Fine-tuning 3D foundation models for geometric object retrieval

Jarne van den Herrewegen, Tom Tourwé, Maks Ovsjanikov, Francis Wyffels

Computers & Graphics 122 (2024) 103993

Contents lists available at ScienceDirect

Computers & Graphics

journal homepage: www.elsevier.com/locate/cag

Special Section on 3DOR2024

Fine-tuning 3D foundation models for geometric object retrieval

Jarne Van den Herrewegen [a,b,*], Tom Tourwé [a], Maks Ovsjanikov [c], Francis wyffels [b]

[a] Oqton AI, Belgium
[b] AI and Robotics Lab, IDLab-AIRO, Ghent University - imec, Belgium
[c] LIX, Ecole Polytechnique, France

ARTICLE INFO

Keywords:
Object retrieval
Deep learning
3D
Transfer learning
Foundation models
Self-supervised learning

ABSTRACT

Foundation models, such as ULIP-2 (Xue et al., 2023) recently projected forward the field of 3D deep learning. These models are trained with significantly more data and show superior representation learning capacity in many downstream tasks like 3D shape classification and few-shot part segmentation.

A particular characteristic of the recent 3D foundation models is that they are typically *multi-modal*, and involve image (2D) as well as caption (text) branches. This leads to an intricate interplay that benefits all modalities. At the same time, the nature of the *3D* encoders alone, involved in these foundation models is not well-understood. Specifically, there is little analysis on the utility of both pre-trained 3D features provided by these models, or their capacity to adapt to new downstream 3D data. Furthermore, existing studies typically focus on label-oriented downstream tasks, such as shape classification, and ignore other critical applications, such as 3D content-based object retrieval.

In this paper, we fill this gap and show, for the first time, how 3D foundation models can be leveraged for strong 3D-to-3D retrieval performance on seven different datasets, on par with state-of-the-art view-based architectures. We evaluate both the pre-trained foundation models, as well as their fine-tuned versions using downstream data. We compare supervised fine-tuning using classification labels against two self-supervised label-free fine-tuning methods. Importantly, we introduce and describe a methodology for fine-tuning, as we found this to be crucial to make transfer learning from 3D foundation models work in a stable manner.

1. Introduction

Content-based object retrieval is an essential task in Computer-Aided Design/Manufacturing, medicine, AR/VR and game development among other domains. Modern retrieval systems represent 3D objects with feature vectors, allowing fast comparison of 3D objects in this vector space. Good features can capture geometric and potentially semantic properties of 3D shapes. First, such features were defined in a purely axiomatic manner [1], while more recently, learned features have gained prominence [2].

Initial learning-based approaches were trained on subsets of target datasets, and were thus data and task-specific. Later works have advocated for *transferring* features trained for one task to other downstream applications in the 3D setting [3,4]. These approaches do not offer general-purpose task-agnostic features, hence 3D neural networks are typically still initialized with random weights instead of pre-trained weights. This is in contrast to other fields, like 2D vision, where foundation models that are pre-trained on internet-scale datasets have been shown to lead to powerful generic features that can be used in a wide range of downstream tasks [5]. The benefits of transferring such generic features include faster convergence, higher performance and improved data efficiency.

In 3D, several candidates for foundation models have been proposed recently [6–8], demonstrating strong generalization capabilities. However, the utility of the 3D features that these models offer has not been studied in detail. This is particularly the case since these foundation models are multi-modal (with multiple branches for different modalities). Most attention went to multi-modal capabilities and there has been little analysis on the nature of *the 3D features* trained with these models. Specifically for 3D-to-3D retrieval, there exists no work which explores the application of these foundation models.

Given the strong performance in other domains, these models have the potential to bring a new approach to content-based 3D object retrieval, potentially delivering to this area as in image or natural language processing. In our work we provide the first in-depth study of the utility of features from recent foundation models in the context of 3D object retrieval.

Among the proposed 3D foundation models, we focus on the Point-BERT [9] architecture presented in ULIP-2 [6]. Compared to other

* Corresponding author at: AI and Robotics Lab, IDLab-AIRO, Ghent University - imec, Belgium.
E-mail addresses: jarne.vandenherrewegen@oqton.com, jarne.vandenherrewegen@ugent.be (J. Van den Herrewegen), tom.tourwe@oqton.com (T. Tourwé), maks@lix.polytechnique.fr (M. Ovsjanikov), francis.wyffels@ugent.be (F. wyffels).

https://doi.org/10.1016/j.cag.2024.103993
Received 12 April 2024; Received in revised form 3 June 2024; Accepted 25 June 2024
Available online 3 July 2024

works like Uni3D [8] and Openshape [7], the ULIP-2 PointBERT strikes a good balance between performance (stronger than Openshape) and accessibility (more practical than Uni3D).

We test this model extensively on seven datasets from various domains (household, medical, manufacturing, scans) to observe a very strong generalization performance of learned features. We compare the performance of the pre-trained model to training from scratch on each dataset as well as when *adapting* the model to specialized domains. For this we evaluate three different fine-tuning approaches across all datasets, while comparing to numerous baselines and sharing our insights on the training dynamics.

To summarize our contributions are the following:

- We show that the features provided by the recent 3D foundation models are *out-of-the-box* competitive with existing point cloud-based architectures for retrieval tasks on datasets from various domains.
- We demonstrate that, with appropriate methodology, these features can be fine-tuned to a target domain through supervised classification for unprecedented, state-of-the-art results on 3D object retrieval with point cloud-based architectures, even considering *only* the 3D branch, and without any image or text information. The presented advancements in 3D object retrieval bring a point-cloud based architecture (PointBERT [6,9]) on par with the state-of-the-art performance of view-based methods, with superior practical applicability.
- Next to supervised training, we also investigate the possibility to fine-tune the foundation model with two self-supervised methods. While the gains are not as large as for supervised fine-tuning, we still observe an improvement of the performance beyond the previous state-of-the-art.
- We also remark that the fine-tuning process can lead to training instabilities, and propose an adapted methodology to address this issue.

In the remainder of this paper, we first highlight related works and give background information on 3D object retrieval. Next, we detail the fine-tuning methods used in this paper. Finally, we cover the experiments and the new-found insights.

2. Background & related work

To motivate our work, we highlight existing literature on 3D object retrieval, foundation models for 3D data and self-supervised learning.

2.1. 3D object retrieval

Content-based 3D retrieval allows 3D artists, designers, and engineers to initiate their workflow from an existing reference object, significantly reducing the required time for a design task [10]. Vector-based methods have been one of the important 3D lookup techniques in the past 30 years [11].

Before 2010, extracting feature vectors happened through algorithms based on 2D projections [12] or harmonical decompositions [13] for example. For an extensive overview, we refer to Tangelder et al. [1]. Later, advancements in the deep learning domain made feature extraction with neural networks accessible [14]. Neural networks can *learn* useful vector representations of 3D objects given raw input as 3D voxels [15], 2D projections [2], point clouds [16], meshes [17] and signed distance fields [18].

In recent years, the field of object retrieval has been dominated by neural networks as feature extractors. There are two common families of architectures: view-based methods such as MVCNN [2], and point cloud-based models such PointNet++ [16]. In this work, we will focus on point cloud-based architectures, as we are interested in the pure 3D features learned by the 3D foundation models. For practical application, these architectures also come with no rendering overhead.

State-of-the-art models for 3D shape similarity [19–21] have typically used classification labels in their training scheme. Obtaining these labels is a time-consuming and costly process in the development of new retrieval systems. 3D foundation models could enable transfer learning in the 3D domain and improve the label efficiency significantly [22] for developing new retrieval systems. The goal of this work is to investigate the transfer learning capabilities of recent point cloud-based foundation models, either out-of-the-box or by fine-tuning them to specialized domains.

2.2. Multi-modal foundation models for 3D data

In the past months, several works, including ULIP-2 [6], OpenShape [7] and Uni3D [8], showed impressive results in 3D classification, zero-shot classification and 3D segmentation with multi-modal training approaches. Each of these works trained 3D encoders to align with pre-trained image-text encoders in latent space. The image-text encoders, e.g., CLIP [22] or SLIP [23], consist of a text encoder and an image encoder that are trained to project image-text pairs close in latent space. The training procedure of such models involves several hundred million to several billion image-text pairs, resulting in models with broad semantic knowledge. To align 3D encoders to the same latent space as the image-text encoders, triplets with image, text and point clouds are necessary. The triplets are typically generated by rendering images from 3D meshes, sampling the meshes for the point cloud and captioning [24,25] the renders to obtain the text. Through the multi-modal alignment approach, it is possible to leverage these powerful models in the 3D field.

The second key element to the success of multi-modal approaches in the 3D field, is the Objaverse [26] dataset. This web-scraped dataset contains 800k qualitative 3D objects, which is much more than most other accessible datasets such as ShapeNetCore55 [27] or the Mechanical Components Benchmark [28]. Importantly, most samples are *textured* meshes, which makes it possible to render images that are colorized (as opposed to the ABC dataset [29]). Colorized images are closer to the input distribution of the existing image-text encoders. To observe the large improvement brought by the Objaverse dataset, we refer to ULIP [30] (trained on ShapeNet) and to ULIP-2 [6] (trained on Objaverse).

In this work we focus on the ULIP-2 pre-trained PointBERT as it provides a good trade-off between performance and practical use. The ULIP-2 PointBERT has 21M parameters compared to 5M parameters in the OpenShape PointBERT, which is significantly more but still practical to run and to fine-tune on one GPU. However the Uni3D transformers are much larger, available models range 80M to 1B parameters, which are more unstable for fine-tuning and are harder to fit on one GPU.

2.3. Self-supervised 3D deep learning

To overcome labeling efforts in the development of 3D retrieval systems, self-supervised methods can provide alternative training objectives with competitive performance [34]. Self-supervised methods include auto-encoding, contrastive learning and generative modeling. These methods use training tasks for which training targets can be synthetically generated, given a collection of (unlabeled) 3D objects. In this work, we apply two forms of contrastive learning as alternative to the supervised classification fine-tuning. There are two motivations for the usage of such methods for object retrieval: (1) 3D contrastive methods are trained with a similarity metric for vector representations of 3D objects, which is a close proxy for the downstream task and (2) 3D contrastive methods have shown strong results in the past for training similarity models from scratch [34–36].

The first form is multi-modal contrastive learning as introduced in Section 2.2 and further detailed in Section 3.2. In the past, several

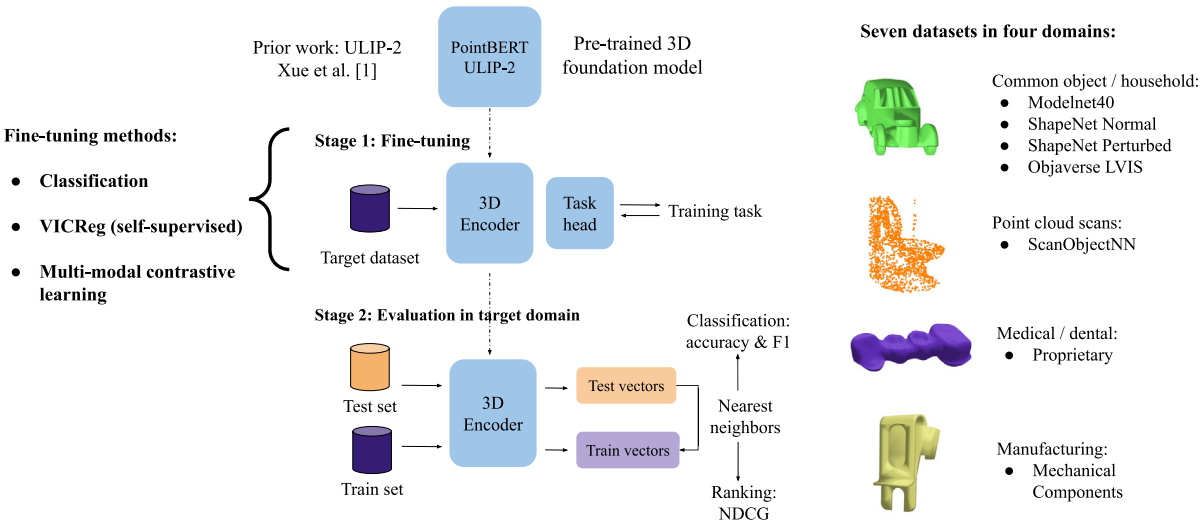


Fig. 1. A paper overview showing how the ULIP-2 [6] PointBERT foundation model is fine-tuned for object retrieval in Stage 1 and evaluated in Stage 2. In Stage 1, one of three fine-tuning methods is used: supervised classification, self-supervised VICReg [31] or multi-modal constrastive learning [30]. Each method has its own specific task head that is removed after fine-tuning. In Stage 2, the 3D encoder is evaluated by encoding the test and train split of a dataset and performing nearest neighbor search on the vectors. The retrieval performance is measured through nearest neighbor classification (NN acc, NN F1) and the NDCG ranking metric. The fine-tuning and evaluation stage are applied to seven datasets from various domains: ModelNet40 [32], ShapeNet [27], a subset of Objaverse- LVIS [26], ScanObjectNN [33], a proprietary dataset and the Mechanical Components Benchmark [28].

works have combined the 2D modality with 3D architectures for self-supervised learning [35,36]. The successes of image-text encoders in CLIP-style models [22] have drawn interest toward multi-modal self-supervised learning with 2D, text *and* 3D, i.e., PointCLIP [37] and the first ULIP [30] work. With the release of the Objaverse dataset [26], these methods gained significantly more attention for pre-training large 3D encoders. In our setting, we use the image/text/point-cloud contrastive learning as one of the *fine-tuning* methods. Given that the pre-trained 3D encoder in our experiments was originally trained with multi-modal contrastive learning, the method forms an interesting candidate for further fine-tuning the model.

Secondly, we will apply VICReg [31], a contrastive learning technique based on Siamese networks [38,39]. A Siamese network setup projects *pairs* of similar samples in the same latent space and learns to bring the latent vectors as close as possible. In the self-supervised setting, these pairs are created artificially [40] instead of manually annotating paired samples as (dis)similar. The key challenge for Siamese networks is to prevent the neural network from collapsing to the trivial solution where it always predicts the same vector. The VICReg method addresses this challenge through auxiliary loss functions that penalize the neural network for not having enough variance in each prediction batch. Historically, methods using negative samples [41] have been more popular, but require either more resources to accommodate large batch sizes or bring more complexity for negative mining.

Next to the multi-modal contrastive learning, VICReg is an interesting candidate for fine-tuning because (1) it has proven to give strong results in 3D object retrieval [34] and (2) the practical implementation is less complex (no need for large image-text models during training, nor for rendering images and generating text descriptions).

3. Method

As mentioned, our key goal is to evaluate the utility of pre-trained foundation 3D models in the context of content-based object retrieval. For this, we both evaluate the accuracy of the "out of the box" pre-trained model, as well as its fine-tuned versions using downstream data in several different datasets. To achieve accurate results, the fine-tuning process plays a critical role. Below we detail the three different fine-tuning methods applied during our experiments. Each of the three methods adds a different head to the pre-trained backbone from the foundation model, uses a different training objective and handles the training data in a different manner. The final 3D encoder that is used in the retrieval system always has the same architecture however, as shown in Fig. 1.

3.1. Fine-tuning with classification

3D feature extractors for object retrieval have primarily been trained with a classification objective in the past and has consistently delivered state-of-the-art performance [19,20,42]. To fine-tune a 3D encoder through classification in our experiments, we add a small MLP to the backbone as head and use the cross-entropy loss. The head has two hidden layers and an output layer with dimensions 512, 256 and the number of classes. Before each fully-connected layer (except the first), there is a batch normalization layer, an ReLU activation function and a dropout layer with a dropout probability of 0.2%.

3.2. Multi-modal fine-tuning

The second fine-tuning method used in this work is multi-modal alignment in latent space, similar to the pre-training method of the recent 3D foundation models such as ULIP-2 [6]. Starting from a dataset with only 3D objects, we render images from the 3D objects and generate captions for the images. With the image/text/point-cloud triplets, a 3D encoder is fine-tuned to align with pre-trained image-text encoders. The image and text encoders used in this work, are the same as in the ULIP-2 work and originate from SLIP [23]. We detail the architecture, contrastive loss and data generation in the following subsections.

3.2.1. Architecture

In the multi-modal fine-tuning setup, the 3D encoder is extended with one linear layer, the *projection* layer. The same goes for the image encoder (Vision Transformer [43], Base model) and the text encoder (Transformer [44] defined in SLIP [23]). Both the pre-trained image and text encoders do not receive updates to their weights during our

fine-tuning process and are solely used to provide training targets in latent space for the 3D encoder. The shared latent space has 512 dimensions.

3.2.2. Multi-modal contrastive loss

Given a image/text/point-cloud triplet $(X^{\mathrm{I}}, X^{\mathrm{T}}, X^{\mathrm{P}})$, the three encoders respectively return latent vectors $(\mathbf{z}^{\mathrm{I}}, \mathbf{z}^{\mathrm{T}}, \mathbf{z}^{\mathrm{P}})$ that have the same dimensionality (512) and share the same latent space. Similar to the training objective for CLIP-style models, the multi-modal objective for 3D foundation models uses contrastive learning. A batch of N triplets is first mapped by the three encoders to the latent vectors $([\mathbf{z}^{\mathrm{I}}_1, \dots, \mathbf{z}^{\mathrm{I}}_N], [\mathbf{z}^{\mathrm{T}}_1, \dots, \mathbf{z}^{\mathrm{T}}_N], [\mathbf{z}^{\mathrm{P}}_1, \dots, \mathbf{z}^{\mathrm{P}}_N])$. Next, each point cloud vector is forced to be close to the image and the text vector from the same triplet. At the same time, each point cloud vector is also pushed away from the image and the text vector originating from other triplets in the batch. The multi-modal constrastive loss L_{MMCL} is formulated as follows, assuming all vectors $\mathbf{z}$ are normalized:

$$L_{\mathrm{MMCL}} = \frac{1}{4}\sum_{i=1}^{N}\left(\log\frac{\exp(\mathbf{z}_i^{\mathrm{P}}\cdot\mathbf{z}_i^{\mathrm{T}})}{\sum_{j=1}^{N}\exp(\mathbf{z}_i^{\mathrm{P}}\cdot\mathbf{z}_j^{\mathrm{T}})} + \log\frac{\exp(\mathbf{z}_i^{\mathrm{T}}\cdot\mathbf{z}_i^{\mathrm{P}})}{\sum_{j=1}^{N}\exp(\mathbf{z}_i^{\mathrm{T}}\cdot\mathbf{z}_j^{\mathrm{P}})} + \log\frac{\exp(\mathbf{z}_i^{\mathrm{P}}\cdot\mathbf{z}_i^{\mathrm{I}})}{\sum_{j=1}^{N}\exp(\mathbf{z}_i^{\mathrm{P}}\cdot\mathbf{z}_j^{\mathrm{I}})} + \log\frac{\exp(\mathbf{z}_i^{\mathrm{I}}\cdot\mathbf{z}_i^{\mathrm{P}})}{\sum_{j=1}^{N}\exp(\mathbf{z}_i^{\mathrm{I}}\cdot\mathbf{z}_j^{\mathrm{P}})}\right). \tag{1}$$

In each term, a vector in one modality, e.g. $\mathbf{z}_i^{\mathrm{P}}$ for point cloud X_i^{P} in the first term, is compared to each vector of a different modality with the cosine similarity metric – represented here as the dot product between normalized vectors. The other operations in each term are a softmax function and a cross-entropy loss combined, penalizing the model when the inferred vector in one modality, e.g. $\mathbf{z}_i^{\mathrm{P}}$, has a low cosine similarity to its counterpart $\mathbf{z}_i^{\mathrm{T}}$ in a different modality, text T in the first term, compared to the text vectors $\mathbf{z}_j^{\mathrm{T}}$ from other triplets. There are four loss components in total, combining the point cloud modality with either text or image in one of two possible directions. The combination of text and image is not considered as the text-image encoders are kept frozen.

3.2.3. Data generation

Obtaining image/text/point-cloud triplets through manual labeling is a labor-intensive task. In our work, we generate the images and text in an automated manner, similar to the methodology in ULIP-2 [6]. Given a dataset with meshes, images are rendered from several viewpoints and text descriptions are generated for each render using a captioning model. Specifically, we render 4 images for each mesh from equally spaced view angles with the Objaverse [26] rendering library[1] based on Blender [45]. For describing the images, we tried the BLIP-2 [24] model as done in ULIP-2. However, we found that BLIP-2 captions are basic, are not very detailed and that in certain cases, the model starts repeating itself. We explored more recent vision-language models and settled with Llava [25], a multi-modal conversation model. While the Llava 1.5 7B model also shows hallucinations, the provided descriptions of images contain more details than BLIP-2 and the model output is easy to manipulate through prompting, see Fig. 2 for an example.

For each rendered image, we generate 4 descriptions with 4 different prompts:

- *What do you see in this image?*
- *This is a render of a 3D model. Ignore the background. Can you give a detailed description of the 3D model in the image?*
- *What do you see in the image? Can you give a detailed description of the 3D model in the image? Do not say anything about the colors in the image, nor about the background. Say things about the geometry of the object in the image.*
- *What do you see in the image? Give 1 sentence with the category of the object and 1 sentence about the shape and geometry of the object.*

[1] github.com/allenai/objaverse-rendering.

Prompt: "This is a render of a 3D model. Ignore the background. Can you give a detailed description of the 3D model in the image?"

BLIP-2: "The white chair has a metal frame and an open seat with two wheels and two wheels."

Llava 1.5 7B: "In the image, there is a white plastic chair with a silver metal frame. The chair is designed to be lightweight and portable, making it suitable for various settings. The chair is currently empty, with no one sitting on it. The overall appearance of the chair is clean and minimalistic, making it a versatile and functional piece of furniture."

Fig. 2. A captioning example showing how Llava [25] vision-language model returns more elaborate descriptions than BLIP-2 [24].

3.3. Self-supervised fine-tuning with VICReg

The final fine-tuning method is VICReg [31], a Siamese network setup that learns representations in a self-supervised manner. Where an unregularized Siamese network is unstable to train [39], VICReg uses auxiliary loss functions to stabilize the training procedure. In what follows, we discuss the Siamese network scheme, the regularizing loss functions and the self-supervised data generation setup.

3.3.1. Siamese network setup

A Siamese network learns to predict similar vectors for paired samples. Each sample in the pair is encoded separately and is then compared in latent space through a differentiable similarity function. Specifically, each pair with point clouds $X_1, X_2 \in \mathbb{R}^{N\times 3}$ is projected onto latent vectors $\mathbf{z}_1, \mathbf{z}_2 \in \mathbb{Z}^D$ by a neural network f_θ. Following the original VICReg [31] work, we use the mean-squared Euclidean distance as differentiable similarity loss L_{Eucl}.

The model f_θ consists of a 3D backbone and a projection head. The head consists of two hidden linear layers and an output layer, all of dimension 1024. The head uses batch normalization and the ReLU activation function. While one may consider to keep the training head for inference – it learns a similarity metric during training that might be good for 3D object retrieval –, empirical evidence and theory have shown how this does not give the best results [46].

3.3.2. Auxiliary losses

Applying a Siamese network setup as described so far, would result in a model that collapses to predicting a constant vector. Always predicting the same vector, regardless of the input, satisfies the similarity objective L_{Eucl}. VICReg brings regularizing loss functions to avoid the trivial solution. VICReg stands for *Variance-Invariance-Covariance Regularization*, indicating there are three components in the loss function. The *Invariance* component is the similarity loss L_{Eucl} that was described in Section 3.3.1.

The *Variance* loss is the main regularizer in VICReg and forces the network output to have a minimal amount of variance in each batch. This effectively pushes the network away from the constant solution. The variance loss L_{var} measures the standard deviation in each vector dimension in a batch and penalizes the network when the measured value is below 1 in a dimension. For a batch of N output vectors $[\mathbf{z}_1, \dots, \mathbf{z}_n]$, the variance loss L_{var} is calculated as follows:

$$L_{\mathrm{var}}([\mathbf{z}_1, \dots, \mathbf{z}_n]) = \frac{1}{D}\sum_{i=1}^{D}\max(1 - std(\mathbf{z}^i), 0)\,, \tag{2}$$

were $\mathbf{z}^i$ are the N values of dimension i in the batch.

The *Covariance* loss learns the network to decorrelate the dimensions in vector space as much as possible. In other words, the covariance loss encourages the network to maximize the information content

per dimension. This improves the quality of the learned representations and improves the stability of the training. The covariance loss L_{cov} is calculated by first estimating the covariance matrix in a batch Z with N output vectors $[\mathbf{z}_1, \ldots, \mathbf{z}_n]$. Next, L_{cov} is calculated as the sum of the squared covariances, the off-diagonal elements in the covariance matrix., which can be found on the off-diagonal elements.

Finally, all VICReg loss components are combined in a weighted sum, where we follow the values in the VICReg [31] paper and give a relative weight of 25 to the Euclidean loss L_{Eucl} and the variance loss L_{var}.

3.3.3. Data generation

The key factor to self-supervised learning with Siamese networks, lies in the generation of similar pairs [40]. Pairs are created by applying randomized augmentations twice to one object. As the two new objects originate from the same object, they are grouped in a pair that is labeled as *similar*. This artificial way of generating labels allows training a neural network without requiring any human annotation. In this work, we use the following stochastic point cloud transformations as augmentations, assuming the point cloud is normalized to the unit sphere:

1. Point cloud subsampling from 16 000 to 8192 points.
2. Rotation around all axes or one axis, depending on the alignment in the dataset.
3. Anisotropic scaling, with factors t_x, t_y, $t_z \sim \mathcal{U}^3(0.7, 1.25)$.
4. Isotropic scaling the object, with factor $s \sim \mathcal{U}(0.7, 1.25)$.
5. Jittering each point in the point cloud, with displacements d_x, d_y, $d_z \sim \mathcal{N}^3(0, 0.005)$ with a maximum absolute value of 0.05.
6. Translating all points with a global displacement vector v_x, v_y, $v_z \sim \mathcal{U}^3(-0.07, 0.07)$.

4. Experiments

Our experiments demonstrate how the recent 3D foundation models can be leveraged for unprecedented results in 3D-to-3D retrieval with point cloud-based architectures. We quantify the retrieval performance of the learned vector representations with nearest neighbor classification and ranking performance. The full evaluation protocol for retrieval models is explained in Section 4.1. To test the generalization capabilities of the presented approaches, seven different datasets are considered from various domains (household, medical, manufacturing, scans), detailed in Section 4.2. Our derived insights are discussed in Section 4.3. Finally we extend on the intricacies of the fine-tuning process in Section 4.4.

4.1. Evaluation protocol

To compare the three different fine-tuning methods for object retrieval, we evaluate trained 3D encoders as shown in Fig. 1. The task head is removed from the model and the backbone encoder is evaluated as embedding model in a retrieval system. To evaluate an encoder on a dataset, the test and train samples are encoded as vectors. Next, we do a nearest neighbor search with the cosine distance for each test sample in vector space to obtain a ranked list of retrieval results, as in Stage 2 Fig. 1. We calculate three different metrics with the ranked lists: nearest neighbor classification accuracy (NN acc), F1-score (NN F1) and Normalized Discounted Cumulative Gain (NDCG).

As a proxy for retrieval performance, **nearest neighbor classification** is a popular evaluation task. We consider accuracy and F1-score as metrics for scoring nearest neighbor classification. Concretely, we retrieve the closest neighbor in the train set for each test sample and use the classification label of the retrieved sample as prediction for the test sample class. The accuracy is the percentage of correct predictions over all samples. The F1-score is the averaged harmonic mean of the accuracy and the precision per class.

The **ranking performance** is evaluated through the NDCG metric. This involves retrieving the N closest training objects in the latent space for each test sample and ranking them based on their distance. The Discounted Cumulative Gain metric assigns a basic score of 1 to each training sample with the same class as the test query sample, with the score discounted logarithmically based on its ranking position. The highest score is achieved when all correct class training samples top the list. The sum of the basic scores is normalized with the maximum possible score to obtain the Normalized Discounted Cumulative Gain (NDCG). The NDCG value reported is the average across all test samples in an evaluation dataset. We use $N = 100$ in all of our experiments.

4.2. Datasets

To have an extensive empirical evaluation of the different fine-tuning methods, we consider seven datasets in our experiments. These datasets differ in volume, alignment and source domain. See Fig. 3 for samples.

The **ModelNet40** [32] (M40) dataset, Fig. 3(a), contains household objects: chairs, tables, stairs ... Several other common types are also present such as airplanes and cars. There is a total of 12K samples in the dataset, of which 9843 in the train set and 2468 in the test set. The total number of classes is 40. Each object class has a canonical orientation which implies it is better to not use rotation augmentations [34,47].

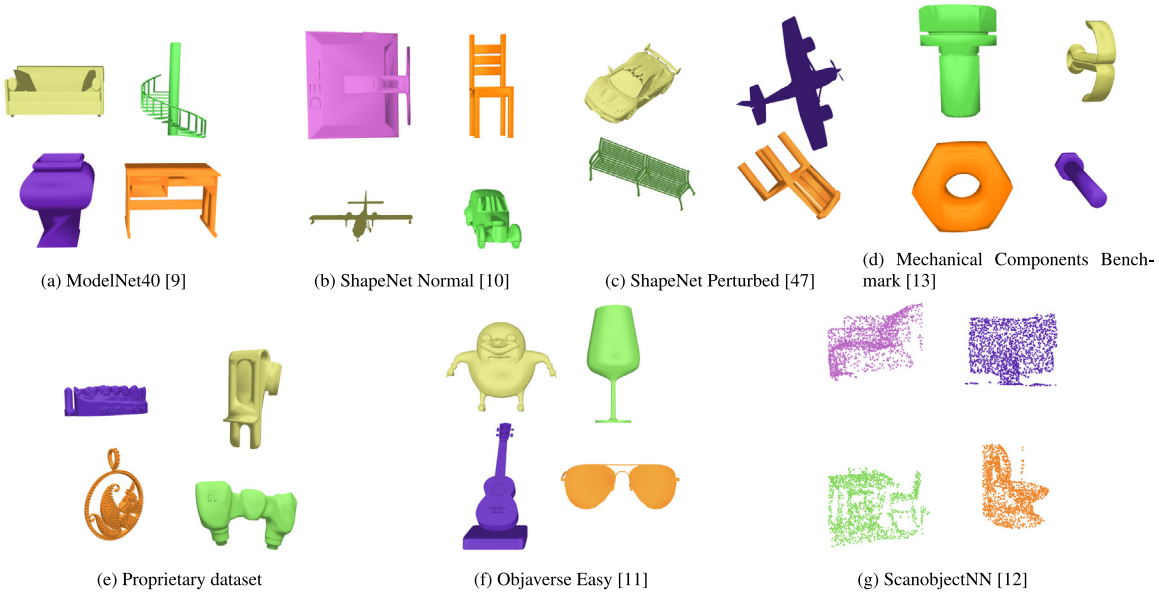
ShapeNet Normal (SN Norm), Fig. 3(b), and **ShapeNet Perturbed** (SN Per), Fig. 3(c), are datasets sampled from the ShapeNet repository [27]. There are 55 classes in each dataset, with similar classes to ModelNet40: household scenery and everyday objects. While ShapeNet Normal is an aligned dataset, ShapeNet Perturbed is a modified version of ShapeNet Normal, where all samples underwent a random rotation [48]. During any training in our experiments, we used rotation augmentations for the perturbed version but not for training with the normal version. Each dataset contains 51K samples, split into 41K training samples and 10K test samples.

To diversify the data in our experiments, we included the **Mechanical Components Benchmark** [28] (MCB) dataset, Fig. 3(d), offering objects related to manufacturing and construction: gears, nuts, bearings ... There is a total 58K objects divided over 68 classes. The training split has 46K samples and the test split has 11K samples. While many objects have a canonical pose, not all objects are aligned and therefore we use rotation augmentations during our experiments with MCB.

Amongst our experiments, we present the results on a **proprietary dataset** (Prop), Fig. 3(e), with objects mostly from the dental domain (80%) and other domains (20%) such as manufacturing and jewelry. The dataset contains 16K objects over 49 classes, with 12K samples in the train set and 3401 in the test set. We used rotation augmentations for this dataset as the objects are not aligned.

While the **Objaverse** [26] work mainly focuses on making 3D data available in a large volume, the authors also presented a small subset with crowdsourced classification labels, Objaverse-LVIS. The subset contains 46K samples over 1156 classes and has predominantly been used as an evaluation dataset for zero-shot classification of multi-modal 3D foundation models. Given the heavy class imbalance, class confusion and label noise, we reduced this dataset size to 7993 by only including classes with at least 100 samples. We call this subset **Objaverse Easy** (Obja Easy), Fig. 3(f), throughout our experiments. The dataset represents 76 classes with 6.142 training samples and 833 test samples. The target domain is household objects and animals. Generally, the objects in Objaverse are aligned. As such, we did not use rotation augmentations for our experiments with this dataset.

ScanObjectNN [33] is a benchmark for classifying noisy point cloud scans, Fig. 3(g). The dataset holds 2902 samples in 15 categories. As all other datasets in our experiments use synthetic 3D data, ScanObjectNN forms an interesting complement. These scans are aligned to a common gravitational axis (y-axis), we included rotations around this axis in our experiments. There exist several subsets of ScanObjectNN, each increasing amount of perturbations. We use the vanilla *OBJ_ONLY* subset.



(a) ModelNet40 [9] (b) ShapeNet Normal [10] (c) ShapeNet Perturbed [47] (d) Mechanical Components Benchmark [13]

(e) Proprietary dataset (f) Objaverse Easy [11] (g) ScanobjectNN [12]

Fig. 3. Examples from each of the seven datasets. We discern four source domains for the 3D objects: household, medical, manufacturing and scans. Household objects and other common objects are found in ModelNet40, the ShapeNet datasets and Objaverse Easy (see Section 4.2 for a definition). The Mechanical Components Benchmark contains manufacturing data. The proprietary dataset contains medical samples, in particular dental-related. The scanned data can be found in ScanObjectNN.

Table 1
Overview of retrieval performance expressed with the nearest neighbor classification test accuracy (NN acc) metric as explained in Section 4.1. The bottom section shows the strong results for the pre-trained PointBERT ULIP-2 model. The PointNet++ section provides the baseline results for prior work with point cloud-based architectures. The PointBERT section in the center of the table serves as a comparison with the bottom section to see the effect of the ULIP-2 multi-modal pre-training. Each cell of the table shows the test result of a separate training using the training method + model on the left of the row and the target dataset on the top of the column. There are three training methods considered in this table: supervised classification, self-supervised VICReg [31] and multi-modal contrastive learning (MMCL) [30], we refer to Section 3 for more detail. The **bold** numbers represent the highest (best) score for each dataset.

Training method\Dataset	MN40	SN Norm	SN Per	MCB	Prop	Obja Easy	ScanObjectNN
PointNet++							
Random weights	76.0	74.9	38.3	83.1	55.9	55.9	41.7
Classification	90.2	84.7	73.1	95.4	71.0	72.4	79.8
VICReg	88.7	83.4	74.2	94.7	71.7	73.4	75.2
PointBERT							
Random weights	78.0	75.9	40.4	88.4	56.8	57.1	43.2
Classification	89.6	85.2	78.3	96.4	71.7	75.0	82.3
VICReg	85.5	81.2	68.1	95.1	64.3	68.8	56.3
MMCL	86.6	83.5	76.4	95.6	66.5	75.6	–
PointBERT ULIP-2							
Pre-trained	88.7	86.6	78.5	96.3	77.6	85.1	76.3
FT w/ classification	**93.1**	**89.1**	**84.8**	**97.4**	**82.6**	**88.1**	**93.2**
FT w/ VICReg	90.7	86.4	81.9	96.1	79.4	84.0	78.2
FT w/ MMCL	91.5	86.5	80.7	96.4	78.1	87.1	–

Table 2
Comparison between state-of-the-art view-based methods View-GCN [20], MVTN [21] and our best performing point cloud-based approach expressed with the nearest neighbor classification test accuracy (NN acc) metric as explained in Section 4.1. The three methods are competitive, with a slight advantage for the classification fine-tuned ULIP-2 PointBERT in 4 out of 7 cases. Each cell represents a separate training of the fine-tuning method on the left and the dataset on the top. The **bold** numbers represent the highest (best) score for each dataset.

Training method\Dataset	MN40	SN Norm	SN Per	MCB	Prop	Obja Easy	ScanObjectNN
View-GCN [20]							
FT w/ classification	**93.1**	86.9	83.1	97.3	80.3	**88.4**	–
MVTN [21]							
FT w/ classification	92.7	88.9	**85.2**	97.1	79.5	87.8	91.8
PointBERT ULIP-2							
FT w/ classification (ours)	**93.1**	**89.1**	84.8	**97.4**	**82.6**	88.1	**93.2**

4.3. Retrieval results discussion

To investigate the possibilities of ULIP-2 as foundation model for object retrieval, we compare the three fine-tuning approaches to each other, to baseline methods and to existing art. In Table 1 the results over seven datasets are presented for point cloud-based architectures using the nearest neighbor classification accuracy metric (NN acc). Table 2 presents a comparison of our best-performing approach with

state-of-the-art view-based architectures. Each cell in the tables is the outcome of the training method on the left in its row fine-tuned to the dataset on top of the column

We consider three sections in Table 1: first the PointNet++ backbone and the PointBERT backbone, both initialized with random weights, and thirdly the ULIP-2 pre-trained PointBERT, with a weight initialization from ULIP-2 pre-training. The PointNet++ section represents the prior art in point cloud-based 3D retrieval. As multi-modal contrastive learning with the three modalities was not part of the prior art, we leave this out of scope. The randomly initialized PointBERT section in the center of the table is added as a baseline for its pre-trained counterpart on the bottom. In the following discussion, we mainly consider the results in Table 1 with the nearest neighbor classification accuracy as we found that the other metrics in Table 3 and Table 5 show are highly correlated and do not influence the final conclusions.

No training baseline. We compare the pre-trained ULIP- 2 PointBERT "out of the box" to random weight baselines in Table 1. As a sanity check, two random baselines are included in the table: *PointNet++ random weights* and *PointBERT random weights*. These rows show the performance of random features from PointNet++ and PointBERT. The *PointBERT ULIP-2 pre-trained* shows an improvement of 7.9–38.1% over the best results from the random baselines, indicating a strong effect from the multi-modal pre-training.

Comparing ULIP-2 pre-trained features to fully-trained baseline models. We pay close attention to the performance of the pre-trained PointBERT (without any fine-tuning) compared to *all* PointNet++ and PointBERT models in the first two sections of Table 1. We observe how the pre-trained PointBERT outperforms all other models for 4 out of 7 datasets. This comparison shows that the pre-trained ULIP-2 PointBERT is competitive "out of the box" with existing point cloud-based methods, which require a training process. For practical applications, this has far-reaching consequences: it is possible to use strong 3D encoders without requiring a training dataset or a computational intensive training process.

Supervised fine-tuning of ULIP-2 gives a significant advancement for point cloud-based architectures. Moving further, we investigate if the strong pre-trained features can be fine-tuned for even better performance in 3D-to-3D retrieval, beyond the existing art. As detailed in Section 3, three fine-tuning approaches are discussed: classification, VICReg and MMCL. In Table 1, we note that classification achieves the best performance on all datasets and shows a 1.1–16.9% increase over the pre-trained baseline. Compared to PointNet++ and the regular PointBERT baselines, we make a significant leap forward on all benchmarks. In particular, we highlight the improved results for the datasets containing more unconventional objects, such as MCB (mechanical components), the proprietary dataset (Prop – dental shapes) and ScanObjectNN (point cloud scans). These strong results on these datasets demonstrate the transfer learning capabilities of the ULIP-2 PointBERT model towards specialized domains.

Supervised fine-tuning of ULIP-2 matches state-of-the-art view-based architectures. To give a comparison to the broader field of 3D object retrieval, we include the state-of-the-art results for the official View-GCN [20] (20 views) and MVTN [21] (12 views) implementations on all seven datasets, see Table 2. It is important to note that view-based architectures use image encoders as backbones, that are pre-trained on ImageNet [14]. For this reason we also refer to training View-GCN and MVTN as fine-tuning with classification in Table 2. While view-based architectures have outperformed point cloud-based methods for 3D object retrieval in the deep learning era, we show that our approach now closes this gap. In Table 2, we see how the ULIP-2 PointBERT backbone fine-tuned with classification slightly outperforms the other architectures in 4 out of 7 cases. The differences are not large however, to make a definitive conslusion the experiments should be repeated several times to eliminate the stochasticity of the process. We conclude that our approach brings point cloud-based architectures at least on par with view-based methods for 3D object retrieval.

Practical advantage for point cloud-based architectures. In addition to state-of-the-art retrieval performance, point cloud-based methods have the advantage of not requiring mesh rendering. This gives a significant advantage over view-based methods regarding the practical complexity of a retrieval system. Furthermore, the fine-tuning of the ULIP-2 PointBERT takes only 3–5 h depending on the dataset size, while fine-tuning View-GCN takes 10–20 h and MVTN required 20–36 h on one Tesla V100 (32 GB) GPU. These are important considerations for developing and maintaining retrieval systems.

Self-supervised fine-tuning of ULIP-2 through VICReg. As classification is a form of supervised learning, it requires labeling effort, making the method more time-consuming and costly to apply in practice. Therefore we explore the effect of fine-tuning with two methods not requiring human annotation, VICReg and multi-modal contrastive learning (MMCL). Table 1 shows that using VICReg as fine-tuning method improves the NN accuracy of the pre-trained model with 1.8–3.4% for M40, SN Per, Prop and ScanObjectNN. These are not large improvements, but when there is no access to sufficient labeled data the additional performance is nice to have. For the other three datasets, retrieval performance does not increase or even slightly degrades performance: Obja Easy shows a 1.1% loss in NN accuracy. We suspect the latter is the case because the model was pre-trained on this exact data distribution, but with another training objective. Although, the degraded performance is not observable for the NDCG metric in Table 3. In practice, we also noticed that VICReg is particularly sensitive to the learning rate schedule, see Section 4.4 and Fig. 5(c).

Fine-tuning of ULIP-2 through MMCL. Recent 3D foundation models were all trained in a multi-modal contrastive learning setup as explained in Section 3.2. Given the strong pre-training performance, we apply MMCL as fine-tuning method in our experiments, except for the ScanObjectNN dataset for which it is not possible to generate image/text/point-cloud triplets. The fine-tuning results show 0.5–2.8% improvements over the out-of-the-box pre-trained ULIP-2 PointBERT except for ShapeNet Normal.

Retrieval examples. Fig. 4 shows a visual comparison between PointBERT with random weights, PointBERT pre-trained in the ULIP-2 work and the ULIP-2 PointBERT fine-tuned through classification training on the proprietary dataset (Prop). The purple 4-unit dental bridge on the left is a sample from the test set and is used as query for each of the three models. The orange samples are the retrieval results, with increasing distance to the query object from left to right. The baseline model does not retrieve any 4-unit dental bridge, and even shows a dental stent and a dental ortho band: 2 classes which are not close to the query class. The ULIP-2 pre-trained PointBERT model gives significantly better retrieval results and only shows dental bridges, of which 2 are 4-unit bridges. The fine-tuned PointBERT model only shows 4-unit dental bridges and is clearly superior to the other two models. These results demonstrate that (1) the "out of the box" pre-trained model is able to return useful results for a rather specialized domain and that (2) fine-tuning the pre-trained model leads to significantly better results.

A practical perspective on fine-tuning methods for PointBERT. While fine-tuning the foundation model through classification yields the best results for 3D-to-3D retrieval across all datasets, we would like to highlight the other approaches from different practical perspectives. In a situation where no training data is available, the pre-trained ULIP-2 model offers a great starting point as we have shown that it is competitive with existing state-of-the-art. If training data is available, but without classification labels, VICReg or MMCL can allow to adapt the foundation model towards a specialized domain. We note that VICReg has less implementation complexity over MMCL, but shows less consistency over the different benchmarks in our experiments.

4.4. Ablation: training dynamics for fine-tuning foundation models

Getting improved performance out of the pre-trained model proved to be non-trivial. In a normal setting where the model is trained from

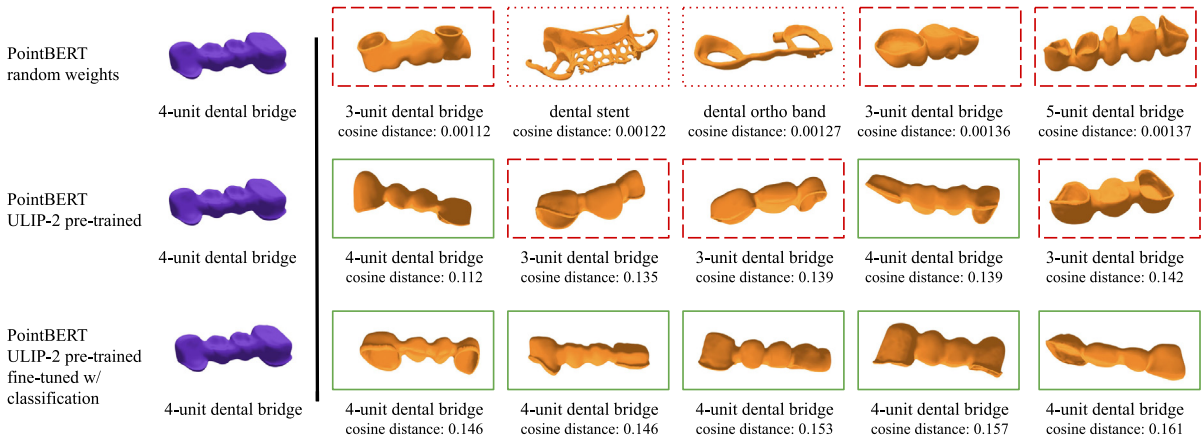


Fig. 4. A qualitative comparison of a baseline PointBERT with random weights, PointBERT pre-trained in the ULIP-2 work [6] and the ULIP-2 PointBERT fine-tuned through classification training on the proprietary dataset (Prop). The purple 4-unit dental bridge on the left is a sample from the test set and is used as query for each of the three models. The orange samples are the retrieval results, with increasing distance to the query object from left to right. A full green line indicates a perfect match (4-unit bridge), a red dashed line indicates dental bridges with a different amount of units than the query part, the dotted line shows objects from completely irrelevant classes. The baseline model does not retrieve any 4-unit dental bridge, and even shows a dental stent and a dental ortho band: 2 classes which are not close to the query class. The ULIP-2 pre-trained PointBERT model gives significantly better retrieval results and only shows dental bridges, of which 2 are 4-unit bridges. The fine-tuned PointBERT model only shows 4-unit dental bridges and is clearly superior to the other two models.

random initialization, AdamW [49] is commonly used as optimizer with a learning rate of $3e{-}4$ and a cosine annealing schedule. Applying this setting to fine-tune the pre-trained models showed various forms of unwanted training behavior.

For example in Fig. 5(a), the blue curve shows the fine-tuning of ULIP-2 PointBERT on ModelNet40 with the common learning rate of $3e{-}4$ with a cosine annealing schedule. The performance shoots up in the first epoch towards 93.0% NN accuracy and afterwards remains between 92.0–93.0%. The network achieves its peak very fast. Fig. 5(b) visualizes the loss score of the hold-out set for the same experiment. The blue curve shows an unstable and increasing trend, indicating overfitting behavior as opposed to generalization. Similar behavior can be observed for the fine-tuning with VICReg in Fig. 5(c). In this case the common learning rate setting only leads to deteriorating performance.

To stabilize training dynamics during fine-tuning, we lowered the learning rate to $5e{-}5$. We found that reducing the learning rate even further would lead to significant underfitting of the network. To further stabilize the initial stages of training, we also introduced a linear warm-up schedule where the learning rate is gradually increased from 0 to $5e{-}5$ in the first 20% of the training. Such warmup schedules have proven to prevent divergence in the early stages of training deep networks due to the random initialization of momentum in optimizers [50]. The effect of our adaptations to the schedule are shown by the orange curves in Fig. 5. The classification fine-tuning experiment shows improved stability in the loss curve and improved performance in the NN test accuracy plot. The VICReg fine-tuning now shows improving performance instead of deteriorating performance.

4.5. Implementation details

Several details regarding the technical implementation are included in this section. We used the AdamW [49] optimizer in all experiments with a learning rate of $3e{-}4$ (PointNet++, regular PointBERT) or $5e{-}5$ (pre-trained PointBERT) and a cosine annealing schedule (+ linear warmup for the pre-trained model). PointNet++ was trained with batch size 64 for classification and batch size 128 for VICReg. All PointBERT (regular and pre-trained) were trained with batch size 64 for classification, 48 for VICReg and 32 for MMCL. The PointNet++ and the regular PointBERT models were trained for 100 epochs on the larger datasets (ShapeNet and MCB) and 200 epochs on the other datasets. For the pre-trained PointBERT this was 50 for the larger datasets and 100 for the smaller datasets. We used point clouds normalized to the unit sphere. We sampled 16K points from each mesh and subsample 2048 or 8192 points during training, respectively for PointNet++ and PointBERT. For the View-GCN architecture and the MMCL fine-tuning method, the ScanObjectNN dataset was not evaluated as the methods do not support point cloud rendering. All experiments were carried out on one Nvidia Tesla V100 GPU (32 GB). The code and checkpoints for the best performing model is available on GitHub.[2]

5. Conclusion

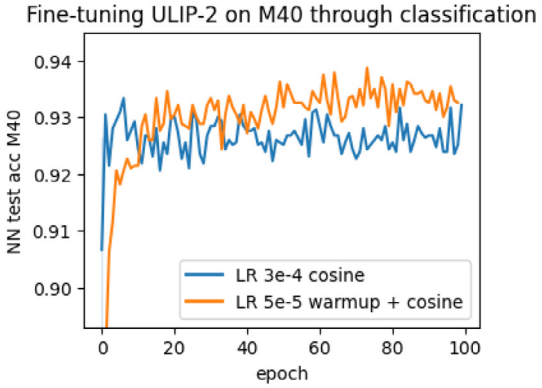
Recently, multi-modal training approaches leveraging the large Objaverse dataset and strong image-text encoders led to the first 3D foundation models. These models have shown unprecedented performance in tasks like 3D shape classification and few-shot part segmentation. The quality of the pure 3D features however, remained unexplored for many other downstream tasks such as 3D-to-3D retrieval. In this work we showed that these pre-trained models significantly advance the potential of 3D encoders for retrieval purposes.

First we discussed the direct application of a foundation model and show that, without any fine-tuning, the model is competitive with existing point cloud-based architectures. For the practical development of retrieval systems, a foundation model is a strong starting point which does not require a training dataset nor a training process. Further, we explored two self-supervised approaches to fine-tuning a foundation model for 3D-to-3D retrieval. These approaches showed improved results over the “out of the box” performance, without requiring any labeled data. With labeled data however, it is possible to fine-tune these models to the state-of-the-art performance of view-based architectures, as demonstrated on seven different datasets. Finally we shared insights on appropriate learning rate schedules for stabilizing the non-trivial fine-tuning process.

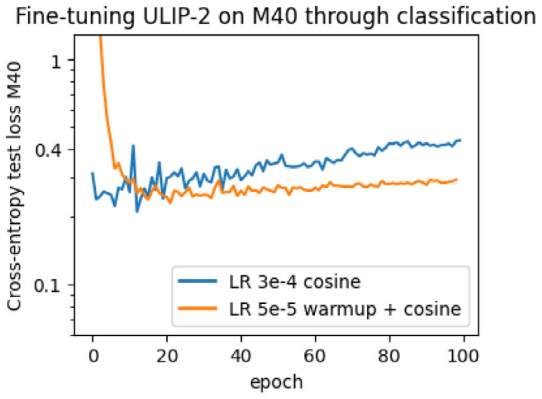
CRediT authorship contribution statement

Jarne Van den Herrewegen: Writing – original draft, Validation, Project administration, Methodology, Investigation, Funding acquisition, Data curation, Conceptualization. **Tom Tourwé:** Writing – review

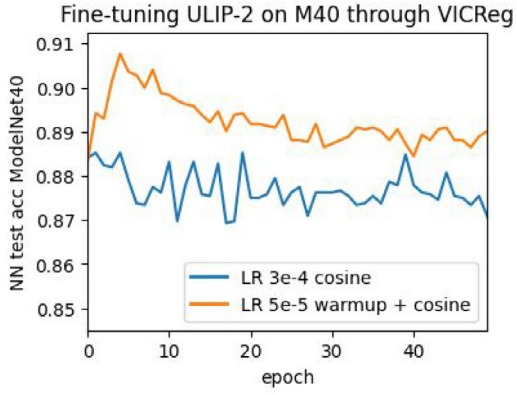
[2] github.com/Anon3DOR/3DOR_Finetuning.



(a) Without a lower learning rate and a warmup schedule (blue curve), the fine-tuning of a ULIP-2 PointBERT through classification shoots up in performance in the first epoch and oscillates afterwards, not achieving full performance. The orange curve shows a smoother training process and higher end result as a consequence of a more careful learning rate schedule.



(b) Loss curve for the same experiment as in Figure 5a. The loss for the hold-out almost exclusively increases, indicating overfitting behavior. The orange curve shows a smoother training process and better result as a consequence of a more careful learning rate schedule.



(c) Applying a typical schedule using cosine annealing with base learning rate $3e-4$ to a ULIP-2 PointBERT in VICReg fine-tuning, only shows deteriorating performance (blue curve). An improved learning rate schedule can allow positive changes (orange curve).

Fig. 5. Ablation study demonstrating the impact of a cosine annealing learning rate schedule with warmup on the training dynamics.

& editing, Supervision, Project administration, Methodology, Funding acquisition, Conceptualization. **Maks Ovsjanikov:** Writing – review & editing, Methodology, Conceptualization. **Francis wyffels:** Writing – review & editing, Supervision, Project administration, Methodology, Funding acquisition, Conceptualization.

Declaration of competing interest

The authors declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper.

Data availability

Link to code is in the paper. The repository contains code, checkpoints and instructions for downloading the data.

Declaration of Generative AI and AI-assisted technologies in the writing process

In the writing process, the author(s) used AI-assisted technology (1) for spell-checking and (2) for summarizing and reviewing the paper, leading to improved structure.

During the preparation of this work the author(s) used Grammarly in order to do spell-checking and OpenAI - GPT4 in order to summarize and to review drafts of the paper, improving the overall structure and clarity. After using this tool/service, the author(s) reviewed and edited the content as needed and take(s) full responsibility for the content of the publication.

Acknowledgments

We would like to thank Anthony Rathé and Axel Vlaminck for their suggestions regarding vision-language models. This work was supported by the Flemish Institute for Innovation and Entrepreneurship (VLAIO), Belgium, grant number HBC.2022.0165.

Parts of this work were supported by the ERC Consolidator Grant 101087347 (VEGA) and ANR AI Chair AIGRETTE, France.

Appendix A. Supplementary data

Supplementary material related to this article can be found online at https://doi.org/10.1016/j.cag.2024.103993.

References

[1] Tangelder JW, Veltkamp RC. A survey of content based 3D shape retrieval methods. Multimedia Tools Appl 2008;39(3):441–71.

[2] Su H, Maji S, Kalogerakis E, Learned-Miller E. Multi-view convolutional neural networks for 3d shape recognition. In: Proceedings of the IEEE international conference on computer vision. 2015, p. 945–53.

[3] Xie S, Gu J, Guo D, Qi CR, Guibas L, Litany O. Pointcontrast: Unsupervised pre-training for 3d point cloud understanding. In: Computer vision–ECCV 2020: 16th European conference, glasgow, UK, August 23–28, 2020, proceedings, part III 16. Springer; 2020, p. 574–91.

[4] Hassani K, Haley M. Unsupervised multi-task feature learning on point clouds. In: Proceedings of the IEEE/CVF international conference on computer vision. 2019, p. 8160–71.

[5] Oquab M, Darcet T, Moutakanni T, Vo HV, Szafraniec M, Khalidov V, Fernandez P, HAZIZA D, Massa F, El-Nouby A, et al. DINOv2: Learning robust visual features without supervision. Trans Mach Learn Res 2023.

[6] Xue L, Yu N, Zhang S, Li J, Martín-Martín R, Wu J, Xiong C, Xu R, Niebles JC, Savarese S. Ulip-2: Towards scalable multimodal pre-training for 3d understanding. 2023, arXiv preprint arXiv:2305.08275.

[7] Liu M, Shi R, Kuang K, Zhu Y, Li X, Han S, Cai H, Porikli F, Su H. Openshape: Scaling up 3d shape representation towards open-world understanding. Adv Neural Inf Process Syst 2024;36.

[8] Zhang B, Yuan J, Shi B, Chen T, Li Y, Qiao Y. Uni3d: A unified baseline for multi-dataset 3d object detection. In: Proceedings of the IEEE/CVF conference on computer vision and pattern recognition. 2023, p. 9253–62.

[9] Yu X, Tang L, Rao Y, Huang T, Zhou J, Lu J. Point-bert: Pre-training 3d point cloud transformers with masked point modeling. In: Proceedings of the IEEE/CVF conference on computer vision and pattern recognition. 2022, p. 19313–22.

[10] Li Z, Liu M, Ramani K. Review of product information retrieval: representation and indexing. In: International design engineering technical conferences and computers and information in engineering conference. Vol. 46970, 2004, p. 971–9.

[11] Cybenko G, Bhasin A, Cohen KD. Pattern recognition of 3d cad objects: Towards an electronic yellow pages of mechanical parts. Int J Smart Eng Syst Des 1997;1(1):1–13.

[12] Kuo C-T, Cheng S-C. 3D model retrieval using principal plane analysis and dynamic programming. Pattern Recognit 2007;40(2):742–55.

[13] Saupe D, Vranić DV. 3D model retrieval with spherical harmonics and moments. In: Pattern recognition: 23rd DAGM symposium munich, Germany, September 12–14, 2001 proceedings 23. Springer; 2001, p. 392–7.

[14] Krizhevsky A, Sutskever I, Hinton GE. Imagenet classification with deep convolutional neural networks. Adv Neural Inf Process Syst 2012;25.

[15] Maturana D, Scherer S. Voxnet: A 3d convolutional neural network for real-time object recognition. In: 2015 IEEE/RSJ international conference on intelligent robots and systems. IROS, IEEE; 2015, p. 922–8.

[16] Qi CR, Yi L, Su H, Guibas LJ. Pointnet++: Deep hierarchical feature learning on point sets in a metric space. Adv Neural Inf Process Syst 2017;30.

[17] Hanocka R, Hertz A, Fish N, Giryes R, Fleishman S, Cohen-Or D. MeshCNN: a network with an edge. ACM Trans Graph 2019;38(4). http://dx.doi.org/10.1145/3306346.3322959.

[18] Park JJ, Florence P, Straub J, Newcombe R, Lovegrove S. Deepsdf: Learning continuous signed distance functions for shape representation. In: Proceedings of the IEEE/CVF conference on computer vision and pattern recognition. 2019, p. 165–74.

[19] Kanezaki A, Matsushita Y, Nishida Y. Rotationnet: Joint object categorization and pose estimation using multiviews from unsupervised viewpoints. In: Proceedings of the IEEE conference on computer vision and pattern recognition. 2018, p. 5010–9.

[20] Wei X, Yu R, Sun J. View-gcn: View-based graph convolutional network for 3d shape analysis. In: Proceedings of the IEEE/CVF conference on computer vision and pattern recognition. 2020, p. 1850–9.

[21] Hamdi A, Giancola S, Ghanem B. Mvtn: Multi-view transformation network for 3d shape recognition. In: Proceedings of the IEEE/CVF international conference on computer vision. 2021, p. 1–11.

[22] Radford A, Kim JW, Hallacy C, Ramesh A, Goh G, Agarwal S, Sastry G, Askell A, Mishkin P, Clark J, et al. Learning transferable visual models from natural language supervision. In: International conference on machine learning. PMLR; 2021, p. 8748–63.

[23] Mu N, Kirillov A, Wagner D, Xie S. Slip: Self-supervision meets language-image pre-training. In: European conference on computer vision. Springer; 2022, p. 529–44.

[24] Li J, Li D, Savarese S, Hoi S. Blip-2: Bootstrapping language-image pre-training with frozen image encoders and large language models. In: International conference on machine learning. PMLR; 2023, p. 19730–42.

[25] Liu H, Li C, Li Y, Lee YJ. Improved baselines with visual instruction tuning. 2023, arXiv preprint arXiv:2310.03744.

[26] Deitke M, Schwenk D, Salvador J, Weihs L, Michel O, VanderBilt E, Schmidt L, Ehsani K, Kembhavi A, Farhadi A. Objaverse: A universe of annotated 3d objects. In: Proceedings of the IEEE/CVF conference on computer vision and pattern recognition. 2023, p. 13142–53.

[27] Chang AX, Funkhouser T, Guibas L, Hanrahan P, Huang Q, Li Z, Savarese S, Savva M, Song S, Su H, et al. Shapenet: An information-rich 3d model repository. 2015, arXiv preprint arXiv:1512.03012.

[28] Kim S, Chi H-g, Hu X, Huang Q, Ramani K. A large-scale annotated mechanical components benchmark for classification and retrieval tasks with deep neural networks. In: Computer vision–ECCV 2020: 16th European conference, glasgow, UK, August 23–28, 2020, proceedings, part XVIII 16. Springer; 2020, p. 175–91.

[29] Koch S, Matveev A, Jiang Z, Williams F, Artemov A, Burnaev E, Alexa M, Zorin D, Panozzo D. Abc: A big cad model dataset for geometric deep learning. In: Proceedings of the IEEE/CVF conference on computer vision and pattern recognition. 2019, p. 9601–11.

[30] Xue L, Gao M, Xing C, Martín-Martín R, Wu J, Xiong C, Xu R, Niebles JC, Savarese S. Ulip: Learning a unified representation of language, images, and point clouds for 3d understanding. In: Proceedings of the IEEE/CVF conference on computer vision and pattern recognition. 2023, p. 1179–89.

[31] Bardes A, Ponce J, LeCun Y. Vicreg: Variance-invariance-covariance regularization for self-supervised learning. 2021, arXiv preprint arXiv:2105.04906.

[32] Wu Z, Song S, Khosla A, Yu F, Zhang L, Tang X, Xiao J. 3D shapenets: A deep representation for volumetric shapes. In: Proceedings of the IEEE conference on computer vision and pattern recognition. 2015, p. 1912–20.

[33] Uy MA, Pham Q-H, Hua B-S, Nguyen T, Yeung S-K. Revisiting point cloud classification: A new benchmark dataset and classification model on real-world data. In: Proceedings of the IEEE/CVF international conference on computer vision. 2019, p. 1588–97.

[34] Van den Herrewegen J, Tourwé T, et al. Self-supervised learning for robust object retrieval without human annotations. Comput Graph 2023;115:13–24.

[35] Afham M, Dissanayake I, Dissanayake D, Dharmasiri A, Thilakarathna K, Rodrigo R. Crosspoint: Self-supervised cross-modal contrastive learning for 3d point cloud understanding. In: Proceedings of the IEEE/CVF conference on computer vision and pattern recognition. 2022, p. 9902–12.

[36] Jing L, Zhang L, Tian Y. Self-supervised feature learning by cross-modality and cross-view correspondences. In: Proceedings of the IEEE/CVF conference on computer vision and pattern recognition. 2021, p. 1581–91.

[37] Zhang R, Guo Z, Zhang W, Li K, Miao X, Cui B, Qiao Y, Gao P, Li H. Pointclip: Point cloud understanding by clip. In: Proceedings of the IEEE/CVF conference on computer vision and pattern recognition. 2022, p. 8552–62.

[38] Becker S, Hinton GE. Self-organizing neural network that discovers surfaces in random-dot stereograms. Nature 1992;355(6356):161–3.

[39] Bromley J, Guyon I, LeCun Y, Säckinger E, Shah R. Signature verification using a“ siamese” time delay neural network. Adv Neural Inf Process Syst 1993;6.

[40] Hadsell R, Chopra S, LeCun Y. Dimensionality reduction by learning an invariant mapping. In: 2006 IEEE computer society conference on computer vision and pattern recognition. CVPR’06, 2, IEEE; 2006, p. 1735–42.

[41] Chen T, Kornblith S, Norouzi M, Hinton G. A simple framework for contrastive learning of visual representations. In: International conference on machine learning. PMLR; 2020, p. 1597–607.

[42] Fan L, Ge Y, Li W, Duan L. Multi-view token clustering and fusion for 3D object recognition and retrieval. In: 2023 IEEE international conference on multimedia and expo. ICME, IEEE; 2023, p. 1145–50.

[43] Dosovitskiy A, Beyer L, Kolesnikov A, Weissenborn D, Zhai X, Unterthiner T, Dehghani M, Minderer M, Heigold G, Gelly S, et al. An image is worth 16x16 words: Transformers for image recognition at scale. 2020, arXiv preprint arXiv:2010.11929.

[44] Vaswani A, Shazeer N, Parmar N, Uszkoreit J, Jones L, Gomez AN, Kaiser Ł, Polosukhin I. Attention is all you need. Adv Neural Inf Process Syst 2017;30.

[45] Community BO. Blender - a 3D modelling and rendering package. Stichting Blender Foundation, Amsterdam: Blender Foundation; 2018, URL http://www.blender.org.

[46] Xue Y, Gan E, Ni J, Joshi S, Mirzasoleiman B. Investigating the benefits of projection head for representation learning. 2024, arXiv preprint arXiv:2403.11391.

[47] Van den Herrewegen J, Tourwé T, et al. Point cloud classification with ModelNet40: What is left? In: DMLR, data-centric machine learning research workshop at the 40 th international conference on machine learning. 2023.

[48] Savva M, Yu F, Su H, Aono M, Chen B, Cohen-Or D, Deng W, Su H, Bai S, Bai X, et al. Shrec16 track: largescale 3d shape retrieval from shapenet core55. In: Proceedings of the eurographics workshop on 3D object retrieval. Vol. 10, 2016, p. 13.

[49] Loshchilov I, Hutter F. Decoupled weight decay regularization. 2017, arXiv preprint arXiv:1711.05101.

[50] Gotmare A, Keskar NS, Xiong C, Socher R. A closer look at deep learning heuristics: Learning rate restarts, warmup and distillation. In: International conference on learning representations. 2018.