Self-Supervised 3D Mesh Object Retrieval

Kajal Sanklecha, Prayushi Mathur, P. J. Narayanan
Center for Visual Information Technology, International Institute of Information Technology, Hyderabad

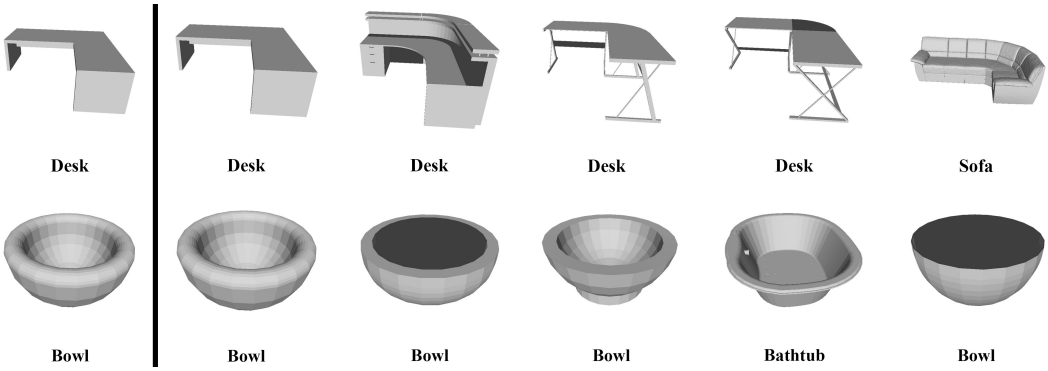

Figure 1: Given a query object (left), the top-5 objects returned by our system, ranked from left to right, have strong similarity in shape. The learned embedding seems to capture the underlying shape well. Labels are shown only for reference. Our system is trained in a self-supervised manner and doesn't use the labels.

ABSTRACT

Digital representations of 3D objects are increasingly being used for engineering, entertainment, education, etc. Efforts to search and retrieve digital 3D models from a collection have not attracted sufficient attention, unlike digital representations of documents, images, etc. Supervised methods are not feasible to solve this problem as a large collection of labelled 3D objects is difficult to create. This paper presents a self-supervised method to learn efficient embeddings of 3D mesh objects for ranked retrieval of similar objects. We propose a simple representation of mesh objects and an encoder-decoder architecture to learn the embedding. Extensive experiments show that our method is competitive with methods that need supervision while being more scalable to different object collections.

CCS CONCEPTS

• **Computing methodologies** → **Shape modeling**; **Learning latent representations**.

KEYWORDS

Object Retrieval, 3D Triangle Mesh, Self-Supervision, Embedding Space, Ranked Retrieval, Mesh Analysis

ACM Reference Format:
Kajal Sanklecha, Prayushi Mathur, P. J. Narayanan. 2023. Self-Supervised 3D Mesh Object Retrieval. In *Proceedings of 14th Indian Conference on Computer Vision, Graphics and Image Processing (ICVGIP'23).* ACM, New York, NY, USA, 10 pages. https://doi.org/10.1145/3627631.3627657

ICVGIP'23, December 2023, Ropar, India

ACM ISBN 978-x-xxxx-xxxx-x/YY/MM.
https://doi.org/10.1145/3627631.3627657

1 INTRODUCTION

Digitally represented 3D models are used widely for designing, visualizing and communicating complex objects and environments. The use of digital 3D models has become increasingly important in various fields such as virtual and augmented reality, gaming, medical imaging, design, scientific visualization, training, education, etc. As the trend continues, the demand for models of greater variety and detail will increase in the future. Advanced authoring tools are the major source of creating rich 3D models. Computer vision algorithms are employed to capture complex models from the world using regular and depth cameras. For 3D models, the triangle (or polygon) mesh representation is the most common and versatile, though point clouds [46, 50, 58, 59, 84], implicit surfaces, voxels [18, 77], signed distance fields, etc., have found several applications.

Rich tools and techniques to create, process, analyze, and render 3D models are available today. However, efforts to search and retrieve similar objects from a collection of 3D models given a query model lag behind such efforts involving other digital assets. Text retrieval has been an aid in searching a large collection of documents quickly and efficiently. Search engines, recommendation systems, and other tools have been indispensable in everyday life. Large numbers of 3D models are accessible via the internet and in specialised databases today and the collections are growing rapidly. A large number of 3D models, mostly represented as triangular meshes, are available on repositories such as Thingiverse [5], GrabCAD [2], Sketchfab [3], TurboSquid [4] and 3D Warehouse [1]. Several open 3D datasets are available today, including ModelNet40 [77], ShapeNet [17], Objaverse [24], CADSketchNet [52] and ObjectNet3D [79]. We will need methods to search and retrieve 3D models like search engines for other types of data.

Early methods for retrieval used hand-designed descriptors computed from the geometric data, such as Spin Images [40], Heat Kernel Signatures [69], etc. Descriptors based on activations of neural

networks have become dominant recently due to their discriminative power, compactness of representation, and search efficiency. There have only been a few such modern efforts for 3D retrieval. The related problems of classification and recognition of 3D models have gotten more attention. Supervised methods for retrieval cannot scale up due to the effort involved in annotation. In this paper, we explore creating a self-supervised system for the 3D retrieval task along the lines of image retrieval [16, 21, 36, 78] and document retrieval, classification or summarization [23, 75].

Supervised Siamese network [70] architectures that learn to recognize and match images in an embedding space are used in image recognition [70]. Such networks learn feature embeddings such that similar items are together and dissimilar ones are farther away from one another. This embedding can also be used for retrieval from large databases. We extend this idea to learn an embedding space of the 3D mesh objects in a self-supervised manner without the need of annotating the objects with their class or other properties.

Current 3D retrieval methods can be categorized into view-based and model-based methods. View-based methods retrieve using multi-view images of the 3D object, but they take only two dimensions of the three-dimensional information of the object, which results in data loss. Model-based 3D retrieval methods find and retrieve 3D models based on their shape and structure. They use descriptors, bag-of-features, shape context, graph-based or spectral methods, and deep learning-based approaches like CNNs for efficient retrieval.

We propose a model-based 3D retrieval method that learns an embedding for 3D shapes using an appropriate encoder-decoder architecture. Our model uses self-supervised training. Shape representation is a critical aspect of geometry processing. We design our system for triangular mesh models. A triangle mesh consists of a set of triangles with no standardized ordering. Without addressing the hard problem of ordering, we seek an embedding for mesh objects that is useful to retrieve them. Our simple embedding scheme, inspired by MeshNet++ [67], is trained on ModelNet40 [77] dataset in a self-supervised way without using category labels. We show that the embedding can be used effectively to retrieve the top-k matches, as shown in Figure 1 from a database of 3D models. We achieve a retrieval mAP of 87% for top-10 matches on ModelNet40. Our major contributions are:

- A self-supervised retrieval system for 3D mesh objects, we believe, is the first of its kind in the literature. We believe the self-supervised method is an advantage as it can scale to larger and diverse collections easily.
- A simple input representation of 3D mesh objects that keeps computation and storage requirements feasible making this a method that can be generalised and widely used.
- An encoder-decoder architecture with fully connected MLPs to learn the embedding space of each object that is trained in a self-supervised manner for a complicated yet information-rich mesh representation of a 3D object.

2 RELATED WORK

The literature on classification, retrieval, self-supervised learning, and mesh analysis is vast. We discuss recent relevant efforts that have contributed to our work on self-supervised 3D object retrieval.

2.1 Mesh Analysis

Mesh analysis involves analyzing and processing 3D polygonal meshes, which are a common digital representation of 3D shapes for computer graphics, computer vision, etc. Traditional geometric processing of meshes goes back a long time, while Deep Learning on meshes has a shorter history. We discuss some of the recent deep learning based methods. MeshCNN [33] collapses edges of the mesh by combining convolution and pooling operations for classification. ExMeshCNN [41] learns the geodesic and geometric characteristics of a mesh to analyze them. Picasso [44] learns features on a mesh to get GPU-accelerated mesh decimation. MeshMAE [47] is a transformer-based mesh analysis for classification and segmentation and Markov Random Fields [43] is a graphical probabilistic model to improve mesh analysis. Mesh analysis techniques can be used for a variety of tasks such as mesh segmentation, mesh simplification, mesh smoothing, mesh parameterization, mesh deformation and mesh repair. Thus, we use mesh in our method to extract maximum information from the 3D object. Meshes are not entities defined on a 2D or 3D lattice. They need to be represented carefully before they can be processed by neural networks.

2.2 Object Classification

Classification of 3D objects traditionally uses supervised methods, with objects paired with their class or category label. Retrieval involves assigning the class of query objects. Classical methods used for 3D object classification involve spin images [40], quasi-spin images [73], 3D SIFT [62] and 3D SURF [42]. They are based on hand-crafted features computed from the object model and matching in the feature space. Machine learning and neural network-based methods have become popular in recent years due to their ability to learn discriminative features directly from raw data. Popular deep learning-based 3D object classification methods include MeshNet [28], MeshCNN [33], MeshNet++ [67], and MeshMAE [47].

2.3 Self-Supervision

Self-supervised learning involves algorithms learning directly from the data itself, without the need of human-annotated labels. The algorithms typically use some auxiliary task to learn the useful representation of the data. Autoencoders are common to learn embeddings of the input data that can be used for a variety of tasks. Self-supervised learning has become increasingly popular in image retrieval [21, 30, 34], signal processing [9, 54] and document retrieval [6, 7, 22] due to its ability to learn useful features without requiring explicit supervision and are used for different applications.

There are methods which use self-supervision or unsupervised networks for 3D point cloud processing. A few methods which generate local features for the 3d objects are Deep Spatiality [32], L2G Auto-encoder [49] and 3D local features [71] among others. Some try to learn a representation for the entire 3D point cloud like Deep Shape descriptor [66], few-shot learning [11, 64] , sampling invariance [19], pretraining of 3D features [85] and graph topology [20].

2.4 Retrieval

Search and retrieval help identify useful data from large collections. Classical methods used for text retrieval are probabilistic [8], latent

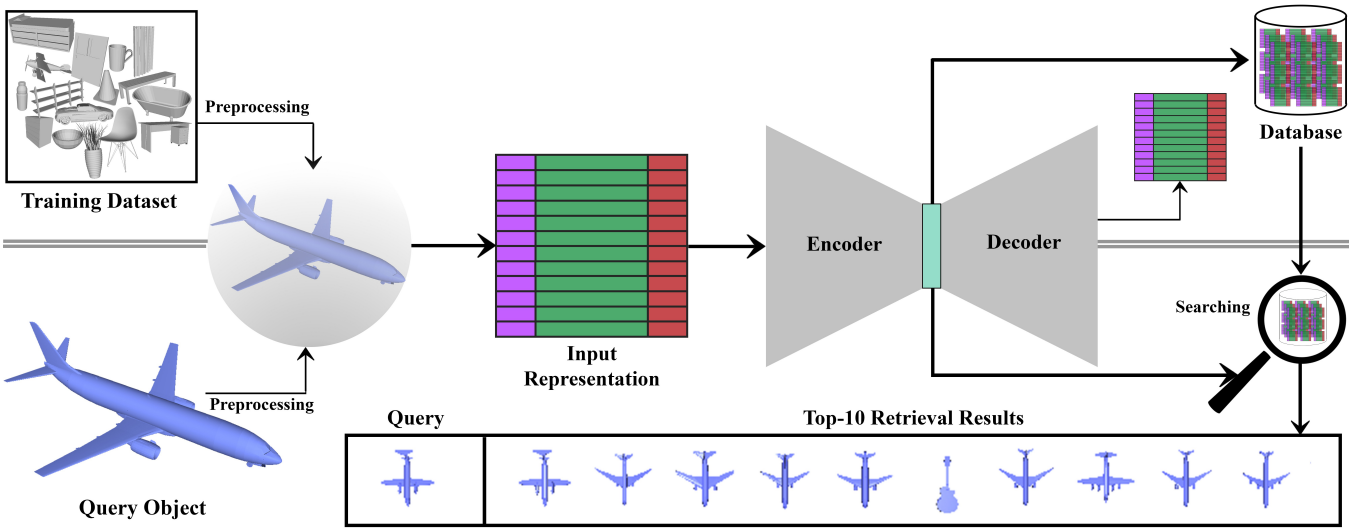

Figure 2: The overall pipeline of our system. The top part shows the training and database building steps done offline. The bottom part shows the query step done online while querying the system. Both use the same representation and embedding provided by the encoder trained in a self-supervised manner. Preprocessing for normalization and the input representation (Figure 3) form the major portion for training the network to generate representations for the 3D objects so that the online querying can take place in a blink of an eye.

semantic indexing [26], okapi BM25 [61] and page rank algorithm [12]. Several neural networks and machine learning algorithms have also been proposed, such as ranking documents by relevance [15] and BERT [83]. Well-known classical methods for image retrieval include bag-of-visual-words [65], vector of locally aggregated descriptors [38], fisher vector [56], etc. State-of-the-art methods for image retrieval using deep-learned embedding include text-image search [76] and fine-tuned CNN [60]. There are methods which utilize text and image embeddings to retrieve 3D models and generate 3D model embeddings like ULIP [81], OpenShape [48] and ULIP2 [82]. Extending these retrieval techniques to the 3D domain is a non-trivial task [25]. As a result, methods particularly need to be developed for 3D object retrieval.

Query-based 3D retrieval methods use different types of input queries. View-based query methods [10, 29, 35] use one or more views of the query object to retrieve similar objects from a database. Sketch-based query methods [27, 45, 57, 72, 74] use 2D sketches or drawings of an object to retrieve similar objects from a database. Object-based query methods [13, 28, 67, 80] for 3D object retrieval involve using 3D models or parts of objects to retrieve similar objects from a database. Ours is an attempt to develop a model-based retrieval system.

Classification and retrieval methods also use other properties of the objects to be retrieved, such as rigid objects [10, 28, 67, 72, 73] and non-rigid objects [13, 63, 80]. These can also be divided on the basis of complete objects [10, 28, 31, 67], and local shape or part-based [10, 72, 73] objects being looked-up for. Shape Google [13] and Deep Shape [80] use Heat Kernel Signature (HKS) [14] as features. Part-based retrieval uses parts or components of an object to retrieve objects with similar parts from a database. The idea is to decompose an object into its constituent parts and use these parts as queries to find other objects with similar parts. In complete object retrieval, the entire 3D object is matched at a time for comparison. Microshapes [72], and QUICCI [73] use hamming distance for feature matching, while GIFT uses inverted files for matching and re-ranking.

3 METHOD

The overall pipeline of our method is shown in Figure 2. All mesh objects are first pre-processed to fit into a unit sphere and reduced to meshes with n = 512 triangles. We use a simple representation for each face of a triangular mesh which is inspired by the representation of MeshNet++ [67]. The representation of 3D objects consisting of the coordinates of the center, corners and normal of the triangles suffices for this purpose. We use a fully-connected encoder-decoder to learn the embedding, used to create a database of embeddings. The encoder is used to embed query objects. Objects can be ranked and retrieved solely based on geometric similarity to the query object. We learn an embedding suitable for the collection of mesh objects in the collection of databases using a self-supervised network.

3.1 Preprocessing

Rich mesh representations of 3D models contain a large number of triangles. For easier processing, we reduce the number of triangles in each mesh to n = 512. This reduces the size of the mesh and thus computational cost. Decimation helps to concentrate on major shape properties, overlooking fine local variations. Manifold [37] facilitates mesh decimation, generating watertight meshes not required for our method but is required for methods like MeshNet [28] and MeshNet++ [67]. The triangle mesh is normalized in size to fit into a unit sphere. The center of the object is shifted to the origin. These steps achieve generalizability for all meshes.

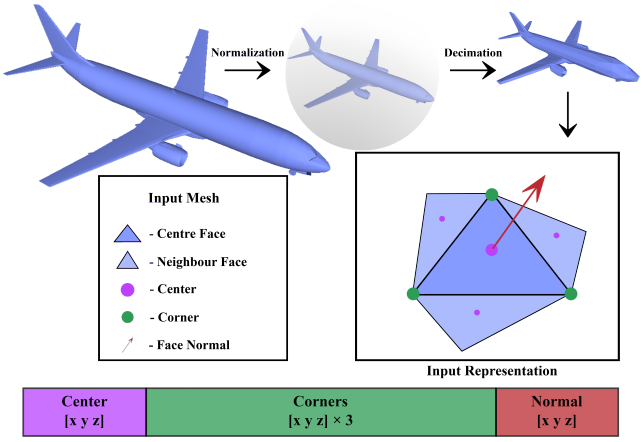

Figure 3: Preprocessing and the simple input representation consisting of center, corners, and normal for each triangle in the mesh

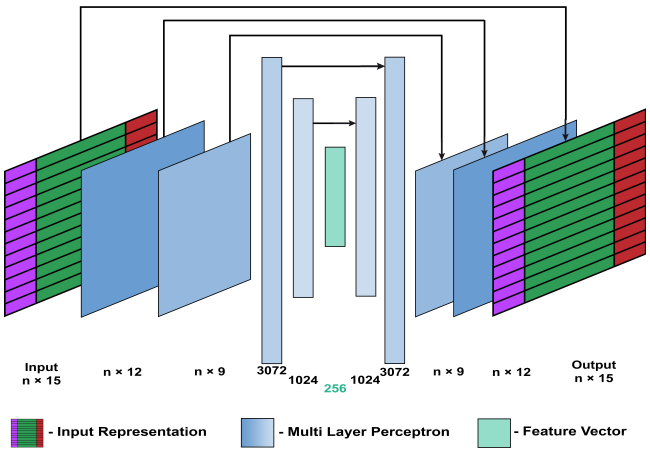

Figure 4: The architecture of our encoder-decoder system. The input for a mesh is a list of the simple representations of its triangles. The 5 network layers each of encoder and decoder are fully connected and the bottleneck layer has 256 dimensions.

3.2 Input Representation

Compared to other 3D object representations, triangle mesh is intuitive, rich in information, and compact in size. Many tools are available to create, edit, and visualize meshes. Other 3D representations such as point cloud, SDF [55] and NeRF [53] can be converted into a mesh using marching cubes [51]. We choose face as the key component from all the information of a mesh [28, 67].

Based on detailed experimentation (presented in Section 4.4), we settled on a simple and reduced representation for each triangle of the mesh, consisting of its center, corners, and normal as deduced from Table 3. A 2D matrix representing the 3D object is generated. The number of rows in the 2D matrix is equal to the number of faces, n in the mesh being encoded. The descriptor of each triangle in the mesh consists of the following components (see Figure 3):

- **Center**: The (x, y, z) coordinates of the center of the triangle represent the global structure of the mesh in the 3D space.
- **Corners**: The (x, y, z) coordinates of the three corners of the triangle.
- **Normal**: The (x, y, z) coordinates of the unit normal vector of the triangle directed outwards from the object.

3.3 Encoder-Decoder Architecture

For model-based 3D object retrieval, feature vectors are learnt in embedding space representing the whole mesh object. To learn the embeddings, we propose a simple encoder-decoder network based on the Siamese network architecture [70]. In the image domain, 2D convolution layers are used by encoders-decoders for recognition and retrieval. Such convolutions are not defined clearly for mesh representations. Therefore, instead of using convolutional layers, we use fully connected MLP layers. Our encoder-decoder network architecture is shown in Figure 4. Both encoder and decoder have five MLP layers each. We use skip connections to help the flow of gradients and to avoid the vanishing gradient problem. The embedding vector obtained from the bottleneck layer of the encoder is a 256 length vector.

3.4 Object Retrieval

Retrieval using the learned embedding space can handle large datasets efficiently by using nearest neighbour search techniques that scale well with the size of the dataset. Once the encoder-decoder model is trained, all objects of the database are embedded by feeding them through the encoder network to obtain their representations in the embedding space. The object retrieval process is divided into database generation and retrieval as we explain now.

3.4.1 Database Generation. Database generation for retrieval using embedding space involves preparing a dataset of inputs and their corresponding feature vectors, which can be used to perform efficient retrieval of similar inputs. To generate the database, we use the encoder model and obtain the embedding vector for each mesh object. This comprises the database for our retrieval model. Each of the embeddings is a 256 length vector used for comparing to obtain retrieval results as explained further.

3.4.2 Retrieval. Whenever a query object is given, an embedding is generated by encoding the preprocessed query mesh model and searching the database to find objects closest to the query. The retrieved inputs are then sorted by their L2 distance to the preprocessed query input representation, and the top-ranked results are returned as the retrieval results. Similar inputs are represented as points that are close to each other in the embedding space , while dissimilar inputs are represented as points that are far apart as shown in Figure 6.

4 EXPERIMENTS AND RESULTS

4.1 Dataset

The dataset used in this paper for 3D object retrieval is ModelNet40 [77]. It contains 12,311 labelled 3D mesh objects of 40 common categories. All the objects in the original dataset are in Object File Format (OFF). The number of faces in each of the mesh can go as high as 1,451,244 faces. This can lead to high computational complexity. Therefore, ModelNet40 is preprocessed to contain 512

triangles as mentioned in Section 3.1. The objects which do not decimate to n = 512 faces are discarded. Omitting these meshes, the dataset used for our retrieval system had 11,920 meshes. The train-test split of the dataset is 83%-17% and the individual class-split is given in the supplementary material.

We also evaluate our method on 569 objects from Shapenet [17]. This set of 569 objects from the entire dataset are generated using consistent preprocessing methods, yielding watertight models. These objects are further decimated to 512 faces without the need for any additional pre-processing steps. Objects from both ModelNet and Shapenet have consistent ordering of triangles among themselves.

4.2 Implementation Details

Train set of ModelNet40, as elaborated in Section 4.1, is used for training the model for database generation for 150 epochs. The model is trained with stochastic gradient descent and multi-step learning rate scheduler for a learning rate of 0.05 and batch size of 8. The loss used in the model is L2 loss with ReLU activation function on each layer. The model size is 1.1GB when trained on Nvidia RTX 3090. Once trained, the embeddings of all the database objects are saved for query retrieval. For the experiments during random rotation, the input is randomly rotated within a given range of angles whereas the target is the object in its original orientation.

During object retrieval process, a query object is passed through the trained encoder which generates an embedding of the query object for further comparison and retrieval. Top-k (generally, k=5, 10) mAP scores are generated based on ranked retrieval of the object.

4.3 Results

In object retrieval, a query object is the input object for which the user needs potential matches. The retrieved object instances from the model are ranked based on their similarity to the query object and the top-ranked objects are returned as the results. We evaluate our method on ModelNet40 test dataset as detailed in Section 4.1, unless mentioned otherwise. The quantitative and qualitative validation of the experiments are described below. See the supplementary material for more results and analysis.

4.3.1 Quantitative validation. Table 1 shows performance of various methods of 3D shape classification and retrieval. Our method shows better performance against the state-of-the-art methods for retrieval. All other methods are supervised on object class labels. On the contrary, our method is completely self-supervised. The mean average precision (mAP) score obtained on retrieving the top-10 results by our model for the ModelNet40 test dataset is 87.88% on top-10 and 94.87% on top-5 results. We also evaluate our method on objects from other than ModelNet40. We evaluate on 569, 3D mesh objects from ShapeNet [17] and achieve an mAP score of 80.46% for top-10 and 91.28% for top-5 retrieval matches. Table 2 shows the performance of our proposed model while learning on varying number of epochs and random rotation as explained in Section 4.2.

4.3.2 Qualitative validation. One of the crucial characteristics of our method is that it has the capability to understand the shape of 3D objects as demonstrated in Figure 5. This is the visual result of learning the embeddings to understand the structure and shape of the query objects. For more analysis on the retrieved query results, please refer the supplementary material.

To get the class-wise performance of our method, confusion matrix and t-distributed stochastic neighbor embedding (tSNE) clusters are analyzed. tSNE clusters of all the object embeddings from the test set are generated to visualize the embeddings generated by our model. As shown in Figure 6, the individual classes form distinctive clusters. A blob is seen in each of the class clusters while some of the embeddings are scattered around. This is because every class of objects have majority of objects with similar shapes while there are outliers which denote the uniqueness in the shape of a few objects. The confusion matrix demonstrates the retrieved results against the class labels of the query objects as shown in Figure 7. The order of

Method	Modality	Classification Accuracy (%)	mAP (%)
Methods that need supervision			
LFD[18]	Voxels	75.5	40.9
MVCNN[68]	Multiview	90.1	80.2
3D ShapeNets[77]	Voxels	77.3	49.2
GIFT[10]	Multiview	-	81.9
PointNet[58]	Points	89.2	-
PointNet++[59]	Points	90.7	-
PointCNN[46]	Points	91.8	-
PVNet[84]	Points	93.2	88.5
MLVCNN[39]	Multiview	94.2	92.2
DensePoint[50]	Points	93.2	88.5
MeshNet[28]	Mesh	88.9	81.9
Meshnet++[67]	Mesh	91.6	-
MVTN[31]	Multiview	93.8	92.9
MeshMAE[47]	Mesh	92.5	-
Methods that need no supervision			
Ours Top-10	Mesh	-	87.88
Ours Top-5	**Mesh**	**-**	**94.87**

Table 1: Comparison of overall classification accuracy and retrieval results (mAP scores) for different methods on ModelNet40. Our method is the only one that is self-supervised with state-of-the-art methods, but performs on par or better.

Experiment		mAP	
		Top-5	Top-10
Random Rotation	0	0.9487	0.8788
	10	0.9499	0.8776
	20	0.9518	0.8817
	30	0.9516	0.8825
Number of epochs	50	0.9398	0.8702
	100	0.9453	0.8757
	150	0.9487	0.8788

Table 2: Results of our method on Modelnet40 train for random rotations of objects upto the given angles (top) and for different training iterations (bottom)

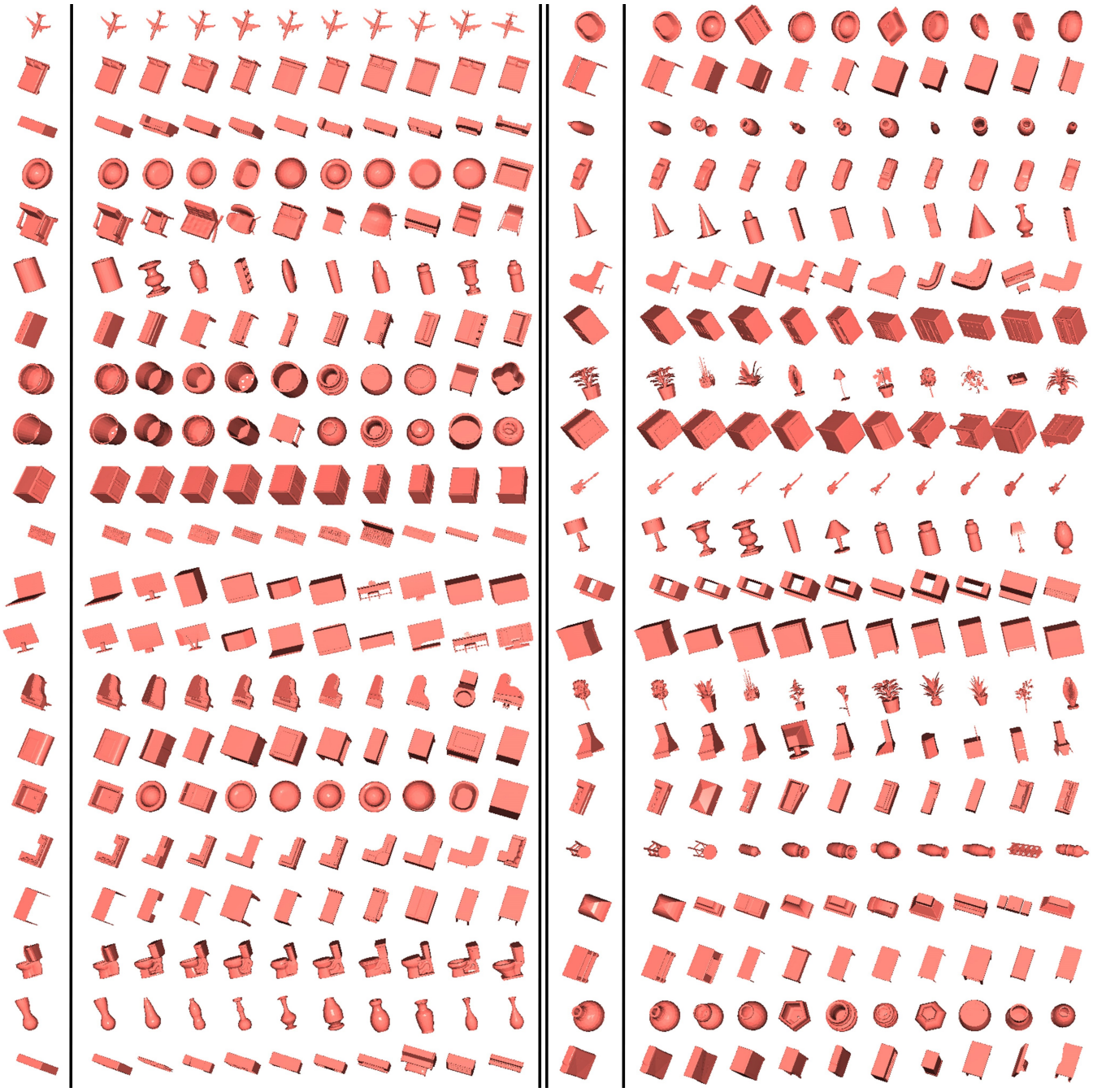

Figure 5: Retrieval results for different query objects. For each query object (left column), the top 10 closest objects, ranked from left to right, from the database are shown to its right. Left and right halves show 22 different queries each. The models are shown in top view. More results are in the supplementary document.

class labels is sorted based on normalized retrieved top-10 results of the same class. This reflects the capability of our method to learn about different classes in the absence of their respective class labels.

For example, in the tSNE clusters, furniture objects like sofa, table, night stand, mantel, TV stand, bookshelf overlap with each other due to their structural similarity. Similarly, flower pot, vase and bottle overlap with each other. In the confusion matrix, apart from objects matching their own class, it is seen that they also match visually similar objects. For example, the next top object class label

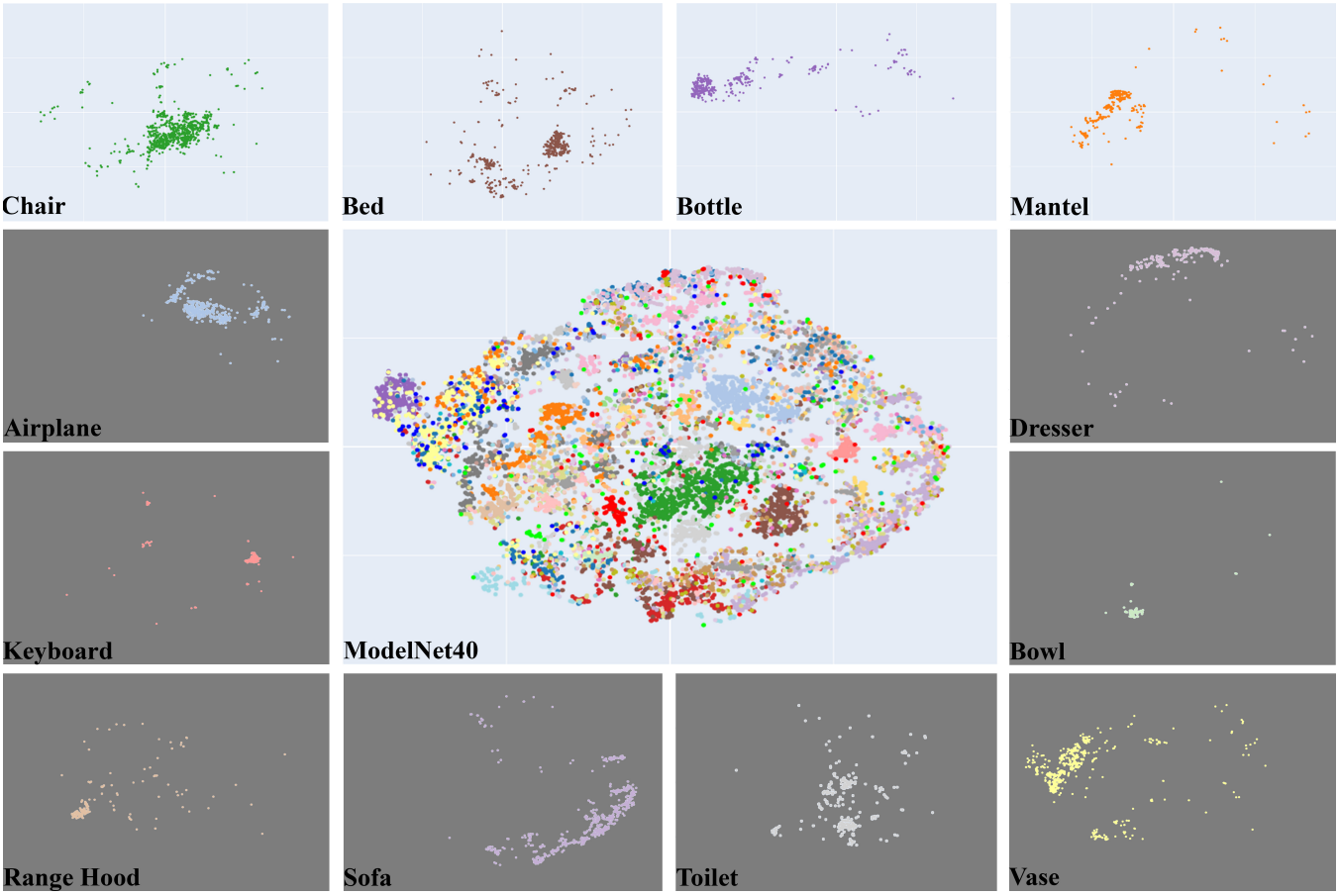

Figure 6: tSNE Visualization of the embedding space (centre). We also show clusters of several object classes for clarity in the periphery. Please note our system is self-supervised and doesn't use the class labels. Objects of the same class seem to be close and separated from other classes. For better visualization, the background colors have been adjusted accordingly.

for flowerpot is a vase. As a result, our method learns the shape, class and mainly, visual similarity between different objects.

4.4 Ablation Study

Our input representation contains face center, despite of them being derived from face corners, as they provide better understanding for the overall spatial structure of the meshes. Apart from centers, corners, and normals of the mesh faces, MeshNet++ [67] also include vectors to points in the neighbouring faces. Our model being a self-supervised encoder-decoder for retrieval didn't learn the embeddings well when the neighbour vectors were provided as backed by the results shown in Table 3. The encoder-decoder architecture tries to learn the entire input given by adjusting the weights according to back-propagation. When we include neighbour vectors into our input representation, the model gets confused as the neighbour vectors occupy a larger portion of the input representation. This is because the positional information of a face like centers, corners and normals give more information about the structure of a mesh as compared to it's neighbour vectors. From the experimentation results mentioned in Table 3, mesh decimation to 512 faces is chosen on the basis of mAP score. If we consider number of faces more than 512, results are comparable with the ones having 512 faces. While meshes containing number of faces less than 512 decimate a lot which become unnaturally coarse and hence cannot be taken into account for complex objects.

In our experiment, we aim to test the invariance of our method towards the sequence of the input vector, which represents the faces in a given mesh. It is well-known that Multi-Layer Perceptrons (MLPs) are not inherently invariant to the order of the input elements. Since a mesh is designed to optimize the graphics pipeline, the arrangement of faces ensures that front faces are rasterized first, and back-face culling can be easily implemented. To assess permutation invariance, we conducted controlled permutations of the input vector in two ways: circular rolling of the rows and patch-wise shuffling of the rows. The results of these experiments are presented in Table 4. Upon analyzing the results, we observed that our method demonstrates a certain degree of permutation invariance. However, it is essential to note that complete invariance might not be achieved, as the nature of MLPs inherently retains some sensitivity to input sequence variations. Nonetheless, our approach shows promising results and can

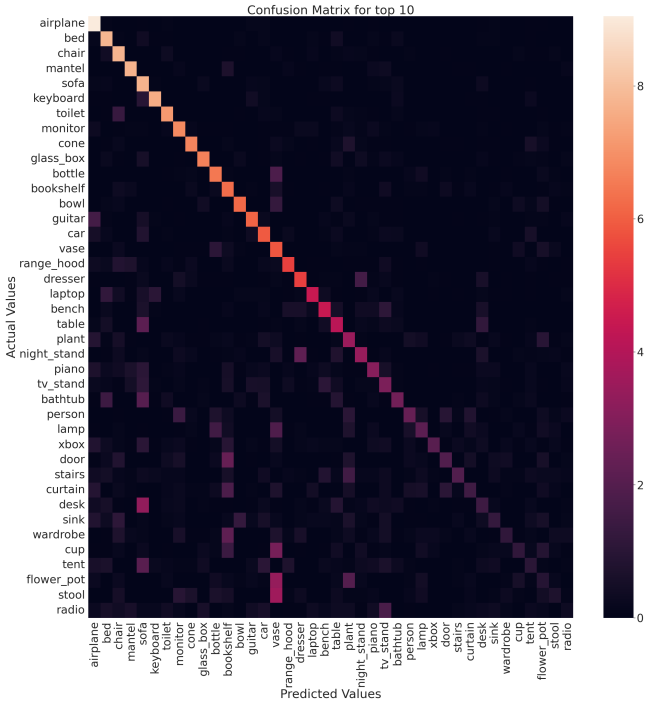

Figure 7: Confusion Matrix for the top-10 retrieval of the 40 classes of ModelNet40. The classes like flower pot, stool, and radio have fewer samples in the dataset.

handle input vector permutations to a significant extent. Experiments were also performed by changing various parameters as mentioned in Table 3. As a result, we settled on a 256-dimensional feature without neighbour triangles with objects represented using 512 faces for the best retrieval results.

4.5 Limitations and Future Work

Our method works directly on triangle-meshes, the most popular 3D object representation scheme. We currently decimate objects to 512 triangles primarily due to memory constraints on the GPU. This works well and captures the broad shape as is desirable for the problem but may not capture some of the fine details. Also, creating water-tight meshes with 512 triangles may be challenging. A drawback of our method is its limited robustness to triangle permutations. Mesh representation is permutation invariant in theory. Objects created using popular tools.however, do follow a scheme to order its triangles which is critical to the performance of our simple method. However, more work needs too be carried out to bring explicit permutation invariance to our method. We will explore pre-procesing schemes that reduce all meshes to a canonical ordering for our method to work well.

Our method is trained on mostly rigid objects as the standard datasets consist of them overwhelmingly. It will be interesting to explore how our simple scheme performs on non-rigid objects. A specific, simple case will be articulated objects. How does our embedding capture a mesh-represented object that are in two different poses? Can we bring pose invariance to the embedding using additional training?

Experiment		mAP	
		Top-5	Top-10
Number of faces	**512**	**0.9487**	**0.8788**
	768	0.9484	0.8752
	1024	0.9466	0.8734
Embedding Vector Dimension	128	0.9463	0.8591
	256	**0.9478**	**0.8621**
	512	0.9476	0.8635
	1024	0.9471	0.8641
Input Parameters	No centers	0.9472	0.8641
	No corners	0.9457	0.8591
	No normals	0.9461	0.8598
	No neighbours	**0.9501**	**0.8808**
	1-ring Neighbours	0.9469	0.8688
	2-ring Neighbours	0.9471	0.8641

Table 3: Results of various ablation studies. 1024 faces, 2-ring neighbours, 1024 dimensional embedding, centers, corners, and normal are used unless otherwise mentioned. Based on these results, we settle on 512 faced meshes, 256 dimensions for embedding and no neighbours to represent each triangle.

Experiment		mAP	
		Top-5	Top-10
Circular Rotation by n rows	0	0.9488	0.8792
	2	0.9511	0.8769
	4	0.9462	0.8746
	6	0.9283	0.8562
	8	0.8665	0.8070
Patch-wise shuffle of n rows	1	0.9488	0.8792
	2	0.9484	0.8799
	4	0.9485	0.8781
	8	0.9480	0.8776
	16	0.9499	0.8781
	32	0.9433	0.8714

Table 4: Results on Modelnet40 circular rotation and patch-wise shuffle of the input vector for limited permutation invariance.

5 CONCLUSIONS

Current 3D object retrieval methods rely on class labels in both model-based and view-based retrievals. We propose a surprisingly simple self-supervised 3D mesh object retrieval method that achieves an mAP score of 87.88% for the top 10 retrieved results and 94.87% for the top 5 on the ModelNet40 dataset. Our method retrieves on the basis of the shape of the object to get results that may not belong to the same category but look similar in shape using the embeddings generated by our model. Our method is simple but is effective at capturing inherent similarity in shape, as demonstrated by the results shown in Figure 5.

REFERENCES

[1] [n.d.]. 3D Warehouse. https://3dwarehouse.sketchup.com/?hl=en.
[2] [n.d.]. GrabCad Community Library. https://grabcad.com/library.
[3] [n.d.]. Sketchfab 3D Models Store. https://sketchfab.com/store/3d-models.
[4] [n.d.]. Turbosquid by Shutterstock. https://www.turbosquid.com/.
[5] [n.d.]. UltiMaker Thingiverse. https://www.thingiverse.com/.
[6] Amin Abolghasemi, Suzan Verberne, and Leif Azzopardi. 2022. Improving BERT-based query-by-document retrieval with multi-task optimization. In *Advances in Information Retrieval: 44th European Conference on IR Research, ECIR 2022, Stavanger, Norway, April 10–14, 2022, Proceedings, Part II*. Springer, 3–12.
[7] Ashutosh Adhikari, Achyudh Ram, Raphael Tang, and Jimmy Lin. 2019. Docbert: Bert for document classification. *arXiv preprint arXiv:1904.08398* (2019).
[8] Gianni Amati and Cornelis Joost Van Rijsbergen. 2002. Probabilistic models of information retrieval based on measuring the divergence from randomness. *ACM Transactions on Information Systems (TOIS)* 20, 4 (2002), 357–389.
[9] Alexei Baevski, Yuhao Zhou, Abdelrahman Mohamed, and Michael Auli. 2020. wav2vec 2.0: A framework for self-supervised learning of speech representations. *Advances in neural information processing systems* 33 (2020), 12449–12460.
[10] Song Bai, Xiang Bai, Zhichao Zhou, Zhaoxiang Zhang, and Longin Jan Latecki. 2016. Gift: A real-time and scalable 3d shape search engine. In *Proceedings of the IEEE conference on computer vision and pattern recognition*. 5023–5032.
[11] Aditya Bharti, NB Vineeth, and CV Jawahar. 2020. Few shot learning with no labels. *arXiv preprint arXiv:2012.13751* (2020).
[12] Sergey Brin and Lawrence Page. 1998. The anatomy of a large-scale hypertextual web search engine. *Computer networks and ISDN systems* 30, 1-7 (1998), 107–117.
[13] Alexander M Bronstein, Michael M Bronstein, Leonidas J Guibas, and Maks Ovsjanikov. 2011. Shape google: Geometric words and expressions for invariant shape retrieval. *ACM Transactions on Graphics (TOG)* 30, 1 (2011), 1–20.
[14] Michael M Bronstein and Iasonas Kokkinos. 2010. Scale-invariant heat kernel signatures for non-rigid shape recognition. In *2010 IEEE computer society conference on computer vision and pattern recognition*. IEEE, 1704–1711.
[15] Chris Burges, Tal Shaked, Erin Renshaw, Ari Lazier, Matt Deeds, Nicole Hamilton, and Greg Hullender. 2005. Learning to rank using gradient descent. In *Proceedings of the 22nd international conference on Machine learning*. 89–96.
[16] Mathilde Caron, Ishan Misra, Julien Mairal, Priya Goyal, Piotr Bojanowski, and Armand Joulin. 2020. Unsupervised learning of visual features by contrasting cluster assignments. *Advances in neural information processing systems* 33 (2020), 9912–9924.
[17] Angel X. Chang, Thomas Funkhouser, Leonidas Guibas, Pat Hanrahan, Qixing Huang, Zimo Li, Silvio Savarese, Manolis Savva, Shuran Song, Hao Su, Jianxiong Xiao, Li Yi, and Fisher Yu. 2015. *ShapeNet: An Information-Rich 3D Model Repository*. Technical Report arXiv:1512.03012 [cs.GR]. Stanford University — Princeton University — Toyota Technological Institute at Chicago.
[18] Ding-Yun Chen, Xiao-Pei Tian, Yu-Te Shen, and Ming Ouhyoung. 2003. On visual similarity based 3D model retrieval. In *Computer graphics forum*, Vol. 22. Wiley Online Library, 223–232.
[19] Haolan Chen, Shitong Luo, Xiang Gao, and Wei Hu. 2021. Unsupervised learning of geometric sampling invariant representations for 3d point clouds. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*. 893–903.
[20] Siheng Chen, Chaojing Duan, Yaoqing Yang, Duanshun Li, Chen Feng, and Dong Tian. 2019. Deep unsupervised learning of 3D point clouds via graph topology inference and filtering. *IEEE transactions on image processing* 29 (2019), 3183–3198.
[21] Ting Chen, Simon Kornblith, Mohammad Norouzi, and Geoffrey Hinton. 2020. A simple framework for contrastive learning of visual representations. In *International conference on machine learning*. PMLR, 1597–1607.
[22] Kevin Clark, Minh-Thang Luong, Quoc V Le, and Christopher D Manning. 2020. Electra: Pre-training text encoders as discriminators rather than generators. *arXiv preprint arXiv:2003.10555* (2020).
[23] Andrew M Dai, Christopher Olah, and Quoc V Le. 2015. Document embedding with paragraph vectors. *arXiv preprint arXiv:1507.07998* (2015).
[24] Matt Deitke, Dustin Schwenk, Jordi Salvador, Luca Weihs, Oscar Michel, Eli VanderBilt, Ludwig Schmidt, Kiana Ehsani, Aniruddha Kembhavi, and Ali Farhadi. 2023. Objaverse: A universe of annotated 3d objects. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*. 13142–13153.
[25] David L Donoho et al. 2000. High-dimensional data analysis: The curses and blessings of dimensionality. *AMS math challenges lecture* 1, 2000 (2000), 32.
[26] Susan T Dumais et al. 2004. Latent semantic analysis. *Annu. Rev. Inf. Sci. Technol.* 38, 1 (2004), 188–230.
[27] Mathias Eitz, Ronald Richter, Tamy Boubekeur, Kristian Hildebrand, and Marc Alexa. 2012. Sketch-based shape retrieval. *ACM Transactions on graphics (TOG)* 31, 4 (2012), 1–10.
[28] Yutong Feng, Yifan Feng, Haoxuan You, Xibin Zhao, and Yue Gao. 2019. Meshnet: Mesh neural network for 3d shape representation. In *Proceedings of the AAAI conference on artificial intelligence*, Vol. 33. 8279–8286.
[29] Zan Gao, Haixin Xue, and Shaohua Wan. 2020. Multiple discrimination and pairwise CNN for view-based 3D object retrieval. *Neural Networks* 125 (2020), 290–302.
[30] Jean-Bastien Grill, Florian Strub, Florent Altché, Corentin Tallec, Pierre Richemond, Elena Buchatskaya, Carl Doersch, Bernardo Avila Pires, Zhaohan Guo, Mohammad Gheshlaghi Azar, et al. 2020. Bootstrap your own latent-a new approach to self-supervised learning. *Advances in neural information processing systems* 33 (2020), 21271–21284.
[31] Abdullah Hamdi, Silvio Giancola, and Bernard Ghanem. 2021. Mvtn: Multi-view transformation network for 3d shape recognition. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*. 1–11.
[32] Zhizhong Han, Zhenbao Liu, Chi-Man Vong, Yu-Shen Liu, Shuhui Bu, Junwei Han, and CL Philip Chen. 2018. Deep spatiality: Unsupervised learning of spatially-enhanced global and local 3D features by deep neural network with coupled softmax. *IEEE Transactions on Image Processing* 27, 6 (2018), 3049–3063.
[33] Rana Hanocka, Amir Hertz, Noa Fish, Raja Giryes, Shachar Fleishman, and Daniel Cohen-Or. 2019. Meshcnn: a network with an edge. *ACM Transactions on Graphics (TOG)* 38, 4 (2019), 1–12.
[34] Kaiming He, Haoqi Fan, Yuxin Wu, Saining Xie, and Ross Girshick. 2020. Momentum contrast for unsupervised visual representation learning. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*. 9729–9738.
[35] Xinwei He, Tengteng Huang, Song Bai, and Xiang Bai. 2019. View n-gram network for 3d object retrieval. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*. 7515–7524.
[36] R Devon Hjelm, Alex Fedorov, Samuel Lavoie-Marchildon, Karan Grewal, Phil Bachman, Adam Trischler, and Yoshua Bengio. 2018. Learning deep representations by mutual information estimation and maximization. *arXiv preprint arXiv:1808.06670* (2018).
[37] Jingwei Huang, Hao Su, and Leonidas Guibas. 2018. Robust watertight manifold surface generation method for shapenet models. *arXiv preprint arXiv:1802.01698* (2018).
[38] Hervé Jégou, Matthijs Douze, Cordelia Schmid, and Patrick Pérez. 2010. Aggregating local descriptors into a compact image representation. In *2010 IEEE computer society conference on computer vision and pattern recognition*. IEEE, 3304–3311.
[39] Jianwen Jiang, Di Bao, Ziqiang Chen, Xibin Zhao, and Yue Gao. 2019. MLVCNN: Multi-loop-view convolutional neural network for 3D shape retrieval. In *Proceedings of the AAAI Conference on Artificial Intelligence*, Vol. 33. 8513–8520.
[40] Andrew E Johnson. 1997. Spin-images: a representation for 3-D surface matching. (1997).
[41] Seonggyeom Kim and Dong-Kyu Chae. 2022. ExMeshCNN: An Explainable Convolutional Neural Network Architecture for 3D Shape Analysis. In *Proceedings of the 28th ACM SIGKDD Conference on Knowledge Discovery and Data Mining*. 795–803.
[42] Jan Knopp, Mukta Prasad, Geert Willems, Radu Timofte, and Luc Van Gool. 2010. Hough transform and 3D SURF for robust three dimensional classification. In *Computer Vision–ECCV 2010: 11th European Conference on Computer Vision, Heraklion, Crete, Greece, September 5-11, 2010, Proceedings, Part VI 11*. Springer, 589–602.
[43] Guillaume Lavoué and Christian Wolf. 2008. Markov Random Fields for Improving 3D Mesh Analysis and Segmentation.. In *3DOR@ Eurographics*. 25–32.
[44] Huan Lei, Naveed Akhtar, and Ajmal Mian. 2021. Picasso: A CUDA-based library for deep learning over 3d meshes. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*. 13854–13864.
[45] Bo Li, Yijuan Lu, Azeem Ghumman, Bradley Strylowski, Mario Gutierrez, Safiyah Sadiq, Scott Forster, Natacha Feola, and Travis Bugerin. 2015. 3D sketch-based 3D model retrieval. In *Proceedings of the 5th ACM on International Conference on Multimedia Retrieval*. 555–558.
[46] Yangyan Li, Rui Bu, Mingchao Sun, Wei Wu, Xinhan Di, and Baoquan Chen. 2018. Pointcnn: Convolution on x-transformed points. *Advances in neural information processing systems* 31 (2018).
[47] Yaqian Liang, Shanshan Zhao, Baosheng Yu, Jing Zhang, and Fazhi He. 2022. MeshMAE: Masked Autoencoders for 3D Mesh Data Analysis. In *Computer Vision–ECCV 2022: 17th European Conference, Tel Aviv, Israel, October 23–27, 2022, Proceedings, Part III*. Springer, 37–54.
[48] Minghua Liu, Ruoxi Shi, Kaiming Kuang, Yinhao Zhu, Xuanlin Li, Shizhong Han, Hong Cai, Fatih Porikli, and Hao Su. 2023. OpenShape: Scaling Up 3D Shape Representation Towards Open-World Understanding.
[49] Xinhai Liu, Zhizhong Han, Xin Wen, Yu-Shen Liu, and Matthias Zwicker. 2019. L2g auto-encoder: Understanding point clouds by local-to-global reconstruction with hierarchical self-attention. In *Proceedings of the 27th ACM International Conference on Multimedia*. 989–997.
[50] Yongcheng Liu, Bin Fan, Gaofeng Meng, Jiwen Lu, Shiming Xiang, and Chunhong Pan. 2019. Densepoint: Learning densely contextual representation for efficient point cloud processing. In *Proceedings of the IEEE/CVF international conference on computer vision*. 5239–5248.

[51] William E Lorensen and Harvey E Cline. 1987. Marching cubes: A high resolution 3D surface construction algorithm. *ACM siggraph computer graphics* 21, 4 (1987), 163–169.

[52] Bharadwaj Manda, Shubham Dhayarkar, Sai Mitheran, VK Viekash, and Ramanathan Muthuganapathy. 2021. 'CADSketchNet'-An annotated sketch dataset for 3D CAD model retrieval with deep neural networks. *Computers & Graphics* 99 (2021), 100–113.

[53] Ben Mildenhall, Pratul P Srinivasan, Matthew Tancik, Jonathan T Barron, Ravi Ramamoorthi, and Ren Ng. 2021. Nerf: Representing scenes as neural radiance fields for view synthesis. *Commun. ACM* 65, 1 (2021), 99–106.

[54] Aaron van den Oord, Yazhe Li, and Oriol Vinyals. 2018. Representation learning with contrastive predictive coding. *arXiv preprint arXiv:1807.03748* (2018).

[55] Jeong Joon Park, Peter Florence, Julian Straub, Richard Newcombe, and Steven Lovegrove. 2019. Deepsdf: Learning continuous signed distance functions for shape representation. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*. 165–174.

[56] Florent Perronnin, Jorge Sánchez, and Thomas Mensink. 2010. Improving the fisher kernel for large-scale image classification. In *Computer Vision–ECCV 2010: 11th European Conference on Computer Vision, Heraklion, Crete, Greece, September 5-11, 2010, Proceedings, Part IV 11*. Springer, 143–156.

[57] Anran Qi, Yi-Zhe Song, and Tao Xiang. 2018. Semantic Embedding for Sketch-Based 3D Shape Retrieval.. In *BMVC*, Vol. 3. 11–12.

[58] Charles R Qi, Hao Su, Kaichun Mo, and Leonidas J Guibas. 2017. Pointnet: Deep learning on point sets for 3d classification and segmentation. In *Proceedings of the IEEE conference on computer vision and pattern recognition*. 652–660.

[59] Charles Ruizhongtai Qi, Li Yi, Hao Su, and Leonidas J Guibas. 2017. Pointnet++: Deep hierarchical feature learning on point sets in a metric space. *Advances in neural information processing systems* 30 (2017).

[60] Filip Radenović, Giorgos Tolias, and Ondřej Chum. 2018. Fine-tuning CNN image retrieval with no human annotation. *IEEE transactions on pattern analysis and machine intelligence* 41, 7 (2018), 1655–1668.

[61] Stephen Robertson, Hugo Zaragoza, et al. 2009. The probabilistic relevance framework: BM25 and beyond. *Foundations and Trends® in Information Retrieval* 3, 4 (2009), 333–389.

[62] Paul Scovanner, Saad Ali, and Mubarak Shah. 2007. A 3-dimensional sift descriptor and its application to action recognition. In *Proceedings of the 15th ACM international conference on Multimedia*. 357–360.

[63] Avinash Sharma, Radu Horaud, Jan Cech, and Edmond Boyer. 2011. Topologically-robust 3D shape matching based on diffusion geometry and seed growing. In *CVPR 2011*. 2481–2488. https://doi.org/10.1109/CVPR.2011.5995455

[64] Charu Sharma and Manohar Kaul. 2020. Self-supervised few-shot learning on point clouds. *Advances in Neural Information Processing Systems* 33 (2020), 7212–7221.

[65] Ravi Shekhar and CV Jawahar. 2012. Word image retrieval using bag of visual words. In *2012 10th IAPR International Workshop on Document Analysis Systems*. IEEE, 297–301.

[66] Yi Shi, Mengchen Xu, Shuaihang Yuan, and Yi Fang. 2020. Unsupervised deep shape descriptor with point distribution learning. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*. 9353–9362.

[67] Vinit Veerendraveer Singh, Shivanand Venkanna Sheshappanavar, and Chandra Kambhamettu. 2021. MeshNet++: A Network with a Face.. In *ACM Multimedia*. 4883–4891.

[68] Hang Su, Subhransu Maji, Evangelos Kalogerakis, and Erik Learned-Miller. 2015. Multi-view convolutional neural networks for 3d shape recognition. In *Proceedings of the IEEE international conference on computer vision*. 945–953.

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

[70] Yaniv Taigman, Ming Yang, Marc'Aurelio Ranzato, and Lior Wolf. 2014. DeepFace: Closing the Gap to Human-Level Performance in Face Verification. In *2014 IEEE Conference on Computer Vision and Pattern Recognition*. 1701–1708. https://doi.org/10.1109/CVPR.2014.220

[71] Ali Thabet, Humam Alwassel, and Bernard Ghanem. 2020. Self-supervised learning of local features in 3d point clouds. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition workshops*. 938–939.

[72] Bart Iver van Blokland and Theoharis Theoharis. 2018. Microshapes: efficient querying of 3D object collections based on local shape. In *Proceedings of the 11th Eurographics Workshop on 3D Object Retrieval*. 9–16.

[73] Bart Iver van Blokland and Theoharis Theoharis. 2020. An indexing scheme and descriptor for 3D object retrieval based on local shape querying. *Computers & Graphics* 92 (2020), 55–66.

[74] Fang Wang, Le Kang, and Yi Li. 2015. Sketch-based 3d shape retrieval using convolutional neural networks. In *Proceedings of the IEEE conference on computer vision and pattern recognition*. 1875–1883.

[75] Hong Wang, Xin Wang, Wenhan Xiong, Mo Yu, Xiaoxiao Guo, Shiyu Chang, and William Yang Wang. 2019. Self-supervised learning for contextualized extractive summarization. *arXiv preprint arXiv:1906.04466* (2019).

[76] Liwei Wang, Yin Li, and Svetlana Lazebnik. 2016. Learning deep structure-preserving image-text embeddings. In *Proceedings of the IEEE conference on computer vision and pattern recognition*. 5005–5013.

[77] Zhirong Wu, Shuran Song, Aditya Khosla, Fisher Yu, Linguang Zhang, Xiaoou Tang, and Jianxiong Xiao. 2015. 3d shapenets: A deep representation for volumetric shapes. In *Proceedings of the IEEE conference on computer vision and pattern recognition*. 1912–1920.

[78] Zhirong Wu, Yuanjun Xiong, Stella X Yu, and Dahua Lin. 2018. Unsupervised feature learning via non-parametric instance discrimination. In *Proceedings of the IEEE conference on computer vision and pattern recognition*. 3733–3742.

[79] Yu Xiang, Wonhui Kim, Wei Chen, Jingwei Ji, Christopher Choy, Hao Su, Roozbeh Mottaghi, Leonidas Guibas, and Silvio Savarese. 2016. Objectnet3d: A large scale database for 3d object recognition. In *Computer Vision–ECCV 2016: 14th European Conference, Amsterdam, The Netherlands, October 11-14, 2016, Proceedings, Part VIII 14*. Springer, 160–176.

[80] Jin Xie, Guoxian Dai, Fan Zhu, Edward K Wong, and Yi Fang. 2016. Deepshape: Deep-learned shape descriptor for 3d shape retrieval. *IEEE transactions on pattern analysis and machine intelligence* 39, 7 (2016), 1335–1345.

[81] Le Xue, Mingfei Gao, Chen Xing, Roberto Martín-Martín, Jiajun Wu, Caiming Xiong, Ran Xu, Juan Carlos Niebles, and Silvio Savarese. 2023. ULIP: Learning a Unified Representation of Language, Images, and Point Clouds for 3D Understanding. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*.

[82] Le Xue, Ning Yu, Shu Zhang, Junnan Li, Roberto Martín-Martín, Jiajun Wu, Caiming Xiong, Ran Xu, Juan Carlos Niebles, and Silvio Savarese. 2023. ULIP-2: Towards Scalable Multimodal Pre-training for 3D Understanding.

[83] Andrew Yates, Rodrigo Nogueira, and Jimmy Lin. 2021. Pretrained transformers for text ranking: BERT and beyond. In *Proceedings of the 14th ACM International Conference on web search and data mining*. 1154–1156.

[84] Haoxuan You, Yifan Feng, Rongrong Ji, and Yue Gao. 2018. Pvnet: A joint convolutional network of point cloud and multi-view for 3d shape recognition. In *Proceedings of the 26th ACM international conference on Multimedia*. 1310–1318.

[85] Zaiwei Zhang, Rohit Girdhar, Armand Joulin, and Ishan Misra. 2021. Self-supervised pretraining of 3d features on any point-cloud. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*. 10252–10263.