Unsupervised Feature Learning for Point Cloud by Contrasting and Clustering with Graph Convolutional Neural Network

Thesis

Submitted in partial fulfillment of the requirements for
the Degree of Master of Computer Engineering

At

The City College of the City University of New York

By

Ling Zhang

May 2019

Approved:

Prof. Zhigang Zhu, Thesis Advisor

Prof. M. Umit Uyar, Director of Computer Engineering Program

Abstract

Recently, deep graph neural networks (GNNs) have attracted significant attention for point cloud understanding tasks, including classification, segmentation, and detection. However, the training of such deep networks still requires large amount of annotated data, which is both expensive and time-consuming. To alleviate the cost of collecting and annotating large-scale point cloud datasets, we propose an unsupervised learning approach to learn features from unlabeled point cloud "3D object" dataset by using part contrasting and object clustering with GNNs. In the contrast learning step, all the samples in the 3D object dataset are cut into two parts and put into a "part" dataset. Then a contrast learning GNN (ContrastNet) is trained to verify whether two randomly sampled parts from the part dataset belong to the same object. In the cluster learning step, the trained ContrastNet is applied to all the samples in the original 3D object dataset to extract features, which are used to group the samples into clusters. Then another GNN for clustering learning (ClusterNet) is trained to predict the cluster IDs of all the training samples. The contrasting learning forces the ContrastNet to learn high-level semantic features of objects but probably ignores low-level features, while the ClusterNet improves the quality of learned features by being trained to discover objects that probably belong to the same semantic categories by using cluster IDs. We have conducted extensive experiments to evaluate the proposed framework on point cloud classification tasks. The proposed unsupervised learning approach obtain comparable performance to the state-of-the-art unsupervised learning methods that used much more complicated network structures. The code of this work is publicly available via: https://github.com/lingzhang1/ContrastNet

Acknowledgements

First, I would like to thank my thesis advisor Dr. Zhigang Zhu, Herbert G. Kayser Professor of Computer Science, who gave me this opportunity to conduct my thesis research with him. Prof. Zhu has not only provided me with professional advice on research, but also guided me to be a better researcher. He always gave me full freedom to exploit innovative ideas but in the meantime steered me in the right direction whenever needed. Under his supervision, my experience of research was actually very enjoyable.

I would also like to thank two of my fellow graduate students, Longlong Jing and Gregory Olmschenk, who hepled me to identify and solve problems patiently. I would also like to thank Professor Jianting Zhang who provided me machines to conduct experiments at the early stage of the thesis. Without their sincere assistance and generous support, the thesis could not have been successfully completed.

Finally, I must express my very profound gratitude to my parents and to my husband for providing me with unfailing support and continuous encouragement throughout my years of study and through the process of researching and writing this thesis. This accomplishment would not have been possible without them.

The work is partially supported by the National Science Foundation via awards #CNS-1737533 and #IIP-1827505, as well as a CUNY-Bentley CRA.

Contents

List of Figures

List of Tables

Chapter 1

Introduction

1.1 Background

With ever increasing applications, point cloud data understanding with deep graph convolutional neural networks (GNNs) has drawn extensive attention [1–4]. Various networks, such as PointNet [1], PointNet++ [3], DGCNN [2] and etc., and datasets such as ModelNet [5], ShapeNet [4], and SUNCG [6], have been proposed for point cloud understanding tasks. With the help of deep models and large-scale labeled datasets, significant progress has been made on point cloud understanding tasks, including classification, segmentation and detection.

GNNs typically have millions of parameters which could easily lead to over-fitting. Large-scale annotated datasets are needed for the training of such deep networks. However, the collection and annotation of point cloud datasets are very time-consuming and expensive since pixel-level annotations are needed. With the powerful ability to learn useful representations from unlabeled data, the unsupervised learning methods, sometimes also known as self-supervised learning methods, have drawn significant attention.

The general pipeline of unsupervised learning with a deep neural network is to design a "pretext" task for the network to solve while the label for this pretext tasks can be automatically generated based on the attributes of the data. After

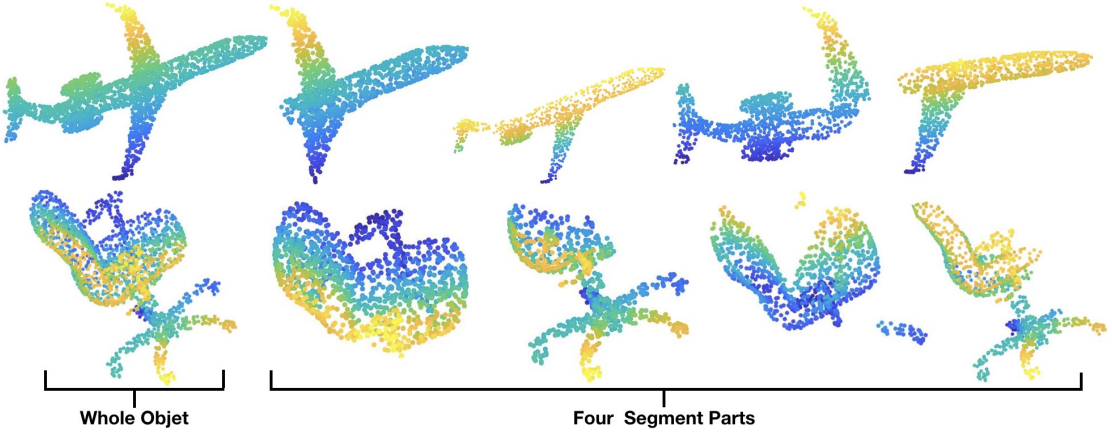

FIGURE 1.1: The Figure of segments of objects. Each row consists of a 3D point cloud object and its four different segments. Human can easily recognize the object and the locations of the segments in the object even for a small segment. Inspired by this observation, we propose to train GNNs to learn features from unlabeled dataset by recognize whether two segments are from the same object.

the network is trained with pretext tasks, the network will be able to capture useful features. Recently, many unsupervised learning methods have been proposed to learn image features by training networks to solve pretext tasks such as playing image jigsaw [7], clustering images [8], predicting image rotations [9], image inpainting [10], generating images with generation adversarial network [11], etc. The unsupervised learning methods for image feature learning have obtained great success and the performance of the unsupervised learning methods sometimes come very close to that of the supervised methods [8, 12].

Some unsupervised learning methods have also been proposed for point cloud unsupervised learning [13–18]. Most of them are based on Auto-Encoder (AE) [15–18]. Various AEs are proposed and the features are obtained by training AEs to reconstruct the 3D point cloud data. Since the main purpose of an AE is to reconstruct the data, the networks may need to memorize the low-level features of the point cloud.

1.2 Overview of the Work

In this paper, we propose an unsupervised feature learning approach for point cloud by training GNNs to solve two pretext tasks consecutively, which are part contrasting and object clustering. Specifically, the network is trained to accomplish two pretext tasks: to compare (contrast) two point cloud cuts and to cluster point cloud objects. First, all the 3D point objects are cut into two parts and a GNN (ContrastNet) is trained to verify whether two randomly sampled parts from the dataset belong to the same object. Second, the point cloud data is clustered into clusters by using the features learned by the ContrastNet, and another GNN (ClusterNet) is trained to predict the cluster ID of each point cloud data. The contrasting learning forces the ContrastNet to learn high-level semantic features that describe the similarities between parts of the same objects, whereas the predicted cluster IDs boost the quality of learned features by training the ClusterNet to discover objects that belong to the same semantic categories.

In summary, our main contributions in this paper are as follows:

- A generalized and effective unsupervised feature learning framework is proposed for point cloud data. By training deep neural networks to solve two pretext tasks, part contrasting and object clustering, the networks are able to learn semantic features related to object parts and categories from point cloud data without using any annotations.
- In particular, aligning pseudo-labels for point clouds using clustering is able to transfer knowledge from pre-training models to fine-tuning models by learning semantic features that put object into the same clusters. This step significantly boosts the classification performance, a 2.9% improvement on ModelNet40.
- The extensive experiments show that our proposed approach outperforms most of the state-of-the-art unsupervised learning methods. With the features learned from the unlabeled dataset, the proposed model obtains 86.8% and 93.8% on ModelNet40 and ModelNet10 dataset respectively.

1.3 Outline of the Thesis

The organization of the thesis is as the following. After an introduction in Chapter 1, related work is discussed in Chapter 2. Then in Chapter 3, the proposed approach is described, with details of two main components: the ContrastNet and the ClusterNet, after the introduction of the base graph CNN model. Chapter 4 provides experimental results and analyses. Finally in Chapter 5 we conclude the work and discuss a few future research directions.

Chapter 2

Related Work

2.1 Point Cloud Understanding

Various approaches have been proposed for point cloud understanding tasks, including classification, segmentation, and recognition and detection. All the approaches can be classified into three categories: hand-crafted methods [14, 19–25], CNNs on regular 3D data [5, 26–33], and CNNs on unordered 3D data (Point Clouds) [1–4].

The first type of methods is hand-crafted based methods. These traditional methods capture the local geometric structure information of point clouds such as intrinsic descriptors [19–21], or extrinsic descriptors [14, 22–25]. Although the hand-crafted methods have been applied to point cloud data, these methods have very limited performance of 3D data analysis.

Applying 3D convolutional neural networks to regular 3D data usually obtained better performance than traditional hand-crafted features. There are several approaches to handle the regular 3D data with CNN: Volumetric methods [5, 26–29] voxelize unordered data to a static 3D grid then 3D CNNs are used to process the data. This kind of methods has a constraint on efficiency and complexity due to the data sparsity and cost of CNNs. Multi-view methods [30, 31] use 2D CNNs

after rendering the 3D data to 2D images, which have obtained significant performance improvement on the classification task. However, this kind of methods has constraints in doing point level task, such as segmentation. Spectral methods [34, 35] apply spectral CNNs on meshes that are constrained by the expandability to other data formats. Feature-extracting methods [32, 33] extract features of 3D data and then apply CNNs to the features, which deeply depend on the quality of the extracted features.

Recently, a number of methods have been proposed for understanding unordered point cloud data [1–4]. Qi *et al.* made the first attempt to designing a deep net architecture, named PointNet [1], for using unordered point cloud to perform 3D shape classification, shape part segmentation and scene semantic parsing tasks. PointNet process each 3D point in a sample individually, therefore disarrangement of the point cloud will not constrain the function of the model. However, because of this, PointNet does not utilize the local structure of point cloud, which limits its ability to recognize fine-grained patterns. Later, they proposed PointNet++ which applied PointNet recursively on a nested partitioning of the input point set [3] to improve the PointNet and address the impact of local information lost. To capturing local structure, Wang *et al.* proposed an edge-convolutional network to specifically model local neighborhood information by applying convolutions over the k nearest neighbors calculated by KNN in metric space, and the k nearest neighbors can be dynamically updated in different layers [2].

2.2 Unsupervised Feature Learning

Various unsupervised learning methods have been proposed to learn features from unlabeled data [8, 13–18]. Girdhar *et al.* proposed the TL-embedding network[15], which consists of an autoencoder that ensures the representation is generative and a convolutional network that ensures the representation is predictable. Sharma *et al.* proposed a full convolutional volumetric autoencoder to learn volumetric representation from noisy data by estimating the voxel occupancy grids [16].

Achlioptas *et al.* proposed LatentGAN by introducing a new deep AutoEncoder (AE) network with state-of-the-art reconstruction quality and generalization ability for point cloud data [17]. Yang *et al.* proposed FoldingNet which is a novel end-to-end autoencoder that is the state-of-the-art for unsupervised feature learning on point clouds [18]. A graph-based enhancement is applied to the encoder to enforce local structures on top of PointNet, and a folding-based decoder deforms a canonical 2D grid onto the underlying 3D object surface of a point cloud.

Most of the deep learning based methods use auto-encoder variations for learning features on unlabeled point cloud data. However, the purpose of the autoencoder is to reconstruct the data and the feature may have a good performance on low-level tasks such as completion, reconstruction, and denoise, but have an inferior performance on tasks demands more high-level semantic meanings. Therefore, we propose the ContrastNet and ClusterNet to learn features by exploring high-level semantic features in the part and object levels respectively. Our method outperforms most of the unsupervised methods on two MoldelNet datasets and only 0.6% lower than the supervised method PointNet on the ModelNet40 dataset.

Chapter 3

Method

3.1 An Overview

To learn features from unlabeled point cloud data, we propose to learn features by training networks to accomplish both of the part contrasting and the object clustering pretext tasks. The pipeline of our framework is illustrated in Fig. 3.1, which includes three major steps: ContrastNet for part contrast learning, clustering using the learned features, and then ClusterNet for object cluster learning using the cluster IDs. Here is a summary of the three modules before we get into details of the ContrastNet and ClusterNet.

a) ContrastNet: Semantic Features from Part Contrasting: The first step is to learn features by training a network called ContrastNet to accomplish the part contrast task. Specifically, the part contrast task is to verify whether two point cloud segments (parts) belong to the same sample (object). The positive pair is drawn by selecting two different segments from the same object, while the negative pair is drawn by selecting two segments from two different objects.

b) Clustering to Obtain Pseudo-labels: After the training with the part contrasting finished, the trained ContrastNet can obtain high-level semantic features from point cloud data. Using the extracted features, the 3D point cloud data samples are clustered into different clusters. Kmeans++ [36] is used as the clustering

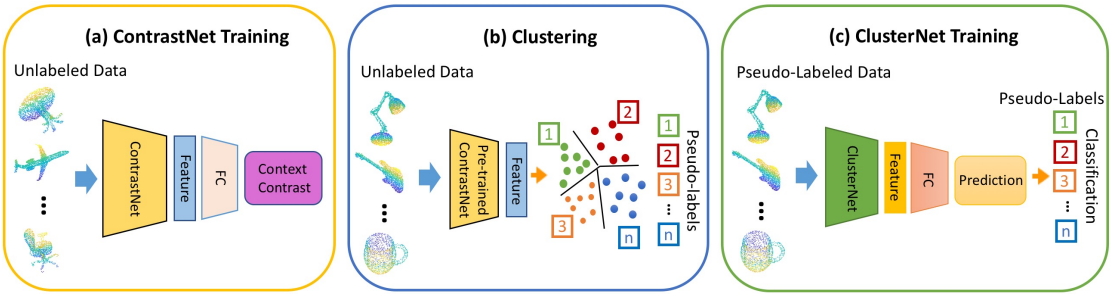

FIGURE 3.1: The Figure of pipeline. The unsupervised feature learning includes three main steps: (a) ContrastNet for part contrast learning, by verifying whether two point cloud cuts belong to the same object; (b) Cluster samples of 3D objects and assign cluster IDs, using the features learned by ContrastNet; (c) ClusterNet for object clustering learning, by training the network with the 3D point cloud data while the labels are the cluster IDs assigned by the clustering step.

algorithm in the paper. The point cloud data from the same cluster have high similarity while the data from different clusters have low similarity.

c) ClusterNet: Semantic Features from Clustering: Once obtained the clusters for the training data by using the Kmeans++ algorithm, the cluster IDs can be used as the pseudo-labels to train another network called ClusterNet, which is also based on the GNN structure in our implementation. The clustering is used to boost the quality of the learned features of the ContrastNet. We would like to note that the architecture of the network for this step does not depend on the previous self-supervised model ContrastNet and therefore it can be flexibly designed as the demands.

Since both of our ContrastNet and ClusterNet use Graph Convolutional Neural Networks (GNNs) as the base models, we will first describe the basis of GNNs, before presenting the ContrastNet and the ClusterNet.

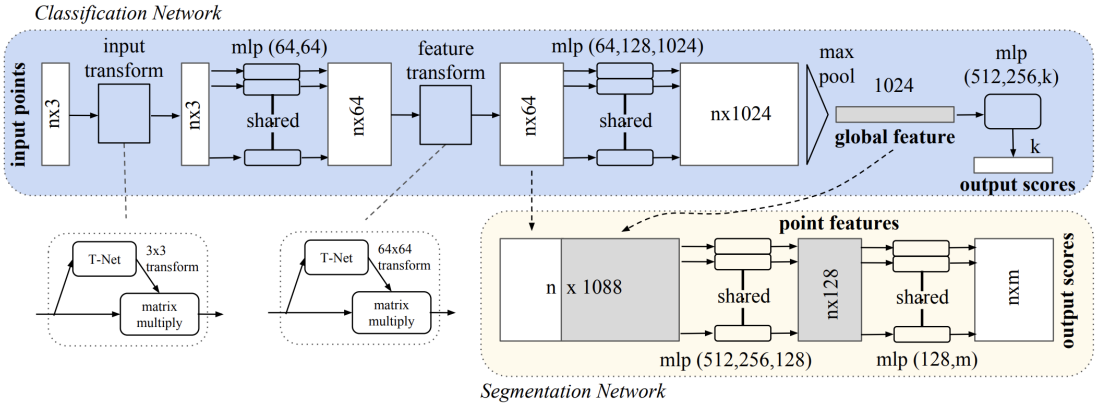

FIGURE 3.2: The architecture of PointNet [1]. For the classification network, the input is a set of n 3D points in the form of (X,Y,Z) which is transformed by T-nets and then predictions are made by several multi-layer perceptrons (mlp). For the segmentation network, the local features and global features are concatenated and the segmentation predictions are made based on the features. In the figure, mlp stands for multi-layer perceptron, and numbers in the parentheses are numbers of kernels for the layers. The figure is originally presented in [1].

3.2 GNN: Graph CNN for Point Cloud

Graph Convolutional Neural Network (GNN) is type of deep neural network to handle unordered data which has been successfully applied for point cloud applications [1–3]. PointNet [1] is the pioneer work for applying a novel deep neural network on point cloud data. The architecture of PointNet is shown in Fig. 3.2. For the classification network, the input is a set of n 3D points which is transformed by 3x3 T-nets and then predictions are made by several fully connected mlp layers. For the segmentation network, the local features and global features are concatenated and the segmentation predictions are made based on the features. With 1×1 convolution kernels to extract local features for each point and then concatenated with the global features extracted by the global pooling, the PointNet obtained state-of-the-art performance on point cloud classification, part segmentation, and semantic segmentation tasks. However, PointNet does not specifically capture the local structure of point cloud data. Later, Qi *et al.* proposed PointNet++ [3] which is an enhanced version of PointNet by involving

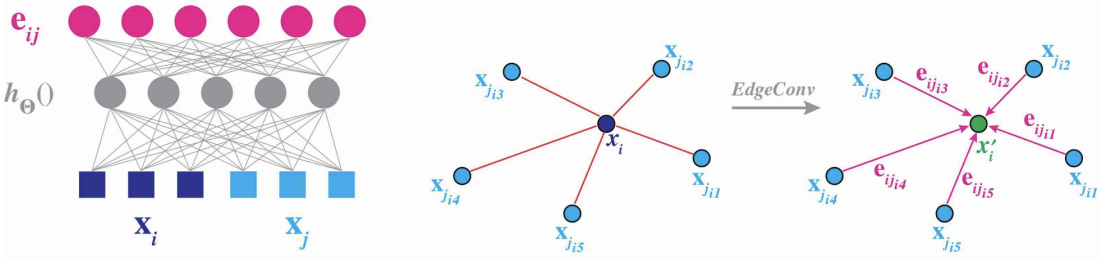

FIGURE 3.3: The figure of EdgeConv proposed in DGCNN [2]. It shows an example of computing an edge feature , e_{ij}, from a point pair, x_i and x_j. The output of EdgeConv is calculated by aggregating the edge features associated with all the edges emanating from each connected vertex. The figure is originally presented in paper DGCNN [2].

local structures. However, the improvement of the point cloud tasks is not significant. To model the local structure of the point cloud data, Wang *et al.* proposed the dynamic graph convolutional neural network (DGCNN) by utilizing a novel convolutional operation called EdgeConv, which specifically model the local structure by using the KNNs over each 3D point data in both the original data space and feature space [11]. With the local structure, the DGCNN achieved significant improvement over PointNet over several point cloud applications.

The illustration of the EdgeConv is shown in Fig. 3.3. Based on the EdgeConv, a deep dynamic graph convolution can be built recursively into a convolutional neural network. The output of one EdgeConv layer can be fed to another EdgeConv layer as the input to extract high-level semantic features. Based on this operation, The DGCNN proposed by Wang *et al.* [2] consists of a number of EdgeConv layers for point cloud tasks. In this paper, the DGCNN is used as the base graph network to extract features in all our experiments. Before getting into details, we would like to note that while DGCNN was designed for supervised classification/segmentation tasks, we use this graph neural network for our two consecutive unsupervised networks: ContrastNet and ClusterNet, to extract features that capture semantic features of object parts and object categories, respectively.

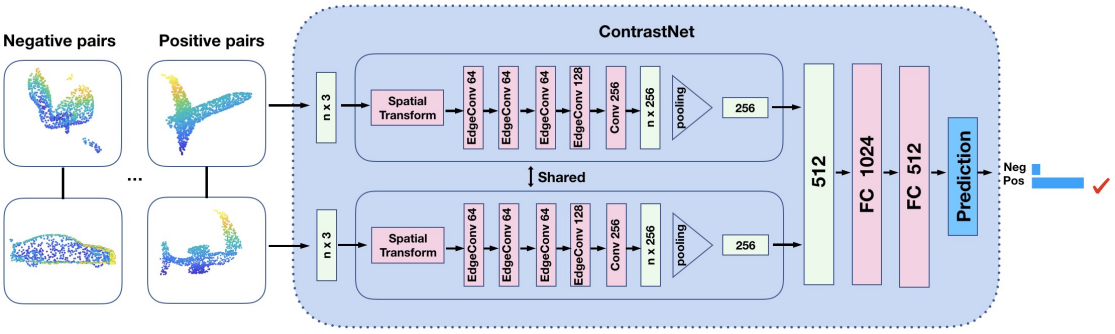

FIGURE 3.4: The architecture of ContrastNet for part contrast learning. The positive pairs are generated by randomly sampling two segments from the same point cloud sample, while the negative pairs are generated by randomly sampling two segments from two different samples. A dynamic graph convolutional neural network (GNN) is used as the backbone network. The features of two segments are concatenated and fed to fully connected layers to make the prediction of positive or negative. The part contrast learning does not require any data annotations by humans.

3.3 ContrastNet: Semantic Features from Parts

When a point cloud data is observed from different views, only a part of the 3D object can be seen. The observable part can be very different based on the view. For example, as shown in Fig. 1.1. For the same airplane, when it is observed from different views, the observed segments can be totally different. However, the different segments still belong to the same object.

Inspired by this observation, we proposed the part (segment) contrast as the pretext task for a GNN to solve. The task is defined as to train a ContrastNet to verify whether two point cloud segments belong to the same object. The positive pair is drawn by selecting two different segments from the same object, while the negative pair is drawn by selecting two segments from two different objects. The illustration of the part contrast task is shown in Fig. 3.4.

We randomly split one object into two segments with almost the same number of 3D points, and then a pair of objects includes 4 segments. If a pair of the segments are segmented from the same object, this pair is a positive instance that will be labeled as 1. Otherwise, if a pair of segments comes from two objects, this pair is a negative instance and will be labeled as 0. For example, if segment A and segment

B come from different parts of object 1, and segment C and segment D come from different parts of object 2, then there will have 6 pairs, AB, CD, AC, AD, BC, and BD. In the 6 pairs, AB and CD are positive pairs since the two segments in the pair are from the same object and the other four pairs are the negative pairs. We model this task as a binary classification problem. As the training goes on, the segments from the same object should have a smaller distance while the segments from different objects have a larger distance. In this way, the semantic features can be learned by this process.

Note that since a pair of objects may belong to the same category, the training of positive and negative has a certain percentage of "noisy data". For example, In ModelNet40 dataset, objects belong to 40 categories. Without using the labels in training ContrastNet, there is a 1/40 (2.5%) error in the input data for verifying positive or negative instances.

As for the network architecture, we choose the DGCNN [2] as the backbone model since this model specifically captures the local structure of the point cloud and yields better performance. The details of the network architecture are shown in Fig. 3.4. There are two branches, one for each point cloud segment, from a pair of input segments. Each branch consists of a spatial transformer network to align the point cloud and followed by 4 EdgeConv layers with $64, 64, 64, 128$ kernel sizes, respectively. After which, one convolutional layer with 256 channels is used to embed the four embeddings obtained by the four EdgeConv layers to high dimensional space. The feature then is pooled into a 256-dimension vector by applying the max-pooling layer. The two feature vectors from the two branches are then concatenated into a vector to be fed to three fully-connected layers (with $1,024, 512, 2$ vector lengths, respectively). The ReLU activation and batch normalization are used for each layer and 50% dropout is used on each fully-connected layer. The cross-entropy loss is optimized by using mini-batch stochastic gradient descent (SGD) and backpropagation to compute the gradient.

3.4 ClusterNet: Semantic Features from Clusters

The underline intuition of clustering is that 3D objects from the same categories have high similarity than those from different categories. After obtaining the clusters of the data by using the Kmeans++, based on the features extracted by ContrastNet. The cluster IDs of the data are used as the "pseudo" labels to train a ClusterNet, so that more meaningful features may be extracted from it. We hope that using cluster IDs as pseudo labels in ClusterNet can provide more powerful self-supervision and therefore, the network can learn more representative features for object classification.

Given any unlabeled point cloud dataset $X = \{x_1, x_2, \ldots, x_N\}$ of N images, The clustering process can be parametrized as [36]:

$$\min_{C \in \mathbb{R}^{d \times k}} \frac{1}{N} \sum_{n=1}^{N} \min_{y_n \in \{0,1\}^k} \|f_\theta(x_n) - C y_n\|_2^2, \qquad y_n^\top 1_k = 1, \tag{3.1}$$

where f_θ is the feature extractor that can map any point cloud data into a vector , θ is the set of corresponding parameters that need to be optimized, y_n is the cluster ID. Solving this clustering problem provides a set of optimal assignments $(y_n^*)_{n \leq N}$ and a centroid matrix C^*. The cluster ID assignments $(y_n^*)_{n \leq N}$ are then used as the pseudo-labels to train the GNN.

The training of ClusterNet, also based on DGCNN [2], with the cluster ID assignments as the pseudo-labels, is described as:

$$\min_{\theta, W} \frac{1}{N} \sum_{n=1}^{N} \ell\left(g_W\left(f_\theta(x_n)\right), y_n\right), \tag{3.2}$$

where the purpose of training is to find the optimal parameters θ^* such that the mapping f_{θ^*} produces good general-purpose features for point cloud data. A parametrized classifier g_W predicts the correct labels of the data based on the features $f_\theta(x_n)$. All the parameters are learned by optimizing this loss function.

In supervised training, parameters θ are optimized with the human-annotated labels while each data x_n is paired with a human-annotated label y_n in $\{0,1\}^k$. In our unsupervised learning training, each data x_n is paired with a pseudo label y_n that is generated by the clustering algorithm. The label y_n indicates the data's membership to one of the k clusters, where k can be specified in the clustering algorithm. In our experiments (below), various number of clusters are tested for comparing the impact on feature extraction. In training the ClusterNet, this loss function is optimized by using mini-batch stochastic gradient descent (SGD) and backpropagation to compute the gradient.

Chapter 4

Experimental Results

We conduct extensive experiments to evaluate the proposed approach and the quality of the learned features for point cloud on the point cloud classification task. After providing a few implementation details of our experiments, a number of datasets that are used in the experiments are described. Three sets of experiments were performed: (1) basic experiments on the effectiveness of ContrastNet and ClusterNet; (2)further studies on the impact of number of clusters on performance, accuracy of clustering, data amount ablation and partial object recognition; and finally (3) comparison with the state of the art approaches.

4.1 Some Implementation Details

ContrastNet: During the part contrast unsupervised learning, each object is cut by randomly generated 15 planes into 30 segments. Each segment has at least 512 points. Any two segments from the same object are treated as the positive samples while any two segments from two different objects are treated as negative samples. The DGCNN is used as the backbone of ContrastNet. During the unsupervised part contrast training phase, the learning rate is 0.001, momentum is 0.9, the optimizer is Adam, the learning rate decay rate is 0.7, and the decay step is 200000.

ClusterNet: The Kmeans++ is used as the clustering algorithm to cluster the data based on the embeddings extracted by the ContrastNet. We test the performance of different cluster numbers to train the ClusterNet. The same DGCNN is used as the backbone in the ClusterNet except that the size of the last dense layer is the cluster number. During the training with pseudo labels, the learning rate is 0.001, momentum is 0.9, the optimizer is Adam, the learning rate decay rate is 0.7, and the decay step is 200000.

All the experiments for both the ContrastNet and ClusterNet are done on three point cloud benchmarks: ModelNet40, ShapeNet, and ModelNet10. The data augmentation including random rotation, shift, and jittering are used during all the training phases.

4.2 Datasets

ModelNet40: The ModelNet40 contains $12,311$ meshed CAD models covering 40 classes. There are $9,843$ and $2,468$ samples in the training and testing splits, respectively. In all our experiments, 1024 points are randomly picked for each model during the training and testing phases. This dataset is used to train and test our unsupervised learning method. During training, the training split of this dataset has been used for learning features without using the class labels. During the testing phase, the testing split of this dataset is used to evaluate the quality of the learned features.

ModelNet10: The dataset contains 10 categories including 3991 meshed CAD models for training and 909 models for testing. We randomly sample 2048 points form the mesh faces and use the (x, y, z) coordinates of sampled points in all experiments. The testing split of this dataset is only used for testing the quality of the learned features.

ShapeNet: The ShapeNet part dataset that contains 16 categories including $12,137$ models for training and $2,874$ for testing from ShapeNet dataset. In all our

experiments, 1024 points are randomly picked for each model during the training and testing phases. The training split of this dataset is used for unsupervised training.

4.3 Basic Experiments

4.3.1 Can ContrastNet Accomplish the Part Contrast Task?

The hypothesis of our idea is that the ContrastNet is able to learn semantic features by accomplishing the "part contrast" pretext task, and then the learned features can be used for other downstream tasks, such as point cloud classification. Therefore, we test the performance of ContrastNet in verifying whether two patches belong to the same object. No human-annotated labels are used during the training phased of ContrastNet.

The performance of ContrastNet in part contrasting is shown in Table. 4.1. The average accuracy of part contrast is more than 90% on the two datasets, with cross-dataset testing. However, for the unsupervised learning methods, we have realized that the network can easily learn trivial solutions such as capturing edges and corners and make predictions based on these specific features. To verify whether the ContrastNet learned useful features, we visualize the testing data by using TSNE.

Training	Testing	Accuracy(%)
ShapeNet	ShapeNet	94.0
ShapeNet	ModelNet40	86.4
ModelNet40	ModelNet40	95.0
ModelNet40	ShapeNet	90.9

TABLE 4.1: Performance of ContrastNet in part contrasting on ShapeNet and ModelNet40 datasets.

The TSNE visualization of the learned features is shown in Fig. 4.1. All of the data covering 10 classes of ModelNet10 is visualized. The figure shows that the

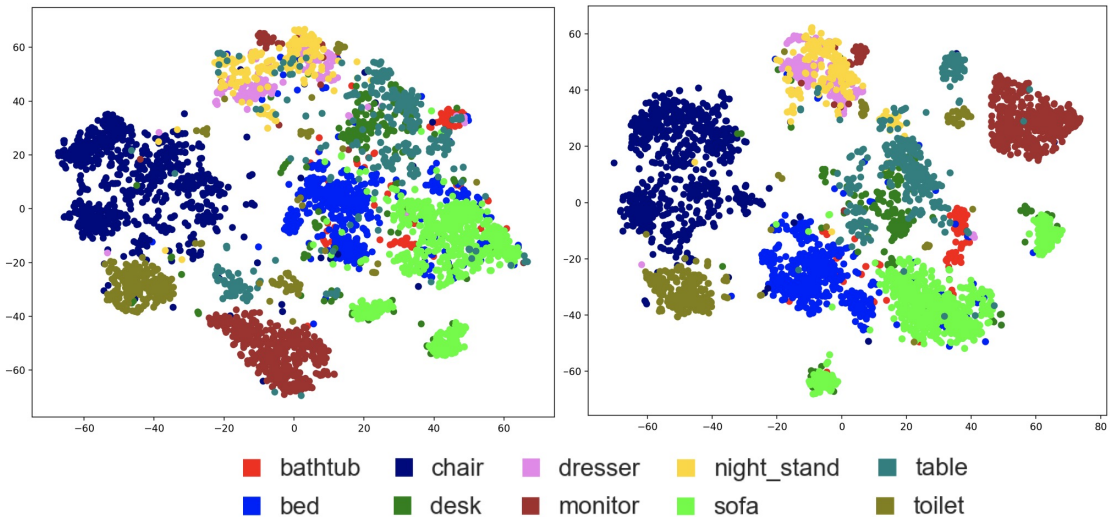

FIGURE 4.1: Visualization of object embedding of the ModelNet10 test data through part contrast training on the ShapeNet dataset. The features are learned by part contrast learning (left) and then boosted by object clustering (right).

ContrastNet indeed learned semantic features and the data from the same class are closer than the data from different classes. Clustering enhances the features more, which will be discussed more in the impact on classification.

4.3.2 Transfer Learned Features to Classification Task

To quantitatively evaluate the quality of the learned features by using the part contrasting pretext task, we conduct experiments on three different datasets: ShapeNet, ModelNet10, and ModelNet40. The features are extracted by the ContrastNet that is only trained with the part contrast task on unlabeled data. A linear classifier SVM is trained based on the features of the training data, and the testing classification accuracy is reported in the column "ContrastNet" in Table 4.2.

Following the practice of previous work [16, 18], we conduct cross-dataset training and testing to verify the generalization ability of features between different datasets. As shown in Table 4.2, when trained only with a linear classifier SVM on one dataset, the ContrastNet trained on ShapeNet is able to achieve 84.1%

and 91.0% on ModelNet40 and ModelNet10 dataset respectively. As a comparison, the model trained on ModelNet40 and tested on the same dataset achieved 85.7%. These results validate the effectiveness of the proposed method and that the learned features by the proposed unsupervised learning method can be transferred among different datasets.

4.3.3 Can Clustering Boost the Performance?

The part contrast learning indeed forces the ContrastNet to learn semantic features. However, the fact that objects belong to the same classes were treated as different objects in part contrasting (without knowing their labels) may have a negative impact on the quality of learned features by ContrastNet. Therefore, clustering is applied to discover the objects with similar appearances and the ClusterNet is trained to learn features by using the cluster IDs as object labels. We hope that the ClusterNet should be able to help the model learn more discriminative features.

The features extracted by the ContrastNet for a dataset are used to group the data into a number of clusters using the Kmeans++ algorithm. A ClusterNet is trained from scratch to predict the cluster ID of each data sample. After the training is finished, the network is tested on the point cloud classification task using the same SVM on the same three benchmarks as above. The classification results are shown in column "ClusterNet" of Table. 4.2, where the number of clusters is selected as 300 (see below further studies for a more detailed discussion).

Training	Testing	ContrastNet(%)	ClusterNet(%)
ShapeNet	ModelNet40	84.1	86.8 (**+2.7**)
ShapeNet	ModelNet10	91.0	93.8 (**+2.8**)
ModelNet40	ModelNet40	85.7	88.6 (**+2.9**)

TABLE 4.2: Comparison of 3D object classification results using ContrastNet and ClusterNet. The classification accuracy of ClusterNet have average 2.8% improvement on all experiments.

As shown in Table 4.2, training the ClusterNet to predict the cluster ID of each data, generated by Kmeans++ based on the features learned by ContrastNet, can significantly boost the point cloud classification accuracy. The clustering boosts the classification accuracy on all the three datasets by at least 2.7%. These improvements validate the effectiveness of using clustering to boost the quality of the learned features, as shown in Fig. 4.1.

4.4 Further Studies

4.4.1 Study on number of clusters

For the Kmeans++ clustering, the numbers of clusters may also affect the quality of the features. Therefore, we also conduct experiments to evaluate the impact of the numbers of clusters on the quality of features. By varying the numbers of clusters, we have examined the point cloud classification accuracy on the three benchmark datasets, as shown in Table 4.3.

Clusters	ShapeNet ModelNet40	ShapeNet ModelNet10	ModelNet40 ModelNet40
16	86.1%	-	-
40	-	-	87.4%
100	86.2%	93.5%	87.7%
200	86.4%	93.6%	88.2%
300	**86.8%**	**93.8%**	**88.6%**
400	86.5%	93.2%	87.8%

TABLE 4.3: The relation of number of clusters and the performance on point cloud classification. The classification performance improved slightly when larger cluster numbers are used.

As shown in Table. 4.3, when the number of clusters for ShapeNet data is 16 and number of clusters for ModelNet40 is 40, which are the exactly the same as the category numbers of these two datasets, the classification accuracy percentages are 86.1 % and 87.6 %, respectively. However, we might not be able to obtain the best performance when the number of clusters equals to the number of categories

of a dataset, and we may not know the number of categories to begin with. Therefore we conduct other experiments by varying the clusters numbers, from 100 to 400, hoping that finer category information will be captured. It turns out that clustering the data into 300 clusters can obtain best results for all three datasets, which are 86.8%, 93.8%, and 88.6% respectively. This means the points cloud classification performance can be improved when larger cluster numbers are used but then is saturated when the numbers are larger than certain values. We speculate that when more clusters are discovered, the fine-grained object group can be discovered which probably leads to more discriminative features. However, we would like to note that the selection of the numbers of clusters is not very critical since the improvement with varying the numbers of clusters is less than 1%.

4.4.2 Study on accuracy of clustering

Pre-training	Clusters	Clustering Acc.	Testing Acc.
ShapeNet	16	83.4%	86.1%
ModelNet40	40	64.2%	87.4%

TABLE 4.4: The accuracy of the cluster results. Clustering using Kmeans++ can effectively cluster the data and the clustering accuracy pretty depends on the number of the categories. The testing accuracy are obtained from ClusterNet using SVM, that has 23.2% improvement on ModelNet40.

To further analysis the quality of learned features, we evaluate the quality of the clusters by calculating the accuracy of each cluster. Specifically, for each cluster, we assign the category label of the majority data as the label and evaluate the accuracy of all the data. We cluster the ShapeNet and ModelNet40 into 16 and 40 clusters respectively, since these numbers equal to the actual numbers of categories in the two datasets. The results are shown in Table 4.4.

As shown in Table 4.4, the cluster accuracy on ShapeNet is 83.4% which means that 83.4% of the data are correctly clustered into the same labels using Kmeans++. The clustering accuracy on ModelNet40 is 64.2%, which is much lower than that of ShapNet, probably because ModelNet has more categories than ShapeNet. However, even with the low clustering accuracy, the testing accuracy obtained after

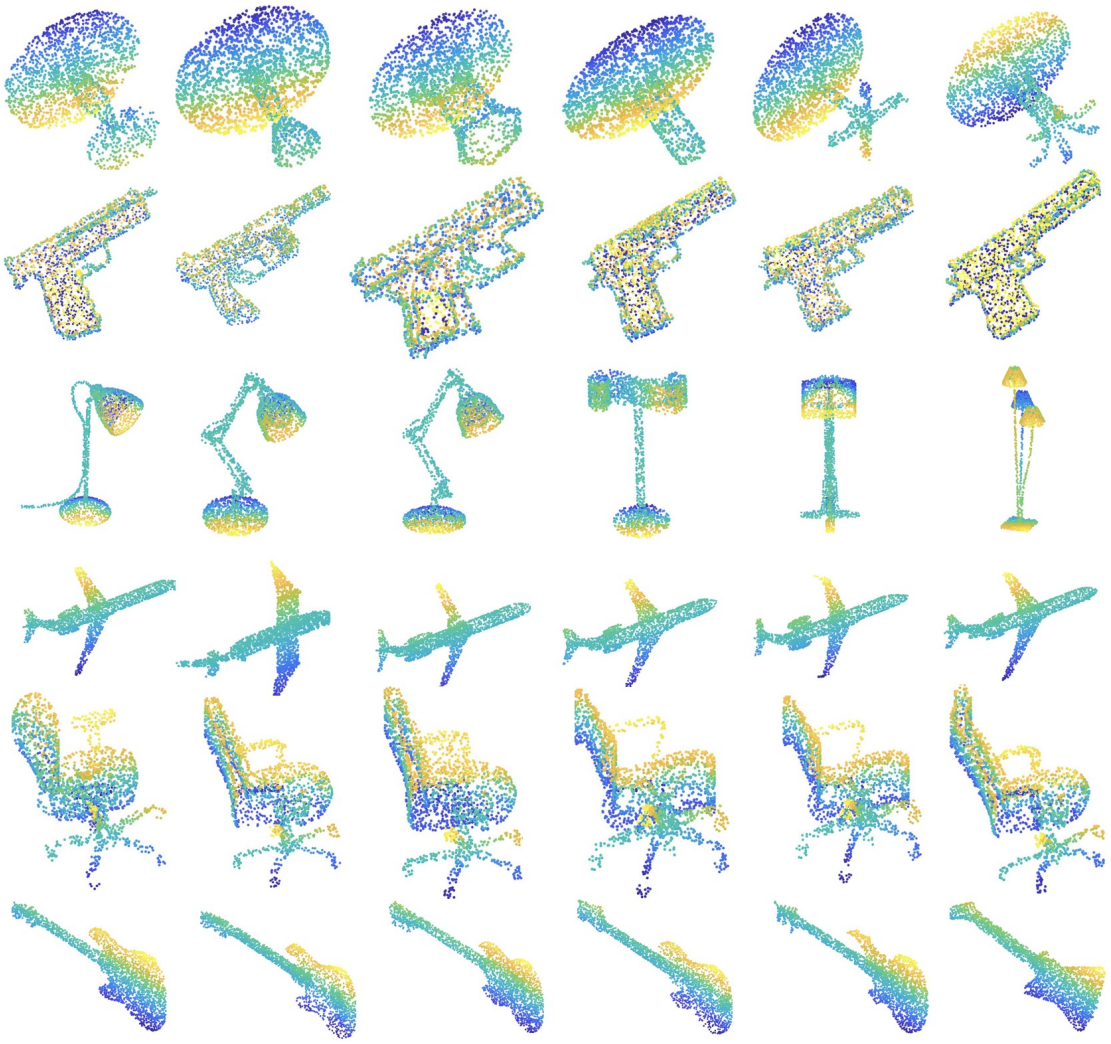

FIGURE 4.2: Visualizing clustering result after applying Kmeans++ on the unlabeled data. The first column is the discovered centrioid of each cluster and the other five columns are the top five closest data to the centroid.

ClusterNet using SVM on ModelNet40 dataset is 87.4%, a 23.2% "improvement" over the training data "accuracy", which means ClusterNet can significantly optimize the quality of the features with a clustering.

The clustering can cluster point cloud objects into groups that the objects from the same groups have smaller distances in the feature space while objects from different groups have larger distances in the feature space. The quality of the cluster indicates the discriminative ability of the learned features. Therefore, we randomly select 6 clusters and show the cluster center and the top 5 objects that closest to the cluster center.

The quality of clustering is visually illustrated in Fig. 4.2. Each row represents the objects from the same cluster discovered by the Kmeans++ based on the unsupervised learning features. As shown in Fig. 4.2, the object from the same cluster have very high similar appearance and geometry.

FIGURE 4.3: Relation of the point cloud object classification accuracy and the training data amount on ClusterNet for classification. The performance of object classification increases along with more annotated training data is used.

4.4.3 Data Amount Ablation Study

We have realized that even with unsupervised feature extraction, we have to know data labels for training a classifier for the classification task. In a real-world scenario, the data annotation is very expensive and sometimes demands special skills. Even a few training examples may take a lot of effort to collect and annotate. Therefore, in this section, we evaluate the impact of the amount of labeled data for SVM training after the ClusterNet feature extraction. We vary the amount of labeled training data for SVM and observe the point cloud classification performance. The performance is shown in Fig. 4.3.

We can see that even if only 5% of the labeled training data are available, the test accuracy is still more than 55%. When 20% of the training data are available, the test classification accuracy is already close to 78%. This demonstrates that when annotated data are extremely limited (a few data per class), the unsupervised learned model can still perform well.

4.4.4 Can ContrastNet Classifiy Part Segments?

Training	Testing	Acc-Part (%)	Acc-Full (%)
ShapeNet	ModelNet40	76.9	84.1
ModelNet40	ModelNet40	80.0	85.7

TABLE 4.5: The accuracy of the part segments classification. The testing accuracy is obtained from the ContrastNet using the SVM. The average accuracy with cross-dataset is more than 80%, which means ContrastNet can effectively learn high-quality features on part segments from a full object. Nevertheless, when all the data are used for the training and testing, the performance is much higher that networks that were trained only using only part segments.

We have known that ContrastNet can learn useful features from the full objects. However, in real-life a sensor can only observe part of the object based on a certain perspective view. Fig. 1.1 shows that different segment parts from one object might be totally different from each other. To verify if ContrastNet can classify part segments, we trained two ContrastNet models using part segments of ModelNet40 and ShapeNet datasets, respectively, and tested the classification of part segments of the objects on ModelNet40. Note that the features were extracted by each ContrastNet model on part segments instead of full objects. Each part segment contains 512 points. A linear SVM is trained based on the features extracted from ContrastNet and we conducted cross-dataset training to evaluate the results. The results are shown in Table 4.5.

The reasults in Table 4.5 shows that the average accuracy (in column "Acc-Part") of part segments classification are more than 80.0% with the cross-dataset, which means even we only feed part of object to ContrastNet model, it still can learning high-quality features for classification task. The results validate the practical

significance of the context contrast task. Nevertheless, we would like to note that when all the data are used for the training and testing (in column "Acc-Full", listed again here from column "ContrastNet" of Table 4.2), the performance is 5% higher that networks that were trained only using only part segments.

4.5 Comparison with the State of the Art

In this section, we compare our approach with both the supervised models [1, 1, 2] and other unsupervised learning models [13–18, 37] on point cloud classification benchmarks ModelNet10 and ModelNet40. In all comparison, we use the ClusterNet.

Following the common practice [16, 18], all the unsupervised models are trained on the ShapeNet data with the same procedure. The methods in [13, 14] are hand-crafted features and methods in [15–18, 37] are deep learning based methods. On the ModelNet10 dataset, our methods outperforms SPH [13], LFD [14], TLNetwork [15], VConv-DAE [37], and 3DGAN [16], and only 0.6% lower than FoldingNet [18] which is the latest work for unsupervised feature learning. On the MoldelNet40 datasets, our method outperforms all the methods except FoldingNet (1.6% lower). We would like to note that our ClusterNet has a much simpler structure and is easier in training.

Models	ModelNet40 (%)	ModelNet10 (%)
SPH [13]	68.2	79.8
LFD [14]	75.5	79.9
T-L Network [15]	74.4	-
VConv-DAE [37]	75.5	80.5
3D-GAN [16]	83.3	91.0
Latent-GAN [17]	85.7	95.3
FoldingNet [18]	88.4	94.4
ClusterNet (Ours)	**86.8**	**93.8**

TABLE 4.6: The comparison on classification accuracy between our ClusterNet and other unsupervised methods on point cloud classificaton dataset ModelNet40 and ModelNet10.

In addition to compare with the unsupervised feature learning methods, we also compare the performance with the recent supervised methods including PointNet [1], PointNet++ [3], and DGCNN [2]. All the parameters of these methods are trained with human-annotated labels, while our results are obtained by training linear SVM based on the features extracted by ContrastNet. The performance comparison is shown in Table. 4.7.

Models	ModelNet40 (%)
POINTNET [1]	89.2
POINTNET++ [3]	90.7
POINTCNN [18]	92.2
DGCNN [2]	92.2
ClusterNet (Ours - unsupervised)	**88.6**

TABLE 4.7: The comparison on classification accuracy between our unsupervised ClusterNet and the supervised methods on point cloud classificaton on ModelNet40.

As shown in Table. 4.7, the supervised methods have better performance because all the parameters are tuned by the hand-annotated labels. With the unsupervised learned features and a linear SVM, the performance of our model (using unsupervised DGCNN) is only 3.6% lower than the supervised DGCNN. These results demonstrate the effectiveness of our unsupervised learning methods.

Chapter 5

Conclusions and Discussions

We have proposed a straightforward and effective method for learning features for point cloud data from unlabeled data. The experiment results demonstrated that proposed pretext tasks (part contrasting and object clustering) are able to provide essential semantic information of the point cloud data for the network to learn semantic features. Using the dynamic graph CNN (DGCNN, designed for supervised classification/segmentation tasks) as the backbone, our two consecutive unsupervised networks - ContrastNet and ClusterNet - can extract features that capture semantic features of object parts and object categories, respectively. Our proposed methods have been evaluated on three public point cloud benchmarks and obtained comparable performance with other state-of-the-art self-supervised learning methods.

There are number of future directions based on the current research.

5.1 Using Larger Datasets

The unsupervised feature learning methods are of great importance to many computer vision tasks since semantic features can be obtained by learning from unlabeled datastets. Past research has shown that the performance of convolutional

neural networks can be boosted up when larger datasets are used [38]. Generally, the performance increase logarithmically with the increase of the amount of training data. Therefore, the quality of the learned features by our framework can be further boosted if more training data are used. In our current work, only the partial ShapeNet dataset with only 16 classes of objects is used for training. We have noticed that there are datasets, such as the full ShapeNet dataset with 55 classes of data objects, and many paper use all of them for unsupervised training [18]. Therefore, the full ShapeNet dataset can be used for training. In addition to ShapeNet, any large-scale point cloud dataset can be used to train our methods.

5.2 Using Perspective Data

Another improvement is to use more practically meaningful data to train the network. The ContrastNet is trained to verify whether two segments are from one object. The network is designed to ignore the low-level features while only focusing the high-level semantic features. The idea is inspired by the facts that even though the same object looks like very different when observed from different views, but they share some kinds of higher level semantic features. The current implementation of cutting an object into two pieces is too artificial. A more reasonable and practical approach would be to use the perspective views of the 3D point cloud data to train the network. The perspective data can be automatically generated by rendering the point cloud points from various different views.

5.3 Automating Data Labeling

There are two networks in our framework: ContrastNet and ClucsterNet. The ContrastNet learned to verify whether two 3D point cloud segments belong to the same object while the ClusterNet learns to cluster the point cloud objects. The number of clusters can be pre-defined. Given smaller cluster numbers, the ClusterNet are trained to automatically cluster data into super clusters, while with larger

cluster numbers, the ClusterNet can discover fine-grain clusters automatically in the data. Therefore, the ClusterNet may be used to automatically discover the clusters in a given dataset. One possible application is to cluster unlabeled data and the other possible application is to automatically label the data, probably with some automatic evaluation and then human cleansing at the end.

Bibliography

[1] Charles R Qi, Hao Su, Kaichun Mo, and Leonidas J Guibas. Pointnet: Deep learning on point sets for 3d classification and segmentation. In *CVPR*, pages 652–660, 2017.

[2] 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.

[3] Charles Ruizhongtai Qi, Li Yi, Hao Su, and Leonidas J Guibas. Pointnet++: Deep hierarchical feature learning on point sets in a metric space. In *NIPS*, pages 5099–5108, 2017.

[4] Angel X Chang, Thomas Funkhouser, Leonidas Guibas, Pat Hanrahan, Qixing Huang, Zimo Li, Silvio Savarese, Manolis Savva, Shuran Song, Hao Su, et al. Shapenet: An information-rich 3d model repository. *arXiv preprint arXiv:1512.03012*, 2015.

[5] Zhirong Wu, Shuran Song, Aditya Khosla, Fisher Yu, Linguang Zhang, Xiaoou Tang, and Jianxiong Xiao. 3d shapenets: A deep representation for volumetric shapes. In *CVPR*, pages 1912–1920, 2015.

[6] Shuran Song, Fisher Yu, Andy Zeng, Angel X Chang, Manolis Savva, and Thomas Funkhouser. Semantic scene completion from a single depth image. In *CVPR*, pages 1746–1754, 2017.

[7] Mehdi Noroozi and Paolo Favaro. Unsupervised learning of visual representations by solving jigsaw puzzles. In *ECCV*, pages 69–84. Springer, 2016.

[8] Mathilde Caron, Piotr Bojanowski, Armand Joulin, and Matthijs Douze. Deep clustering for unsupervised learning of visual features. In *ECCV*, pages 132–149, 2018.

[9] Longlong Jing and Yingli Tian. Self-supervised spatiotemporal feature learning by video geometric transformations. *arXiv preprint arXiv:1811.11387*, 2018.

[10] Deepak Pathak, Philipp Krahenbuhl, Jeff Donahue, Trevor Darrell, and Alexei A Efros. Context encoders: Feature learning by inpainting. In *CVPR*, pages 2536–2544, 2016.

[11] Alec Radford, Luke Metz, and Soumith Chintala. Unsupervised representation learning with deep convolutional generative adversarial networks. *arXiv preprint arXiv:1511.06434*, 2015.

[12] Longlong Jing and Yingli Tian. Self-supervised visual feature learning with deep neural networks: A survey. *arXiv preprint arXiv:1902.06162*, 2019.

[13] Michael Kazhdan, Thomas Funkhouser, and Szymon Rusinkiewicz. Rotation invariant spherical harmonic representation of 3 d shape descriptors. In *Symposium on geometry processing*, volume 6, pages 156–164, 2003.

[14] Ding-Yun Chen, Xiao-Pei Tian, Yu-Te Shen, and Ming Ouhyoung. On visual similarity based 3d model retrieval. In *Computer graphics forum*, volume 22, pages 223–232. Wiley Online Library, 2003.

[15] Rohit Girdhar, David F Fouhey, Mikel Rodriguez, and Abhinav Gupta. Learning a predictable and generative vector representation for objects. In *ECCV*, pages 484–499. Springer, 2016.

[16] Jiajun Wu, Chengkai Zhang, Tianfan Xue, Bill Freeman, and Josh Tenenbaum. Learning a probabilistic latent space of object shapes via 3d generative-adversarial modeling. In *NIPS*, pages 82–90, 2016.

[17] Panos Achlioptas, Olga Diamanti, Ioannis Mitliagkas, and Leonidas Guibas. Representation learning and adversarial generation of 3d point clouds. *arXiv preprint arXiv:1707.02392*, 2(3):4, 2017.

[18] Yaoqing Yang, Chen Feng, Yiru Shen, and Dong Tian. Foldingnet: Point cloud auto-encoder via deep grid deformation. In *CVPR*, pages 206–215, 2018.

[19] Mathieu Aubry, Ulrich Schlickewei, and Daniel Cremers. The wave kernel signature: A quantum mechanical approach to shape analysis. In *ICCVW*, pages 1626–1633. IEEE, 2011.

[20] Jian Sun, Maks Ovsjanikov, and Leonidas Guibas. A concise and provably informative multi-scale signature based on heat diffusion. In *Computer graphics forum*, volume 28, pages 1383–1392. Wiley Online Library, 2009.

[21] Michael M Bronstein and Iasonas Kokkinos. Scale-invariant heat kernel signatures for non-rigid shape recognition. In *2010 IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, pages 1704–1711. IEEE, 2010.

[22] Radu Bogdan Rusu, Nico Blodow, Zoltan Csaba Marton, and Michael Beetz. Aligning point cloud views using persistent feature histograms. In *2008 IEEE/RSJ International Conference on Intelligent Robots and Systems*, pages 3384–3391. IEEE, 2008.

[23] Radu Bogdan Rusu, Nico Blodow, and Michael Beetz. Fast point feature histograms (fpfh) for 3d registration. In *ICRA*, pages 3212–3217. IEEE, 2009.

[24] Haibin Ling and David W Jacobs. Shape classification using the inner-distance. *TPAMI*, 29(2):286–299, 2007.

[25] Andrew E. Johnson and Martial Hebert. Using spin images for efficient object recognition in cluttered 3d scenes. *TPAMI*, 21(5):433–449, 1999.

[26] Daniel Maturana and Sebastian Scherer. Voxnet: A 3d convolutional neural network for real-time object recognition. In *IROS*, pages 922–928. IEEE, 2015.

[27] Maxim Tatarchenko, Alexey Dosovitskiy, and Thomas Brox. Octree generating networks: Efficient convolutional architectures for high-resolution 3d outputs. In *ICCV*, pages 2088–2096, 2017.

[28] Roman Klokov and Victor Lempitsky. Escape from cells: Deep kd-networks for the recognition of 3d point cloud models. In *ICCV*, pages 863–872, 2017.

[29] 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*, pages 5648–5656, 2016.

[30] Hang Su, Subhransu Maji, Evangelos Kalogerakis, and Erik Learned-Miller. Multi-view convolutional neural networks for 3d shape recognition. In *ICCV*, pages 945–953, 2015.

[31] Manolis Savva, Fisher Yu, Hao Su, M Aono, B Chen, D Cohen-Or, W Deng, Hang Su, Song Bai, Xiang Bai, et al. Shrec16 track: largescale 3d shape retrieval from shapenet core55. In *Proceedings of the eurographics workshop on 3D object retrieval*, pages 89–98, 2016.

[32] Yi Fang, Jin Xie, Guoxian Dai, Meng Wang, Fan Zhu, Tiantian Xu, and Edward Wong. 3d deep shape descriptor. In *CVPR*, pages 2319–2328, 2015.

[33] Kan Guo, Dongqing Zou, and Xiaowu Chen. 3d mesh labeling via deep convolutional neural networks. *TOG*, 35(1):3, 2015.

[34] Joan Bruna, Wojciech Zaremba, Arthur Szlam, and Yann LeCun. Spectral networks and locally connected networks on graphs. *arXiv preprint arXiv:1312.6203*, 2013.

[35] Jonathan Masci, Davide Boscaini, Michael Bronstein, and Pierre Vandergheynst. Geodesic convolutional neural networks on riemannian manifolds. In *ICCVW*, pages 37–45, 2015.

[36] John A Hartigan and Manchek A Wong. Algorithm as 136: A k-means clustering algorithm. *Journal of the Royal Statistical Society. Series C (Applied Statistics)*, 28(1):100–108, 1979.

[37] Abhishek Sharma, Oliver Grau, and Mario Fritz. Vconv-dae: Deep volumetric shape learning without object labels. In *European Conference on Computer Vision*, pages 236–250. Springer, 2016.

[38] Deepak Pathak, Ross Girshick, Piotr Dollár, Trevor Darrell, and Bharath Hariharan. Learning features by watching objects move. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 2701–2710, 2017.