Deep Distance Metric Learning for Similarity Preserving Embedding of Point Clouds

Ahmed Abouelazm, Igor Vozniak, Nils Lipp, Pavel Astreika and Christian Mueller
Deutsches Forschungszentrum für Künstliche Intelligenz (DFKI), Saarbruecken, Germany
{*first name.last name*}*@dfki.de*

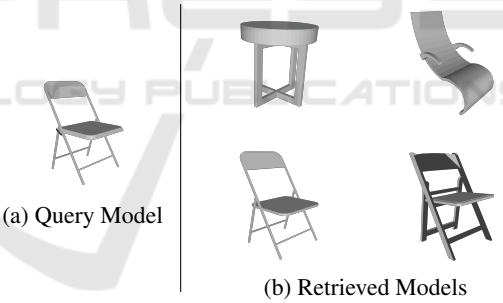
Keywords: Point Clouds, 3D Deep Learning, Distance Metric Learning, Similarity Preserving Embedding.

Abstract: Point cloud processing and 3D model retrieval methods have received a lot of interest as a result of the recent advancement in deep learning, computing hardware, and a wide range of available 3D sensors. Many state-of-the-art approaches utilize distance metric learning for solving the 3D model retrieval problem. However, the majority of these approaches disregard the variation in shape and properties of instances belonging to the same class known as intra-class variance, and focus on semantic labels as a measure of relevance. In this work, we present two novel loss functions for similarity-preserving point cloud embedding, in which the distance between point clouds in the embedding space is directly proportional to the ground truth distance between them using a similarity or distance measure. The building block of both loss functions is the forward passing of n-pair input point clouds through a Siamese network. We utilize ModelNet 10 dataset in the course of numerical evaluations under classification and mean average precision evaluation metrics. The reported quantitative and qualitative results demonstrate enhancement in retrieved models both quantitatively and qualitatively by a significant margin.

1 INTRODUCTION

Point Cloud Data (PCD) is a data representation obtained by LiDAR sensors, and is one of the most prominent 3D data structures. PCD depicts a scanned object as a set of discrete points scattered in a Euclidean space. Compared to 2D images, PCDs are substantially more resistant to changes in lighting conditions and accurately capture object depth. Due to the obvious inherent advantages of PCDs, they have been used in a variety of application domains such as robotics and autonomous driving. PCDs have been employed in a range of applications, including classification, scene segmentation, model retrieval, and reconstruction (Qi et al., 2017a; Qi et al., 2017b; Wang et al., 2019; Uy and Lee, 2018; Mescheder et al., 2019; Peng et al., 2020; Park et al., 2019).

In the scope of this work, we are particularly interested in classification and reconstruction applications of PCDs. The majority of these pipelines require efficient and meaningful encoding of the PCDs (Qi et al., 2017a; Mescheder et al., 2019). Encoding PCD into a feature space is thus a bottleneck of such approaches, motivating us to explore further into the resulting embedding and strategies to improve their quality. One of the most dominant methods for judging the quality of an embedding space is investigating the model retrieval problem over such a space (Hadsell et al., 2006). **Model retrieval** is a search problem in which a query model is given, and its most relevant counterparts are retrieved from a large-scale dataset. In deep learning, features are extracted from N training objects using an encoder architecture (Hadsell et al., 2006; Melekhov et al., 2016; Wang et al., 2014; Hoffer and Ailon, 2015). A data structure, e.g., kdtree, octree is used to fit these features into memory



(a) Query Model (b) Retrieved Models

Figure 1: Problem description: the row-wise retrieval of the query model on the left (a) is presented on the right (b). The upper row results in (b) are from the same class but don't incorporate any visual similarity. In contrast, the bottom row in (b) accounts for intra-class similarity and is a desirable retrieval result. Note: in this work, a sampled point cloud from ModelNet 10 mesh models is used, whereas mesh models are depicted for explanatory purposes only.

Abouelazm, A., Vozniak, I., Lipp, N., Astreika, P. and Mueller, C.
Deep Distance Metric Learning for Similarity Preserving Embedding of Point Clouds.
DOI: 10.5220/0011627100003417
In *Proceedings of the 18th International Joint Conference on Computer Vision, Imaging and Computer Graphics Theory and Applications (VISIGRAPP 2023) - Volume 4: VISAPP*, pages 570-581
ISBN: 978-989-758-634-7; ISSN: 2184-4321

Figure 2: Sample of intra-class variance of the class chair based on ModelNet 10 dataset.

and a query database efficiently. When given a query model, the same network is used to extract a feature vector. A distance metric is then utilized to extract the most relevant objects from the feature database. However, relevance is commonly satisfied when the query object and the matched object are of the same semantic class (Qi et al., 2017a). Based on figure (1), we can conclude that this relevance metric is flawed because it ignores intra-class variation, resulting in an embedding space that disregards similarity between instances of the same class. Intra-class variance is the variance of similarity in structure, appearance, and properties between instances in the same class, an example of such variance on chairs is illustrated in figure (2). This shortcoming encouraged us to investigate the well-established distance metric learning approaches toward improving embedding space quality by accounting for intra-class variance. The ultimate target is to achieve a fine-grained model retrieval over a PCD dataset that takes structural similarity between instances in the same class into account.

The contribution of this work is threefold. **First**, the loosely defined retrieval relevance metric which merely takes the labels of the instances into account is replaced by a tighter metric which only accepts a retrieval if it belongs to the nearest neighbors of the query given a ground truth similarity measure. Chamfer distance is used as a similarity measure between PCDs in this work. **Second**, the deep distance metric learning is utilized for performing the learning directly on the feature space. **Finally**, we propose two novel loss functions, based on contrastive and triplet losses from the state-of-the-art literature (Hadsell et al., 2006; Wang et al., 2014; Hoffer and Ailon, 2015), that take into consideration both, inter- and more essentially intra-class variance, and clarify their training process inside the distance metric learning paradigm, to accomplish our fine-grained model retrieval goal. To the best of our knowledge, there is no instance of a loss function used for distance metric learning that explicitly accounts for intra-class variance prior to this work.

2 RELATED WORK

Model retrieval problem as a search problem is commonly solved by applying distance metric learning on a feature embedding space according to (Hadsell et al., 2006). Distance Metric Learning is a machine learning task that involves automatically inferring a discriminative similarity or distance measures over data instances as defined in (Xing et al., 2002). A discriminative measure has a small distance between similar objects and pushes different objects away from each other. This learning technique has been utilized in a wide range of applications, including nearest-neighbor models, clustering, dimensionality reduction, and model retrieval as will be clarified in this section. The first work on inferring such measures is presented in (Xing et al., 2002), where Mahalanobis distance and linear transformation are used as a distance metric as in equation (1). This approach was further extended in (Chatpatanasiri et al., 2010) by employing kernel learning to learn non-linear transformations over input data.

$$\begin{aligned} D(x_i,x_j) &= \sqrt{(x_i-x_j)^T M (x_i-x_j)} \\ &= \sqrt{(Lx_i-Lx_j)^T (Lx_i-Lx_j)} \end{aligned} \tag{1}$$

where x_i, x_j are two input data instances. M is a positive definite matrix which can be decomposed as $M = L^T L$ with L representing the linear transformation over the input data.

According to (Hadsell et al., 2006), classical distance metric learning has two major drawbacks: it requires meaningful and computable distance measures, and it does not provide an explicit function to map new input data whose relationship to the training data is unknown. Due to the aforementioned limitations, several attempts have been made to address distance metric learning within a deep learning framework. To accomplish distance metric learning, a network is trained to learn non-linear transformations over input data guided by a custom loss function. The majority of deep distance metric learning algorithms feed N instances of training to a neural network and compare between them using the aforementioned loss function.

(Hadsell et al., 2006) is a pioneer work in deep distance metric learning. This paper introduced *contrastive loss*, where the loss function has two training instances as an input. The loss function attempts to cluster instances from the same class in the feature space and separates them from instances belonging to different classes, as shown in equation (2). The input pairs are sampled at random and are used to learn in-

variant mapping achieving dimensionality reduction.

$$Contrastive\ Loss(x_i,x_j) = \frac{1}{2}\, y \times d_f(x_i,x_j)^2 + \frac{1}{2}\,(1-y) \times \left\{max(0, m - d_f(x_i,x_j))\right\}^2 \quad (2)$$

where x_i, x_j are the input training instance, y indicates whether x_i, x_j share the same label and m is the desired margin separating different classes.

In order to improve image-matching results, (Melekhov et al., 2016) employed contrastive loss. Finding matching images across large datasets is a key difficulty in many applications, including multi-view 3D reconstruction and image retrieval. Based on the feature vectors in Euclidean space embedded by a neural network trained via contrastive loss, this work discovers matching and non-matching pairs of images. In addition, a heuristic approach for determining the margin between classes is presented. SigNet (Dey et al., 2017) uses convolutional neural networks and contrastive loss to perform offline signature verification. A minor change to contrastive loss is proposed, which assigns different weights to the pull force between similar instances and the push force between dissimilar ones. When compared to state-of-the-art approaches, SigNet outperformed most of the benchmark signature datasets.

Triplet loss is introduced in (Wang et al., 2014; Hoffer and Ailon, 2015). The input of the triplet loss is extended to be a triplet. Positive-negative triplets are the focus of this loss function. A positive-negative triplet is one in which the first (anchor) and second (positive) members belong to the same class, but the third (negative) member belongs to a different class. Using this loss, the anchor and positive members are simultaneously drawn near each other while being pushed away from the third member, as formulated in equation (3). The first application in which this loss is utilized is image ranking in (Wang et al., 2014).

$$Triplet\ Loss(x_a,x_p,x_n) = \frac{1}{2}\{max(0, d_f(x_a,x_p) + m - d_f(x_a,x_n))\}^2 \quad (3)$$

where x_a, x_p, x_n are the anchor, positive and negative training instances respectively, and m stands for the desired margin separating different classes.

FaceNet (Schroff et al., 2015) achieves state-of-the-art facial recognition performance by combining triplet loss with an inception architecture (Szegedy et al., 2015). Face recognition is investigated by embedding face images in a Euclidean feature space, where face similarity may be measured directly. Using the L_2 norm, the feature space is confined within a d-dimension hyper-sphere. In addition, rather than mining triplets offline using network checkpoints, this work provides a unique technique for mining triplets online. A huge mini-batch of thousands of instances is generated, and the hardest positive-negative triplets are sampled. The hardest triplets are those in which the distance between the anchor and the positive instance is similar to the distance between the anchor and the negative instance. (Nazir et al., 2021) is concerned with utilizing triplet loss for reconstruction and completion challenges in PCD domain. In this work, deep distance metric learning is used to learn global features with adequate separation between different classes. To accomplish discriminative reconstruction of PCDs, the final loss function is a combination of reconstruction loss calculated by Chamfer distance and triplet loss.

In the literature, many variants based on contrastive and triplet losses have been suggested. To accomplish gait recognition, (Xu, 2021) proposes Deep Large Margin Nearest Neighbor (DLMNN) loss. Gait recognition is the challenge of identifying a far away human from their walking manner. The DLMNN loss is a linear combination of the triplet and contrastive losses controlled by weighting, γ as clarified in equation (4). The goal of such a loss function is to guarantee that similar examples are drawn as close to each other as possible, while also learning to distinguish between examples from different classes.

$$DLMNN\ Loss(x_a,x_p,x_n) = Triplet\,Loss(x_a,x_p,x_n) + \gamma \times Contrastive\,Loss(x_a,x_p) \quad (4)$$

PointNetVLAD (Uy and Lee, 2018) provides a novel pipeline for PCD-based retrieval for place recognition that utilizes PointNet and normalizes its outputs using NetVLAD (Arandjelovic et al., 2016). This work proposes two variants of the triplet loss: lazy triplet and quadruplet losses. Both losses sample N negative examples rather than sampling just one. These two loss functions have more stable training and faster convergence. PointNetVLAD is able to achieve state-of-the-art performance on its reference task. In (Wang et al., 2017), the angular loss is introduced as an alternative formulation of a triplet. This loss assumes a triangle, linking the input triplet (x_a, x_p, x_n) and constrains the angle at the triplet's negative member. Angular loss has two inherent advantages over triplet loss: scale invariance and robustness against feature variance. The results indicate that angular loss has faster convergence and outperforms triplet loss on image clustering and retrieval tasks.

Using a modified contrastive loss function, graph proximity loss as in (Bai et al., 2019) learns a graph-level representation. The purpose of this work is to encapsulate an entire graph into a vector space that takes graph-graph proximity into account. The mod-

ified contrastive loss pulls the distance between two training graphs to be identical to their ground truth distances, independent of which class each graph belongs to. To the best of our knowledge, this is the only loss function that explicitly considers intra-class variation. However, it does not provide any separation between different classes. This work achieves competitive results on similarity ranking, and visualization tasks by learning graph representation in unsupervised and inductive manners.

3 METHODOLOGY

3.1 Preliminaries

We establish a consistent mathematical notation that will be utilized throughout the rest of this paper. The introduced notation is as follows:

- Given a labeled dataset with M tuples where each tuple is a PCD and its label, the dataset is noted as $D = \{(x_i, y_i)\}_{i=1}^{M}$.
- A PCD containing N points, where each point is represented just by its coordinates, with no additional attributes taken into account, is denoted as $x_i = \{p_j\}_{j=1}^{N} \in \mathbb{R}^{N\times 3}$. Each PCD is sampled with the same number of points. The same preprocessing as introduced in (Qi et al., 2017a) is applied consistently over all PCDs.
- The discrete label assigned to each PCD x_i is indicated by y_i and can belong to a set of pre-defined labels $\in \{0, 1, ..., C\}$.
- A neural network ϕ parameterized by weights θ maps the input PCD into a Euclidean feature space such that $\phi : x_i \mapsto f_i$. The PCD embedding in the feature space is formulated as follows: $f_i = \phi(x_i; \theta) \in \mathbb{R}^d$.
- $\hat{d}_{i,j}$ is the L_2 distance between two PCDs (x_i, x_j) in the feature space, and it is formulated as follows: (5).

$$\hat{d}_{i,j} = \left\| f_i - f_j \right\|_2 = \left\| \phi(x_i; \theta) - \phi(x_j; \theta) \right\|_2 \quad (5)$$

- Finally, $d_{i,j}$ is the ground truth distance between two PCDs (x_i, x_j) based on a similarity or distance measure.

3.2 Modified Relevance Metric

In this work, we start with a modification of the relevance metric used in evaluating correct retrievals in model retrieval problems. We propose a tighter metric that only accepts retrievals to be correct if they belong to the nearest neighbors of the query object on a ground truth distance metric. The Chamfer distance is used as a ground truth distance metric on PCDs in this work. Chamfer distance is an asymmetric distance measure that has been widely utilized in the literature for comparing PCDs as in (Mescheder et al., 2019; Peng et al., 2020; Park et al., 2019). Equation (6) introduces the Chamfer distance between two PCDs (x_i, x_j). The first term in the equation iterates over each point $p \in x_i$ and calculates the average distance to their corresponding nearest point $q \in x_j$. The second term achieves the same, but with the roles of x_i and x_j are swapped.

$$\begin{aligned} CD(x_i, x_j) = &\frac{1}{|x_i|} \sum_{p\in x_i} \min_{q\in x_j} \|p - q\|_2^2 \\ &+ \frac{1}{|x_j|} \sum_{q\in x_j} \min_{p\in x_i} \|p - q\|_2^2 \end{aligned} \quad (6)$$

The suggested relevance metric necessitates the development of a loss function that accounts for intra-class variation, diversity in structure, and attributes of objects in the same class. To the best of our knowledge, graph proximity loss (Bai et al., 2019) is the pioneer loss function for including intra-class variation. The graph proximity loss takes an input pair (x_i, x_j) and minimizes the difference between their distance in the feature space $\hat{d}_{i,j}$ and their ground truth distance $d_{i,j}$, as shown in equation (7). This loss function is used for learning graph embeddings. Despite having desirable properties related to the fine-grained model retrieval that we are interested in, graph proximity loss is not directly applicable to our work because the loss setting assumes a single global class throughout the entire dataset and does not provide adequate separation between different classes, which is appropriate for graph similarity ranking on a macro level but not for our setting.

$$Graph\ Proximity\ Loss(x_i, x_j) = \frac{1}{2} \times (\hat{d}_{i,j} - d_{i,j})^2 \quad (7)$$

3.3 Proposed Pair Loss

We begin our proposed pair loss design by reviewing the basic contrastive loss formulation as provided in equation (2). Contrastive loss pulls pairs sampled from the same class while repelling different pairs with at least a hyperparameter margin, as simplified in equation (8). Equation (9) clearly shows that the second component in the loss function does not contribute unless the distance between two instances from different classes is smaller than the margin. This term is designed to simplify the training procedure and prevent the network from exerting ad-

ditional pushing force when different classes are adequately separated. Contrastive loss overlooks intra-class variance since it minimizes the distance between examples belonging to the same class regardless of their similarity. Thus, in order to include intra-class variation and accomplish fine-grained model retrieval, contrastive loss formulation must be modified.

$$Loss(x_i, x_j) = \frac{1}{2}\begin{Bmatrix} \hat{d}_{i,j}^2 & ,if\ y_i = y_j \\ max(0, m - \hat{d}_{i,j})^2 & ,if\ y_i \neq y_j \end{Bmatrix} \tag{8}$$

$$max(0, m - \hat{d}_{i,j}) = \begin{Bmatrix} zero & ,if\ \hat{d}_{i,j} \geq m \\ +ve & ,if\ \hat{d}_{i,j} < m \end{Bmatrix} \tag{9}$$

The proposed **intra-class pair loss** modifies contrastive loss, as formulated in equation (10). The first component is modified to minimize the square difference between the distance, in the embedding space, between a pair of (x_i, x_j) and their ground truth distance calculated by Chamfer distance, as in equation (6). The benefit of this improvement is the incorporation of intra-class variation by embedding PCDs in a Euclidean feature space that respects their similarity based on chamfer distance. The second term is maintained owing to its capacity to learn clear separation between different classes and regularization of its contribution depending on the hyperparameter margin.

$$ICPL(x_i, x_j) = \frac{1}{2}\begin{Bmatrix} (\hat{d}_{i,j} - d_{i,j})^2 & ,if\ y_i = y_j \\ max(0, m - \hat{d}_{i,j})^2 & ,if\ y_i \neq y_j \end{Bmatrix} \tag{10}$$

Figures (3, 4, 5) demonstrate the expected embedding of positive and negative PCD pairs using a siamese network and the proposed intra-class pair loss. A positive pair consists of PCDs from the same class, whereas a negative pair consists of PCDs from different classes. A siamese network is a number of copies of a network with identical shared weights. A visually similar positive pair (x_1, x_2) is shown in figure (3). A visually similar pair has a low Chamfer distance between its members, and the proposed loss function promotes the network to learn an embedding in which the distance between the members in the feature space is equal to their Chamfer distance.

The pair of PCDs (x_1, x_3) is passed to the embedding pipeline as shown in figure (4). Since x_1 and x_3 are visually dissimilar, the Chamfer distance between them is considerably greater than the distance between x_1 and x_2. As a result, the network strives to learn an embedding that is true to the ground truth distance guided by the proposed loss. As seen from Figures (3, 4), $\hat{d}_{1,2} < \hat{d}_{1,3}$, indicating that the constructed feature space takes into account intra-class variation.

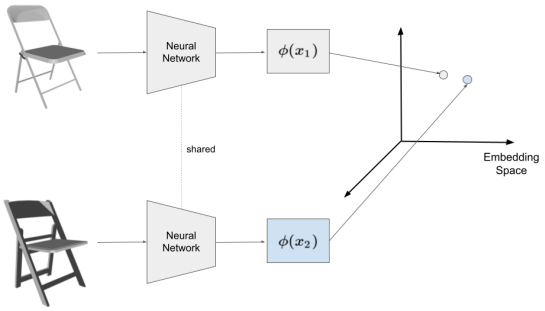


Figure 3: Visually similar positive pair (x_1, x_2) example to illustrate proposed intra-class pair loss behavior.

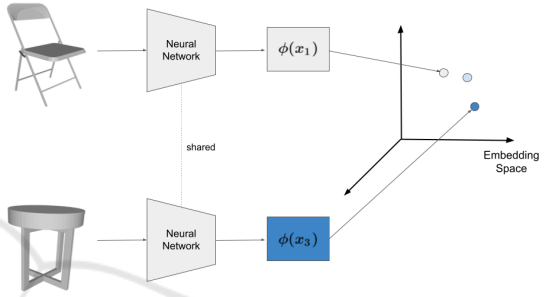


Figure 4: Visually dissimilar positive pair (x_1, x_3) example to illustrate proposed pair loss behavior.

Figure (5) illustrates an example in which the sampled pair is a negative one, because x_1 belongs to the class chair and x_4 belongs to the class toilet. Despite the fact that x_1 and x_4 are visually similar and share some geometric properties, the second term in Equation (8) forces the network to push their embeddings apart from each other by at least the value of the margin. As a result, our loss function instructs the network to learn a reasonable separation between different classes, even if they share similarities.

The feature vector extracted from a PCD x_i is normalized using the L_2 norm, such that $\|f_i\|_2 = 1$. This normalization is initially introduced in (Schroff et al., 2015) to constrain the learned feature vectors within a d-dimensional hyper-sphere. The advantage of using the L_2 norm is that the squared Euclidean distance between two PCDs is confined between $[0, 4]$, making margin selection easier and more meaningful. The approach given in (Melekhov et al., 2016) is utilized to automatically set the margin value to achieve appropriate separation between classes and more efficient training. The margin value is specified to be twice the average distance between PCD pairs computed via random network initialization.

Dataset D has $\binom{M}{2}$ pairs. Choosing pairs that contribute to the loss is critical for network training to achieve fast convergence. Positive pairs are not an

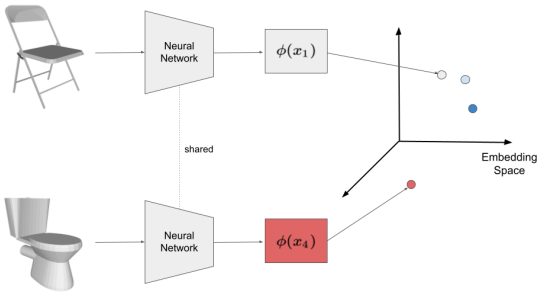


Figure 5: Negative pair (x_1, x_4) example to illustrate proposed intra-class pair loss behavior.

issue because they all contribute to the loss; however, sampling negative pairs that break the constraint, called hard negative pairs, is a challenge. Hard negative pairs are pairs that have members from different classes and have a distance in the feature space that is smaller than the margin value. It is infeasible to sample these pairs over the whole dataset at each training stage. Thus, we adopt the online sampling technique introduced in (Schroff et al., 2015). First, a balanced mini-batch is created with k samples from each class. The intention behind a balanced mini-batch is to have a minimum number of examples from each class in each mini-batch. Using the siamese network of choice, the sampled mini-batch is encoded into the feature space. All positive pairs in the mini-batch are generated, and an equal number of the hardest (nearest) negative pairs are sampled. Even though online sampling requires a high computational effort, it is preferable to offline sampling. Since offline sampling uses a stationary network checkpoint and the sampled pairs may become outdated and cease to contribute to the loss when the current network weights change.

An online sampling strategy by itself is insufficient for efficient network training. Since sampling PCDs presents a problem in that their mini-batch size is limited when compared to work done on images owing to PCDs high memory and computation requirements. In this work, for example, just a hundred samples can be packed into a mini-batch compared to a mini-batch containing 1800 images used in (Schroff et al., 2015). Due to the small mini-batch size, only a few negative pairs contribute to the loss by breaching the margin constraint as training advances, resulting in slower network convergence. Thus, we modified our final loss to be a linear combination of cross entropy loss and the proposed pair loss, with α and γ regulating the trade-off between them as seen in equation (11). By including the cross entropy loss into the proposed loss function, the network is able to learn a decision hyper-plane that compensates for the limited mini-batch size while also improving the discriminative ability of the network as proposed in (Wen et al., 2016; Em et al., 2017).

$$\begin{aligned} Total\ Loss = \alpha \times Cross\ Entropy\ Loss + \\ \gamma \times Intra\ Class\ Pair\ Loss \end{aligned} \tag{11}$$

3.4 Proposed Triplet Loss

(Schroff et al., 2015) introduced the triplet loss, which is used to compare the embedding of three input instances within deep distance metric learning frameworks. The formulation of triplet loss, which only accepts a positive - negative triplet (x_a, x_p, x_n) as an input, is clarified by Equation (3). The objective of this loss function is to learn an embedding in which the difference between the distance between the anchor and positive members $\hat{d}_{a,p}$ and the distance between the anchor and negative members $\hat{d}_{a,n}$ is larger than the margin value. Only when the separation between members is less than the margin value, as shown in Equation (12), can a triplet contribute to the loss. This formulation is preferable since well-separated triplets do not contribute to network weight updates and are instead guided only by ill-separated triplets. Triplet loss guides the network to learn an embedding of a triplet that treats instances from the same class exactly the same, regardless of their similarity. This loss is modified in order to incorporate intra-class variance and learn a fine-grained embedding space based on the similarity between PCDs.

The proposed triplet loss alters the triplet loss as shown in Equation (13). The proposed formulation broadens the kind of triplets accepted as input to the loss function to include both (positive-negative) and (positive-positive) triplets. The (positive-positive) triplets are triplets with the same label for all three members. Part one of equation (13) depicts the loss component responsible for handling (positive-positive) triplets. The goal of this component is to minimize the difference between the ratio of distances in the embedding space compared to their ground truth distances as estimated by Chamfer distance. This component is further clarified in Equation (14), where a triplet ceases contributing to the loss when the ratio of their distances is equivalent to the ground truth distance. The (positive-negative) triplets are handled in the second term of equation (13), which penalizes the difference between the embedding distance ratio compared to the ratio of Chamfer distance between the anchor and the positive members and the margin value. This term leads the network to believe that the Chamfer distance between the anchor and negative members is equal to at least the margin, causing the network to learn an embedding that pushes them further apart. In Equation (15), the case based

$$max(0, \hat{d}_{a,p} + m - \hat{d}_{a,n}) = \begin{Bmatrix} zero & ,if\ \hat{d}_{a,p} + m \leq \hat{d}_{a,n} \\ +ve & ,if\ \hat{d}_{a,p} + m > \hat{d}_{a,n} \end{Bmatrix} \tag{12}$$

$$Intra\ Class\ Triplet\ Loss(x_a, x_i, x_j) = \begin{Bmatrix} (\hat{d}_{a,i} \times d_{a,j} - \hat{d}_{a,j} \times d_{a,i})^2 & ,if\ y_a = y_i\ and\ y_a = y_j \\ max(0, \hat{d}_{a,i} \times m - \hat{d}_{a,j} \times d_{a,i})^2 & ,if\ y_a = y_i\ and\ y_a \neq y_j \end{Bmatrix} \tag{13}$$

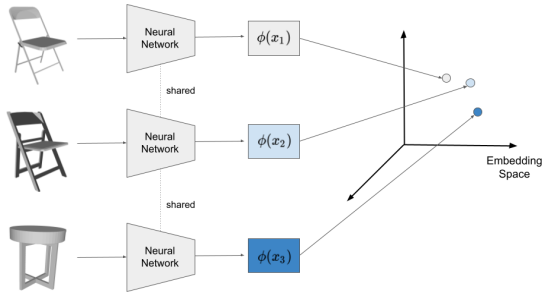


Figure 6: Positive - Positive triplet (x_1, x_2, x_3) example to illustrate proposed intra-class triplet loss behavior.

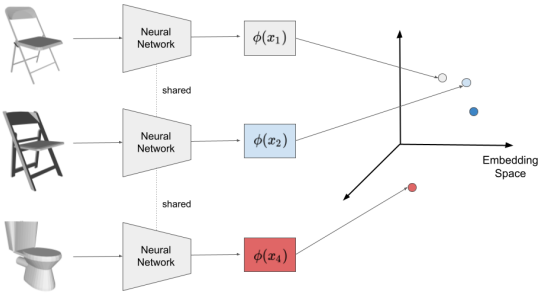


Figure 7: Positive - Negative triplet (x_1, x_2, x_4) example to illustrate proposed intra-class triplet loss behavior.

behavior of the loss for (positive-negative) triplets is formulated. Once members are separated by at least the margin value, this component of the loss function vanishes.

The expected embedding of various PCD triplets using a siamese network guided by the proposed triplet loss is illustrated in Figures (6, 7). In figure (6), a positive - positive triplet (x_1, x_2, x_3) is selected as the siamese network input. When compared to x_3, x_1 and x_2 are more visually similar. As a result, the proposed loss guides the network to learn an embedding that respects the ratio of the members' Chamfer distances. Since the distance between the first and second members of the triplet $\hat{d}_{1,2}$ is less than the distance between the first and third members $\hat{d}_{1,3}$, the resulting vector space preserves intra-class variation.

$$(\hat{d}_{a,i} \times d_{a,j} - \hat{d}_{a,j} \times d_{a,i})^2 = \begin{Bmatrix} zero & ,if\ \dfrac{\hat{d}_{a,i}}{\hat{d}_{a,j}} = \dfrac{d_{a,i}}{d_{a,j}} \\ +ve & ,if\ \dfrac{\hat{d}_{a,i}}{\hat{d}_{a,j}} \neq \dfrac{d_{a,i}}{d_{a,j}} \end{Bmatrix} \tag{14}$$

$$max(0, \hat{d}_{a,i} \times m - \hat{d}_{a,j} \times d_{a,i}) = \begin{Bmatrix} zero & ,if\ \dfrac{\hat{d}_{a,i}}{\hat{d}_{a,j}} \leq \dfrac{d_{a,i}}{m} \\ +ve & ,if\ \dfrac{\hat{d}_{a,i}}{\hat{d}_{a,j}} > \dfrac{d_{a,i}}{m} \end{Bmatrix} \tag{15}$$

A positive - negative triplet is depicted in figure (7), with x_1, x_2 belonging to the class chair and x_4 belonging to the class toilet. The sampled triplet is the siamese network's input. The loss function is designed to build an embedding space that respects the Chamfer distance between examples of the same class (x_1, x_2) while pushing examples from a different class x_4 away with the margin. By separating different classes with a margin, the inter-class variance within the dataset is preserved.

The feature space is confined inside a d-dimensional hyper-sphere by normalizing extracted feature vectors with the L_2 norm. As discussed in section 3.3, the confined space has a favorable influence on the margin selection and distance computation between vectors. However, the margin selection criteria used in the preceding section is not applicable to the proposed triplet loss. Since the proposed method's margin was empirically found to be narrow and does not offer an adequate separation between distinct classes. As a result, the margin selection for this loss is considered a part of the hyperparameter selection process.

M^3 triplets are present in dataset D. It is crucial for stable network training and efficient convergence to choose triplets that contribute to the loss. The selection of (positive-positive) triplets is not problematic, since all of these triplets contribute to the loss. However, sampling (positive-negative) triplets that violate the constraint, known as (positive-hard negative) triplets, poses a challenge. (Positive-hard negative) triplets are triplets where the difference between the feature space distances between the anchor and positive members and the anchor and negative members is less than the margin value. It is impractical to sample these triplets over the whole dataset at each training step.

Section 3.3 examined online and offline sampling strategies. Given its benefits over the offline strategy,

online sampling is well suited for (positive-hard negative) triplets selection. A balanced mini-batch is sampled in the same way as was mentioned in the preceding section. A pre-defined number of (positive-positive) triplets are randomly sampled from the mini-batch, as it is computationally infeasible to sample all these triplets. In addition, an equal number of the (positive-hardest negative) triplets is sampled

Due to the restricted mini-batch size of PCDs, an online sampling technique alone is insufficient for efficient network training. As a result of this constraint, only a few (positive-negative) triplets contribute to the loss by violating the margin constraint at the advanced phases of the training process. As shown in Equation (16), the total loss is adjusted to be a linear combination of cross entropy loss and the proposed intra-class triplet loss, with α and γ regulating the trade-off between both. By integrating the cross entropy loss into the proposed intra-class loss function, the network has an increased discriminative capacity that compensates for the restricted mini-batch, as proposed in Section 3.3.

$$\begin{aligned} Total\ Loss = \alpha \times Cross\ Entropy\ Loss \\ +\gamma \times Intra\ Class\ Triplet\ Loss \end{aligned} \tag{16}$$

3.5 Network Architecture

DGCNN (Wang et al., 2019) is a network architecture that draws inspiration from PointNet (Qi et al., 2017a) and convolution operators. DGCNN, like PointNet, offers a versatile architecture that can be used for a variety of high-level applications. DGCNN presents the Edge-Conv layer, which is a differentiable layer that captures local geometries by considering a point correlation with its neighbors. This layer receives N points as input, with each point represented by d_{in} features. To begin, a $knn-graph$ with a fixed number of neighbors is constructed over the points to emphasize the underlying local geometric structure. The constructed graph is processed using a convolution-like operator to compute a per-point output with d_{out} dimensions. The $knn-graph$ is not fixed and is dynamically modified in each Edge-Conv layer of the network. Dynamically updating implies that the list of k-nearest neighbors in each edge-conv layer is unique and is built from the layer's input features rather than the network's initial input.

The convolution operator learns the edge weight between the center point p_i and its nearest k-neighbors $\{q_1,...,q_k\}$. The edge weights are formulated in Equation (17) to capture local correlations with neighbors and preserve global shape structure. The edge weights capture local geometries based on the difference between a center point features and each of its neighbor features $(q_j - p_i)$ and extract relevant information from it by multiplying with learnable weights θ_1. While the global shape structure is preserved by center point features p_i and information is inferred from these features with learnable weights θ_2. Using Leaky RELU non-linearities, non-linearities are injected into the combined learned features. In contrast to RELU, which does not allow any negative values, Leaky RELU enables a small slope for negative values. When compared to RELU, Leaky RELU produced superior empirical results in DGCNN. DGCNN is one of the best performing point-based models in PCD analysis, owing to the favorable properties of the edge-conv layer. Since neither the graph structure nor the convolution-like operator is order-dependent, edge-conv is permutation invariant. Furthermore, it exhibits partial translation invariance since the difference between features $(q_j - p_i)$ is a relative term that is translation invariant. However, the center point feature p_i term is absolute. Finally, this layer displays non-locality characteristics since the proximity in the feature space is not the same as the proximity in the input space, information is dispersed in a non-local manner.

$$e_{i,j} = LeakyRelu(\theta_1 \times (q_j - p_i) + \theta_2 \times p_i) \tag{17}$$

4 EVALUATIONS

4.1 Dataset

The ModelNet-10 dataset is used in this work to evaluate the proposed loss functions' discriminative and model retrieval performance. ModelNet-10 (Wu et al., 2015), published by Princeton University, is a benchmark for 3D object classification and retrieval, where the dataset includes 4899 CAD-generated meshes that were saved in Object File Format (OFF). The meshes are separated into 3991 training meshes and 908 testing meshes. ModelNet is well-known in the research field as it is a well-structured dataset, including pre-aligned clean shapes picked from several categories. The dataset includes ten classes: bathtub, bed, chair, desk, dresser, monitor, nightstand, sofa, table, and toilet. The output PCD is generated by randomly sampling evenly distributed points from the triangle faces of the CAD-generated mesh. The points that were sampled are normalized and put into a bounding box between $[-1, 1]$.

4.2 Experimental Setting

The implementation of the proposed losses and network architecture is established in Python 3.9 with

PyTorch 1.10.0 and Cuda 11.5 on an Nvidia GeForce RTX 3090 graphics card with 24 GB of vRAM. DGCNN network architecture is implemented as in (Wang et al., 2019). The network is trained with a balanced mini-batch having 10 samples per class for 15 epochs. Besides, it is optimized using an SGD optimizer with an initial learning rate of 0.1, momentum of 0.9, and weight decay of 10^{-4}. The learning rate is decreased using a cosine annealing schedule to reach a final learning rate of 0.001 as recommended in the original implementation. Upon complication of the learning, the training dataset is fitted in *kd*-tree for a faster query during the inference of model retrieval metrics.

4.3 Evaluation Metrics

Two aspects of the designed loss functions will be evaluated in this work: discriminative and retrieval abilities. The discriminative ability is measured using well-known classification metrics such as recall, precision, F−score, and accuracy. The mean average precision (mAP) metric is used to evaluate the model retrieval quality and stands for the mean value of the average precision. mAP is first presented in (Harman, 1993) as metric of information retrieval quality over text. Average Precision (AP) is parameterized by k which is the number of retrieved examples, as seen in Equation (19). Average Precision@k is the product of precision@k, as in Equation (18), and relevance@k, normalized by the number of relevant retrieved instances, also known as Ground Truth Positives (GTP). Relevance@k is an indicator of whether the retrieved instance is correct or not. Precision@k is the number of relevant retrieved instances to the total number of retrieved instances, as illustrated in equation (18).

$$precision\,@\,k = \frac{correct\ results\,@\,k}{k} \tag{18}$$

$$AP\,@\,K = \frac{1}{GTP}\sum_{k=1}^{K} precision\,@\,k \times relevance\,@\,k \tag{19}$$

4.4 Quantitative Results

Table 1 illustrates the discriminative ability of DGCNN trained with various loss functions based on the classification metrics stated earlier. The results indicate that both of our proposed losses outperform the cross entropy loss on the classification task by 2% on average on all metrics. This improvement indicates that the proposed intra-class pair and triplet losses develop a more discriminative hyperspace for classification with good separation between classes in the dataset. When compared to the proposed triplet loss, which is slightly better, the proposed pair loss has a very competitive performance.

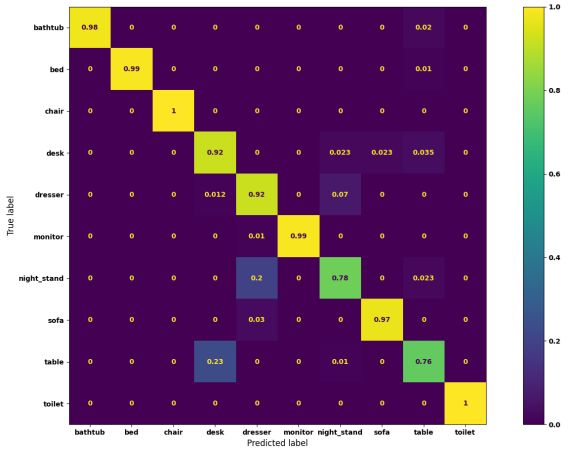


Figure 8: Confusion matrix for DGCNN network trained by the proposed pair loss.

Figures (8, 9) demonstrate the confusion matrix extracted from classification statistics of a DGCNN classifier trained by the proposed intra-class pair and triplet losses, respectively. The highest values always appear on the confusion matrix diagonals, indicating a good classification performance across all classes. The little dispersed values scattered over the confusion matrix indicate the classification error for a certain class distributed across the other classes. A minor categorization error occurs when instances from the class nightstand are classified as belonging to the class dresser, and when instances from the class table are labeled as belonging to the class desk. The fact that both class pairs (nightstand, dresser) and (table, desk) are visually similar and might be difficult for humans to categorize leads to these minor inter-class errors.

Table 2 introduces model retrieval results on mAP metric using labels sharing as relevance measure. The results are calculated across a range of retrieval sizes, starting with 5 and up to 20 with a step size of 5, to provide a better insight into our proposed losses' performance. Two extra baselines are added in this table, specifically the original contrastive and triplet losses formulations, which were created explicitly for the model retrieval task based on the labels. The results demonstrate that our proposed intra-class pair and triplet losses are competitive on the task at hand. On all retrieval sizes except 20, where triplet loss is marginally better, the proposed pair loss performs better than all baselines. We may deduce from this table that our proposed losses do not degrade performance as they keep the inter-class term of the original losses and, in fact, outperform the state-of-the-art losses like

Table 1: Classification results, where the highest values indicate better performance.

Metric	Cross Entropy Loss	Intra-Class Pair Loss (Ours 1)	Intra-Class Triplet Loss (Ours 2)
Accuracy	0.9152	0.9306	0.9317
Average Recall	0.9082	0.9306	0.9301
Average Precision	0.9178	0.9334	0.9335
Average F-score	0.9133	0.9298	0.93

Table 2: DGCNN model retrieval results with labels as relevance measure, where the highest values indicate better performance.

Top - K	Cross Entropy Loss	Contrastive Loss	Triplet Loss	Intra-Class Pair Loss (Ours 1)	Intra-Class Triplet Loss (Ours 2)
5	0.9120	0.9139	0.9191	0.9239	0.9210
10	0.9025	0.9063	0.9115	0.9150	0.9102
15	0.8953	0.8991	0.9055	0.9062	0.9035
20	0.8857	0.8945	0.9005	0.8997	0.8969

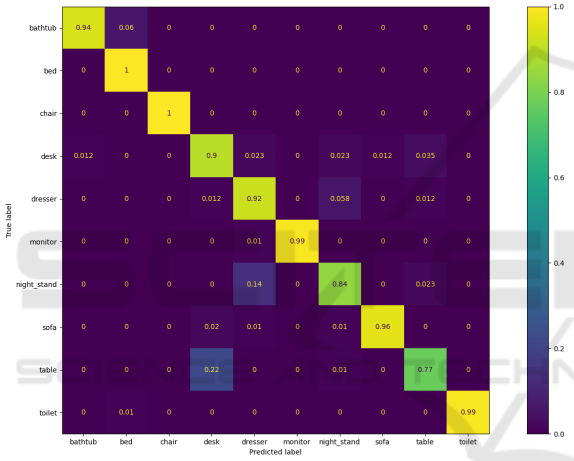


Figure 9: Confusion matrix for DGCNN network trained by the proposed triplet loss.

contrastive and triplet loss.

Furthermore, the model retrieval task with chamfer distance as a relevance metric is examined in Table 3. When compared to all three baselines, the proposed pair and triplet losses are substantially superior, with the proposed intra-class pair loss being slightly better. The nearest loss in performance to our losses is the cross entropy loss. The lack of a term dedicated to addressing intra-class variance in both the original formulations of contrastive and triplet losses hinders performance on this objective.

4.5 Qualitative Results

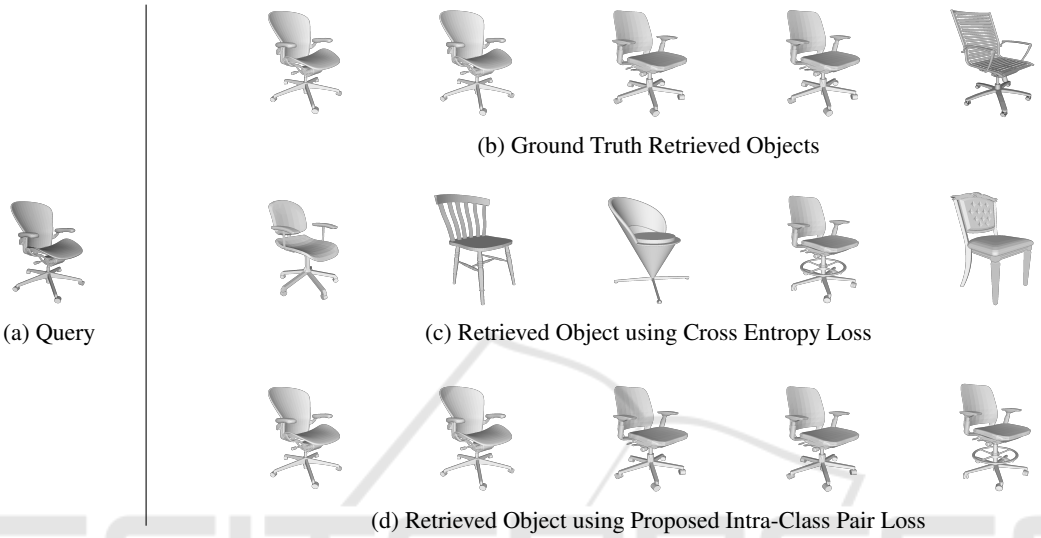
Figure (10) shows an example of model retrieval results using the Chamfer distance as a measure of relevance. Figure(10a) represents the query model for which the most similar counterparts in the dataset should be retrieved. Figure (10) is the ground truth nearest neighbors to the query model PCD with Chamfer distance as a similarity measure. Figure (10) demonstrate retrieved objects using cross-entropy loss. The retrieved models, as shown in the figure, belong to the same class, but they are not structurally or visually similar, and none of them is a member of the ground truth query. The obtained results based on the proposed intra-class pair loss are quite visually similar to the query model, as shown in figure (10). The first four models retrieved are members of the ground truth query, whereas the fifth model retrieved is not part of the ground truth query. Despite the fact that the last retrieval is incorrect, it has a tolerable visual resemblance to the query model. Thus, figure (10) supports our hypothesis that a loss function that accounts for intra-class variance improves model retrieval results.

5 DISCUSSION AND FUTURE WORK

According to empirical results, the network has issues in distinguishing PCDs from classes that are structurally similar, such as the class pairs (nightstand, dresser) and (table, desk), and can not provide sufficient separation between them. The difficulty arises as these class pairs have visually similar instances that are difficult to distinguish even by humans. To solve this deficiency, a more extensive structural comparison between these related instances is necessary. As a result, in future work, a network pre-trained for the part segmentation task or jointly trained for part segmentation will be investigated. This suggestion is based on the notion that the increased level of in-

Table 3: DGCNN model retrieval results with Chamfer distance as relevance measure, where the highest values indicate better performance.

Top - K	Cross Entropy Loss	Contrastive Loss	Triplet Loss	Proposed Pair Loss (Ours 1)	Proposed Triplet Loss (Ours 2)
5	0.6531	0.5974	0.5178	0.6986	0.6951
10	0.6837	0.6363	0.5698	0.7200	0.7152
15	0.6924	0.6417	0.5764	0.7287	0.7271
20	0.6989	0.6479	0.5770	0.7329	0.7311



(a) Query

(b) Ground Truth Retrieved Objects

(c) Retrieved Object using Cross Entropy Loss

(d) Retrieved Object using Proposed Intra-Class Pair Loss

Figure 10: Model retrieval results with Chamfer distance as relevance measure under different loss functions. Note: the qualitative results for the proposed triplet loss were excluded, since intra-class pair loss showed slightly better performance.

formation captured during part segmentation can emphasize the structural differences between similar instances from different classes.

In addition, we suggest using clustering to simplify the similarity-aware embedding task. This simplification is designed to eliminate the combinatorial nature of the proposed embedding task. The combinatorial nature comes from the fact that altering weights based on a mini-batch affects the network objective for all remaining instances. We propose establishing intra-class clusters based on the similarity between instances from the same class to simplify this problem. To begin, a graph is constructed, with each PCD in the class representing a node and edge weights proportional to the RBF kernel of the Chamfer distance between the two PCDs connected by the edge. This graph can be clustered using spectral graph clustering, which searches for the optimal graph cut to cluster the graph. Finally, the network objective is revised to partition the features space into intra-class clusters separated by an intra-class margin, as well as, separating different classes by at least a distance of a margin.

6 CONCLUSION

In this work, two novel loss functions are proposed by modifying contrastive and triplet loss formulations to include a term that handles intra-class variance and generates PCD embeddings that respect the Chamfer distance between PCDs from the same class. DGCNN is the network of choice for this work since it dynamically generates local $knn - graphs$ across the PCD, encapsulating local neighborhoods and features at each layer. Results demonstrate that DGCNN trained with either of the proposed loss functions outperforms all baselines in classification, label-based model retrieval, and similarity-based model retrieval due to the quantitatively and qualitatively enhancement of embedding.

REFERENCES

Arandjelovic, R., Gronat, P., Torii, A., Pajdla, T., and Sivic, J. (2016). Netvlad: Cnn architecture for weakly supervised place recognition. In *Proceedings of the IEEE*

conference on computer vision and pattern recognition, pages 5297–5307.

Bai, Y., Ding, H., Qiao, Y., Marinovic, A., Gu, K., Chen, T., Sun, Y., and Wang, W. (2019). Unsupervised inductive graph-level representation learning via graph-graph proximity. *arXiv preprint arXiv:1904.01098*.

Chatpatanasiri, R., Korsrilabutr, T., Tangchanachaianan, P., and Kijsirikul, B. (2010). A new kernelization framework for mahalanobis distance learning algorithms. *Neurocomputing*, 73(10-12):1570–1579.

Dey, S., Dutta, A., Toledo, J. I., Ghosh, S. K., Lladós, J., and Pal, U. (2017). Signet: Convolutional siamese network for writer independent offline signature verification. *arXiv preprint arXiv:1707.02131*.

Em, Y., Gag, F., Lou, Y., Wang, S., Huang, T., and Duan, L.-Y. (2017). Incorporating intra-class variance to fine-grained visual recognition. In *2017 IEEE International Conference on Multimedia and Expo (ICME)*, pages 1452–1457. IEEE.

Hadsell, R., Chopra, S., and LeCun, Y. (2006). Dimensionality reduction by learning an invariant mapping. In *2006 IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR'06)*, volume 2, pages 1735–1742. IEEE.

Harman, D. K. (1993). *The first text retrieval conference (TREC-1)*, volume 500. US Department of Commerce, National Institute of Standards and Technology.

Hoffer, E. and Ailon, N. (2015). Deep metric learning using triplet network. In *International workshop on similarity-based pattern recognition*, pages 84–92. Springer.

Melekhov, I., Kannala, J., and Rahtu, E. (2016). Siamese network features for image matching. In *2016 23rd international conference on pattern recognition (ICPR)*, pages 378–383. IEEE.

Mescheder, L., Oechsle, M., Niemeyer, M., Nowozin, S., and Geiger, A. (2019). Occupancy networks: Learning 3d reconstruction in function space. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 4460–4470.

Nazir, D., Afzal, M. Z., Pagani, A., Liwicki, M., and Stricker, D. (2021). Contrastive learning for 3d point clouds classification and shape completion. *Sensors*, 21(21):7392.

Park, J. J., Florence, P., Straub, J., Newcombe, R., and Lovegrove, S. (2019). Deepsdf: Learning continuous signed distance functions for shape representation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 165–174.

Peng, S., Niemeyer, M., Mescheder, L., Pollefeys, M., and Geiger, A. (2020). Convolutional occupancy networks. In *European Conference on Computer Vision*, pages 523–540. Springer.

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

Qi, C. R., Yi, L., Su, H., and Guibas, L. J. (2017b). Pointnet++: Deep hierarchical feature learning on point sets in a metric space. *Advances in neural information processing systems*, 30.

Schroff, F., Kalenichenko, D., and Philbin, J. (2015). Facenet: A unified embedding for face recognition and clustering. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 815–823.

Szegedy, C., Liu, W., Jia, Y., Sermanet, P., Reed, S., Anguelov, D., Erhan, D., Vanhoucke, V., and Rabinovich, A. (2015). Going deeper with convolutions. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 1–9.

Uy, M. A. and Lee, G. H. (2018). Pointnetvlad: Deep point cloud based retrieval for large-scale place recognition. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 4470–4479.

Wang, J., Song, Y., Leung, T., Rosenberg, C., Wang, J., Philbin, J., Chen, B., and Wu, Y. (2014). Learning fine-grained image similarity with deep ranking. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 1386–1393.

Wang, J., Zhou, F., Wen, S., Liu, X., and Lin, Y. (2017). Deep metric learning with angular loss. In *Proceedings of the IEEE international conference on computer vision*, pages 2593–2601.

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

Wen, Y., Zhang, K., Li, Z., and Qiao, Y. (2016). A discriminative feature learning approach for deep face recognition. In *European conference on computer vision*, pages 499–515. Springer.

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

Xing, E., Jordan, M., Russell, S. J., and Ng, A. (2002). Distance metric learning with application to clustering with side-information. *Advances in neural information processing systems*, 15.

Xu, W. (2021). Deep large margin nearest neighbor for gait recognition. *Journal of Intelligent Systems*, 30(1):604–619.