Unsupervised Multi-Task Feature Learning on Point Clouds

Kaveh Hassani
Autodesk AI Lab
Toronto, Canada
kaveh.hassani@autodesk.com

Mike Haley
Autodesk AI Lab
San Francisco, USA
mike.haley@autodesk.com

Abstract

We introduce an unsupervised multi-task model to jointly learn point and shape features on point clouds. We define three unsupervised tasks including clustering, reconstruction, and self-supervised classification to train a multi-scale graph-based encoder. We evaluate our model on shape classification and segmentation benchmarks. The results suggest that it outperforms prior state-of-the-art unsupervised models: In the ModelNet40 classification task, it achieves an accuracy of 89.1% and in ShapeNet segmentation task, it achieves an mIoU of 68.2 and accuracy of 88.6%.

1. Introduction

Point clouds are sparse order-invariant sets of interacting points defined in a coordinate space and sampled from surface of objects to capture their spatial-semantic information. They are the output of 3D sensors such as LiDAR scanners and RGB-D cameras, and are used in applications such as human-computer interactions [21], self-driving cars [51], and robotics [60]. Their sparse nature makes them computationally efficient and less sensitive to noise compared to volumetric and multi-view representations.

Classic methods craft salient geometric features on point clouds to capture their local or global statistical properties. Intrinsic features such as wave kernel signature (WKS) [6], heat kernel signature (HKS) [7], multi-scale Gaussian curvature [66], and global point signature [57]; and extrinsic features such as persistent point feature histograms [59] and fast point feature histograms [58] are examples of such features. These features cannot address semantic tasks required by modern applications and hence are replaced by the unparalleled representation capacity of deep models.

Feeding point clouds to deep models, however, is not trivial. Standard deep models operate on regular-structured inputs such as grids (images and volumetric data) and sequences (speech and text) whereas point clouds are permutation-invariant and irregular in nature. One can rasterize the point clouds into voxels [81, 43, 53] but it demands excessive time and memory, and suffers from information loss and quantization artifacts [78].

Some recent deep models can directly consume point clouds and learn to perform various tasks such as classification [78], semantic segmentation [89, 17], part segmentation [78], image-point cloud translation [19], object detection and region proposal [97], consolidation and surface reconstruction [92, 45, 47], registration [16, 74, 34], generation [68, 67], and up-sampling [93]. These models achieve promising results thanks to their feature learning capabilities. However, to successfully learn such features, they require large amounts of labeled data.

A few works explore unsupervised feature learning on point sets using autoencoders [88, 13, 39, 96, 2, 16] and generative models, e.g., generative adversarial networks (GAN) [68, 67, 2], variational autoencoders (VAE) [20], and Gaussian mixture models (GMM) [2]. Despite their good feature learning capabilities, they suffer from not having access to supervisory signals and targeting a single task. These shortcomings can be addressed by self-supervised learning and multi-task learning, respectively. Self-supervised learning defines a pretext task using only the information present in the data to provide a surrogate supervisory signal whereas multi-task learning uses the commonalities across tasks by jointly learning them [95].

We introduce a multi-task model that exploits three regimes of unsupervised learning including self-supervision, autoencoding, and clustering as its target tasks to jointly learn point and shape features. Inspired by [9, 22], we show that leveraging joint clustering and self-supervised classification along with enforcing reconstruction achieves promising results while avoiding trivial solutions. The key contributions of our work are as follows:

- We introduce a multi-scale graph-based encoder for point clouds and train it within an unsupervised multi-task learning setting.
- We exhaustively evaluate our model under various learning settings on ModelNet40 shape classification and ShapeNetPart segmentation tasks.

- We show that our model achieves state-of-the-art results w.r.t prior unsupervised models and narrows the gap between unsupervised and supervised models.

2. Related Work

2.1. Deep Learning on Point Clouds

PointNet [52] is an MLP that learns point features independently and aggregates them into a shape feature. PointNet++ [54] defines multi-scale regions and uses PointNet to learn their features and then hierarchically aggregates them. Models based on KD-trees [37, 94, 20] spatially partition the points using kd-trees and then recursively aggregate them. RNNs [31, 89, 17, 41] are applied to point clouds by the assumption that *"order matters"* [72] and achieve promising results on semantic segmentation tasks but the quality of the learned features is not clear.

CNN models introduce non-Euclidean convolutions to operate on point sets. A few models such as RGCNN [70], SyncSpecCNN [91] and Local Spectral GCNN [75] operate on *spectral* domain. These models tend to be computationally expensive. *Spatial* CNNs learn point features by aggregating the contributions of neighbor points. Pointwise convolution [30], Edge convolution [78] , Spider convolution [84], sparse convolution [65, 25], Monte Carlo convolution [27], parametric continuous convolution [76], feature-steered graph convolution [71], point-set convolution [63], χ-convolution [40], and spherical convolution [38] are examples of these models. Spatial models provide strong localized filters but struggle to learn global structures [70].

A few works train generative models on point sets. Multiresolution VAE [20] introduces a VAE with multiresolution convolution and deconvolution layers. PointGrow [68] is an auto-regressive model that can generate point clouds from scratch or conditioned on given semantic contexts. It is shown that GMMs trained on PointNet features achieve better performance compared to GANs [2].

A few recent works explore representation learning using autoencoders. A simple autoencoder based on PointNet is shown to achieve good results on various tasks [2]. FoldingNet [88] uses an encoder with graph pooling and MLP layers and introduces a decoder of folding operations that deform a 2D grid onto the underlying object surface. PPF-FoldNet [13] projects the points into point pair feature (PPF) space and then applies a PointNet encoder and a FoldingNet decoder to reconstruct that space. AtlasNet[26] extends the FoldingNet to multiple grid patches whereas SONet [39] aggregates the point features into SOM node features to encode the spatial distributions. PointCapsNet [96] introduces an autoencoder based on dynamic routing to extract latent capsules and a few MLPs that generate multiple point patches from the latent capsules with distinct grids.

2.2. Self-Supervised Learning

Self-supervised learning defines a proxy task on unlabeled data and uses the pseudo-labels of that task to provide the model with supervisory signals. It is used in machine vision with proxy tasks such as predicting arrow of time [79], missing pixels [50], position of patches [14], image rotations [23], synthetic artifacts [33], image clusters [9], camera transformation in consecutive frames [3], rearranging shuffled patches [48], video colourization [73], and tracking of image patches[77] and has demonstrated promising results in learning and transferring visual features.

The main challenge in self-supervised learning is to define tasks that relate most to the down-stream tasks that use the learned features [33]. Unsupervised learning, e.g., density estimation and clustering, on the other hand, is not domain specific [9]. *Deep clustering* [4, 44, 86, 28, 83, 22, 61, 87, 29] models are recently proposed to learn cluster-friendly features by jointly optimizing a clustering loss with a network-specific loss. A few recent works combine these two approaches and define deep clustering as a surrogate task for self-supervised learning. It is shown that alternating between clustering the latent representation and predicting the cluster assignments achieves state-of-the-art results in visual feature learning[9, 22].

2.3. Multi-Task Learning

Multi-task learning leverages the commonalities across relevant tasks to enhance the performance over those tasks [95, 18]. It learns a shared feature with adequate expressive power to capture the useful information across the tasks. Multi-task learning has been successfully used in machine vision applications such as image classification [42], image segmentation [12], video captioning [49], and activity recognition [85]. A few works explore self-supervised multi-task learning to learn high level visual features [15, 55]. Our approach is relevant to these models except we use self-supervised tasks in addition to other unsupervised tasks such as clustering and autoencoding.

3. Methodology

Assume a training set $\mathcal{S} = [s_1, s_2, ..., s_N]$ of N point sets where a point set $s_i = \{p_1^i, p_2^i, ..., p_M^i\}$ is an order-invariant set of M points and each point $p_j^i \in \mathbb{R}^{d_{in}}$. In the simplest case $p_j^i = (x_j^i, y_j^i, z_j^i)$ only contains coordinates, but can extend to carry other features, e.g., normals. We define an **encoder** $E_\theta : \mathcal{S} \longmapsto \mathcal{Z}$ that maps input point sets from $\mathbb{R}^{M \times d_{in}}$ into the latent space $\mathcal{Z} \in \mathbb{R}^{d_z}$ such that $d_z \gg d_{in}$. For each point p_j^i, the encoder first learns a point (local) feature $z_j^i \in \mathbb{R}^{d_z}$ and then aggregates them into a shape (global) feature $Z^i \in \mathbb{R}^{d_z}$. It basically projects the input points to a feature subspace with higher dimension to encode richer local information than the original space.

Any parametric non-linear function parametrized by θ can be used as the encoder. To learn θ in unsupervised multi-task fashion, we define three parametric functions on the latent variable Z as follows:

Clustering function $\Gamma_c : \mathcal{Z} \longmapsto \mathcal{Y}$ maps the latent variable into K categories $\mathcal{Y} = [y_1, y_2, ..., y_n]$ such that $y_i \in \{0,1\}^K$ and $y_n^T \mathbf{1}_k = 1$. This function encourages the encoder to generate features that are clustering-friendly by pushing similar samples in the feature space closer and pushing dissimilar ones away. It also provides the model with pseudo-labels for self-supervised learning through its hard cluster assignments.

Classifier function $f_\psi : \mathcal{Z} \longmapsto \hat{\mathcal{Y}}$ predicts the cluster assignments of the latent variable such that the predictions correspond to the hard clusters assignments of Γ_c. In other words, f_ψ maps the latent variable into K predicted categories $\hat{\mathcal{Y}} = [\hat{y}_1, \hat{y}_2, ..., \hat{y}_n]$ such that $\hat{y}_i \in \{0,1\}^K$. This function uses the pseudo-labels generated by the clustering function, i.e., cluster assignments, as its proxy train data. The difference between the cluster assignments and the predicted cluster assignments provides the supervisory signals.

Decoder function $g_\phi : \mathcal{Z} \longmapsto \hat{\mathcal{S}}$ reconstructs the original point set from the latent variable, i.e., maps the latent variable $\mathcal{Z} \in \mathbb{R}^{d_z}$ to a point set $\hat{\mathcal{S}} \in \mathbb{R}^{M \times d_{in}}$. Training a deep model with a clustering loss collapses the features into a single cluster [9]. Some heuristics such as penalizing the minimal number of points per cluster and randomly re-assigning empty clusters are introduced to prevent this. We introduce the decoder function to prevent the model from converging to trivial solutions.

3.1. Training

The model alternates between clustering the latent variable $\mathcal{Z}$ to generate pseudo-labels $\mathcal{Y}$ for self-supervised learning, and learning the model parameters by jointly predicting the pseudo-labels $\hat{\mathcal{Y}}$ and reconstructing the input point set $\hat{\mathcal{S}}$. Assuming *K-means* clustering, the model learns a centroid matrix $C \in \mathbb{R}^{d_z \times K}$ and cluster assignments y_n by optimizing the following objective clustering [9]:

$$\min_{\{C,\theta\}} \frac{1}{N} \sum_{n=1}^{N} \min_{y_n \in \{0,1\}^K} \|z_n - C y_n\|_2^2 \tag{1}$$

where $z_n = E_\theta(s_n)$ and $y_n^T \mathbf{1}_k = 1$. The centroid matrix is initialized randomly. It is noteworthy that: (i) when assigning cluster labels, the centroid matrix is fixed, and (ii) the centroid matrix is updated epoch-wise and not batch-wise to prevent the learning process from diverging.

For the classification function, we minimize the cross-entropy loss between the cluster assignments and the predicted cluster assignments as follows.

$$\min_{\{\theta,\psi\}} \frac{-1}{N} \sum_{n=1}^{N} y_n \log \hat{y}_n \tag{2}$$

where $y_n = \Gamma_c(z_n)$ and $\hat{y}_n = f_\psi(z_n)$ are the cluster assignments and the predicted cluster assignments, respectively.

We use Chamfer distance to measure the difference between the original point cloud and its reconstruction. Chamfer distance is differentiable with respect to points and is computationally efficient. It is computed by finding the nearest neighbor of each point of the original space in the reconstructed space and vice versa, and summing up their Euclidean distances. Hence, we optimize the decoding loss as follows.

$$\min_{\{\theta,\phi\}} \frac{1}{2NM} \sum_{n=1}^{N} \sum_{m=1}^{M} \min_{\hat{p} \in \hat{s}_n} \|p_m^n - \hat{p}\|_2^2 + \min_{p \in s_n} \|\hat{p}_m^n - p\|_2^2 \tag{3}$$

where $\hat{s}_n = g_\phi(z_n)$ and, s_n and $\hat{s}_n$ are the original and reconstructed point sets, respectively. N and M denote the number of point sets in the train set and the number of points in each point set, respectively.

Let's denote the clustering, classification, and decoding objectives by $\mathcal{L}_\Gamma$, $\mathcal{L}_f$, and $\mathcal{L}_g$, respectively. we define the multi-task objective as a linear combination of these objectives: $\mathcal{L} = \alpha\mathcal{L}_\Gamma + \beta\mathcal{L}_f + \gamma\mathcal{L}_g$ and train the model based on that. The training process is shown in Algorithm 1.

We first randomly initialize the model parameters and assume an arbitrary upper bound for the number of clusters. We show through experiments that the model converges to a fixed number of clusters by emptying some of the clusters. This is especially favorable when the true number of categories is unknown. We then randomly select K point sets from the training data and feed them to the randomly initialized encoder and set the extracted features as the initial centroids. Afterwards we optimize the model parameters w.r.t the multi-task objective using mini-batch stochastic gradient descent. Updating the centroids with the same frequency as the network parameters can destabilize the training. Therefore, we aggregate the learned features and the cluster assignments within each epoch and update the centroids after an epoch is completed.

3.2. Architecture

Inspired by *Inception* [69] and Dynamic Graph CNN (DGCNN) [78] architectures, we introduce a graph-based architecture shown in Figure 1 which consists of an encoder and three task-specific decoders. The encoder uses a series of graph convolution, convolution, and pooling layers in a multi-scale fashion to learn point and shape features from an input point cloud jittered by Gaussian noise. For each point, it extracts three intermediate features by applying graph convolutions on three neighborhood radii and concatenates them with the input point feature and its convolved feature. The first three features encode the interactions between each point and its neighbors where as the last two features encode the information about each point. The

Algorithm 1: Unsupervised Multi-task training algorithm.

$\theta, \phi, \psi \longleftarrow Random()$ Initial parameters
$K \longleftarrow K_{UB}$ Upper bound #clusters
$C \longleftarrow E_\theta\left(Choice\left(\mathcal{S}, K\right)\right)$ Initial centroids
for *epoch in epochs* **do**
  while *epoch not completed* **do**
    Forward pass
    $\mathcal{S}_x \longleftarrow Sample(\mathcal{S})$ Mini-batch
    $\mathcal{Z}_x \longleftarrow E_\theta\left(\mathcal{S}_x\right)$ Encoding
    $\mathcal{Y}_x \longleftarrow \Gamma_c\left(\mathcal{Z}_x\right)$ Cluster assignment
    $\hat{\mathcal{Y}}_x \longleftarrow f_\psi\left(\mathcal{Z}_x\right)$ Cluster prediction
    $\hat{\mathcal{S}}_x \longleftarrow g_\phi\left(\mathcal{Z}_x\right)$ Decoding
    $(\mathcal{Z}, \mathcal{Y}) \longleftarrow Aggregate\left(\mathcal{Z}_x, \mathcal{Y}\right)$
    Backwards pass
    $\bigtriangledown_{\theta,\phi,\psi}(\alpha\mathcal{L}_\Gamma(\mathcal{Z}_x, C; \theta)+$ Compute gradients
    $\beta\mathcal{L}_f(\mathcal{Y}_x, \hat{\mathcal{Y}}_x; \theta, \psi)+$
    $\gamma\mathcal{L}_g(\mathcal{S}_x, \hat{\mathcal{S}}_x; \theta, \phi))$
    $Update(\theta, \phi, \psi)$ Update with gradients
  end
  $C \longleftarrow Update(\mathcal{Z}, \mathcal{Y})$ Update centroids
end

concatenation of the intermediate features is then passed through a few convolution and pooling layers to learn another level of intermediate features. These point-wise features are then pooled and fed to an MLP to learn the final shape feature. They are also concatenated with the shape feature to represent the final point features. Similar to [78], we define the graph convolution as follows:

$$z_i = \sum_{p_k \in \mathcal{N}(p_i)} h_\theta\left(\left[p_i \parallel p_k - p_i\right]\right) \tag{4}$$

where z_i is the learned feature for point p_i based on its neighbor contributions, $p_k \in \mathcal{N}(p_i)$ are the k nearest points to the p_i in Euclidean space, h_θ is a nonlinear function parameterized by θ and $\parallel$ is the concatenation operator. We use a shared MLP for h_θ. The reason to use both p_i and $p_k - p_i$ is to encode both global information (p_i) and local interactions ($p_k - p_i$) of each point.

To perform the target tasks, i.e., clustering, classification, and autoencoding, we use the following. For clustering, we use a standard implementation of K-means to cluster the shape features. For self-supervised classification, we feed the shape features to an MLP to predict the category of the shape (i.e., cluster assignment by the clustering module). And for the autoencoding task, we use an MLP to reconstruct the original point cloud from the shape feature. This MLP is denoising and reconstructs the original point cloud before the addition of the Gaussian noise. All these models along with the encoder are trained jointly and end-to-end. Note that all these tasks are defined on the shape features. Because a shape feature is an aggregation of its corresponding point features, learning a good shape feature pushes the model to learn good point features too.

4. Experiments

4.1. Implementation Details

We optimize the network using Adam [36] with an initial learning rate of 0.003 and batch size of 40. The learning rate is scheduled to decrease by 0.8 every 50 epochs. We apply batch-normalization [32] and ReLU activation to each layer and use dropout [64] with $p = 0.5$. To normalize the task weights to the same scale, we set the weights of clustering (α), classification (β), and reconstruction(γ) to 0.005, 1.0, 500, respectively. For graph convolutions, we use neighborhood radii of 15, 20, and 25 (as suggested in [78]) and for normal convolutions we use 1×1 kernels. We set the upper bound number of clusters (K_{UB}) to 500. We also set the size of the MLPs in prediction and reconstruction tasks to [2048, 1024, 500] and [2048, 1024, 6144], respectively. Note that the size of the last layers correspond to the upper bound number of clusters (500) and the reconstruction size (6144: 2048×3). Following [2] we set the shape and point feature sizes to 512 and 1024, respectively.

For preprocessing and augmentation we follow [52, 78] and uniformly sample 2048 points and normalize them to a unit sphere. We also apply point-wise Gaussian noise of $N \sim (0, 0.01)$ and shape-wise random rotations between [-180, 180] degrees along z-axis and random rotations between [-20, +20] degrees along x and y axes.

The model is implemented with Tensorflow [1] on a Nvidia DGX-1 server with 8 Volta V100 GPUs. We used synchronous parallel training by distributing the training mini-batches over all GPUs and averaging the gradients to update the model parameters. With this setting, our model takes 830s on average to train one epoch on the ShapeNet (i.e, ∼55k samples of size 2048×3). We train the model for 500 epochs. At test time, it takes 8ms on an input point cloud with size 2048×3.

4.2. Pre-training for Transfer Learning

Following the experimental protocol introduced in [2], we pre-train the model across all categories of the ShapeNet dataset [10] (i.e., 57,000 models across 55 categories) , and then transfer the trained model to two down-stream tasks including shape classification and part segmentation. After pre-training the model, we freeze its weights and do not fine-tune it for the down-stream tasks.

Following [9], we use Normalized Mutual Information (NMI) to measure the correlation between cluster assignments and the categories without leaking the category in-

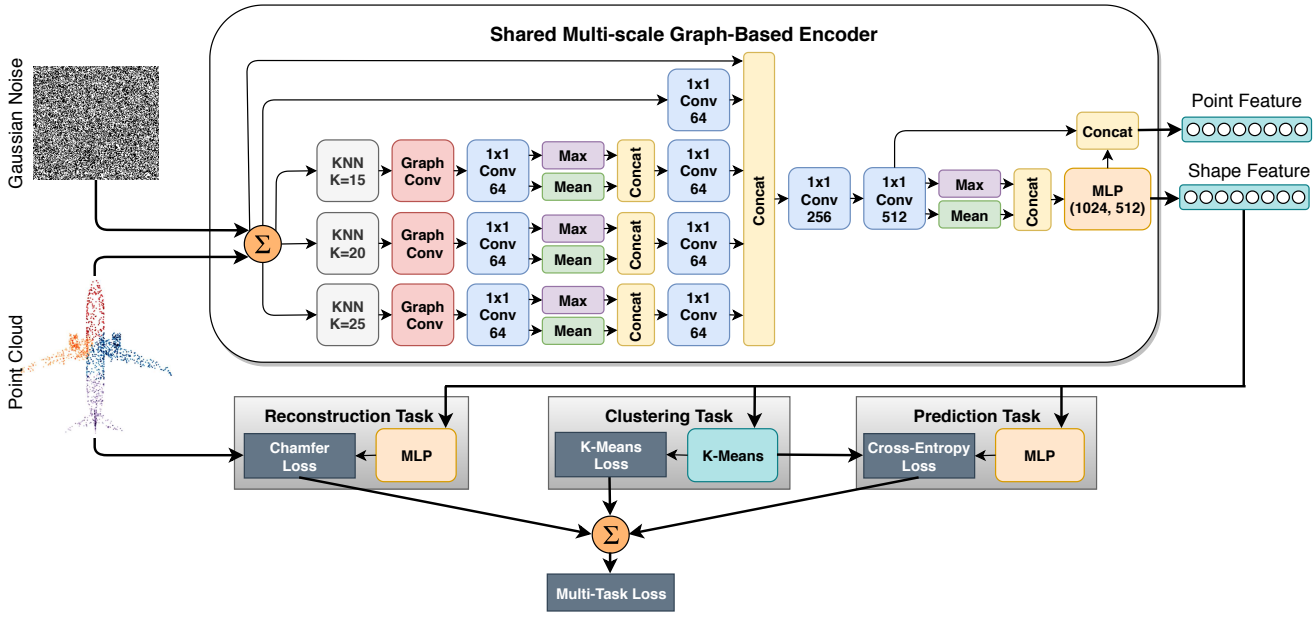

Figure 1. Proposed Architecture for unsupervised multi-task feature learning on point clouds. It consists of a multi-scale graph-based encoder that generates point and shape features for an input point cloud and three task decoders that jointly provide the architecture with a multi-task loss.

formation to the model. This measure gives insight on the capability of the model in predicting category level information without observing the ground-truth labels. The model reaches an NMI of 0.68 and 0.62 on the train and validation sets, respectively which suggests that the learned features are progressively encoding category-wise information.

We also observe that the model converges to 88 clusters (from the initial 500 clusters) which is 33 more clusters compared to the number of ShapeNet categories. This is consistent with the observation that "*some amount of over-segmentation is beneficial*" [9]. The model empties more than 80% of the clusters but does not converge to the trivial solution of one cluster. We also trained our model on the 10 largest ShapeNet categories to investigate the clustering behavior where the model converged to 17 clusters. This confirms that model converges to a fixed number of clusters which is less than the initial upper bound assumption and is more than the actual number of categories in the data.

To investigate the dynamics of the learned features, we selected the 10 largest ShapeNet categories and randomly sampled 200 shapes from each category. The evolution of the features of the sampled shapes visualized using t-SNE (Figure 2) suggests that the learned features progressively demonstrate *clustering-friendly* behavior along the training epochs.

4.3. Shape Classification

To evaluate the performance of the model on shape feature learning, we follow the experimental protocol in [2] and report the classification accuracy on transfer learning from the ShapeNet dataset [10] to the ModelNet40 dataset [82] (i.e., 13,834 models across 40 categories divided to 9,843 and 3,991 train and test samples, respectively). Similar to [2], we extract the shape features of the ModelNet40 samples from the pre-trained model without any fine-tuning, train a linear SVM on them, and report the classification accuracy. This approach is a common practice in evaluating unsupervised visual feature learning [9] and provides insight about the effectiveness of the learned features in classification tasks.

Results shown in Table 1 suggest that our model achieves state-of-the-art accuracy on the ModelNet40 shape classification task compared to other unsupervised feature learning models. It is noteworthy that the reported result is without any hyper-parameter tuning. With random hyper-parameter search, we observed an 0.4 absolute increase in the accuracy (i.e., **89.5%**). The results also suggest that the unsupervised model is competitive with the supervised models. Error analysis reveals that the misclassifications occur between geometrically similar shapes. For example, the three most frequent misclassifications are between (table, desk), (nightstand, dresser), and (flowerpot, plant) categories. A similar observation is reported in [2] and it is suggested that stronger supervision signals may be required to learn subtle details that discriminate these categories.

To further investigate the quality of the learned shape features, we evaluated them in a zero-shot setting. For this purpose, we cluster the learned features using agglomera-

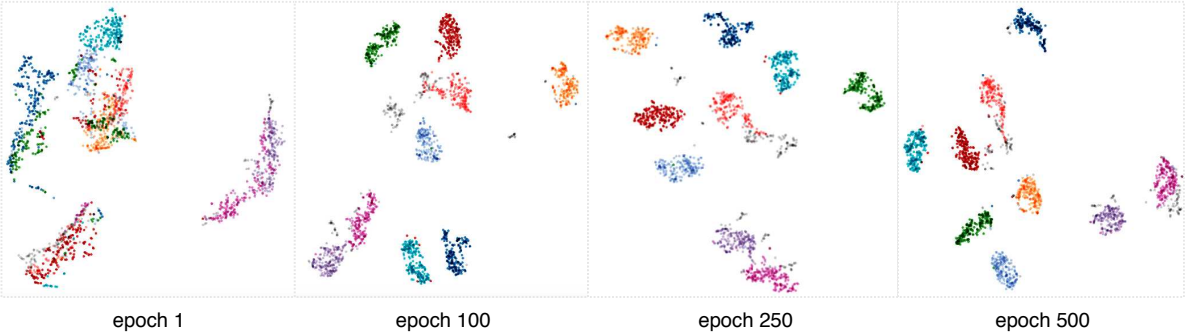

Figure 2. Evolution of the learned features along the training epochs (visualized using t-SNE) showing progressive *clustering-friendly* behavior.

Unsupervised transfer learning		**Supervised learning**	
Model	Accuracy	Model	Accuracy
SPH [35]	68.2	PointNet [52]	89.2
LFD [11]	75.5	PointNet++ [54]	90.7
T-L Network [24]	74.4	PointCNN [30]	86.1
VConv-DAE [62]	75.5	DGCNN [78]	**92.2**
3D-GAN [80]	83.3	KCNet [63]	91.0
Latent-GAN [2]	85.7	KDNet [37]	91.8
MRTNet-VAE [20]	86.4	MRTNet [20]	91.7
FoldingNet [88]	88.4	SpecGCN [75]	91.5
PointCapsNet [96]	88.9		
Ours	**89.1**		

Table 1. **Left**: Accuracy of classification by transfer learning from the ShapeNet on the ModelNet40 data. **Right**: Classification accuracy of supervised learning on the ModelNet40 data. Our model narrows the gap with supervised models.

tive hierarchical clustering (AHC) [46] and then align the assigned cluster labels with the ground truth labels (ModelNet40 categories) based on majority voting within each cluster. The results suggest that the model achieves **68.88%** accuracy on the shape classification task with zero supervision. This result is consistent with the observed NMI between cluster assignments and ground truth labels in the ShapeNet dataset.

4.4. Part Segmentation

Part segmentation is a fine-grained point-wise classification task where the goal is to predict the part category label of each point in a given shape. We evaluate the learned point features on the ShapeNetPart dataset [90], which contains 16,881 objects from 16 categories (12149 train, 2874 test, and 1858 validation). Each object consists of 2 to 6 parts with total of 50 distinct parts among all categories. Following [52], we use mean Intersection-over-Union (mIoU) as the evaluation metric computed by averaging the IoUs of different parts occurring in a shape. We also report part classification accuracy.

Model	1% of train data		5% of train data	
	Accuracy	**IoU**	**Accuracy**	**IoU**
SO-Net[39]	78.0	64.0	84.0	69.0
PointCapsNet[96]	85.0	67.0	86.0	70.0
Ours	**88.6**	**68.2**	**93.7**	**77.7**

Table 2. Results on semi-supervised ShapeNetPart segmentation task.

Following [96], we randomly sample 1% and 5% of the ShapeNetPart train set to evaluate the point features in a semi-supervised setting. We use the same pre-trained model to extract the point features of the sampled training data, along with validation and test samples without any fine-tuning. We then train a 4-layer MLP [2048, 4096, 1024, 50] on the sampled training sets and evaluate it on all test data. Results shown in Table 2 suggest that our model achieves state-of-the-art accuracy and mIoU on ShapeNetPart segmentation task compared to other unsupervised feature learning models. Also comparisons between our model (trained on 5% of the training data) and the fully supervised models are shown in Table 3. The results suggest that our model achieves an mIoU which is only 8% less than the best supervised model and hence narrows the gap with supervised models.

We also performed intrinsic evaluations to investigate the consistency of the learned point features within each category. We sampled a few shapes from each category, stacked their point features, and reduced the feature dimension from 1024 to 512 using PCA. We then co-clustered the features using the AHC method. The result of co-clustering on the *airplane* category is shown in Figure 3. We observed a similar consistent behavior over all categories. We also used AHC and hierarchical density-based spatial clustering (HDBSCAN) [8] methods to cluster the point features of each shape. We aligned the assigned cluster labels with the ground truth labels based on majority voting within each cluster. A few sample shapes along with their ground truth

Model	%train data	Cat. mIoU	Ins. mIoU	Aero	Bag	Cap	Car	Chair	Ear phone	Guitar	Knife	Lamp	Laptop	Motor	Mug	Pistol	Rocket	Skate board	Table
PointNet [52]		80.4	83.7	83.4	78.7	82.5	74.9	89.6	73.0	91.5	85.9	80.8	95.3	65.2	93.0	81.2	57.9	72.8	80.6
PointNet++ [54]		81.9	85.1	82.4	79.0	87.7	77.3	90.8	71.8	91.0	85.9	83.7	95.3	71.6	94.1	81.3	58.7	76.4	82.6
DGCNN [78]		82.3	85.1	**84.2**	83.7	84.4	77.1	**90.9**	**78.5**	91.5	87.3	82.9	96.0	67.8	93.3	82.6	59.7	75.5	82.0
KCNet [63]		82.2	84.7	82.8	81.5	86.4	77.6	90.3	76.8	91.0	87.2	84.5	95.5	69.2	94.4	81.6	60.1	75.2	81.3
RSNet [31]		81.4	84.9	82.7	**86.4**	84.1	78.2	90.4	69.3	91.4	87.0	83.5	95.4	66.0	92.6	81.8	56.1	75.8	82.2
SynSpecCNN [91]	**100%**	82.0	84.7	81.6	81.7	81.9	75.2	90.2	74.9	93.0	86.1	**84.7**	95.6	66.7	92.7	81.6	60.6	82.9	82.1
RGCNN [70]		79.5	84.3	80.2	82.8	**92.6**	75.3	89.2	73.7	91.3	**88.4**	83.3	96.0	63.9	95.7	60.9	44.6	72.9	80.4
SpiderCNN [84]		82.4	85.3	83.5	81.0	87.2	77.5	90.7	76.8	91.1	87.3	83.3	95.8	70.2	93.5	82.7	59.7	75.8	**82.8**
SPLATNet [65]		83.7	**85.4**	83.2	84.3	89.1	80.3	90.7	75.5	92.1	87.1	83.9	96.3	75.6	**95.8**	83.8	64.0	75.5	81.8
FCPN [56]		**84.0**	84.0	84.0	82.8	86.4	**88.3**	83.3	73.6	**93.4**	87.4	77.4	**97.7**	**81.4**	**95.8**	**87.7**	**68.4**	**83.6**	73.4
Ours	**5%**	72.1	77.7	78.4	67.7	78.2	66.2	85.5	52.6	87.7	81.6	76.3	93.7	56.1	80.1	70.9	44.7	60.7	73.0

Table 3. Comparison between our semi-supervised model and supervised models on ShapeNetPart segmentation task. Average mIoU over instances (Ins.) and categories (Cat.) are reported.

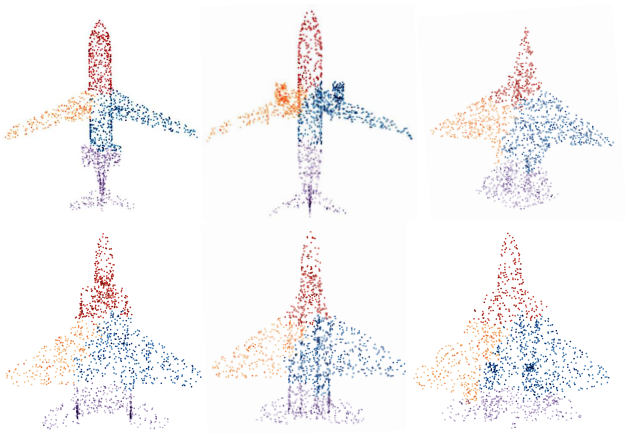

Figure 3. Co-clustering of the learned point features within the Airplane category using hierarchical clustering which demonstrates the consistency of the learned point features within the category.

part labels, predicted part labels by the trained MLP, AHC, and HDBSCAN clustering are illustrated in Figure 4. As shown, HDBSCAN clustering results in a decent segmentation of the learned features in a fully unsupervised setting.

4.5. Ablation Study

We first investigate the effectiveness of the graph-based encoder on the shape classification task. In the first experiment, we replace the encoder with a PointNet [52] encoder and keep the multi-task decoders. We train and test the network with the same transfer learning protocol which results in a classification accuracy of 86.2%. Compared to the graph-based encoder with accuracy of 89.1%, this suggests that our encoder learns better features and hence contributes to the state-of-the-art results that we achieve. To investigate the effectiveness of the multi-task learning, we compare our result against the results reported on a PointNet autoencoder (i.e., single reconstruction decoder) [2] which achieves classification accuracy of 85.7%. This suggests that using multi-task learning improves the quality of the learned features. The summary of the results is shown in Table 4.

Encoder	Decoder	Accuracy
PointNet	Reconstruction	85.7
PointNet	Multi-Task	86.2
Ours	Reconstruction	86.7
Ours	Multi-Task	**89.1**

Table 4. Effect of encoder and multi-task learning on accuracy on the ModelNet40.

We also investigate the effect of different tasks on the quality of the learned features by masking the task losses and training and testing the model on each configuration. The results shown in Table 5 suggest that the reconstruction task has the highest impact on the performance. This is because contrary to [9], we are not applying any heuristics to avoid trivial solutions and hence when the reconstruction task is masked both clustering and classification tasks tend to collapse the features to one cluster which results in degraded feature learning.

Moreover, the results suggest that masking the cross-entropy loss degrades the accuracy to 87.6% (absolute decrease of 1.5%) whereas masking the k-means loss has a less adverse effect (degraded loss of 88.3%, i.e., absolute decrease of 0.8%). This implies that the cross-entropy loss (classifier) plays a more important role than the clustering loss. Furthermore, the results indicate that having both K-means and cross-entropy losses along with the reconstruction task yields the best result (i.e., accuracy of 89.1%). This may seems counter-intuitive as one may assume that using the clustering pseudo-labels to learn a classification function would push the classifier to replicate the K-means behavior and hence the k-means loss will be redundant.

Figure 4. A few sample shapes along with their ground truth part labels, predicted part labels by the trained MLP on 1% of the training data, and predicted part labels by AHC and HDBSCAN methods.

Classification Task	Reconstruction Task	Clustering Task	Overall Accuracy
√	×	×	22.8
×	√	×	86.7
×	×	√	6.9
√	√	×	88.3
√	×	√	15.2
×	√	√	87.6
√	√	√	**89.1**

Table 5. Effect of tasks on the accuracy of classification on the ModelNet40.

However, we think this is not the case because the classifier introduces non-linearity to the feature space by learning non-linear boundaries to approximate the predictions of the linear K-means model which in turn affects the clustering outcomes in the following epoch. K-means loss on the other hand, pushes the features in the same cluster to a closer space while pushing the features of other clusters away.

Finally, we report some of our failed experiments:

- We tried K-Means++ [5] to warm-start the cluster centroids. We did not observe any significant improvement over the randomly selected centroids.

- We tried soft parameter sharing between the decoder and classifier models. We observed that this destabilizes the model and hence we isolated them.

- Similar to [78], we tried stacking more graph convolution layers and recomputing the input adjacency to each layer based on the feature space of its predecessor layer. We observed that this has an adverse effect on both classification and segmentation tasks.

5. Conclusion

We proposed an unsupervised multi-task learning approach to learn point and shape features on point clouds which uses three unsupervised tasks including clustering, autoencoding, and self-supervised classification to train a multi-scale graph-based encoder. We exhaustively evaluated our model on point cloud classification and segmentation benchmarks. The results suggest that the learned features outperform prior state-of-the-art models in unsupervised representation learning. For example, in ModelNet40 shape classification tasks, our model achieved the state-of-the-art (among unsupervised models) accuracy of 89.1% which is also competitive with supervised models. In the ShapeNetPart segmentation task, it achieved mIoU of 77.7 which is only 8% less than the state-of-the-art supervised model. For future directions, we are planning to: (i) introduce more powerful decoders to enhance the quality of the learned features, (ii) investigate the effect of other features such as normals and geodesics, and (iii) adapt the model to perform semantic segmentation tasks too.

References

[1] Martín Abadi, Paul Barham, Jianmin Chen, Zhifeng Chen, Andy Davis, Jeffrey Dean, Matthieu Devin, Sanjay Ghemawat, Geoffrey Irving, Michael Isard, et al. Tensorflow: a system for large-scale machine learning.

[2] Panos Achlioptas, Olga Diamanti, Ioannis Mitliagkas, and Leonidas Guibas. Learning representations and generative models for 3d point clouds. 2018.

[3] Pulkit Agrawal, Joao Carreira, and Jitendra Malik. Learning to see by moving. In *The IEEE International Conference on Computer Vision (ICCV)*, pages 37–45, December 2015.

[4] Elie Aljalbout, Vladimir Golkov, Yawar Siddiqui, and Daniel Cremers. Clustering with deep learning: Taxonomy and new methods. *arXiv preprint arXiv:1801.07648*, 2018.

[5] David Arthur and Sergei Vassilvitskii. K-means++: The advantages of careful seeding. In *Proceedings of the Eighteenth Annual ACM-SIAM Symposium on Discrete Algorithms*, pages 1027–1035, 2007.

[6] Mathieu Aubry, Ulrich Schlickewei, and Daniel Cremers. The wave kernel signature: A quantum mechanical approach to shape analysis. In *2011 IEEE International Conference on Computer Vision Workshops (ICCV Workshops)*, pages 1626–1633, Nov 2011.

[7] 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, June 2010.

[8] Ricardo JGB Campello, Davoud Moulavi, and Jörg Sander. Density-based clustering based on hierarchical density estimates. In *Pacific-Asia conference on knowledge discovery and data mining*, pages 160–172. Springer, 2013.

[9] Mathilde Caron, Piotr Bojanowski, Armand Joulin, and Matthijs Douze. Deep clustering for unsupervised learning of visual features. In *The European Conference on Computer Vision (ECCV)*, pages 132–149, September 2018.

[10] 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.

[11] 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.

[12] Jifeng Dai, Kaiming He, and Jian Sun. Instance-aware semantic segmentation via multi-task network cascades. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 3150–3158, 2016.

[13] Haowen Deng, Tolga Birdal, and Slobodan Ilic. Ppf-foldnet: Unsupervised learning of rotation invariant 3d local descriptors. In *The European Conference on Computer Vision (ECCV)*, September 2018.

[14] Carl Doersch, Abhinav Gupta, and Alexei A. Efros. Unsupervised visual representation learning by context prediction. In *The IEEE International Conference on Computer Vision (ICCV)*, pages 1422–1430, December 2015.

[15] Carl Doersch and Andrew Zisserman. Multi-task self-supervised visual learning. In *Proceedings of the IEEE International Conference on Computer Vision*, pages 2051–2060, 2017.

[16] Gil Elbaz, Tamar Avraham, and Anath Fischer. 3d point cloud registration for localization using a deep neural network auto-encoder. In *The IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 4631–4640, July 2017.

[17] Francis Engelmann, Theodora Kontogianni, Alexander Hermans, and Bastian Leibe. Exploring spatial context for 3d semantic segmentation of point clouds. In *The IEEE International Conference on Computer Vision (ICCV) Workshops*, pages 716–724, Oct 2017.

[18] Andreas Argyriou Theodoros Evgeniou and Massimiliano Pontil. Multi-task feature learning. In *Advances in Neural Information Processing Systems 19: Proceedings of the 2006 Conference*, volume 19, page 41. MIT Press, 2007.

[19] Haoqiang Fan, Hao Su, and Leonidas J. Guibas. A point set generation network for 3d object reconstruction from a single image. In *The IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 605–613, July 2017.

[20] Matheus Gadelha, Rui Wang, and Subhransu Maji. Multiresolution tree networks for 3d point cloud processing. In *The European Conference on Computer Vision (ECCV)*, pages 103–118, September 2018.

[21] Liuhao Ge, Yujun Cai, Junwu Weng, and Junsong Yuan. Hand pointnet: 3d hand pose estimation using point sets. In *The IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 8417–8426, June 2018.

[22] Kamran Ghasedi Dizaji, Amirhossein Herandi, Cheng Deng, Weidong Cai, and Heng Huang. Deep clustering via joint convolutional autoencoder embedding and relative entropy minimization. In *The IEEE International Conference on Computer Vision (ICCV)*, pages 5736–5745, Oct 2017.

[23] Spyros Gidaris, Praveer Singh, and Nikos Komodakis. Unsupervised representation learning by predicting image rotations. In *International Conference on Learning Representations (ICLR)*, 2018.

[24] Rohit Girdhar, David F Fouhey, Mikel Rodriguez, and Abhinav Gupta. Learning a predictable and generative vector representation for objects. In *European Conference on Computer Vision*, pages 484–499. Springer, 2016.

[25] Benjamin Graham, Martin Engelcke, and Laurens van der Maaten. 3d semantic segmentation with submanifold sparse convolutional networks. In *The IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 9224–9232, June 2018.

[26] Thibault Groueix, Matthew Fisher, Vladimir G. Kim, Bryan Russell, and Mathieu Aubry. Atlasnet: A papier-mâché approach to learning 3d surface generation. In *The IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2018.

[27] Pedro Hermosilla, Tobias Ritschel, Pere-Pau Vázquez, Àlvar Vinacua, and Timo Ropinski. Monte carlo convolution for learning on non-uniformly sampled point clouds. *ACM Trans. Graph.*, 37(6):235:1–235:12, 2018.

[28] John R Hershey, Zhuo Chen, Jonathan Le Roux, and Shinji Watanabe. Deep clustering: Discriminative embeddings for segmentation and separation. In *2016 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 31–35, March 2016.

[29] C. Hsu and C. Lin. Cnn-based joint clustering and representation learning with feature drift compensation for large-scale image data. *IEEE Transactions on Multimedia*, 20(2):421–429, Feb 2018.

[30] Binh-Son Hua, Minh-Khoi Tran, and Sai-Kit Yeung. Pointwise convolutional neural networks. In *The IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 984–993, June 2018.

[31] Qiangui Huang, Weiyue Wang, and Ulrich Neumann. Recurrent slice networks for 3d segmentation of point clouds. In *The IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 2626–2635, June 2018.

[32] Sergey Ioffe and Christian Szegedy. Batch normalization: Accelerating deep network training by reducing internal covariate shift. In Francis Bach and David Blei, editors, *Proceedings of the 32nd International Conference on Machine Learning*, Proceedings of Machine Learning Research, pages 448–456, Jul 2015.

[33] Simon Jenni and Paolo Favaro. Self-supervised feature learning by learning to spot artifacts. In *The IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 2733–2742, June 2018.

[34] Felix Jremo Lawin, Martin Danelljan, Fahad Shahbaz Khan, Per-Erik Forssn, and Michael Felsberg. Density adaptive point set registration. In *The IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 3829–3837, June 2018.

[35] Michael Kazhdan, Thomas Funkhouser, and Szymon Rusinkiewicz. Rotation invariant spherical harmonic representation of 3d shape descriptors. In *Proceedings of the 2003 Eurographics/ACM SIGGRAPH Symposium on Geometry Processing*, SGP '03, pages 156–164, Aire-la-Ville, Switzerland, Switzerland, 2003. Eurographics Association.

[36] Diederik P Kingma and Jimmy Lei Ba. Adam: Amethod for stochastic optimization. In *International Conference on Learning Representation (ICLR)*, 2014.

[37] Roman Klokov and Victor Lempitsky. Escape from cells: Deep kd-networks for the recognition of 3d point cloud models. In *The IEEE International Conference on Computer Vision (ICCV)*, pages 863–872, Oct 2017.

[38] Haun Lei, Naveed Akhtar, and Ajmal Mian. Spherical convolutional neural network for 3d point clouds. *arXiv preprint arXiv:1805.07872*, 2018.

[39] Jiaxin Li, Ben M. Chen, and Gim Hee Lee. So-net: Self-organizing network for point cloud analysis. In *The IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 19397–9406, June 2018.

[40] Yangyan Li, Rui Bu, Mingchao Sun, and Baoquan Chen. Pointcnn: Convolution on x-transformed points. In *Advances in Neural Information Processing Systems 32*, pages 828–838. Curran Associates, Inc., 2018.

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

[42] Yong Luo, Yonggang Wen, Dacheng Tao, Jie Gui, and Chao Xu. Large margin multi-modal multi-task feature extraction for image classification. *IEEE Transactions on Image Processing*, 25(1):414–427, 2015.

[43] Daniel Maturana and Sebastian Scherer. 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, Sept 2015.

[44] Erxue Min, Xifeng Guo, Qiang Liu, Gen Zhang, Jianjing Cui, and Jun Long. A survey of clustering with deep learning: From the perspective of network architecture. *IEEE Access*, 6:39501–39514, 2018.

[45] Christian Mostegel, Rudolf Prettenthaler, Friedrich Fraundorfer, and Horst Bischof. Scalable surface reconstruction from point clouds with extreme scale and density diversity. In *The IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 904–913, July 2017.

[46] Daniel Müllner. Modern hierarchical, agglomerative clustering algorithms. *arXiv preprint arXiv:1109.2378*, 2011.

[47] Liangliang Nan and Peter Wonka. Polyfit: Polygonal surface reconstruction from point clouds. In *The IEEE International Conference on Computer Vision (ICCV)*, pages 2353–2361, Oct 2017.

[48] Mehdi Noroozi and Paolo Favaro. Unsupervised learning of visual representations by solving jigsaw puzzles. In *European Conference on Computer Vision (ECCV)*, pages 69–84. Springer, 2016.

[49] Ramakanth Pasunuru and Mohit Bansal. Multi-task video captioning with video and entailment generation. In *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 1273–1283, 2017.

[50] Deepak Pathak, Philipp Krahenbuhl, Jeff Donahue, Trevor Darrell, and Alexei A. Efros. Context encoders: Feature learning by inpainting. In *The IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 2536–2544, June 2016.

[51] Charles R. Qi, Wei Liu, Chenxia Wu, Hao Su, and Leonidas J. Guibas. Frustum pointnets for 3d object detection from rgb-d data. In *The IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 918–927, June 2018.

[52] Charles R. Qi, Hao Su, Kaichun Mo, and Leonidas J. Guibas. Pointnet: Deep learning on point sets for 3d classification and segmentation. In *The IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 652–660, July 2017.

[53] 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 *The IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 5648–5656, June 2016.

[54] Charles Ruizhongtai Qi, Li Yi, Hao Su, and Leonidas J Guibas. Pointnet++: Deep hierarchical feature learning on

point sets in a metric space. In I. Guyon, U. V. Luxburg, S. Bengio, H. Wallach, R. Fergus, S. Vishwanathan, and R. Garnett, editors, *Advances in Neural Information Processing Systems 30*, pages 5099–5108. Curran Associates, Inc., 2017.

[55] Zhongzheng Ren and Yong Jae Lee. Cross-domain self-supervised multi-task feature learning using synthetic imagery. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 762–771, 2018.

[56] Dario Rethage, Johanna Wald, Jurgen Sturm, Nassir Navab, and Federico Tombari. Fully-convolutional point networks for large-scale point clouds. In *The European Conference on Computer Vision (ECCV)*, pages 596–611, September 2018.

[57] Raif M Rustamov. Laplace-beltrami eigenfunctions for deformation invariant shape representation. In *Proceedings of the fifth Eurographics symposium on Geometry processing*, pages 225–233. Eurographics Association, 2007.

[58] Radu Bogdan Rusu, Nico Blodow, and Michael Beetz. Fast point feature histograms (fpfh) for 3d registration. In *2009 IEEE International Conference on Robotics and Automation*, pages 3212–3217, May 2009.

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

[60] Radu Bogdan Rusu, Zoltan Csaba Marton, Nico Blodow, Mihai Dolha, and Michael Beetz. Towards 3d point cloud based object maps for household environments. *Robotics and Autonomous Systems*, 56(11):927–941, 2008.

[61] Uri Shaham, Kelly Stanton, Henry Li, Ronen Basri, Boaz Nadler, and Yuval Kluger. Spectralnet: Spectral clustering using deep neural networks. In *International Conference on Learning Representations (ICLR)*, 2018.

[62] 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.

[63] Yiru Shen, Chen Feng, Yaoqing Yang, and Dong Tian. Mining point cloud local structures by kernel correlation and graph pooling. In *The IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 4548–4557, June 2018.

[64] Nitish Srivastava, Geoffrey Hinton, Alex Krizhevsky, Ilya Sutskever, and Ruslan Salakhutdinov. Dropout: A simple way to prevent neural networks from overfitting. *Journal of Machine Learning Research*, 15:1929–1958, 2014.

[65] Hang Su, Varun Jampani, Deqing Sun, Subhransu Maji, Evangelos Kalogerakis, Ming-Hsuan Yang, and Jan Kautz. Splatnet: Sparse lattice networks for point cloud processing. In *The IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 2530–2539, June 2018.

[66] Jian Sun, Maks Ovsjanikov, and Leonidas Guibas. A concise and provably informative multi-scale signature based on heat diffusion. *Computer Graphics Forum*, 28(5):1383–1392, 2009.

[67] Yongbin Sun, Yue Wang, Ziwei Liu, Joshua E. Siegel, and Sanjay Sarma. Point cloud gan. *arXiv preprint arXiv:1810.05591*, 2018.

[68] Yongbin Sun, Yue Wang, Ziwei Liu, Joshua E. Siegel, and Sanjay Sarma. Pointgrow: Autoregressively learned point cloud generation with self-attention. *arXiv preprint arXiv:1810.05591*, 2018.

[69] Christian Szegedy, Wei Liu, Yangqing Jia, Pierre Sermanet, Scott Reed, Dragomir Anguelov, Dumitru Erhan, Vincent Vanhoucke, and Andrew Rabinovich. Going deeper with convolutions. In *The IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2015.

[70] Gusi Te, Wei Hu, Amin Zheng, and Zongming Guo. Rgcnn: Regularized graph cnn for point cloud segmentation. In *Proceedings of the 26th ACM International Conference on Multimedia*, pages 746–754. ACM, 2018.

[71] Nitika Verma, Edmond Boyer, and Jakob Verbeek. Feastnet: Feature-steered graph convolutions for 3d shape analysis. In *The IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 2598–2606, June 2018.

[72] Oriol Vinyals, Samy Bengio, and Manjunath Kudlur. Order matters: Sequence to sequence for sets. In *International Conference on Learning Representation (ICLR)*, 2015.

[73] Carl Vondrick, Abhinav Shrivastava, Alireza Fathi, Sergio Guadarrama, and Kevin Murphy. Tracking emerges by colorizing videos. In *The European Conference on Computer Vision (ECCV)*, pages 391–408, September 2018.

[74] Jayakorn Vongkulbhisal, Beat Irastorza Ugalde, Fernando De la Torre, and Joo P. Costeira. Inverse composition discriminative optimization for point cloud registration. In *The IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 2993–3001, June 2018.

[75] Chu Wang, Babak Samari, and Kaleem Siddiqi. Local spectral graph convolution for point set feature learning. In *The European Conference on Computer Vision (ECCV)*, pages 52–66, September 2018.

[76] Shenlong Wang, Simon Suo, Wei-Chiu Ma, Andrei Pokrovsky, and Raquel Urtasun. Deep parametric continuous convolutional neural networks. In *The IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 2589–2597, June 2018.

[77] Xiaolong Wang and Abhinav Gupta. Unsupervised learning of visual representations using videos. In *The IEEE International Conference on Computer Vision (ICCV)*, pages 2794–2802, December 2015.

[78] 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.

[79] Donglai Wei, Joseph J. Lim, Andrew Zisserman, and William T. Freeman. Learning and using the arrow of time. In *The IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 8052–8060, June 2018.

[80] 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 D. D. Lee, M. Sugiyama, U. V. Luxburg, I. Guyon, and R. Garnett,

editors, *Advances in Neural Information Processing Systems 29*, pages 82–90. Curran Associates, Inc., 2016.

[81] Zhirong Wu, Shuran Song, Aditya Khosla, Fisher Yu, Linguang Zhang, Xiaoou Tang, and Jianxiong Xiao. 3d shapenets: A deep representation for volumetric shapes. In *The IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 1912–1920, June 2015.

[82] Zhirong Wu, Shuran Song, Aditya Khosla, Fisher Yu, Linguang Zhang, Xiaoou Tang, and Jianxiong Xiao. 3d shapenets: A deep representation for volumetric shapes. In *The IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 1912–1920, June 2015.

[83] Junyuan Xie, Ross Girshick, and Ali Farhadi. Unsupervised deep embedding for clustering analysis. In *Proceedings of The 33rd International Conference on Machine Learning (ICML)*, volume 48, pages 478–487. PMLR, June 2016.

[84] Yifan Xu, Tianqi Fan, Mingye Xu, Long Zeng, and Yu Qiao. Spidercnn: Deep learning on point sets with parameterized convolutional filters. In *The European Conference on Computer Vision (ECCV)*, pages 87–102, September 2018.

[85] Yan Yan, Elisa Ricci, Gaowen Liu, and Nicu Sebe. Egocentric daily activity recognition via multitask clustering. *IEEE Transactions on Image Processing*, 24(10):2984–2995, 2015.

[86] Bo Yang, Xiao Fu, Nicholas D. Sidiropoulos, and Mingyi Hong. Towards k-means-friendly spaces: Simultaneous deep learning and clustering. In *Proceedings of the 34th International Conference on Machine Learning (ICML)*, volume 70, pages 3861–3870. PMLR, August 2017.

[87] Jianwei Yang, Devi Parikh, and Dhruv Batra. Joint unsupervised learning of deep representations and image clusters. In *The IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 5147–5156, June 2016.

[88] Yaoqing Yang, Chen Feng, Yiru Shen, and Dong Tian. Foldingnet: Point cloud auto-encoder via deep grid deformation. In *The IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 206–215, June 2018.

[89] Xiaoqing Ye, Jiamao Li, Hexiao Huang, Liang Du, and Xiaolin Zhang. 3d recurrent neural networks with context fusion for point cloud semantic segmentation. In *The European Conference on Computer Vision (ECCV)*, pages 403–417, September 2018.

[90] Li Yi, Vladimir G. Kim, Duygu Ceylan, I-Chao Shen, Mengyan Yan, Hao Su, Cewu Lu, Qixing Huang, Alla Sheffer, and Leonidas Guibas. A scalable active framework for region annotation in 3d shape collections. *ACM Trans. Graph.*, 35(6):210:1–210:12, Nov. 2016.

[91] Li Yi, Hao Su, Xingwen Guo, and Leonidas J. Guibas. Syncspeccnn: Synchronized spectral cnn for 3d shape segmentation. In *The IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 2282–2290, July 2017.

[92] Lequan Yu, Xianzhi Li, Chi-Wing Fu, Daniel Cohen-Or, and Pheng-Ann Heng. Ec-net: an edge-aware point set consolidation network. In *The European Conference on Computer Vision (ECCV)*, pages 386–402, September 2018.

[93] Lequan Yu, Xianzhi Li, Chi-Wing Fu, Daniel Cohen-Or, and Pheng-Ann Heng. Pu-net: Point cloud upsampling network. In *The IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 2790–2799, June 2018.

[94] Wei Zeng and Theo Gevers. 3dcontextnet: K-d tree guided hierarchical learning of point clouds using local and global contextual cues. In *The European Conference on Computer Vision (ECCV) Workshops*, September 2018.

[95] Yu Zhang and Qiang Yang. A survey on multi-task learning. *arXiv preprint arXiv:1707.08114*, 2017.

[96] Yongheng Zhao, Tolga Birdal, Haowen Deng, and Federico Tombari. 3d point capsule networks. In *The IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2019.

[97] Yin Zhou and Oncel Tuzel. Voxelnet: End-to-end learning for point cloud based 3d object detection. In *The IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 4490–4499, June 2018.