Label-Efficient Learning on Point Clouds using Approximate Convex Decompositions

Matheus Gadelha*, Aruni RoyChowdhury*, Gopal Sharma, Evangelos Kalogerakis,
Liangliang Cao, Erik Learned-Miller, Rui Wang and Subhransu Maji
University of Massachusetts Amherst
{mgadelha,aruni,gopal,kalo,llcao,elm,ruiwang,smaji}@cs.umass.edu

Abstract

The problems of shape classification and part segmentation from 3D point clouds have garnered increasing attention in the last few years. Both of these problems however suffer from relatively small training sets, creating the need for statistically efficient methods to learn 3D shape representations. One way forward is to use existing methods for 3D shape decomposition as an approximate segmentation ground truth in a self-supervised method. In particular, we investigate the use of Approximate Convex Decompositions (ACD) as a self-supervisory signal for label-efficient learning of point cloud representations. In this paper, we show that using ACD to approximate ground truth segmentation provides excellent self-supervision for learning 3D point cloud representations that are highly effective on downstream tasks. We report improvements over the state-of-the-art in unsupervised representation learning on the ModelNet40 shape classification dataset and significant gains in few-shot part segmentation on the ShapeNetPart dataset. Our code will be publicly available.[1]

* *equal contribution.*

1. Introduction

The performance of current deep neural network models on tasks such as classification and semantic segmentation of point cloud data is limited by the amount of high quality labeled data available for training the networks. Since in many situations collecting high quality annotations on point cloud data is time consuming and incurs a high cost, there has been increasing efforts in circumventing this problem by training the neural networks on noisy or weakly labeled datasets [1], or training in completely unsupervised ways [2–6]. A ubiquitous technique in training deep networks is to train the network on one task to initialize its parameters and learn generically useful features, and then fine-tune the network on the final task. In particular, there has been great interest in *self-supervised* tasks for initialization, which do not require any human annotations, allowing the network to be initialized by using various techniques to generate labels automatically – *e.g.* tasks such as clustering, solving jigsaw puzzles, and colorization. There have been a few recent attempts to come up with similar tasks that help with 3D data [2, 6]. The overarching question here is *"what makes for a good self-supervision task?"* – what are the useful inductive biases that our model learns from solving such a task that is beneficial to the actual downstream target task we are interested in solving.

We propose using a classical shape decomposition method, Approximate Convex Decomposition (ACD), as the self-supervisory signal to train neural networks built to process 3D data. We posit that being able to decompose a shape into geometrically simple constituent parts provides

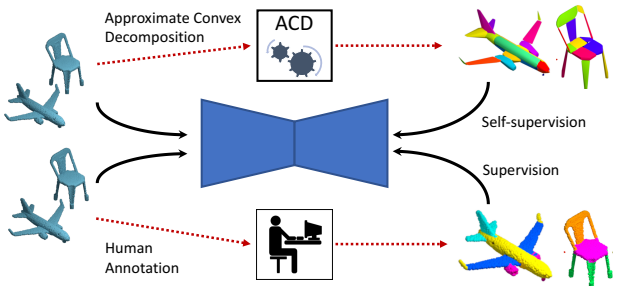

Figure 1. Overview of our method v.s. a fully-supervised approach. ***Top:*** Approximate Convex Decomposition (ACD) can be applied on a large repository of unlabeled point clouds, yielding a *self-supervised* training signal for the neural network without involving any human annotators. ***Bottom:*** the usual *fully-supervised* setting, where human annotators label the semantic parts of point clouds, which are then used as supervision for the neural network. The unsupervised ACD task results in the neural network learning useful representations from unlabeled data, significantly improving performance in shape classification and semantic segmentation when labeled data is scarce or unavailable.

[1] https://github.com/matheusgadelha/PointCloudLearningACD

an excellent self-supervisory learning signal for such purposes. Our approach is illustrated in Figure 1. The main idea is to automatically generate training data by decomposing unlabeled 3D shapes into convex components. Since ACD relies solely on geometric information to perform its decomposition, the process does not require any human intervention. From the model perspective, we formulate ACD as a metric learning problem on point embedding and train the model using a contrastive loss [7,8]. We demonstrate the effectiveness of our approach on standard 3D shape classification and segmentation benchmarks. In classification, we show that the representation learned from performing shape decomposition leads to features that achieve state-of-the-art performance on ModelNet40 [9] unsupervised shape classification ($\mathbf{89}.8\%$). For few-shot part segmentation on ShapeNet [10], our model outperforms the state-of-the-art by **7.5%** mIoU when using 1% of the available labeled training data. Finally, we provide experimental analyses across popular backbone architectures and visualizations demonstrating the role of the ACD self-supervision on the representations learned by neural networks.

2. Method

Decomposing complex shapes as sets of convex components is a well studied problem [11–14]. Given a polyhedron P, the goal is to compute the smallest set of convex polyhedra $\mathcal{C} = \{C_k | k = 1, ..., N\}$, such that the union $\cup_{k=1}^{k=N} C_i$ corresponds to P. In this work, we use an approach called Volumetric Hierarchical Approximate Convex Decomposition (V-HACD) [12]. The reasons for utilizing this approach are threefold. First, as the name suggests, V-HACD performs computations using volumetric representations, which can be easily computed from dense point cloud data or meshes and lead to good results without having to resort to costly decimation and remeshing procedures. Second, the procedure is reasonably fast and can be applied to open surfaces of arbitrary genus. Third, V-HACD guarantees that no two components overlap, which means that there is no part of the surface that is approximated by more than one component. A detailed description of the method can be found in [12].

The output of ACD for every shape is a set of convex components represented by convex meshes. For each shape, we sample points on the original ShapeNet mesh and on the mesh of every ACD component. We then propagate component labels to every point in the original point cloud by using nearest neighbor matching with points in the decomposition. More precisely, given an unlabeled point cloud $\{p_i\}_{i=1}^{N}$, this assigns a component label $\Gamma(p_i, \mathcal{C})$ to each point p_i via

$$\Gamma(p_i, \mathcal{C}) = \underset{k=1...|\mathcal{C}|}{\operatorname{argmin}} \left[\min_{p_j \in C_k} ||p_i - p_j|| \right]. \quad (1)$$

Table 1. Unsupervised shape classification on the ModelNet40 dataset. The representations learned in the intermediate layers by a network trained for the ACD task on ShapeNet data are general enough to be useful for discriminating between shape categories on ModelNet40.

Method	**Accuracy (%)**
VConv-DAE [17]	75.5
3D-GAN [18]	83.3
Latent-GAN [19]	85.7
MRTNet [3]	86.4
PointFlow [5]	86.8
FoldingNet [4]	88.4
PointCapsNet [20]	88.9
Multi-task [2]	89.1
Our baseline (with Random weights)	78.0
With reconstruction term only	86.2
Ours with ACD	89.1
Ours with ACD + Reconstruction	**89.8**

Self-supervision with ACD. The component labels generated by the ACD algorithm are not consistent across point clouds, *i.e.* "component 5" may refer to the *seat* of a chair in one point cloud, while the *leg* of the chair may be labeled as "component 5" in another point cloud. Therefore, the usual cross-entropy loss, which is generally used to train networks for tasks such as semantic part labeling, is not applicable in our setting. We formulate the learning of Approximate Convex Decompositions as a metric learning problem on point embeddings via a *pairwise* or *contrastive loss* [7]. We assume that each point $p_i = (x_i, y_i, z_i)$ in a point cloud $\mathbf{x}$ is encoded as $\Phi(\mathbf{x})_i$ in some embedding space by a neural network encoder $\Phi(\cdot)$, *e.g.* PointNet [15] or PointNet++ [16]. Let the embeddings of a pair of points from a shape be $\Phi(\mathbf{x})_i$ and $\Phi(\mathbf{x})_j$, normalized to unit length (*i.e.* $||\Phi(\mathbf{x})_i|| = 1$), and the set of convex components as described above be $\mathcal{C}$. The pairwise loss is then defined as

$$\begin{aligned}\mathcal{L}^{pair}(\mathbf{x}, p_i, p_j, \mathcal{C}) = \mathcal{I}[\Gamma(p_i, \mathcal{C}) = \Gamma(p_j, \mathcal{C})](1 - \Phi(\mathbf{x})_i^\top \Phi(\mathbf{x})_j) \\ + \mathcal{I}[\Gamma(p_i, \mathcal{C}) \neq \Gamma(p_j, \mathcal{C})](\max(0, \Phi(\mathbf{x})_i^\top \Phi(\mathbf{x})_j - m))\end{aligned} \quad (2)$$

This loss encourages points belonging to the same component to have a high similarity $\Phi(\mathbf{x})_i^\top \Phi(\mathbf{x})_j$, and encourages points from different components to have low similarity, subject to a margin m. $[\cdot]$ denotes the Iverson bracket.

3. Experiments

We demonstrate the effectiveness of the ACD self-supervision across a range of experimental scenarios. For all the experiments in this section we use ACDs computed on all shapes from the ShapeNetCore data [10], which contains 57,447 shapes across 55 categories.

Shape classification on ModelNet40 [9]. In this set of experiments, we show that the representations learned by a

network trained on ACD are useful for discriminative downstream tasks such as classifying point clouds into shape categories. We report results on the ModelNet40 shape classification benchmark, which consists of 12,311 shapes from 40 shape categories in a train/test split of 9,843/2,468. A linear SVM is trained on the features extracted on the training set of ModelNet40. As an initial naïve baseline, we use a PointNet++ network with random weights as our feature extractor. This gives 78% accuracy on ModelNet40. Training this network with ACD, gives a significant boost to performance (78% $\rightarrow$ **89.1**%), demonstrating the effectiveness of our proposed self-supervision task. This indicates some degree of generalization across datasets and tasks – from distinguishing convex components on ShapeNet to classifying shapes on ModelNet40. Inspired by [2], we also investigated if adding a reconstruction component to the loss would further improve accuracy by simply adding an AtlasNet [21] decoder to our model and using Chamfer distance as reconstruction loss. Without the reconstruction term (i.e. trained only to perform ACD using contrastive loss), our result accuracy (89.1%) is the same as the multi-task learning approach presented in [2]. After adding a reconstruction term, we achieve an improved accuracy of **89.8**%. On the other hand, having just reconstruction without ACD yields an accuracy of 86.2%. Approaches for *unsupervised* or *self-supervised* learning on point clouds are listed in the upper portion of Table 1. Our method achieves **89.1%** classification accuracy from purely using the ACD loss, which is met only by the unsupervised multi-task learning method of Hassani *et al*. [2]. We note that our method merely adds a contrastive loss to a standard architecture (PointNet++), without requiring a custom architecture and multiple pretext tasks as in [2], which uses clustering, pseudo-labeling and reconstruction.

Few-shot segmentation on ShapeNet [10]. Table 2 shows the few-shot segmentation performance of our method, versus a fully-supervised baseline. Especially in the cases of very few labeled training samples ($k = 1, \ldots, 10$), having the ACD loss over a large unlabeled dataset provides a consistent and significant gain in performance over purely training on the labeled samples. The performance of recent *unsupervised* and *self-supervised* methods on ShapeNet segmentation are listed in Table 3. Consistent with the protocol followed by the multi-task learning approach of Hassani *et al*. [2], we provide 1% and 5% of the training samples of ShapeNetSeg as the labeled data and report instance-average IoU. Our method clearly outperforms the state-of-the-art unsupervised learning approaches, improving over [2] at both the 1% and 5% settings (68.2 $\rightarrow$ **75.7**% and 77.7 $\rightarrow$ **79.7**%, respectively).

Ablation on backbones. Differently from [2,4,6], the ACD self-supervision does not require any custom network design and should be easily applicable across various backbone architectures. To this end, we use two recent high-performing models – *PointNet++* (with multi-scale grouping [16]) and *DGCNN* [24] – as the backbones, reporting results on ModelNet40 shape classification and few-shot segmentation ($k = 5$) on ShapeNetSeg (Table 4). On shape classification, both networks show large gains from ACD pre-training: 11% for PointNet++ (as reported earlier) and 14% for DGCNN. On few-shot segmentation with 5 samples per category (16 shape categories), PointNet++ improves from 68.14% IoU to 72.3% with the inclusion of the ACD loss. The baseline DGCNN performance with only

Table 2. Few-shot segmentation on the ShapeNet dataset (*class avg. IoU* over 5 rounds). K denotes the number of shots or samples per class for each of the 16 ShapeNet categories used for supervised training. Jointly training with the ACD task reduces overfitting when labeled data is scarce, leading to significantly better performance over a purely supervised baseline.

Samples/cls.	**k=1**	**k=3**	**k=5**	**k=10**
Baseline	53.15 $\pm$ 2.49	59.54 $\pm$ 1.49	68.14 $\pm$ 0.90	71.32 $\pm$ 0.52
w/ ACD	61.52 $\pm$ 2.19	69.33 $\pm$ 2.85	72.30 $\pm$ 1.80	74.12 $\pm$ 1.17
	k=20	**k=50**	**k=100**	**k=inf**
Baseline	75.22 $\pm$ 0.82	78.79 $\pm$ 0.44	79.67 $\pm$ 0.33	81.40 $\pm$ 0.44
w/ ACD	76.19 $\pm$ 1.18	78.67 $\pm$ 0.72	78.76 $\pm$ 0.61	81.57 $\pm$ 0.68

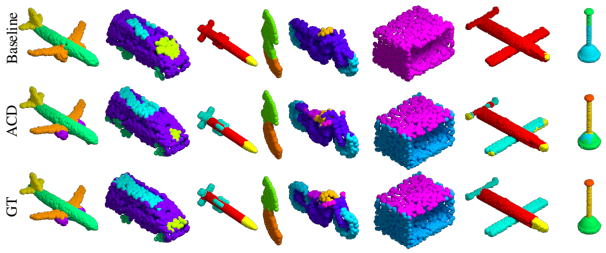

Figure 2. Qualitative comparison on 5-shot ShapeNet [10] part segmentation. The baseline method in the first row corresponds to training using only 5 examples per class, whereas the ACD results in the second row were computed by performing joint training (cross-entropy from 5 examples + contrastive loss over ACD components from ShapeNetCore). The network backbone architecture is the same for both approaches – PointNet++ [16]. The baseline method merges parts that should be separated, *e.g.* engines of the airplane, details of the rocket, top of the table, seat of the motorcycle, etc.

Table 3. Comparison with state-of-the-art semi-supervised part segmentation methods on ShapeNet. Performance is evaluated using *instance-averaged IoU*.

Method	1% labeled **IoU**	5% labeled **IoU**
SO-Net [22]	64.0	69.0
PointCapsNet [20]	67.0	70.0
MortonNet [23]	-	77.1
Multi-task [2]	68.2	77.7
ACD (*ours*)	**75.7**	**79.7**

Table 4. Comparing embeddings from PointNet++ [16] and DGCNN [24] backbones: shape classification accuracy on ModelNet40 (*Class./MN40*) and few-shot part segmentation performance in terms of class-averaged IoU on ShapeNet (*Part Seg./ShapeNet*).

Task / Dataset	Method	PointNet++	DGCNN
Class./MN40	Baseline	77.96	74.11
	w/ ACD	**89.06**	**88.21**
5-shot Seg./ShapeNet	Baseline	68.14 $\pm$ 0.90	64.14 $\pm$ 1.43
	w/ ACD	**72.30** $\pm$ 1.80	**73.11** $\pm$ 0.95

5 labeled samples per class is relatively lower (64.14%), however with the additional ACD loss on unlabeled samples, the model achieves 73.11% IoU, which is comparable to the corresponding PointNet++ performance (72.30%).

References

[1] Sharma, G., Kalogerakis, E., Maji, S.: Learning point embeddings from shape repositories for few-shot segmentation. In: 2019 International Conference on 3D Vision (3DV). (2019) 67–75 1

[2] Hassani, K., Haley, M.: Unsupervised multi-task feature learning on point clouds. In: Proceedings of the IEEE International Conference on Computer Vision. (2019) 8160–8171 1, 2, 3

[3] Gadelha, M., Wang, R., Maji, S.: Multiresolution Tree Networks for 3D Point Cloud Processing. In: ECCV. (2018) 1, 2

[4] Yang, Y., Feng, C., Shen, Y., Tian, D.: Foldingnet: Point cloud auto-encoder via deep grid deformation. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition. (2018) 206–215 1, 2, 3

[5] Yang, G., Huang, X., Hao, Z., Liu, M.Y., Belongie, S., Hariharan, B.: Pointflow: 3d point cloud generation with continuous normalizing flows. In: Proceedings of the IEEE International Conference on Computer Vision. (2019) 4541–4550 1, 2

[6] Chen, Z., Yin, K., Fisher, M., Chaudhuri, S., Zhang, H.: Baenet: branched autoencoder for shape co-segmentation. In: Proceedings of the IEEE International Conference on Computer Vision. (2019) 8490–8499 1, 3

[7] 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). Volume 2., IEEE (2006) 1735–1742 2

[8] Chopra, S., Hadsell, R., LeCun, Y.: Learning a similarity metric discriminatively, with application to face verification. In: 2005 IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR'05). Volume 1., IEEE (2005) 539–546 2

[9] 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) 1912–1920 2

[10] Chang, A.X., Funkhouser, T.A., Guibas, L.J., Hanrahan, P., Huang, Q.X., Li, Z., Savarese, S., Savva, M., Song, S., Su, H., Xiao, J., Yi, L., Yu, F.: Shapenet: An information-rich 3d model repository. CoRR **abs/1512.03012** (2015) 2, 3

[11] Lien, J.M., Amato, N.M.: Approximate convex decomposition of polyhedra. In: Proceedings of the 2007 ACM Symposium on Solid and Physical Modeling. SPM '07 (2007) 2

[12] Mamou, K.: Volumetric approximate convex decomposition. In Lengyel, E., ed.: Game Engine Gems 3. A K Peters / CRC Press (2016) 141–158 2

[13] Kaick, O.V., Fish, N., Kleiman, Y., Asafi, S., Cohen-OR, D.: Shape segmentation by approximate convexity analysis. ACM Trans. Graph. **34**(1) (2014) 2

[14] Zhou Ren, Junsong Yuan, Chunyuan Li, Wenyu Liu: Minimum near-convex decomposition for robust shape representation. In: 2011 International Conference on Computer Vision. (Nov 2011) 2

[15] Su, H., Qi, C., Mo, K., Guibas, L.: PointNet: Deep Learning on Point Sets for 3D Classification and Segmentation. In: CVPR. (2017) 2

[16] Qi, C.R., Yi, L., Su, H., Guibas, L.: PointNet++: Deep hierarchical feature learning on point sets in a metric space. In: Proc. NIPS. (2017) 2, 3, 4

[17] Sharma, A., Grau, O., Fritz, M.: Vconv-dae: Deep volumetric shape learning without object labels. In: European Conference on Computer Vision, Springer (2016) 236–250 2

[18] Wu, J., Zhang, C., Xue, T., Freeman, B., Tenenbaum, J.: Learning a probabilistic latent space of object shapes via 3d generative-adversarial modeling. In: Advances in neural information processing systems. (2016) 82–90 2

[19] Achlioptas, P., Diamanti, O., Mitliagkas, I., Guibas, L.: Representation learning and adversarial generation of 3d point clouds. arXiv preprint arXiv:1707.02392 (2017) 2

[20] Zhao, Y., Birdal, T., Deng, H., Tombari, F.: 3d point capsule networks. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition. (2019) 1009–1018 2, 3

[21] Groueix, T., Fisher, M., Kim, V.G., Russell, B., Aubry, M.: AtlasNet: A Papier-Mâché Approach to Learning 3D Surface Generation. In: Proceedings IEEE Conf. on Computer Vision and Pattern Recognition (CVPR). (2018) 3

[22] Li, J., Chen, B.M., Hee Lee, G.: So-net: Self-organizing network for point cloud analysis. In: Proceedings of the IEEE conference on computer vision and pattern recognition. (2018) 9397–9406 3

[23] Thabet, A., Alwassel, H., Ghanem, B.: MortonNet: Self-Supervised Learning of Local Features in 3D Point Clouds. arXiv (Mar 2019) 3

[24] Wang, Y., Sun, Y., Liu, Z., Sarma, S.E., Bronstein, M.M., Solomon, J.M.: Dynamic graph cnn for learning on point clouds. ACM Transactions on Graphics (TOG) **38**(5) (2019) 1–12 3, 4