3DSSR: 3D Subscene Retrieval

Reza Asad   Manolis Savva
Simon Fraser University
rasad@sfu.ca   msavva@sfu.ca

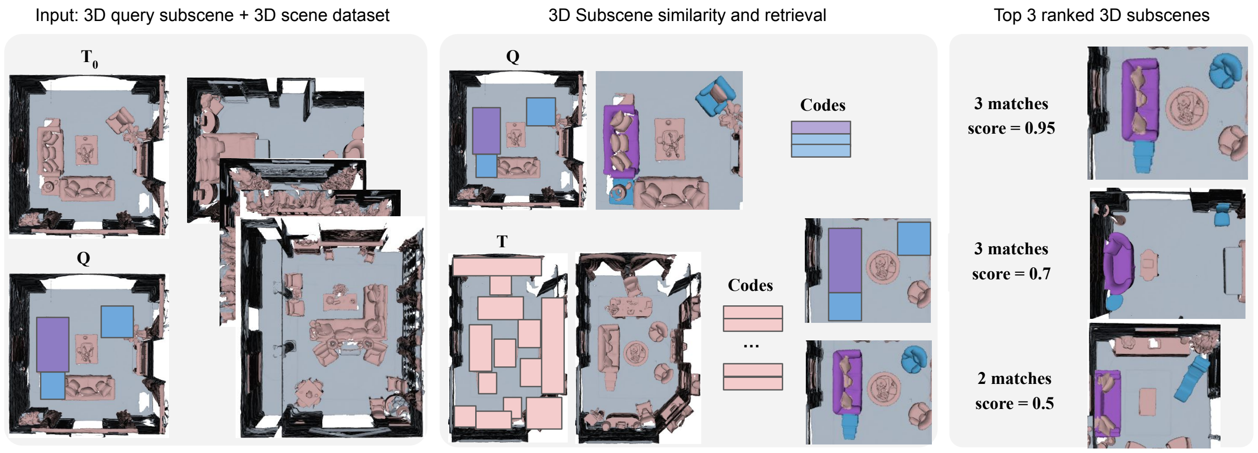

Input: 3D query subscene + 3D scene dataset
3D Subscene similarity and retrieval
Top 3 ranked 3D subscenes
T_0
Q
Q
T
Codes
Codes
...
3 matches
score = 0.95
3 matches
score = 0.7
2 matches
score = 0.5

Figure 1. In the 3DSSR task, a user provides an input query Q specifying a set of axis-aligned bounding boxes (bottom left). The query consists of an object of interest (couch in purple box), and a set of context objects (side table and chair in blue boxes). Then, for each 3D scene T in the dataset we identify a collection of objects (i.e., a subscene) that best correspond to the objects in Q. Our notion of subscene similarity relies on intrinsic and extrinsic properties of the 3D object within each box. For intrinsic similarity, we encode the object within and compare the codes from Q and T. For extrinsic similarity, we measure the overlap between subscene arrangements. The output in 3DSSR is a ranked list of retrieved subscenes (right column).

Abstract

We present the task of 3D subscene retrieval (3DSSR). In this task a user specifies a query object and a set of context objects in a 3D scene. Then, a system retrieves and ranks subscenes from a database of 3D scenes that best correspond to the configuration defined by the query. This formulation generalizes prior work on context-based 3D object retrieval and 3D scene retrieval. To tackle this task we present POINTCROP*: a self-supervised point cloud encoder training scheme that enables retrieval of geometrically similar subscenes without relying on object category supervision. We evaluate* POINTCROP *against alternative methods and baselines through a suite of evaluation metrics that measure the degree of subscene correspondence. Our experiments show that* POINTCROP *training outperforms supervised and prior self-supervised training paradigms by 4.33% and 9.11% in mAP respectively.*

1. Introduction

Recent advances in 3D reconstruction and 3D content creation methods have led to an explosion in the volume of 3D scene data. As the scale of both synthetic and reconstructed 3D scene datasets continues to grow, methods to search and retrieve scenes and their constituent objects become increasingly important. The ability to specify fine-grained constraints on parts of scenes that are to be retrieved is particularly useful in a variety of practical scenarios. For example, if we can retrieve subscenes similar to a square table with two chairs on each side, we can edit the 3D objects in those subscenes for various downstream tasks. In this scenario, one could replace the 3D objects in the retrieved subscenes with geometrically simpler objects, insert additional objects, or delete a subset of objects. This capability can also help to compress a 3D scene database by deduplicating subscenes across many scenes.

Prior work has tackled 3D scene retrieval (i.e. retrieving an entire 3D scene [7]) and object retrieval from 3D scenes (i.e. retrieving a specific object in a 3D scene [6]). This early work in both directions has established versions of these two tasks as separate tasks. However, we argue that the 'contextually meaningful object set' retrieval problem as articulated above is a more practically useful task. Firstly, as the square table with two chairs example illustrates, it is natural and intuitive to specify constraints on a set of objects. Secondly, with increasing 3D scene size and complexity, it is important to localize and select a relevant subset of objects rather than retrieve an entire scene.

We formalize this problem statement as the 3D subscene retrieval task (3DSSR). The input is a 3D query subscene consisting of a query object in the context of a 3D scene (e.g square table with one chair on each side). The output is a ranked list of 3D subscenes retrieved from the database, along with a correspondence map between the 3D objects in the retrieved and query subscenes. This task generalizes single object retrieval and entire scene retrieval and affords a spectrum of options in between. This is a challenging task for a number of reasons. Unlike scene graph-based image retrieval [11], we must reason with 3D scene representations that involve 3D spatial relations between objects. We cannot rely on common viewpoint biases that exist in image data (e.g., that objects of interest are usually centered in the frame, and are less frequently occluded). Moreover, we cannot assume that we have access to large datasets of 3D scenes with annotated object categories.

We avoid supervision through object category labels since we hypothesize their discrete nature is not ideal for subscene retrieval where fine-grained geometric and spatial similarity is important. For example, chairs exhibit high variability in terms of size and geometry. Therefore, we tackle the 3DSSR task without explicit object category supervision. To validate our hypothesis, we train the same point cloud encoder architecture using supervised, prior self-supervised paradigms, and our approach. Furthermore, we compare our model against methods that directly use oracle object categories for subscene retrieval. Our experiments show that our non-contrastive self-supervised encoder (POINTCROP) retrieves geometrically more similar 3D subscenes compared to baselines. We outperform supervised and prior self-supervised paradigms by 4.33% and 9.11% in mean average precision (mAP) of geometric and spatial arrangement similarity between subscenes.

2. Related Work

Scene graph-based image retrieval. Our problem statement is related to work on image retrieval based on scene graph representations. The work of Johnson *et al.* [11] is an early example, employing a conditional random field model to retrieve images that best represent a query scene graph. The use of scene graphs to represent images has been popularized by the VisualGenome dataset [12]. More recent work leveraging such data has focused on image generation based on input scene graphs [10]. Our work deals with 3D scenes, thus our 3D subscene retrieval framework does not rely on viewpoint biases in images.

3D scene graph-based retrieval and dataset organization. The work of Fisher *et al.* [6] on context-based retrieval of 3D objects is an early example of retrieving 3D objects based on pairwise spatial relations in the source scene and a target scene. Followup work used a graph kernel formulation for 3D scene retrieval as well as retrieval of a relevant 3D object by specifying a target node within an input 3D scene [7]. Xu *et al.* [20] focused on the organization of 3D scene datasets by clustering 3D scenes using a graph-based formulation based on a set of detected 'meaningful' focal points in the 3D scene. Ma *et al.* [15] propose a 3D scene synthesis method that takes text as input and generates 3D scenes by retrieving objects from a scene database. This prior work relies on the presence of consistently categorized objects and works with synthetic 3D scene data. Also, prior work does not address our 3D subscene retrieval problem statement where the query consists of an object of interest and a set of contextually relevant surrounding objects.

Self-supervised representation learning for images and 3D scenes. Following the success of transformers in natural language processing, there has been an explosion of self-supervised image representation models. SwAV [2] introduces multi-crop data augmentation to boost the performance of various self-supervised models. He *et al.* [8] introduce momentum encoder to build a dynamic dictionary of codes for contrastive learning. DINO [3] re-purposes the momentum encoder for self-distillation (student-teacher networks) to update teacher parameters without requiring a pre-trained teacher. Recent work is aimed at learning rich representations of 3D point cloud scenes without supervision. PointContrast [19] introduces a point-level contrastive loss. Followup work by Hou *et al.* [9] incorporates spatial context into the training objective. However, both of these contrastive methods rely on data pre-processing to register the 3D data using known camera poses. In this paper, we are focused on leveraging a self-supervised encoder for 3D point cloud representations to tackle the 3DSSR task. To achieve this, we extend Hou *et al.* [9]'s work so that it does not require 3D registration. We systematically compare these approaches to our efficient non-contrastive point cloud encoder (POINTCROP) that is trained through self-distillation [3] and 3D multi-crop data augmentation.

3. 3D Subscene Retrieval (3DSSR) Task

Figure 1 shows an overview of our 3D subscene retrieval pipeline. The first input to 3DSSR is a 3D scene T_0 represented as a triangle mesh. Given T_0, a user specifies a query subscene Q which consists of an object of interest $q \in Q$ (e.g., a couch) and a set of context objects (e.g., a side table next to the couch and a chair in front of it). The user-specified objects in Q are defined by a set of axis-aligned bounding boxes (AABBs) in T_0. We have chosen AABBs over oriented bounding boxes (OBBs) because from a practical perspective, it is easier to acquire AABBs from a user than OBBs. The second input is a database of 3D target scenes $\{T_1, T_2, ..., T_N\}$, where each T_i is a triangle mesh with AABBs localizing the 3D objects in that scene (see Section 4.1). The output is then a list of 3D subscenes $\{S_1, S_2, ..., S_M\}$ ($M \leq N$) extracted from the database and ranked from most to least similar compared to Q. Concretely, each retrieved subscene $S_i \subset T_i$ consists of a subset of the AABBs in T_i. Moreover, each selected AABB in S_i corresponds to exactly one AABB in the query subscene Q. We do not assume the presence of category labels. In Section 4.2, we describe our notion of object-object similarity for retrieving similar objects without the presence of category labels. The desired ranking of retrieved subscenes orders them from most to least similar by counting the number of the *corresponding objects* and the *overall correspondence quality score* (see Section 4.3).

Object Region Extraction. Self-supervised localization of objects in 3D scenes is a challenging task. To tackle 3D subscene retrieval, we first focus on removing the requirement for discrete object categories. Therefore, in the main paper, we assume we have access to the ground truth information on the localization of 3D objects in 3D scenes. Furthermore, we discuss results and future work when object localization is predicted with supervision. We represent the localized 3D objects (ground truth or predicted) using AABBs $\{b_1, b_2, ..., b_k\}$. Given each box b_i, we sample 3D points from the object mesh inside b_i.

Object Region Similarity. To define similarity at the subscene level, we first establish a notion of object-object similarity by encoding the point cloud in each b_i. Alternatively, one could directly use or predict object category labels. In this paper, we avoid category labels, because we hypothesize that their discrete nature poses a problem for the 3D subscene retrieval task. For instance, two 3D objects with the same category (e.g., chair) can have dramatically different geometries and sizes. In the 3DSSR task, we particularly care about retrieving a collection of 3D objects (i.e., 3D subscenes) ranked from most to least similar compared to the query subscene.

3D Subscene Similarity. We then establish a measure of similarity between a 3D query subscene Q with object boxes $\{q_1, q_2, ..., q_n\}$ and a 3D target scene T containing object boxes $\{t_1, t_2, ..., t_m\}$. Each t_i is in a potential correspondence with q_j. We compute the intrinsic similarity between them plus any extrinsic similarity important for the subscene retrieval task at hand. For extrinsic similarity, the geometric overlap of the object boxes when they are mapped to the world coordinate frame might be desirable.

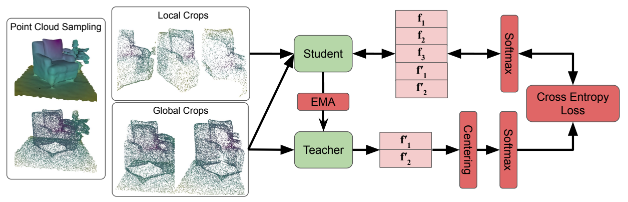


Figure 2. Overview of POINTCROP. We sample 3D points from an input object mesh and take global and local crops of the point cloud. All crops are fed to the student, while the teacher is given only global crops. The teacher and student use the same architecture [1] but different optimization strategies. Unlike the student, the parameters of the teacher are not optimized through gradient descent. Instead, the teacher's parameters are an exponential moving average (EMA) [8] of the student's parameters. To avoid collapse (i.e., assigning the same representation to all crops) we apply a centering operation [3] on the teacher network output. Finally, we apply softmax and compute the cross entropy between the student's output and the teacher's output.

Another example would be the distance and angle of the boxes relative to some anchor objects. Note that the former example using geometric overlap is strict and the latter metric is softer. The exact choice really depends on the downstream application. In this setting, we say t_i corresponds to q_j if their intrinsic similarity ISim$(q_j, t_i) > \tau_1$ and is the highest among all objects in T. Furthermore, we require their extrinsic similarity to satisfy ESim$(q_j, t_i) > \tau_2$. The thresholds τ_1 and τ_2 are hyperparameters that can be determined from validation or training data. Finally, we characterize the goodness of the identified correspondences by summing over the ISims. More concretely, for the identified corresponding boxes $\{(q_1, t_1), (q_2, t_2), ...(q_{\mathcal{N}}, s_{\mathcal{N}})\}$, we compute: $\mathcal{QT} = \sum_{k=1}^{\mathcal{N}} \text{ISim}(q_k, t_k)$.

4. Method

4.1. Object Region Extraction

We extract an AABB for each object using the semantic object instance annotations on the scene mesh in Matterport3D [4]. We compute an axis-aligned bounding box b_i of the object mesh. Each b_i consists of a center parameter c, two orientation vectors v_1 and v_2 (in the global coordinate frame) and three scalars d_1, d_2 and d_3. The scalars represent the dimensions of the box b_i along v_1, v_2 and the vector perpendicular to v_1 and v_2.

4.2. Object Region Similarity

To compute object-object similarity we first sample 3D points from the object mesh in each box b_i and encode them using a point cloud encoder. Point cloud encoding is a convenient choice that has been shown to work well for downstream tasks such as segmentation and detection [9, 17, 19].

Next, we compute the cosine similarity between a pair of codes. More formally, let $f : \mathbb{R}^{N \times 3} \mapsto \mathbb{R}^D$ be a point cloud encoder. Suppose q and t are boxes as defined in Section 4.1. We denote the point clouds sampled from the 3D mesh inside q and t by $\mathcal{P}_q$ and $\mathcal{P}_t$ and the cosine similarity between the codes as $\text{Sim}(f(\mathcal{P}_q), f(\mathcal{P}_t))$.

For the point cloud encoder f we explore two choices: i) POINTCROP: our non-contrastive self-supervised model that follows self-distillation with no labels; and ii) CSC: point-level contrastive learning [9]. Both POINTCROP and CSC divide the input point cloud $\mathcal{P}$ into multiple crops. However, POINTCROP's objective is to classify each crop in $\mathcal{P}$ to similar class concepts without supervision. On the other hand, for a given 3D point $p \in \mathcal{P}$, CSC finds positive and negative 3D point correspondences in $\mathcal{P}$. CSC's objective pushes the representations of the positive examples closer to each other and further away from the negatives. In this paper, we investigate which paradigm leads to better shape representations for 3D subscene retrieval and how they compare to alternative methods.

Self-distillation with no labels. Figure 2 shows an overview of our point cloud encoder POINTCROP. To prepare the input for POINTCROP, we follow the sampling strategy of Osada et al. [16] and sample points proportional to the area of the mesh triangles in each box b_i. Next, we extract global and local crops from each b_i using fixed-size cubes. While a global crop captures most of the 3D object point cloud, a local crop covers a smaller portion of the object. Given the crops, we randomly sample N points from the point cloud of each global crop and $\lfloor N/4 \rfloor$ points from each local crop. We feed all crops to a student network and the global crops to a teacher network for self-distillation. We train the student and teacher so the student classifies each crop to similar class concepts as each global crop fed to the teacher. The similarity here is measured through a cross-entropy loss. The student and teacher networks use the same architecture [1]. Specifically, we use a Point Transformer [21] with the segmentation architecture followed by a global average-pooling across the 3D points and DINO's [3] head module (see supplement for details).

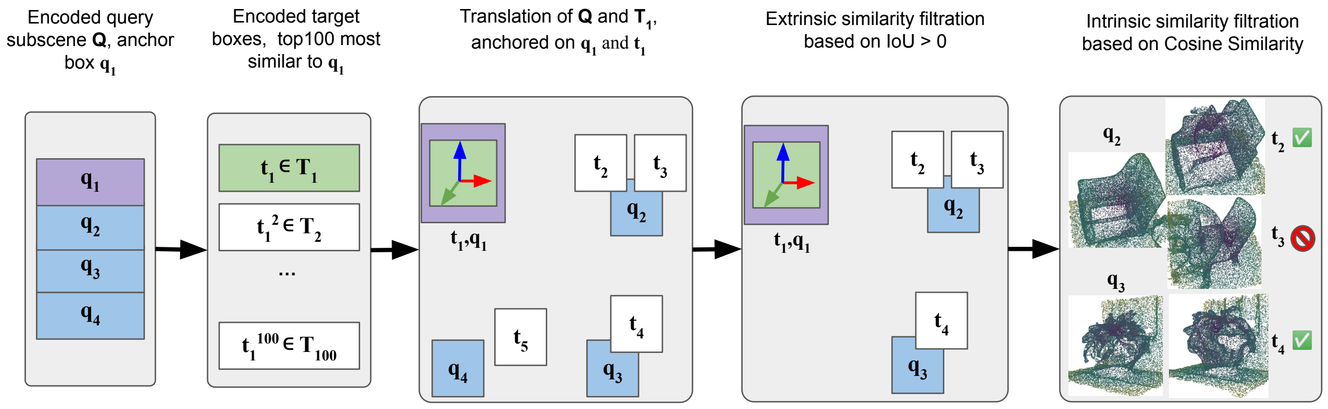
Point-level contrastive learning. The self-supervised CSC [9] model has been shown to outperform PointContrast [19]. The input to both models consists of pairs of partial overlapping point clouds. To derive each pair, both models run a pre-processing step on the RGB-D scans of a 3D scene with known camera poses. We want to encode the 3D objects in each box $\boldsymbol{b_i}$ so we prepare the input data differently. More concretely, we prepare each input as a global crop in POINTCROP. We ensure that pairs of overlapping point clouds satisfy the requirements of CSC [9] (e.g., same percentage of overlapping points). This modification enables us to use CSC without camera poses and registration. For the backbone, in addition to the architectures used in CSC [9] (PointNet++ [17] and Sparse Res-UNet [19]), we also use a Point Transformer [21] (same architecture as POINTCROP). This enables a direct comparison between POINTCROP and CSC.

4.3. 3D Subscene Similarity and Ranking

Figure 3 summarizes the stages of our subscene retrieval strategy. We are given a 3D query subscene Q containing $\{q_1, q_2, q_3, q_4\}$ and a set of 3D target scenes $\{T_1, T_2, ..., T_N\}$. Let q_1 be the object of interest. To retrieve subscenes, we first identify the top 100 boxes from the target scenes that are most similar to q_1. Concretely, for a candidate $t_1 \in T_1$ we use a pre-trained point cloud encoder f from Section 4.2 and compute the cosine similarity $Sim(f(\mathcal{P}_{q_1}), f(\mathcal{P}_{t_1}))$. If $t_1 \in T_1$ is in the top 100, we extract a 3D subscene consisting of t_1 and (potentially) more objects in T_1 that correspond to the other objects in Q.

To find correspondences between other pairs of objects in Q and T_1 we follow the strategy in Section 3 (under 3D subscene similarity). Extrinsic similarity is computed through the intersection-over-union (IoU) between the boxes in a candidate pair when both are translated to the world coordinate frame. More concretely, we translate each box in Q and T_1 so the centroid of the **anchor boxes** q_1 and t_1 are at the origin of the world coordinate frame. Then, for each $t_i \in T_1, i \neq 1$ and $q_j \in Q, j \neq 1$ we accept the candidate pair as a potential match if ESim $= IoU(q_j, t_i) > 0$. Intrinsic similarity uses the same pre-trained point cloud encoder f and computes the cosine similarity between the encoded objects in a candidate pair. At this stage, we accept the pair as a final match if ISim $= Sim(f(\mathcal{P}_{q_j}), f(\mathcal{P}_{t_i})) > \tau$ and is the highest among all remaining boxes in T_1. After repeating this procedure for each remaining $q_j \in Q$ we find the subscene $S_1 \subset T_1$ that best matches the query subscene Q.

To choose the threshold τ, we first compute the cosine similarity between all pairs of encoded objects in the training data. We then apply a k-means clustering algorithm on the computed similarities using two clusters. Next, we compute the centroid for each cluster and use the larger value as the threshold τ. Intuitively, we are clustering the cosine similarities into two groups 'similar' vs 'non-similar', where τ is the average similarity value among the 'similar' cluster. Finally, we rank the retrieved 3D subscenes based on the number of corresponding objects they offer (descending order). For two target subscenes with the same number of corresponding objects, we rank the one with higher correspondence quality score $\mathcal{QT} = \sum_{k=1}^{\mathcal{N}} Sim(f(\mathcal{P}_{q_k}), f(\mathcal{P}_{t_k}))$ higher.



Encoded query subscene **Q**, anchor box q_1
Encoded target boxes, top100 most similar to q_1
Translation of **Q** and T_1, anchored on q_1 and t_1
Extrinsic similarity filtration based on IoU > 0
Intrinsic similarity filtration based on Cosine Similarity
q_1
q_2
q_3
q_4
$t_1 \in T_1$
$t_1^2 \in T_2$
...
$t_1^{100} \in T_{100}$
t_1, q_1
t_2
t_3
t_4
t_5
✅
🚫

Figure 3. Illustration of finding a subscene $S_1 \subset T_1$ that best matches a query subscene Q. Before computing extrinsic similarity, the boxes in Q and T_1 are translated such that the centroids of the anchor boxes (q_1 and t_1) are co-located at the origin (column 3). After computing the extrinsic similarity (column 4), the corresponding candidate t_5 is omitted because it has zero IoU with q_4. At this stage both t_2 and t_3 have positive IoU with q_2 (query chair), but we only take t_2 in column 5. This is because the chair candidate t_2 has a higher cosine similarity (intrinsic similarity) with q_2 than t_3. Both chair and plant candidates t_2 and t_4 pass the cosine similarity threshold required to be labeled as a match. The best matching subscene $S_1 \subset T_1$ consists of t_1, t_2 and t_4.

5. Experiments

5.1. Dataset

We use Matterport3D [4] which contains 2195 3D rooms with a median number of 20 objects. For evaluation and category-dependent baselines, we use the 'mpcat40' category labels. To see a full list of the categories please see the supplement. In total, we have 17174, 2676, and 4802 objects in the train, val, and test sets respectively. To evaluate the generalization of the self-supervised models POINTCROPRANK and CSCRANK, we also train the models on ScanNet [5] and evaluate them on Matterport3D [4] (please see results in the supplement).

Validation and Test data. We prepare point cloud data by first sampling 40960 points from the 3D mesh in each box b_i following Osada *et al.* [16]. Next, we randomly sample 4096 points from this point cloud and save the results for evaluation purposes across all experiments.

Training Data. To enable a fair comparison between POINTCROP and CSC, we prepare the same training data in two different formats. For each box b_i, both models first sample 40960 points as for the val and test data. For POINTCROP, we take 2 global and 8 local cubical crops, where the global and local crops are always 0.7 and 0.4 times the dimensions of b_i respectively. We then randomly sample 4096 points from the point cloud in each global crop and 1024 points from the local crops. The objective function in PointCrop compares each global crop to the other 9 crops, resulting in a total of 18 comparisons. For fairness, we also give 18 pairs of overlapping partial point clouds per b_i to CSC. We use the same overlapping criteria as the original paper and each partial point cloud is prepared exactly like a global crop in POINTCROP.

5.2. Evaluation Metrics

Mean Average Precision. We use mAP to evaluate the ranked list of retrieved subscenes in the 3DSSR task. We compare each query subscene against the top 10 target subscenes provided by a given method. To conduct both coarse and fine differentiation of the point cloud encoders, we introduce two mAP metrics: i) mAP_{cat} considers the object categories; and ii) mAP_{geo} measures geometric similarity of the 3D objects. Furthermore, we denote the precision metric for the two mAPs as P_{cat} and P_{geo} respectively. To compute each metric we translate the query and target subscenes so the centroid of the anchor object boxes are at the origin. See Figure 4 for an example. For mAP_{cat} we directly use the oracle categories of the corresponding objects. A match requires: i) same category; and ii) IoU above a threshold. We try various thresholds by partitioning the open interval between 0 and 1 into k equal subintervals. Note that mAP_{cat} only requires that two objects have the same category (e.g., chair), and completely ignores geometry. In contrast, mAP_{geo} considers geometric similarity, while keeping the IoU threshold condition. Specifically, a match requires Chamfer distance (CD) between the point clouds below a threshold τ_c. Assuming the object from the query subscene has category c, τ_c is a *category-dependent threshold computed for each* c. To find τ_c, we compute the CD for all pairs of boxes in the test data that have category c. We then sort the distances from smallest to largest and choose τ_c to be the distance at top 5%, top 10%, top 20%, or top 40%.

Note that the second matching criteria for both P_{cat} and P_{geo} relies on the IoU of each query object and its corresponding candidate in the world coordinate frame. Rejecting a candidate based on IoU may be strict for some applica-

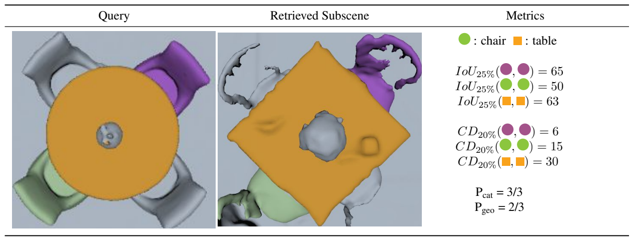


Figure 4. Example illustrating evaluation metrics. For P_{cat}, matches are corresponding objects with the same category and IoU above 25%. This results in a precision of 3/3. For P_{geo} the corresponding tables have a chamfer distance (CD) above the 20% threshold for the table category. Although the IoU values are above the 25% threshold, this mismatch results in a P_{geo} of 2/3.

tions. To evaluate results using a softer metric, we also consider the distance and angle of the query objects and their corresponding candidates relative to the anchor objects. See the supplement for results with the softer matching criteria.

MicroAvg and MacroAvg. A popular approach for evaluating self-supervised image representation models is to first apply the trained model (with frozen weights) on training and test images and extract feature representations per image. Then, a linear classifier is trained on the training features and evaluated on the test features using classification accuracy. However, as suggested in DINO [3], training the linear classifier introduces hyperparameters such as learning rate, number of training epochs, weight initialization/normalization strategies, etc. Therefore, we follow DINO [3] and instead use a weighted k-nearest neighbours [18] on the 3D representations derived from a point cloud encoder. For MICROAVG, we simply compute classification accuracy on test data across all classes. For MACROAVG, we first compute the per category classification accuracy (on test data) and then average those accuracies across all classes.

5.3. Methods

RANDOMRANK: Randomly assigns a box in each target scene to a box in the query subscene. The ranking strategy is the same as Section 4.3, except correspondence quality scores $\mathcal{QT}_m$ are not computed (no intrinsic similarities).

ORACLECATRANK: This baseline directly compares the oracle categories for each box in a corresponding candidate pair and randomly takes a pair with matching categories. The ranking strategy is the same as RANDOMRANK.

ORACLECATRANK[+IOU]: Same as ORACLECATRANK, except for retrieval we also take into account the IoU between the boxes in each candidate pair. More concretely, the candidate pair with matching oracle categories must also have a positive IoU. Note that this is the same extrinsic similarity criteria for POINTCROPRANK and CSCRANK as described in Section 4.3.

ORACLEGKRANK: An extension of Fisher *et al.* [7] graph kernel-based approach. While the original model only allows for 3D scene retrieval, we introduce modifications to enable 3D subscene retrieval. For each node q_i in the query subscene graph, we find a list of candidate nodes (from the target scene graphs) with the same oracle category as q_i. Next, we compute the kernel between each q_i and its corresponding candidate. Here, a kernel is computed by taking a rooted walk starting from the query node in the query subscene graph and a target node in the target graph. We then choose the target candidate with the highest kernel value. Finally, we rank the resulting target subscenes based on the number of corresponding objects they offer. If two subscenes have the same number of corresponding objects, we rank the subscene with the larger average kernel higher. Note that this baseline directly uses the oracle categories.

SUPERVISEDTRANSFORMER: Teacher network from POINTCROP but supervised with focal loss [13].

TRANSFORMERRANKDISC: Predicts category of each target box in a corresponding candidate pair using SUPERVISEDTRANSFORMER. The retrieval and ranking strategy is the same as Section 4.3, except the intrinsic similarity condition is replaced with **category matching**. The result of this design is a pure discrete subscene retrieval model.

TRANSFORMERRANK: Uses SUPERVISEDTRANSFORMER as its point cloud encoder. We use the same retrieval and ranking strategy as Section 4.3. This baseline can be thought of as the continuous version of TRANSFORMERRANKDISC.

CSCRANK: Uses CSC [9] as its point encoder. The retrieval and ranking strategy is the same as Section 4.3.

POINTCROPRANK: Our 3D subscene retrieval model using the POINTCROP teacher as its point cloud encoder, and the retrieval and ranking strategy described in Section 4.3.

POINTCROPRANKV2, **TRANSFORMERRANKV2**, **CSCRANKV2** and **CATRANKV2[+IOU]**: We train Liu *et al.*'s Group-Free 3D object detection model on Matterport3D [4] to localize the 3D objects in test scenes. POINTCROPRANKV2, TRANSFORMERRANKV2, and CSCRANKV2 are the result of applying POINTCROPRANK, TRANSFORMERRANK and CSCRANK on the predicted test boxes. CATRANKV2[+IOU] is the same as ORACLECATRANK[+IOU], except it utilizes the **predicted box categories** instead of the oracle categories.

5.4. Quantitative Evaluation

In this subsection, we compare our subscene retrieval model POINTCROPRANK against alternatives, in the presence or absence of ground truth object boxes. Furthermore, we measure the quality of the point encoders we have used for the 3DSSR task through object classification. In the supplement, we also evaluate the 3DSSR models on 3D object

Method	mAP_{cat}	mAP_{geo}	AUC (mAP_{cat})	AUC (mAP_{geo})
ORACLECATRANK	19.69	7.99	9.97	4.27
ORACLECATRANK[+IOU]	**27.73**	9.16	**13.12**	5.08
ORACLEGKRANK	17.22	5.69	8.70	2.76
TRANSFORMERRANK	16.61	**20.88**	8.87	**10.86**
TRANSFORMERRANKDISC	16.51	6.20	7.54	2.86
RANDOMRANK	0.68	1.45	0.36	0.64
CSCRANK	12.65	16.10	6.98	8.36
POINTCROPRANK	**14.94**	**25.21**	**8.06**	**13.14**

Table 1. Mean average precision and area under the curve (AUC) computed on 50 test queries. All models that use oracle categories are in the top group. For mAP_{cat} and mAP_{geo} our model (POINTCROPRANK) outperforms other models that have no access to oracle categories (bottom group). Furthermore, using mAP_{geo} our model even outperforms the models that directly use oracle categories (e.g., ORACLECATRANK[+IOU]) or are trained with category labels (i.e., TRANSFORMERRANK). Columns four and five show the AUC for each mAP metric at various IoU thresholds. For mAP_{geo}, POINTCROPRANK significantly outperforms all models. Moreover, when evaluating based on AUC of mAP_{cat}, we achieve the best performance among all models that have no access to oracle categories.

retrieval. Moreover, we add a rotation module to the 3DSSR models (so they can take into account rotated subscenes) and evaluate them.

3D Subscene Retrieval. We measure the quality of our model (POINTCROP) against alternatives for the 3DSSR task. We created 50 validation and 50 test query subscenes, isolated from the training data. Each query subscene in the val or test sets is compared against the remaining scenes in the val or test set. Table 1 shows mAP_{cat} and mAP_{geo} on 50 test queries with IoU and Chamfer distance thresholds of 25% and 20% respectively. In the supplement, we share mAP plots for various IoU and Chamfer distance thresholds. To create IoU thresholds, we partition the line from 0.05 and 0.95 into 10 equally spaced values. For Chamfer distance, we consider thresholds of 5%, 10%, 20% and 40%. Given the range of IoU thresholds and a Chamfer distance threshold of 20%, we show the area under the curve (AUC) for each subscene retrieval model in Table 1.

Recall that TRANSFORMERRANK, CSCRANK and POINTCROPRANK all use the same retrieval and ranking strategy from Section 4.3. Moreover, the point cloud encoder used for these subscene retrieval models has the same architecture. The only difference between these retrieval models is the training strategy for their encoder. Notably, the encoder for TRANSFORMERRANK (i.e., SUPERVISEDTRANSFORMER) is trained with category labels as opposed to the self-supervised encoders for CSCRANK and our POINTCROPRANK. This setup results in an apples-to-apples comparison between the training objectives of these encoders for the 3D subscene retrieval task. For mAP_{cat}, the AUC metric in Table 1 (column 4) indicates that our model (POINTCROPRANK) achieves better results than

Method	$\text{mAP}_{\text{cat\&geo}}$	AUC ($\text{mAP}_{\text{cat\&geo}}$)
ORACLECATRANK[+IOU]	9.16	5.08
TRANSFORMERRANK	9.80	**5.49**
CSCRANK	7.91	4.45
POINTCROPRANK	**9.91**	**5.49**
CATRANKV2[+IOU]	2.54	1.42
TRANSFORMERRANKV2	**6.71**	**3.64**
CSCRANKV2	4.46	2.42
POINTCROPRANKV2	5.87	3.40

Table 2. Comparing 3DSSR models when applied on ground truth versus predicted boxes. The top group shows models that retrieve subscene using the oracle test boxes. The bottom group shows results for the same models when applied to predicted test boxes. The performance gap between the two groups indicates that the SOTA 3D object detection models can not help solve 3DSSR in the absence of ground truth object localization.

the self-supervised alternative CSCRANK (15.5%). However, when computing AUC for mAP_{geo}, we observe that POINTCROPRANK significantly outperforms all models including TRANSFORMERRANK (21.0%). Note that mAP_{geo} is an evaluation metric that measures the geometric similarity of the retrieved corresponding objects.

Finally, the superiority of POINTCROPRANK and TRANSFORMERRANK over TRANSFORMERRANKDISC (based on AUC in Table 1) validates our hypothesis that *matching on discrete categories is not ideal for retrieving similar 3D subscenes. Instead, a continuous representation can achieve better results.*

3D Subscene Retrieval with Predicted Boxes. The models in the top group of Table 2 use the ground truth object localization at test time. However, in the bottom group, we apply the same models to the predicted test data boxes. To localize the objects we use the Group-Free [14] 3D object detection model trained on Matterport3D's [4] training scenes. For the evaluation metric, we take an and condition between P_{cat} and P_{geo} and compute $\text{mAP}_{\text{cat\&geo}}$ and its corresponding AUC. Our motivation behind combining P_{cat} and P_{geo} is to ensure the predicted boxes are semantically meaningful and geometrically similar to the query objects. For all results here we use the same IoU and Chamfer distance thresholds as in Table 1. To compute P_{cat} for a predicted box b_i we need a ground truth category. To obtain this, we take the category of all ground truth boxes that have an IoU of at least 0.25 with b_i.

The results in Table 2 demonstrate a performance gap between models that use oracle versus predicted test data boxes for 3DSSR. This indicates that solving 3DSSR in the most general case (i.e., with object localization) is very challenging and the SOTA 3D object detection model can not address that.

Object Classification. To measure the quality of the point

Model	MicroAvg	MacroAvg
SupervisedTransformer	**58.43**	**38.34**
CSC_SparseResUNet	38.90	18.97
CSC_PointNet++	-	-
CSC_Transformer	51.94	32.66
PointCrop	48.21	26.75

Table 3. Object classification on test set using 20-nearest neighbours [18]. We compare PointCrop against SupervisedTransformer and CSC [9] using different backbone architectures. Our model (PointCrop) achieves competitive accuracy.

cloud encoders used in 3DSSR, we evaluate them on a classification task. In Table 3 we compare the performance of PointCrop against CSC and SupervisedTransformer. The results suggest that PointCrop achieves competitive classification accuracies compared to alternatives. An important observation is that although SupervisedTransformer has the highest accuracy, this does not translate to the best performance in subscene retrieval (i.e., the TransformerRank model). In other words, a point cloud encoder with higher classification accuracy does not necessarily retrieve the most geometrically similar subscenes.

5.5. Qualitative Evaluation

Figure 5 shows two sets of test queries comparing our model PointCropRank against TransformerRank and CSCRank. Each scene is displayed as two top-down images showing the entire scene (top left) and the zoomed-in view of the subscene. In the first example, we compare our model PointCropRank against the supervised TransformerRank. Here, we observe that PointCropRank retrieves a subscene at rank 2 and 3 that are the most similar in geometry and object arrangement to the query subscene. Note that for TransformerRank the retrieved tables at all ranks are larger in size (supporting 8 chairs as opposed to 4) compared to the query table.

In the second example, we compare PointCropRank against another self-supervised model, CSCRank. Our model retrieves a chair at ranks 2 and 3 and cushion at rank 2. Here the retrieved subscene that is the most similar to the query subscene is from ours at rank 2. Note the geometric similarity between the chairs at rank 2 (in purple). However, we observe that CSCRank does not find any matches at ranks 1, 2 or 3. Please see the supplemental material for more qualitative examples.

6. Conclusion

Our work is subject to a number of limitations. Our results either assume object detections are given, or rely on an external detection approach. Incorporating 3D object detection into the subscene retrieval pipeline is an interesting direction for future work. Also, our retrieval and ranking strategy is based only on point cloud information and does not account for fine-grained color or texture appearance of the objects in each scene.

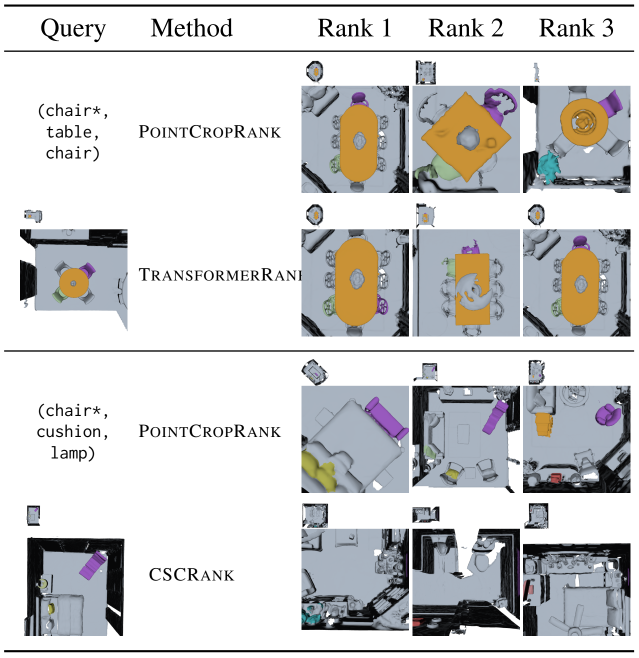


Figure 5. Qualitative results for two test set queries. Each row compares the ranked list of retrieved 3D subscenes for the query subscene shown on the left. Each scene is displayed as a pair of top-down images with the top-left showing the entire scene, and the bottom left showing a zoomed-in view of the subscene. The query and target objects (i.e., anchor objects) are in purple, while other context objects in each subscene are colored according to their category. We note that our PointCropRank approach retrieves subscenes from the 3D scene dataset that are more similar in geometry and object arrangement to the query subscene compared to supervised (TransformerRank) or self-supervised (CSCRank) approaches.

We presented the 3D subscene retrieval task, generalizing prior work on context-based 3D object retrieval and 3D scene retrieval. We believe that 3D subscene retrieval is a useful task that will find increasing use as 3D scene datasets continue to grow. Our key finding in building 3DSSR is that using a good continuous 3D shape representation leads to geometrically more similar retrieval results compared to using discrete object categories. In this paper, we presented PointCrop, our non-contrastive self-supervised encoder and used it for 3D subscene retrieval. Our experiments show that PointCropRank outperforms supervised approaches (trained using category labels) and prior self-supervised training frameworks on the 3D subscene retrieval task.

References

[1] Rohan Anil, Gabriel Pereyra, Alexandre Passos, Robert Ormandi, George E Dahl, and Geoffrey E Hinton. Large scale distributed neural network training through online distillation. *arXiv preprint arXiv:1804.03235*, 2018.

[2] Mathilde Caron, Ishan Misra, Julien Mairal, Priya Goyal, Piotr Bojanowski, and Armand Joulin. Unsupervised learning of visual features by contrasting cluster assignments. *Advances in Neural Information Processing Systems*, 33:9912–9924, 2020.

[3] Mathilde Caron, Hugo Touvron, Ishan Misra, Hervé Jégou, Julien Mairal, Piotr Bojanowski, and Armand Joulin. Emerging properties in self-supervised vision transformers. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 9650–9660, 2021.

[4] Angel Chang, Angela Dai, Thomas Funkhouser, Maciej Halber, Matthias Niessner, Manolis Savva, Shuran Song, Andy Zeng, and Yinda Zhang. Matterport3d. *International Conference on 3D Vision (3DV)*, 2017.

[5] Angela Dai, Angel X. Chang, Manolis Savva, Maciej Halber, Thomas Funkhouser, and Matthias Nießner. Scannet: Richly-annotated 3d reconstructions of indoor scenes. In *Proc. Computer Vision and Pattern Recognition (CVPR), IEEE*, 2017.

[6] Matthew Fisher and Pat Hanrahan. Context-based search for 3d models. *ACM SIGGRAPH Asia 2010 papers*, pages 1–10, 2010.

[7] Matthew Fisher, Manolis Savva, and Pat Hanrahan. Characterizing structural relationships in scenes using graph kernels. *ACM SIGGRAPH 2011 papers*, pages 1–12, 2011.

[8] Kaiming He, Haoqi Fan, Yuxin Wu, Saining Xie, and Ross Girshick. Momentum contrast for unsupervised visual representation learning. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 9729–9738, 2020.

[9] Ji Hou, Benjamin Graham, Matthias Nießner, and Saining Xie. Exploring data-efficient 3d scene understanding with contrastive scene contexts. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 15587–15597, 2021.

[10] Justin Johnson, Agrim Gupta, and Li Fei-Fei. Image generation from scene graphs. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 1219–1228, 2018.

[11] Justin Johnson, Ranjay Krishna, Michael Stark, Li-Jia Li, David Shamma, Michael Bernstein, and Li Fei-Fei. Image retrieval using scene graphs. *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 3668–3678, 2015.

[12] Ranjay Krishna, Yuke Zhu, Oliver Groth, Justin Johnson, Kenji Hata, Joshua Kravitz, Stephanie Chen, Yannis Kalantidis, Li-Jia Li, David A Shamma, et al. Visual genome: Connecting language and vision using crowdsourced dense image annotations. *International journal of computer vision*, 123(1):32–73, 2017.

[13] Tsung-Yi Lin, Priya Goyal, Ross Girshick, Kaiming He, and Piotr Dollár. Focal loss for dense object detection. *Proceedings of the IEEE international conference on computer vision*, pages 2980–2988, 2017.

[14] Ze Liu, Zheng Zhang, Yue Cao, Han Hu, and Xin Tong. Group-free 3d object detection via transformers. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 2949–2958, 2021.

[15] Rui Ma, Akshay Gadi Patil, Matthew Fisher, Manyi Li, Sören Pirk, Binh-Son Hua, Sai-Kit Yeung, Xin Tong, Leonidas Guibas, and Hao Zhang. Language-driven synthesis of 3d scenes from scene databases. *ACM Transactions on Graphics (TOG)*, 37(6):1–16, 2018.

[16] Robert Osada, Thomas Funkhouser, Bernard Chazelle, and David Dobkin. Shape distributions. *ACM Transactions on Graphics (TOG)*, 21(4):807–832, 2002.

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

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

[19] Saining Xie, Jiatao Gu, Demi Guo, Charles R Qi, Leonidas Guibas, and Or Litany. Pointcontrast: Unsupervised pre-training for 3d point cloud understanding. In *European conference on computer vision*, pages 574–591. Springer, 2020.

[20] Kai Xu, Rui Ma, Hao Zhang, Chenyang Zhu, Ariel Shamir, Daniel Cohen-Or, and Hui Huang. Organizing heterogeneous scene collection through contextual focal points. *ACM Transactions on Graphics, (Proc. of SIGGRAPH 2014)*, 33(4):35:1–35:12, 2014.

[21] Hengshuang Zhao, Li Jiang, Jiaya Jia, Philip H.S. Torr, and Vladlen Koltun. Point transformer. In *Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV)*, pages 16259–16268, October 2021.