Contrastive and Angle based Deep Clustering of Point Clouds

Nader Alfares [1] **Guangmingmei Yang** [1 2] **David J. Miller** [1 2] **George Kesidis** [1 2]

Abstract

Motivated by a problem in Generative Specification-Producing AIs (SPAIs), we focus on a problem of deep clustering of three-dimensional shapes specified by point clouds. After reviewing deep clustering, we propose a novel approach involving angle based clustering and semisupervised contrastive penalties. The proposed approach is evaluated on the ModelNet dataset and compared against an unsupervised approach leveraging autoencoding.

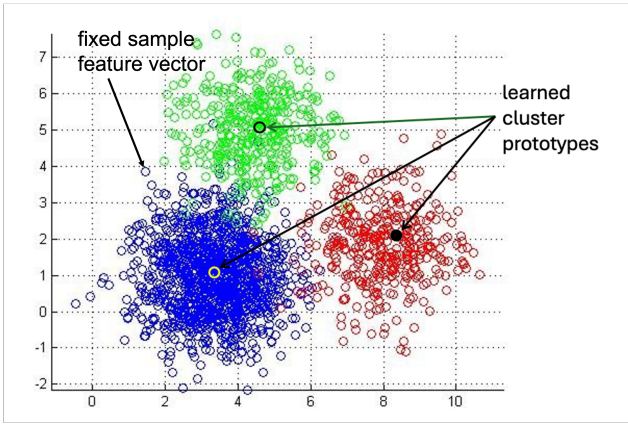

Figure 1. Illustrative example of clustering scenario with $K = 3$ clusters.

1. Introduction

We herein focus on the problem of deep clustering of three-dimensional shapes. Our motivation is the problem of Generative Specification-Producing AIs (SPAIs) (Dodds et al., 2024). SPAIs may have multimodal prompts (as transformer VLMs (Singh et al., 2018)) with diagrams (shapes) and text conveying design requirements. That is, a description of a new design is input and a "formal" design specification is output. The output specification is informed by the designs used to train (or fine-tune) a supervised-learned SPAI. A good design specification for a given set of requirements will be informed by similar ones used to train the SPAI. So, we are concerned with feature maps (encoders) that are useful for multimodal clustering to identify the similar designs used for training, i.e., a retrieval function (Xiao et al., 2021).

In this paper, we review deep clustering, focusing on deep clustering of shapes in particular, using highly discriminative encodings as, e.g., given by a DGCNN, e.g., (Zhang & Zhu, 2019). Shapes are typically specified either by point clouds or tessellations on their surfaces, where these two formats can be translated from one to the other. Three-dimensional shapes are invariant to point-order specification (Charles et al., 2017), translations, rotations, and uniform dilations (or contractions) (Small, 1996).

This paper is organized as follows. We first give an overview of deep clustering. We then describe two main approaches: One is based on unsupervised autoencoding involving either first fixing the encoder feature map then applying K-means (or training a nearest-prototype layer (LeCun, 1998)), or jointly learning the encoder and prototypes. The other is a semisupervised contrastive learning approach involving **fixed prototypes** in the embedded feature (latent, activation) space. Both methods are designed to avoid cluster collapse to which *deep* clustering is particularly prone. There are, of course, hybrid approaches too.

The main contribution is a novel approach to contrastive and angle based deep clustering of shapes. We report results for experiments conducted using the ModelNet10 dataset (ModelNet), which pose challenges because they are highly imbalanced.

2. Background on Clustering

Given an unlabeled training set X of point clouds, one can apply K-means clustering (e.g., (Duda et al., 2001)) using the Chamfer distance, d_c, between pairs of point clouds. Given X, K-means clustering uses an iterative expectation-maximization (map-reduce) to determine the K cluster prototypes $\{\pi_1, ..., \pi_K\}$ by minimizing the cluster-distortion

[1]School of EECS, Penn State, University Park, PA, USA [2]Anomalee Inc., State College, PA, USA. Correspondence to: George Kesidis <gik2@psu.edu>.

Preprint. January 31, 2026.

loss,

$$L_0(\pi) = \frac{1}{|X|} \sum_{x \in X} d_{\mathrm{c}}(x, \pi(x)), \quad \text{where}$$
$$\pi(x) = \arg\min_{\pi_m} d_{\mathrm{c}}(x, \pi_m) \quad \text{and, e.g.,}$$
$$d_{\mathrm{c}}(x, y) = \frac{1}{|x|} \sum_i \min_j \|x_i - y_j\| + \frac{1}{|y|} \sum_j \min_i \|y_j - x_i\|$$

where $x_i, y_j \in \mathbb{R}^3$ respectively are the points of (point clouds) x, y and $\|\cdot\|$ is the Euclidean norm. See Figure 1 for a simple illustrative example with $K = 3$ clusters.

Clearly, the cluster-distortion loss can be zero when $K \geq |X|$. To optimize generalization performance (i.e., avoid overfitting), the number of clusters K can be set using a BIC penalty (Schwarz, 1978; Xiang et al., 2019),

$$\frac{1}{2} K D \log_2 |X|, \tag{1}$$

to the cluster-distortion loss, where here D be the common or average dimension of each encoded shape representation. Alternatively, one can apply a heuristic approach such as the "elbow" method.

On the other hand, **cluster collapse** is a phenomenon where the training samples are assigned to only a few (even just one) cluster. K-means may exhibit cluster collapse, e.g., when the cluster prototypes are poorly initialized.

Deep clustering involves a front-end feature map (encoder) and a back-end clustering. The motivation for applying clustering within a deep learning setting is that, for certain types of data, clustering applied directly to the "raw" input data may produce poor results. For example, consider image domains. The same object may appear in different locations, at different scales, from different perspectives, and in the presence of noise and occlusion, within an image dataset. Applying clustering directly to the raw images is not likely to assign images representing the same object (but subject to these highly variable factors) to the same cluster. However, a deep neural network can learn embedded feature representations that are relatively invariant to these variable factors. Thus, a clustering method applied to deep learning embedded representations of images is much more likely to assign the same object to a common cluster.

As another example, an attempt to detect poisoned samples in the training dataset of a DNN-based classifier has been based on clustering of embedded representations of the training data samples (Tran et al., 2018; Chen et al., 2019; Xiang et al., 2019). (Tran et al., 2018; Chen et al., 2019) employ a two-component clustering approach (simple 2-means clustering in the case of (Chen et al., 2019)), while (Xiang et al., 2019) employs a Gaussian mixture model whose number of components is determined by BIC. As another example, (Kulis & Jordan, 2013) also consider mixture models and penalizes the number of clusters.

Cluster collapse is more problematic for deep clustering because the feature map (encoder) and cluster prototypes (in the range of the feature map) both need to be learned. There are two main ways to proceed:

- Unsupervised learning leveraging an autoencoding to create the encoder and then apply K-means (or attach a nearest prototype layer as in LeNet and learn it by gradient-based back-propagation (LeCun, 1998)).
- Semisupervised with contrastive loss penalties wherein "must link" and "cannot link" decisions are determined by an expert regarding the cluster membership for a small subset of the training dataset.

In both cases an encoder of the form of, e.g., a Deep Graph Convolutional Neural Network (DGCNN) (Zhang & Zhu, 2019) or FoldingNet (Yang et al., 2018).

One alternative supervised approach involves learning an encoder, followed by a decoder and nearest prototype layer in parallel with a training objective being a combination of the reconstruction loss (decoding) and cluster distortion loss (clustering), e.g., (Opochinsky et al., 2020).

In the semisupervised approach, the encoder and nearest prototype layer are jointly learned using **fixed prototypes** ($\pi \in \mathbb{R}^D$) with a cluster-distortion loss objective penalized by the contrastive must-link and, particularly, cannot-link terms regarding pairs of training examples.

Regarding deep clustering of shapes in particular, training-set augmentation is used to ensure invariance of point-order specification, translation, rotation, and uniform dilation (or contraction). Thus, training-set augmentation will create additional must link requirements.

Note that the point cloud *classifier* PointNet (Charles et al., 2017) has a (point-order specification invariant) encoder that is not very expressive but relies on *supervising* class labels. Thus its classification error is low[1]

On the other hand, for purposes of clustering, an autoencoder can be computationally costly, especially for large-scale datasets, with an overly expressive encoder for clustering purposes. one can try to "early stop" the autoencoder training or deprecate how reconstruction loss is weighed against the cluster-distortion loss.

Several works have proposed graphical encoders, including (Yang et al., 2018; Hassani & Haley, 2019; Zhang & Zhu, 2019), where the former two references incorporate autoencoding mechanisms to prevent cluster collapse. In partic-

[1]PointNet's classification accuracy for ModelNet40 is reasonably high at 89.2%, though this is significantly smaller than 93.5% for classification based on a DCGNN encoder.

ular,(Hassani & Haley, 2019) optimizes a joint objective combining cluster-distortion, autoencoding and classification terms. On the other hand, (Zhang & Zhu, 2019) use an unsupervised contrastive approach based on object slicing to learn point-cloud features.

More specifically, in (Zhang & Zhu, 2019): Every training shape is randomly sliced in half and a binary classifier on random slices is trained using a DGCNN encoder assigning 1 to pairs of slices from the same object (must links) and 0 otherwise (cannot links). This trained DGCNN encoder is then applied to the whole training shapes x. K-means on these encodings gives (one of K) pseudo-labels for each $x \in X$. Finally, a second DGCNN based network is trained on the whole shapes using the K-means pseudo-labels, refining the embedding space for cleaner separation. Presumably, cannot links (0) are supposed to avoid cluster collapse, but they include slices from different shapes of the same class in this unsupervised setting. Using planar-aligned slicing may yield cross-sections that are more indicative of the (axis aligned) shape, but this may not improve accuracy considering the cannot links. So, the first DGCNN may only learn about similar slices (cross-sections) of shapes, which may be why it's retrained on whole shapes x. But retraining will just reinforce the K-means decisions based only on matching cross-sections of slices (as the prediction task of (Hassani & Haley, 2019)).

3. ModelNet Dataset for Clustering

ModelNet40 (ModelNet) consists of 12,311 pre-aligned shapes from 40 categories (classes), which are split into 9,843 (80%) for training and 2,468 (20%) for testing, where: each shape is a 3D triangle (surface) mesh in the .OFF format (see Figure 2), and each shape was normalized by inscribing in a unit sphere and converted to a point cloud. The ModelNet10 dataset is a subset of ModelNet40, containing 4,899 shapes from 10 categories. ModelNet10 shares the same preprocessing pipeline as ModelNet40: each mesh is normalized to a unit sphere and converted to a point cloud with 1,024 points sampled from the surface.

For unsupervised clustering, we combined ModelNet40's test and training set.

To assess clustering accuracy after training, training-set class plurality is ascertained for each cluster (i.e., using the class labels which were **not** used for training). That is, if the plurality of training samples clustered to prototype π belong to class $\kappa(\pi)$, and $c(x)$ is the ground-truth class of point cloud x, then the estimated error rate is

$$\frac{1}{|X \backslash U|} \sum_{x \in X \backslash U} \mathbf{1}\{c(x) \neq \kappa(\pi(x))\} \tag{2}$$

where U is the set of training samples involved in contrastive learning (if any). Performance metrics are discussed further below.

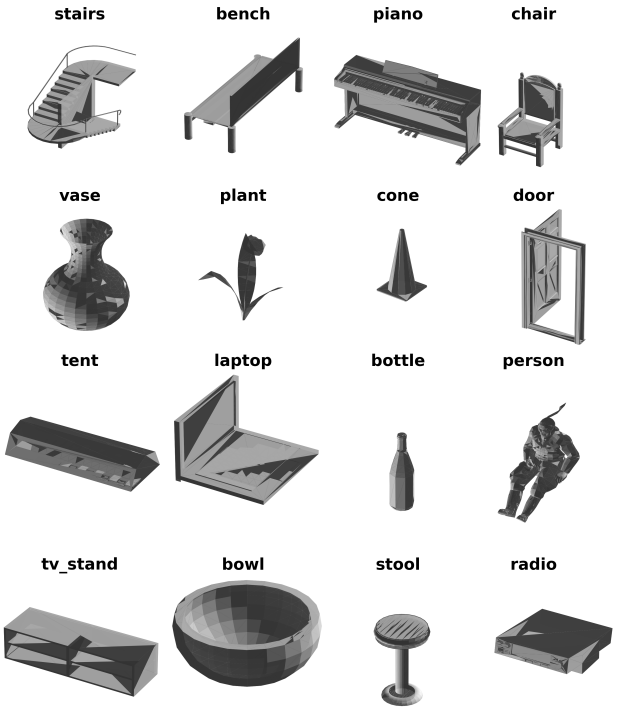

Figure 2. Some example ModelNet shapes and their labeled category.

We conduct our experiments on the ModelNet10 (MN10) dataset (ModelNet).

We evaluate our approach using the following metrics:

- **Accuracy (ACC)**: Clustering accuracy measures the percentage of samples correctly assigned to clusters using a plurality matching strategy. For each predicted cluster j, we find the most common (plurality) true label within that cluster, then count how many samples in the cluster have that label. To clarify (2), let $\mathcal{C}_j = \{i : c_i = j\}$ be the set of samples indexed i assigned to cluster j, and let $y_{\mathcal{C}_j}$ be the ground-truth labels of samples in cluster j. The plurality label for cluster j is:

$$y_j^* = \arg\max_{y \in \mathcal{Y}} |\{i \in \mathcal{C}_j : y_i = y\}| \tag{3}$$

where $\mathcal{Y}$ is the set of true class labels. The number of matching ground-truth and plural label assignments m in cluster j is:

$$m_j = |\{i \in \mathcal{C}_j : y_i = y_j^*\}| \tag{4}$$

The clustering accuracy is then:

$$\text{ACC} = \frac{\sum_{j=1}^{K} m_j}{N} \tag{5}$$

where K is the number of clusters and $N = \sum_j |\mathcal{C}_j|$ the total number of training samples. This metric ranges from 0 to 1, with 1 indicating perfect clustering.

- **Normalized Mutual Information (NMI)**: This metric measures the information-theoretic agreement between true labels and predicted clusters. It quantifies how much information is shared between the two labelings, normalized by their individual entropies. The mutual information between true labels $\mathbf{y}$ and predicted clusters $\mathbf{c}$ is:

$$I(\mathbf{y}; \mathbf{c}) = \sum_{y \in \mathcal{Y}} \sum_{j=1}^{K} p(y, j) \log \frac{p(y, j)}{p(y)p(j)} \tag{6}$$

where $p(y, j) = |\{i : y_i = y,\ c_i = j\}|/N$ is the joint probability, $p(y) = |\{i : y_i = y\}|/N$ is the marginal probability of true label y, and $p(j) = |\{i : c_i = j\}|/N$ is the marginal probability of cluster j.
The entropies are:

$$H(\mathbf{y}) = -\sum_{y \in \mathcal{Y}} p(y) \log p(y) \tag{7}$$

$$H(\mathbf{c}) = -\sum_{j=1}^{K} p(j) \log p(j) \tag{8}$$

The normalized mutual information is:

$$\text{NMI} = \frac{I(\mathbf{y}; \mathbf{c})}{\sqrt{H(\mathbf{y})H(\mathbf{c})}} \tag{9}$$

This metric ranges from 0 to 1, where 1 indicates perfect agreement and 0 indicates independence. NMI is symmetric and does not require label-to-cluster matching, making it robust to different numbers of clusters and classes.

- **Adjusted Rand Index (ARI).** this measures the pairwise similarity between two partitions while correcting for the agreement that would be expected by chance. ARI evaluates how many pairs of samples are clustered together in both labelings, compared to the expected number under a random assignment with the same cluster-size distribution. Let $n_{ij} = |\{k : y_k = i \wedge c_k = j\}|$ be the contingency table count of samples with true class i and predicted cluster j. Let $a_i = \sum_j n_{ij}$ denote the size of true class i, and $b_j = \sum_i n_{ij}$ the size of predicted cluster j. The number of sample pairs that appear in the same class *and* the same predicted cluster is:

$$A = \sum_{i,j} \binom{n_{ij}}{2}. \tag{10}$$

The expected number of such same agreements under random labeling (with fixed class sizes $\{a_i\}$ and cluster sizes $\{b_j\}$) is:

$$B = \frac{\sum_i \binom{a_i}{2} \sum_j \binom{b_j}{2}}{\binom{N}{2}}. \tag{11}$$

The average number of same cluster pairs in the two partitions is:

$$C = \frac{1}{2}\left[\sum_i \binom{a_i}{2} + \sum_j \binom{b_j}{2}\right]. \tag{12}$$

The Adjusted Rand Index is then defined as:

$$\text{ARI} = \frac{A - B}{C - B}. \tag{13}$$

ARI equals 1 when the two partitions are identical, equals 0 when the agreement matches the expected value under random labeling, and can be negative when the agreement is worse than random. ARI is symmetric and permutation invariant.

4. Semi-supervised using Contrastive Learning

In this section, we describe a semisupervised approach to deep clustering by contrastive learning. The K prototypes on the unit sphere in the embedded feature space $\mathbb{R}^D$ are **fixed** by initializing them as vertices of a regular simplex (SIMPLEX_COORDINATES), which maximizes the minimum Euclidean distance (equivalently, cosine similarity) between any pair of them.

The regular simplex construction proceeds as follows: for K prototypes in $\mathbb{R}^D$, we construct a regular k-simplex where the first vertex is placed at $\mathbf{v}_0 = (1, 0, 0, \ldots, 0)$, and each subsequent vertex $\mathbf{v}_i$ for $i = 1, \ldots, K-1$ is constructed such that its first i coordinates are set to $-1/\sqrt{i(i+1)}$ and its i^{th} coordinate is set to $\sqrt{i/(i+1)}$, with all remaining coordinates set to zero. All vertices are then normalized to lie on the unit sphere, ensuring that the prototypes $\pi_k \in \mathbb{R}^D$ are unit vectors with maximum pairwise separation.

Another approach for constructing the prototypes is to use Tamme's algorithm which maximizes the minimum Euclidean distance (equivalently, cosine similarity) between any pair of them (Erber & Hockney, 1997). Using angle based clustering, the encoder $\pi_k \in \mathbb{R}^D$ is then learned by minimizing the cluster-distortion loss objective C_K over the training dataset using angle-based clustering (Zhu et al., 2016; Beer et al., 2020; 2024):

$$C_K = -\frac{1}{|X|} \sum_{x \in X} \max_{k=1,\ldots,K} \frac{\langle \phi(x),\ \pi_k \rangle}{\|\phi(x)\|} \tag{14}$$

Because the prototypes are all on the surface of a unit sphere, this is equivalent to clustering using Euclidean distance. Because of the regular simplex, the domain of every cluster is an infinitely tall regular pyramid (i.e., a cone with flat size) emanating from the origin (its apex) with its prototype on its axis.

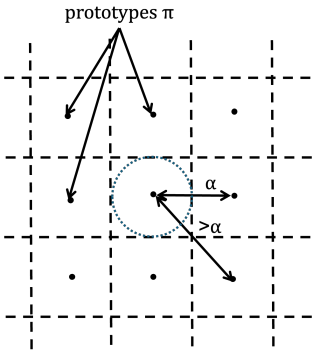

Figure 3. Regular tessellation of the plane with prototypes fixed by Tamme's algorithm producing a regular square tessellation. Note that the angle α corresponding to the largest cosine similarity between such prototypes, α, is smaller than the angle between diagonal neighbors. That is, there are some points in a square tile whose cosine similarity with the tile's prototype (center) is $< \cos(\alpha/2)$. Note that for a square tessellation of the sphere, only three squares meet at a vertex (as a cube).

The overall objective combines a clustering term with supervised contrastive and augmentation consistency penalties:

$$\mathcal{L}_{\text{total}} = C_K + \mathcal{L}_{\text{con}} + \mathcal{L}_{\text{aug}} \tag{15}$$

Consider the minimum cosine similarity between pairs of prototypes ($\in \mathbb{R}^D$) of a regular simplex,

$$\cos(\alpha) \;:=\; \min_{i \neq j} \frac{\langle \pi_i,\ \pi_j \rangle}{\|\pi_i\|\ \|\pi_j\|} \tag{16}$$

If the cosine similarity of y and $\pi(x)$ is $\leq \cos(\alpha/2)$ then y and x are in the same cluster, but the converse is not necessarily true; see Figure 3. That is, to ensure x and y are in different clusters, then their the cosine similarity of $\pi(x)$ and y needs to be $< \cos(\alpha/2)$. In the following, the angle α' is such that $\cos(\alpha'/2) < \cos(\alpha/2)$.

The augmentation consistency term $\mathcal{L}_{\text{aug}}$ enforces that augmented views A_x of the same point cloud x remain within a narrower cone around their prototype:

$$\frac{\lambda}{|X| \cdot |A|} \sum_{y \in A_x,\, x \in X} \left(\cos\left(\frac{\alpha}{2}\right) - \frac{\langle \pi(x), \phi(y) \rangle}{\|\phi(y)\|} \right)^+ \tag{17}$$

where $|A| = |A_x|$ for all $x \in X$, penalty parameter $\lambda > 0$, and $(z)^+ := \max\{z, 0\}$.

A small subset of the training set $U \subset X \times X$ is processed and pairs of samples (point-clouds) in U that must ($V_+ \subset U \times U$) or must not ($V_- \subset U \times U$) be in the same cluster are identified. The supervised contrastive term $\mathcal{L}_{\text{con}}$ penalizes pairs in the subset U whose prototypes are too close in angle despite being in different clusters:

$$\frac{\lambda_-}{|V_-|} \sum_{(x,y) \in V_-} \left(\frac{\langle \pi(x),\ \phi(y) \rangle}{\|\phi(y)\|} - \cos\left(\frac{\alpha'}{2}\right) \right)^+ \tag{18}$$

with $\lambda_- > 0$. $\mathcal{L}_{\text{con}}$ also includes a term similar to that of $\mathcal{L}_{\text{aug}}$ for pairs of samples in U that are deemed to be in the same cluster (with penalty parameters $\lambda_+ > 0$):

$$\frac{\lambda_+}{|V_+|} \sum_{(x,y) \in V_+} \left(\cos\left(\frac{\alpha}{2}\right) - \frac{\langle \pi(x),\ \phi(y) \rangle}{\|\phi(y)\|} \right)^+ \tag{19}$$

Finally, note that during the deep learning process (i.e., back propagation (Kingma & Ba, 2014)), the cluster prototypes for the training samples are updated before the gradients of L_{total} are calculated.

5. Experimental Result for ModelNet10

We first give results for the autoencoder/K-means unsupervised approach to act as a baseline for our semisupervised contrastive learning approach.

5.1. KMeans++ on FoldingNet Encoder

We train a FoldingNet autoencoder (Yang et al., 2018) on our split of the ModelNet dataset. The encoder takes as input 2,048 points uniformly sampled from each shape in the dataset, and the decoder outputs a 2,025-point reconstruction, corresponding to the size of the 2D grid that is "folded" into the target shape. The minimizing objective of the autoencoder is the averaged bidirectional Chamfer distance between the source and target shape. Table 1 shows clustering results for KMeans++ on encoder embeddings. The reconstruction loss (Chamfer distance before applying KMeans++) was 0.0931.

Table 1. Autoencoding/KMeans++ Clustering Results on ModelNet10

# Clusters (K)	**NMI**	**ARI**	**ACC**
2	0.372	0.217	0.356
4	0.576	0.382	0.516
6	0.665	0.576	0.654
8	0.697	0.702	0.762
10	0.684	0.627	0.775
12	0.670	0.582	0.772
14	0.663	0.542	0.787

5.2. Contrastive Semisupervised Deep Clustering

Experimentally, we employed the class labels to indicate only cannot links, i.e., if two examined training samples have different class label then then they can't be in the same cluster, where the converse is not true because a class can consist of multiple clusters.

The only must links are augmentations of the same training sample.

We evaluate the performance of our semi-supervised angle-

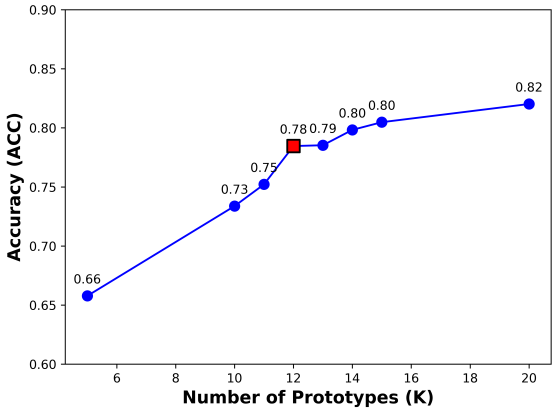

Figure 4. Relationship between the number of prototypes K and classification accuracy on ModelNet10 using the semi-supervised angle based clustering approach. The optimal configuration ($K = 12$, indicated by the red square) was determined using the "elbow method" to identify the point of maximum marginal utility.

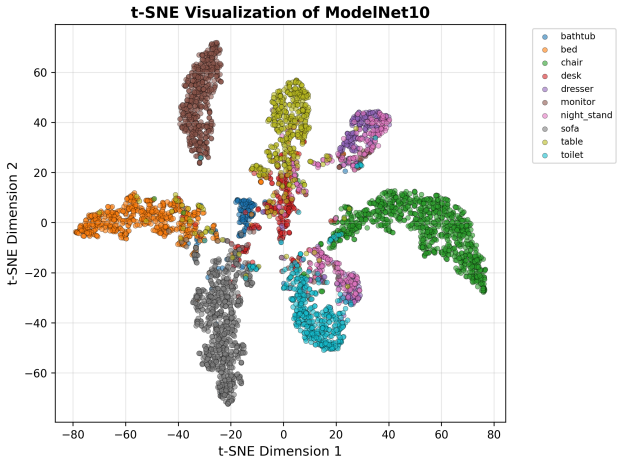

Figure 5. t-SNE visualization of ModelNet10 embeddings using semi-supervised clustering. Each point is a 3D shape from ModelNet10, colored by their true classes. The clustering shows the model learns class-separable representations.

# Clusters (K)	**NMI**	**ARI**	**ACC**
{5, 10}	0.6740	0.5889	0.6578
	0.7282	0.7105	0.7338
11–15	0.7325	0.6934	0.7522
	0.7242	**0.7400**	**0.7846**
	0.7522	0.7501	0.7853
	0.7511	0.7693	0.7983
	0.7631	0.7662	0.8048
20	0.7612	0.7848	0.8202

Table 2. Performance of the semi-supervised, contrastive, angle based clustering method on ModelNet10. Rows are grouped to reflect different step sizes in the number of clusters. Moreover, the bolded row indicates the elbow point of the performance curves.

based clustering approach on the ModelNet10 dataset, examining both quantitative metrics and qualitative visualizations of the learned representations. 10% of the training samples were involved in the contrastive learning terms. Figure 4 illustrates the relationship between the number of prototypes K and classification accuracy, revealing a clear trend of improving performance with increasing K. The accuracy increases from 0.66 at $K = 5$ to 0.82 at $K = 20$, demonstrating that the model benefits from a larger number of prototypes. However, the rate of improvement diminishes significantly after $K = 12$, where the accuracy reaches 0.78. We identify $K = 12$ as the optimal configuration using the elbow method, which balances performance gains with model complexity.

The quantitative performance metrics across different values of K are summarized in Table 2. At the optimal configuration of $K = 12$, our method achieves an NMI of 0.7242, an ARI of 0.7400, and an accuracy (ACC) 0.7846. These results demonstrate strong clustering performance, with the ARI score of 0.7400 indicating substantial agreement between the predicted clusters and the true class labels.

Note that at $K = 10$ to 12, semisupervised contrastive and unsupervised autoencoding give comparable performance.

The quality of the learned representations is further validated through the t-SNE visualization shown in Figure 5, which projects the high-dimensional embeddings into a two-dimensional space while preserving local neighborhood structures. The visualization reveals ten distinct, well-separated clusters corresponding to the ten object classes in ModelNet10, demonstrating that the model successfully learns class-separable representations. Each cluster is clearly identifiable and spatially distinct, with minimal overlap between different classes. Some minor intermingling occurs between semantically similar classes, such as table (yellow) and desk (red), or night stand (pink) and dresser (purple), which is expected given their geometric similarities. Overall, the clear separation of clusters in the t-SNE visualization corroborates the quantitative results, indicating

that the semi-supervised angle-based clustering approach effectively captures the underlying class structure of the 3D shape data.

6. Summary

This paper concerns deep clustering of three-dimensional shapes specified by point clouds. Two general deep-clustering approaches are discussed (with possible hybrids) particularly to avoid cluster collapse: the first is unsupervised involving a feature-map (encoder) through an autoencoding mechanism and K-means clustering on the encodings, while the second is semisupervised involving fixed cluster centers and angle-based cluster-distortion loss training objective with contrastive-loss penalties. We herein focused on a novel semisupervised approach. Experimental results are provided for ModelNet10 point cloud dataset, together with baseline performance for the simple unsupervised approach. Comparing the two general approaches for large-scale datasets, note that the heightened computational complexity of training an autoencoder for the unsupervised approach is traded-off against the effort required to cannot-link label pairs of training examples for the semisupervised approach.

Acknowledgements

This material is based upon work supported by the National Science Foundation under Grant Number 2317987. Also under NSF Grant Number 2415752.

References

Beer, A., Seeholzer, D., Schuler, N.-S., and Seid, T. Angle-Based Clustering. In *Proc. ACM Int'l Conf. on Similarity Search and Applications (SISAP)*, 2020.

Beer, A., Weber, P., Miklautz, L., Leiber, C., Durani, W., B ohm, C., and Plant, C. SHADE: Deep Density-based Clustering. https://arxiv.org/abs/2410.06265, 2024.

Charles, R. Q., Su, H., Kaichun, M., and Guibas, L. J. PointNet: Deep Learning on Point Sets for 3D Classification and Segmentation. In *Proc. IEEE CVPR*, 2017.

Chen, B., Carvalho, W., Baracaldo, N., Ludwig, H., Edwards, B., Lee, T., Malloy, I., and Srivastava, B. Detecting backdoor attacks on deep neural networks by activation clustering. In *Proc. AAAI Workshop on Artificial Intelligence Safety (SafeAI)*, 2019.

Dodds, M., Kiniry, J., and Sweet, S. Generative AI for Specifications. https://www.galois.com/articles/generative-ai-for-specifications, 2024.

Duda, R., Hart, P., and Stork, D. *Pattern classification*. Wiley, 2001.

Erber, T. and Hockney, G. M. Complex systems: Equilibrium configurations of N equal charges on a sphere ($2 \leq N \leq 112$). *Adv. Chem. Phys.*, 98:495–594, 1997.

Hassani, K. and Haley, M. Unsupervised multi-task feature learning on point clouds. In *Proc. IEEE/CVF ICCV*, Seoul, 2019.

Kingma, D. and Ba, J. Adam: A method for stochastic optimization. https://arxiv.org/abs/1412.6980, 2014.

Kulis, B. and Jordan, M. Revisiting K-means: New Algorithms via Bayesian Nonparametrics. In *Proc. ICML*, 2013.

LeCun, Y. LeNet-5, convolutional neural networks. http://yann.lecun.com/exdb/lenet/, 1998.

ModelNet. Princeton ModelNet. https://modelnet.cs.princeton.edu.

Opochinsky, Y., Chazan, S. E., Gannot, S., and Goldberger, J. K-autoencoders deep clustering. In *Proc. IEEE ICASSP*, 2020.

Schwarz, G. Estimating the dimension of a model. *Annals of Stats*, 6:461–464, 1978.

SIMPLEX_COORDINATES. Coordinates of regular simplex in m dimensions. https://people.math.sc.edu/Burkardt/c_src/simplex_coordinates/simplex_coordinates.html.

Singh, J., Ying, V., and Nutkiewicz, A. Attention on Attention: Architectures for Visual Question Answering (VQA). https://arxiv.org/abs/1803.07724, 2018.

Small, C. *The Statistical Theory of Shape*. Springer, 1996.

Tran, B., Li, J., and Madry, A. Spectral signatures in backdoor attacks. In *Proc. NIPS*, 2018.

Xiang, Z., Miller, D., and Kesidis, G. A Benchmark Study of Backdoor Data Poisoning Defenses for Deep Neural Network Classifiers and A Novel Defense. In *Proc. IEEE MLSP*, Pittsburgh, 2019.

Xiao, X., Joshi, S., and Cecil, J. Critical assessment of Shape-Retrieval Tools (SRTs). *Int'l Journal of Advanced Manufacturing Technology*, 2021.

Yang, Y., Feng, C., Shen, Y., and Tian, D. Foldingnet: Point cloud auto-encoder via deep grid deformation. https://arxiv.org/abs/1712.07262, 2018.

Zhang, L. and Zhu, Z. Unsupervised feature learning for point cloud by contrasting and clustering with graph convolutional neural network. In *Proc. IEEE Int'l Conf. on 3D Vision*, Quebec City, 2019.

Zhu, Y., Wang, N., Miller, D., and Wang, Y. Convex Analysis of Mixtures for Separating Non-negative Well-grounded Sources. *Sci Rep 6, 38350*, 2016.