based approach which uses the Contrast-ClusterNet model as the backbone. Our snapshot-based model for unsupervised feature learning works on both dedicated object classification datasets and complex scenes without using labels or even assuming points come from the the same objects. A snapshot is defined as a collection of points, without knowing their labels, sampled from a point cloud scene (Figure 1). It could be a real view of a local 3D scan directly captured from the real scene, or a "virtual" view of such a local 3D scan from a large 3D point cloud dataset. In our work, the captured snapshots first go through the self-supervised pipeline of a two step feature learning using ContrastNet and ClusterNet consecutively. Then a weakly-supervised approach is implemented by training an SVM classifier on the learned features with a small portion of labeled data. We also evaluate the weakly-supervised approach for point cloud classification by using various numbers of labeled data and study the minimal numbers of labeled data for a successful classification.

2 RELATED WORK

Deep Learning on Point Cloud: Inspired by the success of deep learning on grid data, such as 2D images made up of pixels, the 3D vision community has been devoting efforts in adopting deep learning to unorganized data such as 3D point cloud. Due to its nature of being a collection of unordered points, the traditional convolutional operations cannot be directly applied without being first transformed into other grid representations. This observation encouraged a series of works on the development of both voxel-based models (Maturana and Scherer, 2015; Wu et al., 2015b; Hackel et al., 2017; Huang and You, 2016), which voxelize the unordered point cloud data to 3D grids to enable 3D covolutional feature extraction, and the 2D multi-views models (Su et al., 2015; Qi et al., 2016; Novotny et al., 2017; Kanezaki et al., 2018), which take the projections of a 3D point cloud to 2D views from multiple directions, so that traditional convolutional networks can be applied. However, both approaches have some limitations: The voxel representations often have high demands on the memory and computational power and the multi-view methods often lead to the loss of geometrical information during the dimension reduction, despite its efficiency.

PointNet (Qi et al., 2017a) is the first deep network proposed to directly work with point cloud data without transforming the raw point cloud into voxels or 2D multi-views representations. This is achieved by extracting local features of each point and combine them as a global feature vector with a symmetric mapping function, such as max-pooling. However this method suffers from its limited capability of processing complex scene datasets due to its plain structure. This problem is addressed in the work of PointNet++ (Qi et al., 2017b) by proposing a hierarchical network based on the PointNet. Comparatively, PointNet++ is capable of extracting more robust features from different scales.

In addition to the three approaches above, the fourth one introduces the concept of graph to main-

tain the geometrical relations of the point cloud and operate convolutions on such graphs. In general the convolution operations are defined in two manners on point clouds: spectral based (Yi et al., 2017) and non-spectral based (Wang et al., 2019). Compared to the aforementioned approaches, the graph-based models provide more powerful tools to exploit local structures of the 3D data. Therefore, graph CNN has draw significant attention among the community.

Self- and weakly-supervised Learning on Point Cloud: Self-supervised learning aims to predict for output labels that are generated from the intrinsic information of the data, so to reduce the amount of manually labeled data needed as for fully supervised learning. Unlike 2D data, the labeling of large scale 3D point cloud data is much more difficult and laborious. To alleviate the use of labeled data, a number of self-supervised models have been proposed lately (Achlioptas et al., 2018; Sauder and Sievers, 2019; Yang et al., 2018; Zhang and Zhu, 2019). Zhang and Zhu proposed the Contrast-ClusterNet (Zhang and Zhu, 2019) with pre-text tasks of first predicting whether two segments are from the same object, leading to the ContrastNet for obtaining self-learned features, which are then used for separating the objects into different clusters using KMeans++, for training another network called ClusterNet to obtain refined self-learned features. Their work has shown the capability of learning effective features in a self-supervised manner by conducting object classification using an SVM classifier. However, this process still requires to know a set of 3D points belong to a single object. In training the SVMs, the same amount of labeled data as in supervised models is used, therefore decreasing the benefits of leaving out annotations in self-supervised learning.

Inspired by the works of weakly-supervised learning (Xu et al., 2019; Xu and Lee, 2020), we study weakly-supervised 3D point classification by utilizing few labeled data to train the SVM classifier with the learned features through the aforementioned self-supervised learning. Furthermore, our method is capable of performing on complex scene data (Munoz et al., 2009; Hackel et al., 2017) without prior assumptions of 3D points belong to the same objects. These two improvements greatly unleash the power of the self-supervised learning model on 3D point cloud data understanding.

3 METHOD

We propose a new snapshot-based approach using the Contrast-ClusterNet (Zhang and Zhu, 2019) as the backbone. Our approach consists of two major components: Snapshot-based self-supervised feature learning, and weakly-supervised classification. The Snapshot-based self-supervised feature learning, as an analogy to taking snapshots with a camera, captures small areas of the scene point cloud to train the model. We then use the trained model for feature extraction and apply weak-supervision to the extracted features for classification tasks.

3.1 Self-supervised Learning with Snapshots

Self-supervised learning often requires prior knowledge about the input data to ensure the intrinsic information of the data, from which the labels are derived, is consistent across all samples. In the case of the Contrast-ClusterNet (Zhang and Zhu, 2019), such prior assumption is that each training sample must be from a single point cloud object to enable part contrasting, which essentially predicts whether two parts are split from the same object. This assumption is kept true on datasets like ModelNet (Wu et al., 2015a) and ShapeNet (Chang et al., 2015), where each data sample is a synthetic CAD model. However, this soon becomes a limitation on real-life point cloud datasets, such as the Okaland (Munoz et al., 2009) and Semantic3D (Hackel et al., 2017) data, where an entire point cloud is a complex scene.

The pipeline: To address this issue, we propose the snapshot-based method for the self-supervised feature learning, which has three major steps: snapshot capturing, feature learning, and feature extraction (Figure 2). Given a real-life point cloud dataset, a *snapshot capturing* step generate local collections of points as snapshot samples (Figure 1(a)). In each sampling, an anchor point is randomly selected from the point cloud at first, then the kNN pioints are collected to the anchor point, where k defines the snapshot sampling area, as a snapshot. Each 'snapshot' is treated as a single point-cloud object and is fed into the three-stage Contrast-ClusterNet for *feature learning*(Figure 1(b)). In the training of a ContrastNet, we follow the random split procedure, as described in (Zhang and Zhu, 2019): two segments from the same snapshot make up a positive pair with the label 1, and a negative pair consisting of two segments from two different snapshots is labeled as 0. Then the learned features are used in obtaining pseduo-labels of snapshots with KMeans++, which then are used to train a ClusterNet for extracting more fine-grained features. Finally in *feature extraction* (Figure 1(c)), features of the snapshots are extracted using the pre-trained ClusterNet.

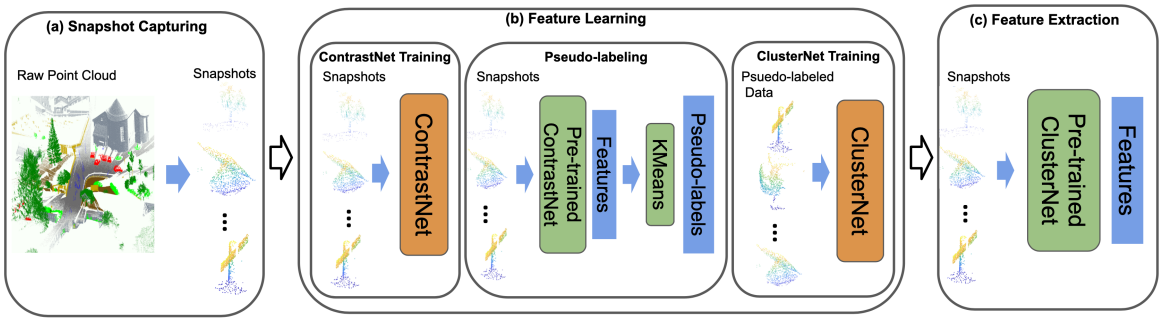

Figure 2: The Snapshot-based self-supervised feature learning pipeline: (a) Snapshot sampling from raw point cloud; (b) Feature learning by part contrasting, snapshot clustering and cluster classification; (c) Snapshot feature extraction by the previously trained ClusterNet.

Impurity of snapshots: Since the anchor point selection happens randomly, it is possible to have the anchor sitting close to the border between objects of different semantic classes. This introduces a certain degree of *impurity* to the snapshot by including some points from other classes. Compared to the object based part contrasting, where the points of a sample need to be of exactly from the same object even though its class label is not used during the training, this method fundamentally sees each snapshot as a collection of points that represent a small region of the bigger scene, and statistically, a large portion of the points in a snapshot belongs to one semantic class.

In our experiment section, we will show how the impurity of data will affect the performance of feature learning for later snapshot classification. To quantify the impurity of the snapshots, we present a metric to evaluate our snapshot sampling quality. For easy understanding, we take the opposite of the impurity, namely purity. When sampling from the Oakland (Munoz et al., 2009) and Semantic3D (Hackel et al., 2017), we utilize the original labels in these datasets to approximate the class label of each snapshot for the sake of snapshot classification; in our future work we will explore scene clustering and segmentation without using any label information.

A label is assigned to a snapshot by voting from all associated points in that sample. The class that most points agree on is chosen as the label for the snapshot, the voting procedure can be parameterized as:

$$C_x = \underset{i}{argmax} \sum_{j=1}^{K} I(y_j = i), \tag{1}$$

where x is the snapshot sample, y_j is the point-wise label for x (j=1,...,K), K represents the number of points in the snapshot x, and I is an indicator function for the class of each point. Thus the purity is given by

$$P(x) = \frac{\sum_{j=1}^{k} I(y_j = C_x)}{k} \tag{2}$$

The statistics of the voted class labels and the purity for each sample will be further discussed in Section 4.1 using real examples.

3.2 Classification with Fewer Labels

End to end fully-supervised learning requires a significant amount of labeled data for training purposes, and frequently in real life scenarios, large amount of labels are not available right away when needed for full-supervision. However, obtaining a few labeled data in a short notice is relatively cost effective. Therefore, we further explore a weakly-supervised approach for classification, where far fewer labels are required to fulfil the tasks.

After the self-supervised feature learning, a classifier is trained on the extracted features of labeled training data for classification. We choose the SVM as our classifier and train it with various numbers of labeled data, whose features are extracted from the training samples by the pre-trained ClusterNet. The numbers of samples for training are organized as declining percentages of the total training data: 100%, 50%, 20%, 10%, 5%, and 1%. As a comparison, the fully-supervised model DGCNN (Wang et al., 2019) is trained on each percentage of samples alongside our model to show the performance comparison.

4 RESULTS

Extensive experiments are conducted to evaluate the effectiveness of our proposed snapshot-based approach for both self-supervised feature learning and weakly-supervised point cloud classification. The implementation and experimental results are described in details in the following sub-sections, including (1) datasets and snapshot generation/evaluation; (2)

snapshot-based feature learning; (3) point cloud classification with fewer labeled data; and (4) evaluation of clustering with snapshots. The code can be found at https://github.com/Aleexy/snapshot_clusternet.

4.1 Datasets and Snapshots

Datasets: In this work, experiments are carried out on three point cloud datasets: the Oakland point cloud dataset (Munoz et al., 2009), the Semantic3D large scale point cloud classification benchmark (Hackel et al., 2017), and the ModelNet40 (Wu et al., 2015a). Both the Oakland and Semantic3D datasets are complex scene point cloud data. The Oakland dataset has only one single scene, containing 1.6 million points, divided into five classes: scatter misc, default wire, utility pole, load bearing, and facade. The Semantic3D consists of a variety of scenes across eight classes: man-made terrain, natural terrain, high vegetation, low vegetation, buildings, hard scape, scanning artefacts, and cars (Figure 1). Considering the huge scale of this dataset (4 billion points) is beyond our computational capacity, we only choose two scenes, named 'untermaederbrunnen3' and 'bildstein3' for our experiments. The ModelNet40 dataset is a collection of synthetic point cloud objects across 40 classes, where there are roughly 9840 single objects in the training set and 2468 in the test set.

Sample balancing: Compared to the synthetic dataset ModelNet40, the two real life datasets are heavily unbalanced toward certain classes. To thoroughly evaluate our method, we first apply two resampling schemes to the Semantic3D dataset before taking on the proposed sampling method. The original unbalanced dataset is notated as **OUD**, on which two resampling schemes are applied:

(1) Upsampling and downsampling based on the class distributions. Points from each class are evenly removed or duplicated with some noises according to the size of each class. The resulted new balanced dataset (**BD**) has all eight classes with numbers of points close to each other.

(2) Shifting duplicated points, and shrinking distances between downsampled points. When upsampling a class, all included points are duplicated and then shifted around as a whole. While shifting ensures that the class density is not destructed by cluttering a large amount of duplicates in the same region, the shrinking process also approximately maintains the original sampling rate. We notate the balanced dataset by shifting and shrinking as **SSBD**.

Snapshot generation: The snapshot based sampling is applied to the Oakland dataset and all three configurations of the Semantic3D benchmark, namely OUD, BD, and SSBD. We generate 5000 samples from the Oakland dataset for training and 500 samples for testing, with 1024 points in each sample. Similarly, we generate 8000 samples from each chosen scene of the Semantic3D as training set and 800 each for testing. Due to the high resolution of the Semantic3D, the snapshot sampling described above is slightly altered to capture larger area. We collect 10 times of points that are nearest to the anchor point through kNN, which forms a mega snapshot of 10240 points. To reduce the size of the mega snapshot while maintaining the new sparse sampling rate, the mega snapshot is further subsampled five times at the size of 1024 points, and the resulting five snapshots are kept as the training or test samples.

Snapshot purity: Based on the proposed purity metric, we run the snapshot sampling procedure 100 times on both the Okaland and Semantic3D datasets for sampling statistics. As shown in Table 1, the snapshot purity of a class is correlated with the numbers of snapshots being sampled. From the point-wise perspective, when choosing an anchor point, each point in the scene has equal chance being selected due to random sampling. However, class-wise speaking, points from less abundant classes have smaller probabilities being selected as the anchor, which leads to smaller numbers of generated samples. Further more, when collecting points surrounding a small class anchor, the chance of including inter-class points is relatively higher. Therefore, the purity of snapshots can be improved by re-balancing the distribution of the dataset, and this is illustrated in Table 2, where samples from the resampled BD and SSBD have higher purity than the OUD samples.

We can see that the overall snapshot purity of both datasets, despite different configurations, is above 90%. This meets our assumption that a snapshot is highly capable of resembling a piece of from one object class.

Snapshots versus objects: To comparatively evaluate our method, we also apply the same sampling procedure on each class grouped by the original labels, in contrast to the whole scene of point cloud. Consequently, samples obtained exclusively from a class have 100% purity, and we refer to them as 'objects' in contrast to 'snapshots'. The object-based approach is mainly for comparison, even though it may have its own value if obtaining objects are possible.

4.2 Snapshot-based Feature Learning

To verify our snapshot based approach for self-supervised feature learning, we conduct experiments on both snapshots and objects derived from the

	Scatter Misc	Default Wire	Utility Poles	Load Bearing	Facade	Overall
Purity (%)	91.3 ± 0.42	35.08 ± 20.74	1.74 ± 6.93	98.37 ± 0.13	89.92 ± 0.78	96.08 ± 0.15
Samples	1092.93 ± 31.93	1.49 ± 1.27	0.06 ± 0.24	3476.3 ± 35.71	429.22 ± 18.73	5000

Table 1: Statistics of snapshot sampling on the Oakland dataset, where 5000 snapshots are sampled at each run, for 100 runs. The purity (mean and variance in percentile) stands for the percentage of points agree on the voted class of one snapshot. Samples represents the number of snapshots being sampled for each class (mean and variance).

Purity (%)		Man Made Terrain	Natural Terrain	High Vegetation	Low Vegetation	Buildings	Hard Scape	Scanning Artefacts	Cars	Total
Scene 1	OUD	98.45 ±0.28	57.47 ±10.23	95.68 ±2.91	92.69 ±1.94	99.48 ±1.45	97.02 ±0.8	73.2 ±32.41	87.85 ±4.46	98.39 ±0.19
	BD	94.64 ±0.93	92.83 ±0.89	98.16 ±0.47	95.51 ±0.83	97.02 ±0.63	96.17 ±0.76	92.99 ±0.81	98.5 ±0.51	95.74 ±0.28
	SSBD	98.16 ±0.45	82.69 ±1.23	97.44 ±0.65	93.03 ±0.88	95.68 ±0.59	98.22 ±0.65	82.7 ±1.31	92.63 ±1.0	92.93 ±0.32
Scene 2	OUD	94.45 ±0.89	96.46 ±0.36	95.13 ±0.75	91.59 ±1.22	94.22 ±1.1	94.25 ±0.74	-	85.59 ±2.81	94.96 ±0.28
	BD	85.95 ±1.38	88.33 ±1.39	91.27 ±1.03	95.09 ±0.71	94.5 ±0.79	91.7 ±0.87	91.14 ±0.94	98.56 ±0.44	92.37 ±0.37
	SSBD	88.96 ±1.18	98.31 ±0.54	86.69 ±1.3	95.84 ±0.72	91.33 ±0.9	93.06 ±1.07	78.14 ±1.5	88.95 ±0.95	90.63 ±0.4

Table 2: Statistics of snapshot sampling on all three configurations on the two scenes from the Semantic3D dataset (Scene 1: 'Untermaederbrunnen3'; Scene 2:'Bildstein3'). The sampling is conducted 100 times, with 8000 samples per time. '-' indicates no snapshots being sampled.

datasets of Oakland and Semantic3D. The evaluations are based on the classification accuracy on the testing samples of an SVM (Linear kernel) trained on the extracted features of training samples.

On the same scenes: Table 3 shows that, our snapshot-based method is able to achieve higher accuracy than the DGCNN (a supervised method) using 100% and 10% of the training samples. It is also shown that the snapshot-based method is more robust than the object-based ClusterNet and the DGCNN when the labeled training data reduces. On the Oakland data, the accuracy drops by 4.6% from 90.6% on the snapshot-based method when the training data decreases from 100% to 10% whereas the DGCNN drops 9.58% from 85.21%. On the OUD, our method performs only slightly worse (0.88%) with 10 times fewer training data. As a comparison, the DGCNN performs 5.25% worse on the snapshots and 14.75% worse on the objects, and the object-based ClusterNet also drops 3.25% on the accuracy. The shown results validate our hypothesis that the proposed snapshot-based method is able to learn effective features from the raw point cloud complex scene in a self-supervised manner.

We would like to point out that class labels were used in re-sampling the data to generate balanced datasets. This seems to contradict to our assumption of not using labels for generating snapshots, but the goal is to show if our snapshot approach can work as well on unbalanced data as on balanced data. Results in Table 3 suggest that the performance on the original unbalanced data (OUD) is similar or even better than on the resampled and thus balanced datasets BD and SSBD. This indicates that even without resampling, our model is robust for feature learning, therefore resampling is not necessary in practice: despite the two classes in the OUD have low purity score (Table2), the overall accuracy still shows that this model is robust enough to learn from the "noisy" snapshot inputs.

	Ours		DGCNN	
Data Percentage	100%	10%	100%	10%
Oakland				
Snapshots	90.6%	86.0%	85.21%	75.63%
Objects	98.4%	94.4%	100%	86.88%
OUD				
Snapshots	96.38%	95.5%	86.5%	71.5%
Objects	98.13%	94.88%	85.88%	71.13%
BD				
Snapshots	97.75%	95.25%	91.88%	75.5%
Objects	98.63%	93.5%	87.75%	77.13%
SSBD				
Snapshots	97.25%	92.75%	90.13%	80.38%
Objects	94.88%	87.88%	90.00%	69.13%

Table 3: Classification performance on the snapshots and objects from the Oakland dataset and the three configurations of the scene, 'untermaederbrunnen3' in the Semantic3D dataset, using the our method and DGCNN. Data percentage represents the amount of labeled data used in the training.

On different scenes: We further evaluate the generalization ability of the learned features with and without fine tuning the pre-trained ClusterNet on increasing percentages of samples from another scene in the Semantic3D dataset. The model is first trained

Data Percentage	0%	1%	5%	10%	20%
OUD					
Snapshots	89.38	91.88	91.75	92.13	92.13
Objects	98.25	97.38	98.25	98.63	98.5
BD					
Snapshots	95.88	96.38	96.5	96.88	96.88
Objects	98.13	98.75	98.5	98.0	98.5
SSBD					
Snapshots	94.88	96.13	96.5	97.0	97.13
Objects	98.5	98.25	98.5	98.63	98.63

Table 4: Classification performance of our method on the scene 'Bildstein', using ClusterNet pre-trained on the scene 'Untermaederbrunnen3' and fine-tuned on increasing percentages of samples from 'Bildstein'.

on snapshots or objects from the scene 'Untermaederbrunnen3', then fine tuned and tested on samples from the scene 'Bildstein'. For this task, we use all 100% of the training data in the first scene to train the SVM classifier. The results are shown in Table 4. We can see that the accuracy of classifying OUD snapshots starts from 89.38% when fine tuning is not applied (0% fine tuning data), and it increases to 92.13% when further trained on 20% of the fine tuning data, a 2.75% improvement. In comparison, the performance difference of classifying OUD objects without and with fine-tuning is neglectable (0.25%). Hence, such a boost on performance by fine tuning on small portion of training data is more significant on the snapshots compared to the objects. This is also seen from the BD and SSBD data, where the performance differences without and with 20% fine-tuning are larger in snapshots than objects, which are 1.00% vs 0.37%, and 2.25% vs 0.13%, respectively.

It is worth noting that when evaluating the pre-trained model directly on the testing snapshots from another scene, the yielded accuracy is much lower on the OUD than the BD and SSBD, compared to testing on the same scene (shown in Table 3). This is due to the different class distribution between the OUD of the two scenes. Without resampling the dataset to balance out the class distribution, an under-sampled class could be over-sampled in another scene and contributes to feature bias.

4.3 Classification with Fewer Labels

After the self-supervised feature learning, an SVM classifier is trained on the features extracted by the pre-trained ClusterNet. To verify the effectiveness of the proposed weakly supervised classification, we gradually reduce the amount of labels involved in the training of the SVM. As a comparison, we train the end-to-end model DGCNN in the same manner. Experiments are conducted on the ModelNet40, Oakland dataset, and the three configurations of the Semantic3D.

On ModelNet40: Figure 3 illustrates how the performance is different between our method and the DGCNN on a decreasing number of training samples on ModelNet40. DGCNN begins with a higher accuracy than the ClusterNet when trained on all labeled data, but the performance drops rapidly when the data supply reduces. With 20% of the labeled data, the ClusterNet starts to outperform the DGCNN and the advantage becomes larger when the labeled data further reduces. On 5% labeled data, the DGCNN is at a 72.61% accuracy while our method maintains a much higher performance (78.48%).

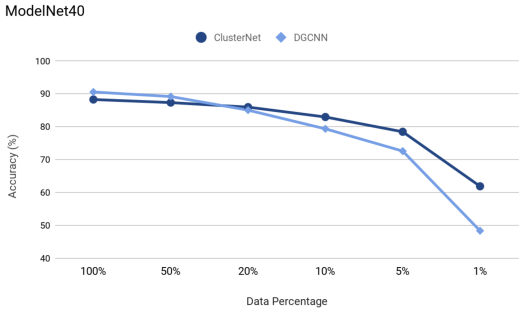

Figure 3: Comparison of the classification accuracies between the ClusterNet and DGCNN on decreasing numbers of labeled data from the ModelNet40

On Oakland and Semantic3D datasets: Figure 4 shows the same comparison on the real life datasets. Our snapshot-based method shows an advantage on classifying snapshots than the DGCNN on all four training scenarios. The ClusterNet performance also shows a higher resistance on the reducing of labeled data. Using labeled data from 100% to 5%, the ClusterNet accuracy only drops by 8.0%, 1.5%, 4.25%, and 6.25% for (a) Oakland, (b) OUD, (c) BD and (d) SSBD, to 82.6%, 94.88%, 93.5% and 91.0%, respectively. In comparison, the similar trend of the DGCNN shown on the ModelNet40 is observed on the BD and SSBD cases, that the accuracy declines drastically with fewer labeled data available for training.

4.4 Evaluate Clustering Performance

We also present the feature embedding using t-SNE (Figure 5) of the snapshots to show the capability of the snapshot-based self-supervised features for conducting clustering task when no labels are available. It can be seen that snapshots from the same class tend to stay closer to each other. There are classes being separated into different clusters, however, most clus-

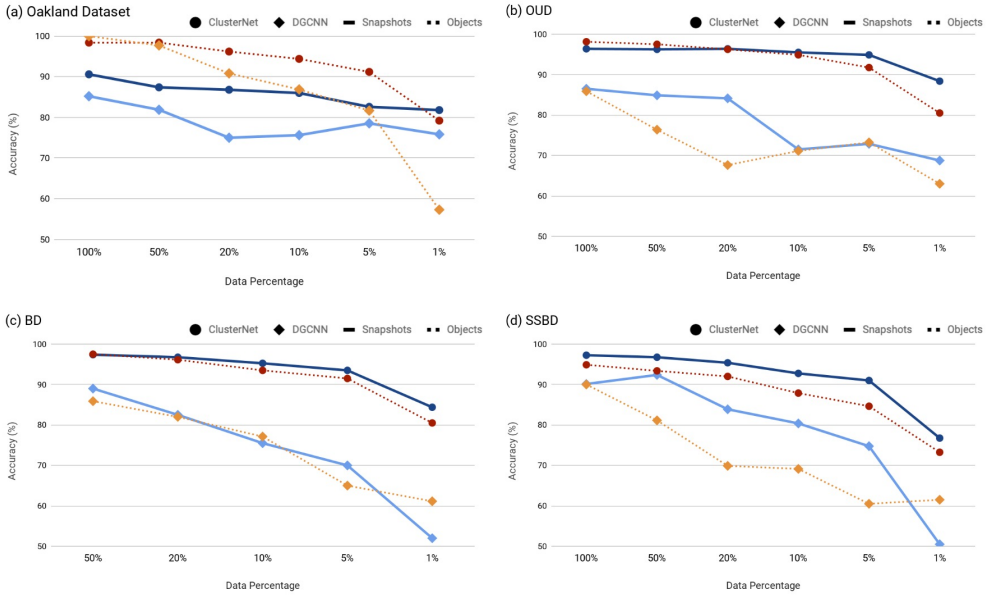

Figure 4: Comparison of the snapshots and objects classification accuracies between the ClusterNet and DGCNN on decreasing numbers of labeled data on complex scene point cloud datasets. The OUD, BD, and SSBD are three sampling configurations derived from the Semantic3D benchmark. Solid lines are results on the snapshots and dotted lines represent the objects.

ters can be well separated from each other, which verifies the discriminative power of the features learned by the proposed method.

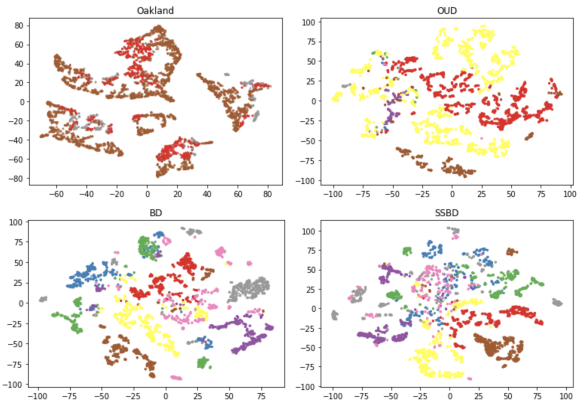

Figure 5: Visualization of the feature embedding of the snapshots from the Oakland dataset and the three Semantic3D configurations. The clusters are colored on the labels for visualization.

5 CONCLUSION

In this work, we have proposed a snapshot-based self-supervised model for feature learning on the complex scene point cloud dataset, and a weakly-supervised method for point cloud classification. The proposed methods are evaluated and verified on a synthetic point cloud dataset and two real life complex scene datasets. The experimental results indicate that our method is capable of learning effective features from unlabeled complex scene point cloud dataset, even without resampling the data. Moreover, the weakly-supervised classification method is shown to be able to outperform the end-to-end model DGCNN when training with fewer labeled data. In future works, the impact of snapshot sampling rates on feature quality will be further investigated, which could potentially be adopted with a 3D Lidar to capture properly sized snapshots for real time learning.

6 ACKNOWLEDGMENTS

The research is supported by National Science Foundation through Awards PFI #1827505 and SCC-Planning #1737533, and Bentley Systems, Incorporated, through a CUNY-Bentley CRA 2017-2020. Additional support is provided by a CCNY CEN Course Innovation Grant from Moxie Foundation, and the Intelligence Community Center of Academic Excellence (IC CAE) at Rutgers University (Awards #HHM402-19-1-0003 and #HHM402-18-1-0007).

REFERENCES

Achlioptas, P., Diamanti, O., Mitliagkas, I., and Guibas, L. (2018). Learning representations and generative models for 3d point clouds. In *International conference on machine learning*, pages 40–49. PMLR.

Apple Inc. (2020). iPad Pro. https://www.apple.com/ipad-pro/specs/.

Bhargava, P., Kim, T., Poulton, C. V., Notaros, J., Yaacobi, A., Timurdogan, E., Baiocco, C., Fahrenkopf, N., Kruger, S., Ngai, T., Timalsina, Y., Watts, M. R., and Stojanović, V. (2019). Fully integrated coherent lidar in 3d-integrated silicon photonics/65nm cmos. In *2019 Symposium on VLSI Circuits*, pages C262–C263.

Caron, M., Bojanowski, P., Joulin, A., and Douze, M. (2018). Deep clustering for unsupervised learning of visual features. In *Proceedings of the ECCV*, pages 132–149.

Chang, A. X., Funkhouser, T., Guibas, L., Hanrahan, P., Huang, Q., Li, Z., Savarese, S., Savva, M., Song, S., Su, H., Xiao, J., Yi, L., and Yu, F. (2015). Shapenet: An information-rich 3d model repository.

Gong, J. (2013). Mobile lidar data collection and analysis for post-sandy disaster recovery. In *Computing in Civil Engineering (2013)*, pages 677–684.

Gong, J., Zhou, H., Gordon, C., and Jalayer, M. (2012). Mobile terrestrial laser scanning for highway inventory data collection. In *Computing in Civil Engineering (2012)*, pages 545–552.

Hackel, T., Savinov, N., Ladicky, L., Wegner, J. D., Schindler, K., and Pollefeys, M. (2017). SEMANTIC3D.NET: A new large-scale point cloud classification benchmark. In *ISPRS Annals of the Photogrammetry, Remote Sensing and Spatial Information Sciences*, volume IV-1-W1, pages 91–98.

Hu, X. and Gong, J. (2018). Advancing smart and resilient cities with big spatial disaster data: Challenges, progress, and opportunities. In *Data Analytics for Smart Cities*, pages 53–90. Auerbach Publications.

Huang, J. and You, S. (2016). Point cloud labeling using 3d convolutional neural network. In *2016 23rd International Conference on Pattern Recognition (ICPR)*, pages 2670–2675. IEEE.

Jing, L., Yang, X., Liu, J., and Tian, Y. (2018). Self-supervised spatiotemporal feature learning via video rotation prediction. *arXiv preprint arXiv:1811.11387*.

Kanezaki, A., Matsushita, Y., and Nishida, Y. (2018). Rotationnet: Joint object categorization and pose estimation using multiviews from unsupervised viewpoints. In *Proceedings of the IEEE CVPR*, pages 5010–5019.

Maturana, D. and Scherer, S. (2015). Voxnet: A 3d convolutional neural network for real-time object recognition. In *2015 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pages 922–928. IEEE.

Munoz, D., Bagnell, J. A. D., Vandapel, N., and Hebert, M. (2009). Contextual classification with functional max-margin markov networks. In *Proceedings of IEEE CVPR*.

Noroozi, M. and Favaro, P. (2016). Unsupervised learning of visual representations by solving jigsaw puzzles. *Lecture Notes in Computer Science*, page 69–84.

Novotny, D., Larlus, D., and Vedaldi, A. (2017). Learning 3d object categories by looking around them. *2017 IEEE International Conference on Computer Vision (ICCV)*.

Qi, C. R., Su, H., Mo, K., and Guibas, L. J. (2017a). Pointnet: Deep learning on point sets for 3d classification and segmentation. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 652–660.

Qi, C. R., Su, H., Nießner, M., Dai, A., Yan, M., and Guibas, L. J. (2016). Volumetric and multi-view cnns for object classification on 3d data. In *Proceedings of the IEEE CVPR*, pages 5648–5656.

Qi, C. R., Yi, L., Su, H., and Guibas, L. J. (2017b). Pointnet++: Deep hierarchical feature learning on point sets in a metric space. In *Advances in neural information processing systems*, pages 5099–5108.

Sauder, J. and Sievers, B. (2019). Context prediction for unsupervised deep learning on point clouds. *arXiv preprint arXiv:1901.08396*.

Su, H., Maji, S., Kalogerakis, E., and Learned-Miller, E. G. (2015). Multi-view convolutional neural networks for 3d shape recognition. In *Proc. ICCV*.

Wang, Y., Sun, Y., Liu, Z., Sarma, S. E., Bronstein, M. M., and Solomon, J. M. (2019). Dynamic graph cnn for learning on point clouds. *ACM Transactions on Graphics (TOG)*, 38(5):1–12.

Wu, Z., Song, S., Khosla, A., Yu, F., Zhang, L., Tang, X., and Xiao, J. (2015a). 3d shapenets: A deep representation for volumetric shapes. In *Proceedings of the IEEE CVPR*, pages 1912–1920.

Wu, Z., Song, S., Khosla, A., Yu, F., Zhang, L., Tang, X., and Xiao, J. (2015b). 3d shapenets: A deep representation for volumetric shapes. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 1912–1920.

Xu, K., Yao, Y., Murasaki, K., Ando, S., and Sagata, A. (2019). Semantic segmentation of sparsely annotated 3d point clouds by pseudo-labelling. In *2019 International Conference on 3D Vision (3DV)*, pages 463–471.

Xu, X. and Lee, G. H. (2020). Weakly supervised semantic point cloud segmentation: Towards 10x fewer labels. In *Proceedings of the IEEE/CVF CVPR*, pages 13706–13715.

Yang, Y., Feng, C., Shen, Y., and Tian, D. (2018). Foldingnet: Point cloud auto-encoder via deep grid deformation. In *Proceedings of the IEEE CVPR*, pages 206–215.

Yi, L., Su, H., Guo, X., and Guibas, L. (2017). Syncspeccnn: Synchronized spectral cnn for 3d shape segmentation. *2017 IEEE CVPR*.

Zhang, L. and Zhu, Z. (2019). Unsupervised feature learning for point cloud understanding by contrasting and clustering using graph convolutional neural networks. In *2019 International Conference on 3D Vision (3DV)*, pages 395–404. IEEE.