Self-Supervised Learning for Domain Adaptation on Point Clouds

Idan Achituve
Bar-Ilan University
Ramat Gan, Israel
idan.achituve@biu.ac.il

Haggai Maron
NVIDIA
Tel-Aviv, Israel
hmaron@nvidia.com

Gal Chechik
Bar-Ilan University, Israel
NVIDIA, Israel

Abstract

Self-supervised learning (SSL) is a technique for learning useful representations from unlabeled data. It has been applied effectively to domain adaptation (DA) on images and videos. It is still unknown if and how it can be leveraged for domain adaptation in 3D perception problems. Here we describe the first study of SSL for DA on point clouds. We introduce a new family of pretext tasks, Deformation Reconstruction, inspired by the deformations encountered in sim-to-real transformations. In addition, we propose a novel training procedure for labeled point cloud data motivated by the MixUp method called Point cloud Mixup (PCM). Evaluations on domain adaptations datasets for classification and segmentation, demonstrate a large improvement over existing and baseline methods.

1. Introduction

Self-supervised learning (SSL) was recently shown to be very effective for learning useful representations from unlabeled images [9, 10, 16, 28, 29] or videos [12, 27, 48, 50]. The key idea is to define an auxiliary, “pretext” task, train using supervised techniques, and then use the learned representation for the main task of interest. While SSL is often effective for images and videos, it is still not fully understood how to apply it to other types of data. Recently, there have been some attempts at designing SSL pretext tasks for point cloud data for representation learning [18, 38, 43, 62], yet this area of research is still largely unexplored. Since SSL operates on unlabeled data, it is natural to test its effectiveness for unsupervised domain adaptation (UDA).

Domain Adaptation (DA) has attracted significant attention recently [13, 36, 45, 46]. In UDA, one aims to classify data from a Target distribution, but the only labeled samples available are from another, Source, distribution. This learning setup has numerous application, including “sim-to-real”, where a model is trained on simulated data in which labels are abundant and is tested on real-world data. Recently, SSL was successfully used in learning across domains [4, 11, 33] and in domain adaptation for visual tasks such as object recognition and segmentation [41, 53]. While SSL has been used to adapt to new domains in images, it is unknown if and how SSL applies to DA for other data types, particularly for 3D data.

The current paper addresses the challenge of developing SSL for point clouds in the context of DA. We describe an SSL approach for adapting to new point cloud distributions. Our approach is based on a multi-task architecture with a multi-head network. One head is trained using a classification or segmentation loss over the source domain, while a second head is trained using a new SSL loss.

To learn a representation that captures the structure of the target domain, we develop a new family of pretext-tasks, called Deformation Reconstruction (DefRec). We design it to address common deformations that are encountered in scanned point clouds. Scanning objects in their natural environments often leads to missing parts of the objects due to occlusion (see Figure 3, third column). The key idea behind the new pretext tasks is as follows: It deforms a region of the shape by dislocating some of the points; then, the network has to map back those points to their original location, re-
constructing the missing regions of the shape. Importantly, success in this task requires the network to learn the underlying statistical structures of objects.

In this paper, we provide an extensive study of different approaches to deform a shape. We group these approaches into three types: (1) **Volume-based deformations:** selecting a region based on proximity in the input space $\mathbb{R}^3$, (2) **Feature-based deformations:** selecting regions that are semantically similar by leveraging deep point embeddings; and (3) **Sampling-based deformations:** selecting a region based on three simple sampling schemes.

As a separate contribution, we propose a training procedure for labeled point cloud data motivated by the MixUp method [61], called **Point Cloud Mixup** (PCM). PCM is applied to source objects during training instead of the standard classification task. Together with DefRec, PCM yields large improvements over the SoTA of domain adaptation in a benchmark classification dataset in this area [32].

Finally, we designed a new DA benchmark for point cloud segmentation based on a dataset published by [25]. We show that DefRec can be extended easily to segmentation tasks, leading to improved performance compared to baseline methods.

This paper makes the following novel contributions. (1) This is the first paper that studies SSL for domain adaptation on point clouds. (2) We describe DefRec, a new family of pretext tasks for point cloud data, motivated by the type of distortions encountered in sim-to-real scenario. (3) We achieve a new SoTA performance for domain adaptation on point clouds, including a large improvement over previous approaches in a sim-to-real tasks. (4) We develop a new variant of the Mixup method for point cloud data. (5) A new DA benchmark for point cloud segmentation.

2. Related work

**Deep learning on point clouds.** Following the success of deep neural networks on images, powerful deep architectures for learning with 3D point clouds were designed. Early methods, such as [26, 31, 52], applied volumetric convolutions to occupancy grids generated from point clouds. These methods suffer from limited performance due to the low resolution of the discretization of 3D data. The seminal work of [30, 60] described the first models that work directly on a point cloud representation. Following these studies, a plethora of architectures was suggested, aiming to generalize convolutions to point clouds [3, 20, 22, 23, 39, 49]. We refer the readers to a recent survey [17] for more details.

**Self-supervised learning for point clouds.** Recently, several studies suggested using self-supervised tasks for learning meaningful representations of point cloud data, mostly as a pre-training step. In [38], it is suggested to generate new point clouds by splitting a shape into $3 \times 3 \times 3$ voxels and shuffling them. The task is to find the voxel assignment that reconstructs the original point cloud. [43] proposed a network that predicts the next point in a space-filling sequence of points that covers a point cloud. [62] generated pairs of half shapes and proposed to learn a classifier to decide whether these two halves originate from the same point cloud. [18] advocates combining three tasks: clustering, prediction, and point cloud reconstruction from noisy input. [6] learns a point cloud auto-encoder that also predicts pairwise relations between the points. [42] suggested learning local geometric properties by training a network to predict the point normal vector and curvature. In a concurrent work, [2] leveraged the SSL task proposed by [38], as an auxiliary task for learning a variety of point cloud tasks. Compared to these studies, our work provides a systematic study of point cloud reconstruction pretext tasks specifically for DA on point clouds, a setup that was not addressed by any of the studies mentioned above.

**Domain adaptation for point clouds.** PointDAN [32] designed a dataset based on three widely-used point cloud datasets: ShapeNet [5], ModelNet [52] and ScanNet [8]. They proposed a model that jointly aligns local and global point cloud features for classification. [40] proposed a generic module to embed information from different domains in a shared space for object detection. Several other studies considered domain adaptation for LiDAR data with methods that do not operate directly on the unordered set of points [21, 34, 37, 51, 57]. [21] and [57] suggested DA methods for point cloud segmentation from a sparse voxel representation. [21] requires a paring of the point cloud and image representation of the scenes, and [57] suggested to apply segmentation on recovered 3D surfaces from the point clouds. [34] suggested a method for DA on voxelized points input using an object region proposal loss, point segmentation loss, and object regression loss. [37] addressed the task of vehicle detection from a bird’s eye view (BEV) using a CycleGAN. [51] designed a training procedure for object segmentation of shapes projected onto a spherical surface.

**Self-supervised learning for domain adaptation.** SSL for domain adaptation is a relatively new research topic. Existing literature is mostly very recent, and is applied to the image domain, which is fundamentally different from unordered point clouds. [15] suggested using a shared encoder for both source and target samples followed by a classification network for source samples and a reconstruction network for target samples. [53] suggested using SSL pretext tasks like image rotation and patch location prediction over a feature extractor. [41] extended the solution to a multi-task problem with several SSL pretext tasks. [4] advocated the use of a Jigsaw puzzle [28] pretext task for domain generalization and adaptation. Our approach is similar to these approaches in the basic architectural design, yet it is different in the type of data and pretext tasks. [35] addressed the problem of universal domain adaptation by learning to clus-
Deep learning of point cloud reconstruction and completion. Numerous methods were suggested for point cloud completion and reconstruction. Most of these studies focus on high-quality shape reconstruction and completion. Our paper draws inspiration from these studies and suggests effective pretext reconstruction tasks for domain adaptation. [1] suggested to learn point clouds representations with an Autoencoder (AE) based on the architecture proposed in [30]. In Section 5 we show that our method compares favorably to theirs. [7] proposed to reconstruct point clouds by training a GAN on a latent space of unpaired clean and partial point clouds. Training GANs may be challenging because of common pitfalls such as mode collapse, our method, on the other hand, is much easier to train. [58] suggested architecture for up-sampling a point cloud by learning point features and replicating them. [59] suggested an object completion network of partial point clouds from a global feature vector representation. [47] extended [59] approach by a cascaded refinement of the reconstructed shape.

3. Approach

In this section, we present the main building blocks of our approach. We first describe our general pipeline and then explain in detail our main contribution, namely, DefRec, a family of SSL tasks. We conclude the section by describing PCM, a training procedure inspired by the Mixup method [61] that we found to be effective when combined with DefRec. For clarity, we describe DefRec in the context of a classification task. An extension of DefRec to segmentation is detailed in Section 5.4.1

3.1. Overview

We tackle unsupervised domain adaptation for point clouds. Here, we are given labeled instances from a source distribution and unlabeled instances from a different, target, distribution. Both distributions of point clouds are based on objects labeled by the same set of classes. Given instances from both distributions, the goal is to train a model that correctly classifies samples from the target domain.

We follow a common approach to tackle this learning setup for DA, learning a shared feature encoder, trained on two tasks [55]. (1) A supervised task on the source domain; and (2) A self-supervised task that can be trained on both source and target domains. To this end, we propose a new family of self-supervised tasks. In our self-supervised tasks, termed Deformation Reconstruction (DefRec), we first deform a region/s in an input point cloud and then train our model to reconstruct it.

More formally, let \( \mathcal{X}, \mathcal{Y} \) denote our input space and label space accordingly. Let \( S \subset \mathcal{X} \times \mathcal{Y} \) represent labeled data from the source domain, and \( T \subset \mathcal{X} \) represent unlabeled data from the target domain. We denote by \( x \in \mathbb{R}^{n \times 3} \) the input point cloud and \( \hat{x} \in \mathbb{R}^{n \times 3} \) the deformed version of it, where \( n \) is the number of points. Our training scheme has two separate data flows trained in an alternating fashion and in an end-to-end manner. Supervised data flow and self-supervised data flow. Both data flows use the same feature encoder \( \Phi \) which is modeled by a neural network for point clouds. After being processed by the shared feature encoder, labeled source samples are further processed by a fully connected sub-network (head) denoted by \( h_{sup} \) and a supervised loss is applied to their result (either the regular cross-entropy loss or a mixup variant that will be described in Section 3.3). Similarly, after the shared feature encoder, the unlabeled source/target samples are fed into a different head, denoted \( h_{SSL} \) which is in charge of producing a reconstructed version of \( \hat{x} \). A reconstruction loss is then applied to the result as we explain in the next subsection. The full architecture is depicted in Figure 1.

3.2. Deformation reconstruction

When designing a self-supervision task, several considerations should be taken into account. First, the task should encourage the model to capture the semantic properties of the inputs. The scale of these properties is important: a task that depends on local features may not capture the semantics, and a task that depends on full global features may be over permissive. In general it is useful to focus on mesoscale features, capturing information at the scale...
of “regions” or parts.

Second, for the specific case of designing SSL for DA, we want the SSL task to “bridge” the distribution gap from the Source to the Target distribution. Intuitively, it would be beneficial if the SSL deformation of target samples can imitate the same deformations that are observed from source to target because then the learned representation tends to be invariant to these gaps. We designed DefRec, our family of SSL tasks, with these intuitions in mind.

The main idea of our SSL family of tasks is to reconstruct deformed input samples. However, a key question remains: which deformations and reconstruction tasks would produce meaningful representations for domain adaptation? We examine three types of region selection methods: Volume-based, Feature-based and Sampling-based. In all cases, the deformation is achieved by selecting a subset of points and deforming them by sampling new points from an isotropic Gaussian distribution with a small standard deviation. Figure 2 illustrates all types of deformations.

Volume-based deformations. Perhaps the simplest and most intuitive way to define a region is based on proximity in the input space. We propose two alternatives to generate distorted point clouds. (1) split the input space (say, the box which bounds all points in the cloud) to \( k \times k \times k \) equally-sized voxels and pick one voxel \( v \) uniformly at random (we found that \( k = 3 \) works well). (2) the deformation region is a sphere with a fixed radius \( r \) that is centered around a single data point \( p \) selected at random. For both alternatives, we replace all the points with new points sampled from a Gaussian distribution around the center of \( v \) (for the first method) or \( p \) (for the second method).

Feature-based deformations. Going beyond input-space proximity, we wish to deform regions defined by their semantics. We follow [49], which showed that distances of point features taken from deeper layers capture semantic similarity more than input-space proximity. Given an input point cloud \( x \in \mathbb{R}^{n \times 3} \) we obtain a representation of the points \( \Phi_l(x) \) at layer \( l \), pick one point uniformly at random and based on the representations take its \( k \) nearest neighbors. We replace all selected points with points sampled around the origin.

Sample-based deformations. In this case, a region is defined based on points sampled according to three common sampling protocols inspired by [19]: (1) Split: Randomly selecting a hyperplane that traverses the shape and separate it into two half-spaces. All points from the smaller part are taken, and points from the second part are randomly sampled with probability \( p \) that is drawn from a uniform distribution on \([0, 1]\) for each input; (2) Gradient: Sampling points with a likelihood that decreases linearly along the largest axis of the shape; and (3) Lambertian: A sampling method that depends on the normal orientation. For each input, we fix a “view” direction (drawn uniformly at random). The probability of sampling a point is proportional to the clamped inner product between the surface normal (which is estimated based on neighboring points) and the fixed “view” direction. In all methods, we limit the number of sampled points to be smaller than a constant to prevent large deformations. Sampled points are relocated and scattered around the origin.

Data flow and loss function. The self-supervised data flow starts with generating a new input-label pair \((\tilde{x}, x) \in \tilde{S} \cup \tilde{T} \subset \mathcal{X} \times \mathcal{X}\) by using any of the methods suggested above. The deformed input \( \tilde{x} \) is first processed by \( \Phi \), producing a representation \( \Phi(\tilde{x}) \). This representation is then fed into \( h_{\text{SSL}} \), which is in charge of producing a reconstructed version of \( \tilde{x} \). A reconstruction loss \( L_{\text{SSL}} \), which penalizes deviations between the output \( h_{\text{SSL}}(\Phi(\tilde{x})) \) and the original point cloud \( x \) is then applied.

We chose the loss function \( L_{\text{SSL}} \) to be the Chamfer distance between the set of points in \( x \) that falls in the deformed region \( R \) and their corresponding outputs. More explicitly, if \( I \subset \{1, \ldots, n\} \) represents the indices of the points in \( x \cap R \), the loss takes the following form:

\[
L_{\text{SSL}}(\tilde{S} \cup \tilde{T} ; \Phi, h_{\text{SSL}}) = \sum_{(\tilde{x}, x) \in \tilde{S} \cup \tilde{T}} d_{\text{Chamfer}} \left( \{x_i\}_{i \in I}, \{h_{\text{SSL}}(\Phi(\tilde{x}))_i\}_{i \in I} \right)
\]

(1)

where \( x_i \) is the \( i \)-th point in the point cloud \( x \), and

\[
don(\tilde{A}, \tilde{B}) = \sum_{a \in A} \min_{b \in B} \|a-b\|^2 + \sum_{b \in B} \min_{a \in A} \|b-a\|^2
\]

(2)

is the symmetric Chamfer distance between \( A, B \subseteq \mathbb{R}^3 \). Since the Chamfer distance is computed only on within-region points, it does not burden the computation.

In our experiments, we found that applying DefRec only to target samples yields better results. Therefore, unless stated otherwise, that is the selected approach.

3.3. Point cloud mixup

We now discuss an additional contribution that is independent of the proposed SSL task, but we find to operate well with DefRec. The labeled samples from the source domain are commonly used in domain adaptation with a standard cross-entropy classification loss. Here, we suggest an alternative loss motivated by the Mixup Method [61]. Mixup is based on the Vicinal Risk Minimization principle, as opposed to Empirical Risk Minimization, and can be viewed as an extension of data augmentation that involves both input samples and their labels. Given two images and their "one-hot" labels \((x, y), (x', y')\), the Mixup method generates a new labeled sample as a convex combination of the inputs \((\gamma x + (1-\gamma)x', \gamma y + (1-\gamma)y')\), where \(\gamma\) is sampled from a Beta distribution with fixed parameters.

126
We generalize this method to point clouds. Since point clouds are arbitrarily ordered, a naive convex combination of two points (similarly to pixels) may result in arbitrary positions. Hence, such combination of two point clouds may be meaningless. Instead, we propose the following Point Cloud Mixup (PCM) procedure. Given two point clouds \( x, x' \in \mathbb{R}^{n \times 3} \), first sample a Mixup coefficient \( \gamma \sim \text{Beta}(\alpha, \beta) \) (\( \alpha, \beta = 1 \) worked well in our case) and then form a new shape by randomly sampling \( \gamma \cdot n \) points from \( x \) and \( (1 - \gamma) \cdot n \) points from \( x' \). The union of the sampled points yields a new point cloud, \( \tilde{x} \in \mathbb{R}^{3\cdot n} \). As in the original Mixup method, the label is a convex combination of the one-hot label vectors of the two point clouds \( \gamma y + (1 - \gamma)y' \). See Figure 1 (green box) for illustration.

To summarize, using PCM, the supervised data flow starts with sampling two labeled point clouds \((x, y), (x', y') \in S \). They are then combined into a new labeled point cloud \((\tilde{x}, \tilde{y}) \). \( \tilde{x} \) is fed into the shared encoder \( \Phi \) to produce a point-cloud representation \( \Phi(\tilde{x}) \in \mathbb{R}^{d} \). This representation is further processed by a fully connected sub-network (head) \( h_{\text{sup}} \). A cross-entropy loss \( L_{\text{ce}} \) is then applied to the output of \( h_{\text{sup}} \) and the new label \( \tilde{y} \).

**Mixup for DA.** Extensions of the Mixup method were offered as a solution for DA on images data [24, 54, 56]. We, on the other hand, propose to use SSL methods, and in particular DefRec. Our formulation of Mixup for point-cloud data can be applied to any classification task and not necessarily for DA. We found that PCM improves the accuracy of various baselines (Section C.1 in the Appendix), and was particularly beneficial when combined with DefRec.

### 3.4. Overall loss

The overall loss is a linear combination of a supervised loss and an SSL loss:

\[
L(S, T; \Phi, h_{\text{sup}}, h_{\text{sup}}) = L_{\text{ce}}(S; \Phi, h_{\text{sup}}) + \lambda L_{\text{SSL}}(S \cup T; \Phi, h_{\text{SSL}}),
\]

where \( \lambda \) is a parameter that controls the importance of the self-supervised term. To use PCM, \( L_{\text{ce}}(S; \Phi, h_{\text{sup}}) \) can be replaced with \( L_{\text{ce}}(\tilde{x}; \Phi, h_{\text{sup}}) \).

**4. Experiments**

We evaluate our method on PointDA-10, a DA dataset designed by [32] for classification of point clouds. For segmentation, we introduce a benchmark dataset, PointSegDA.

#### 4.1. PointDA-10

PointDA-10 ([32]) consists of three subsets of three widely-used datasets: ShapeNet [5], ModelNet [52] and ScanNet [8]. All three subsets share the same ten distinct classes (like chair, table, bed). ModelNet-10 (called ModelNet hereafter) contains 4183 train samples and 856 test samples sampled from clean 3D CAD models. ShapeNet-10 (called ShapeNet hereafter), contains 17,378 train samples and 2492 test samples sampled from several online repositories of 3D CAD models. ScanNet-10 (called ScanNet hereafter) contains 6110 train and 1769 test samples. ScanNet-10 is an RGB-D video dataset of scanned real-world indoor scenes. Samples from this dataset are significantly harder to classify because: (i) Many objects have missing parts, due to occlusions and, (ii) some objects are sampled sparsely. See Figure 3 for a comparison of typical shapes from all the datasets mentioned above. Further details about the experimental setup are provided in Section B in the Appendix.

#### 4.2. PointSegDA

For evaluating point cloud segmentation we built a new benchmark dataset called PointSegDA. It is based on a dataset of meshes of human models proposed by [25], which consists of four subsets: ADOBE, FAUST, MIT; and SCAPE. They share eight classes of human body parts (feet, hand, head, etc.) but differ in point distribution, pose and, scanned humans. Point clouds were extracted and labeled from the meshes to accommodate our setup. PointSegDA differs from PointDA-10 in the type of the domain shifts (different humans, poses and discretizations in each subset), the actual shapes and in the fact that it represent deformable objects. Thus, PointSegDA allows us to show the applicability of DefRec in a fundamentally different setup. Further details are provided in Appendix A and B.2.
Table 1: Test accuracy on PointDA-10 dataset, averaged over three runs (± SEM).

<table>
<thead>
<tr>
<th>Method</th>
<th>ModelNet to ShapeNet</th>
<th>ModelNet to ScanNet</th>
<th>ShapeNet to ModelNet</th>
<th>ShapeNet to ScanNet</th>
<th>ScanNet to ModelNet</th>
<th>ScanNet to ShapeNet</th>
</tr>
</thead>
<tbody>
<tr>
<td>Supervised-T</td>
<td>93.9 ± 0.2</td>
<td>78.4 ± 0.6</td>
<td>96.2 ± 0.1</td>
<td>78.4 ± 0.6</td>
<td>96.2 ± 0.1</td>
<td>93.9 ± 0.2</td>
</tr>
<tr>
<td>Supervised</td>
<td>89.2 ± 0.6</td>
<td>76.2 ± 0.6</td>
<td>93.4 ± 0.6</td>
<td>74.7 ± 0.7</td>
<td>93.2 ± 0.3</td>
<td>88.1 ± 0.7</td>
</tr>
<tr>
<td>Unsupervised</td>
<td>83.3 ± 0.7</td>
<td>43.8 ± 2.3</td>
<td>75.5 ± 1.8</td>
<td>42.5 ± 1.4</td>
<td>63.8 ± 3.9</td>
<td>64.2 ± 0.8</td>
</tr>
<tr>
<td>DANN [14]</td>
<td>75.3 ± 0.6</td>
<td>41.5 ± 0.2</td>
<td>62.5 ± 1.4</td>
<td>46.1 ± 2.8</td>
<td>53.3 ± 1.2</td>
<td>63.2 ± 1.2</td>
</tr>
<tr>
<td>PointDAN [32]</td>
<td>82.5 ± 0.8</td>
<td>47.7 ± 1.0</td>
<td>77.0 ± 0.3</td>
<td>48.5 ± 2.1</td>
<td>55.6 ± 0.6</td>
<td>67.2 ± 2.7</td>
</tr>
<tr>
<td>RS [38]</td>
<td>81.5 ± 1.2</td>
<td>35.2 ± 5.9</td>
<td>71.9 ± 1.4</td>
<td>39.0 ± 0.7</td>
<td>61.0 ± 3.3</td>
<td>63.6 ± 3.4</td>
</tr>
<tr>
<td>DAE-Global [18]</td>
<td>83.5 ± 08</td>
<td>42.6 ± 1.4</td>
<td>74.8 ± 0.8</td>
<td>45.5 ± 1.6</td>
<td>64.9 ± 4.4</td>
<td>67.3 ± 0.6</td>
</tr>
<tr>
<td>DAE-Point</td>
<td>82.5 ± 0.4</td>
<td>40.2 ± 1.6</td>
<td>76.4 ± 0.7</td>
<td>50.2 ± 0.5</td>
<td>66.3 ± 1.5</td>
<td>66.1 ± 0.5</td>
</tr>
<tr>
<td>DefRec (ours)</td>
<td>83.3 ± 0.2</td>
<td>46.6 ± 2.0</td>
<td>79.8 ± 0.5</td>
<td>49.9 ± 1.8</td>
<td>70.7 ± 1.4</td>
<td>64.4 ± 1.2</td>
</tr>
<tr>
<td>DefRec + PCM (Ours)</td>
<td>81.7 ± 0.6</td>
<td>51.8 ± 0.3</td>
<td>78.6 ± 0.7</td>
<td>54.5 ± 0.3</td>
<td>73.7 ± 1.6</td>
<td>71.1 ± 1.4</td>
</tr>
</tbody>
</table>

4.3. Network architecture

The input to the network is a point cloud with 1024 points (for PointDA-10) or 2048 points (for PointSegDA). For a feature extractor and the supervised task head, we used DGCNN [49] with the same configurations as in the official implementation. As for the SSL head, differently from common solutions in the literature (e.g. [1, 18]), it takes as input the global feature vector (of size 1024) concatenated to the feature representations of the points from the backbone network. We consistently found that it generates better solutions. Additional details on the architecture are presented in Appendix B.

5. Results

We now discuss the results of using DefRec and PCM. The same pre-processing and experimental setup was applied to all methods (ours and baselines). In all experiments we report the mean accuracy and standard error of the mean (SEM) across three runs with different seeds. For each of the three deformation types of DefRec we examined different hyper-parameters, such as radii size for volume-based deformations or layer depth for feature-based deformations. See detailed explanation in Appendix B.

5.1. Classification accuracy

We compared DefRec with the following baselines: (1) Unsupervised, using only labeled source samples. (2) DANN [14], a baseline commonly used in DA for images. Training with a domain classifier to distinguish between source and target. (3) PointDAN [32] that suggested to align features both locally and globally. The global feature alignment is implemented using the method proposed in [36] and therefore this baseline can also be seen as an extension of it. (4) RS, using the SSL task for point clouds suggested in [38] instead of DefRec. Split the space to $3 \times 3 \times 3$ equally sized voxels, shuffling them and, assign the network to reconstruct the original order. (5) Denoising Auto-Encoder (DAE)-Global [18], reconstruction from a point cloud perturbed with i.i.d Gaussian noise. Since this method proposed to reconstruct from a global feature vector we also compared to (6) DAE-Point, reconstruction from a point cloud perturbed with i.i.d. Gaussian noise with the same input to $h_{SSL}$ as DefRec; a concatenation of the global feature vector to the point features. We also present two upper bounds: (1) Supervised-T, training on target domain only and, (2) Supervised, training with labeled source and target samples. DefRec, RS, DAE-Point and DANN all have 2.5M parameters, compared with 11.1M in PointDAN and 12.6M parameters in DAE-Global.

We test several deformation types, each with its own variants (such as the radius size for volume-based deformations). To get a unified measure of performance, we treated the deformation type and its variants as hyperparameters. Then, for each adaptation, we followed a stringent protocol and picked the model that maximized accuracy on the validation set of the source domain.

Table 1 shows the classification accuracy of all methods on the six adaptation tasks. Figure 4 shows the average accuracy per method across the six adaptation tasks. As can be seen, our methods outperform all baselines in 5 out of 6 adaptations. The average across six adaptations of both of our methods (DefRec and DefRec + PCM) is the highest. DefRec + PCM improve by 5% compared to the best baseline and by 5.5% compared to PointDAN, the natural competitor on this dataset. Also, DefRec is more accurate on sim-to-real adaptations (ModelNet-to-ScanNet and ShapeNet-to-ScanNet). This observation validates our intuition: (i) DefRec promotes learning semantic properties of the shapes and, (ii) DefRec helps the model in generalizing to real data that has missing regions/parts.

Appendix C.1 further quantifies the effect of combining PCM with baseline methods. PCM boosts the performance of several baselines. However, our proposed DefRec+PCM is still superior. Appendix C.2 present a variant of DefRec in which the deformation types are combined efficiently.
5.2. Analysis

We now analyze the three deformation types of DefRec. See Appendix D and E for further analysis of the representation learned and demonstration of shapes reconstruction.

5.2.1 Accuracy by deformation category

Table 2 shows the test accuracy for the three types of deformations: Volume-based, Feature-based and Sample-based. For each type, per adaptation, we selected the best model among all variants of that family according to source-validation accuracy. As seen from the table, deforming based on proximity in the input space yields the highest accuracy on average. In fact, across all adaptations, variants of the volume-based deformation type also had the highest source-validation accuracy. Note that when considering the three types of deformation separately, namely considering each type of deformation as a separate method (unlike the results in Table 1 in which we treated the type as a hyper-parameter), DefRec has the highest accuracy across all adaptation tasks.

5.2.2 Volume-based deformations

The effectiveness of volume-based deformation is sensitive to the size of a deformed region. Small regions may be too easy for the network to reconstruct, while large regions may be very hard. To quantify this sensitivity Figure 5a shows the mean accuracy gain averaged over 6 adaptations as a function of deformation radius. All values denote the gain compared with a baseline of deforming the full shape ($r=2.0$). Accuracy is maximized with small to mid-sized regions, with an optimum at $r=0.2$. This suggest that deformations at the scale of object parts are superior to global deformations, in agreement with the intuition that mid-scale regions capture the semantic structures of objects.

5.2.3 Feature-based deformations

Figure 5b traces the accuracy as a function of the number of points when deforming based on proximity in feature space. As in Section 5.2.2, we find that deforming large regions degrades the performance, particularly with more than 300 points. Also, layer 4 is dominated by layer 3 by a small gap. Overall, the model is largely robust to the choice of layer and the number of points for small enough regions.

5.2.4 SSL vs data augmentation

In this paper we promote the use SSL tasks for bridging the domain gap. An interesting question arises: could this gap be bridged using deformations as a data augmentation mechanism? In this case, we may use only the labeled source samples for supervision. As an example, consider a sim-to-real adaptation task. The sampling procedures suggested in this paper can be used for data augmentation. These methods sample some parts of the object more densely and other parts more sparsely. As a result, the augmented shapes may resemble to shapes from the target data.

To test this idea we use the sample-based deformations in two fashions: (i) As an SSL task, the method advocated in this paper and, (ii) as a data augmentation procedure for source samples. Figure 5c compares these alternatives on the six adaptations tasks with the three sampling procedures. Most data points (11/18) are below the diagonal line $y = x$, five of which are on sim-to-real adaptations tasks. This result suggests that using the sampling procedures as an SSL task should be preferred over data augmentation.
5.4. DefRec for segmentation

To examine how DefRec generalizes beyond classification tasks, we tested it in segmentation tasks using PointSegDA dataset. It is easy to extend DefRec to this task. Similarly to classification, the network is trained jointly with two losses: one to segment labeled source objects and the second to reconstruct deformed target objects. Since labels are now provided per point, not per object, we adapted PCM to segmentation in the following way. Similarly to PCM for classification, we generate a new point cloud from a random split of two point clouds in the batch. Here, however, each point from the original shape migrates with its associated label. The overall loss is the mean cross-entropy calculated for all points in the new mixed shape.

Table 4 compares the mean Intersection over Union (IoU) of DefRec with the unsupervised baseline, RS [38], and Adapt-SegMap [44], a strong DA baseline for semantic segmentation of images which applies adversarial training in the output space. The experiment shows that: (i) Adding auxiliary SSL tasks helps to bridge the domain gap for segmentation and, (ii) DefRec (either with or without PCM) achieves the highest results on most adaptations, validating the applicability of DefRec to segmentation tasks.

6. Conclusions

We tackled the problem of domain adaptation on 3D point clouds. We designed DefRec, a novel family of self-supervised pretext tasks inspired by the kind of deformations encountered in real 3D point cloud data. In addition, we designed PCM, a new training procedure for 3D point clouds based on the Mixup method that can be applied to any classification or segmentation task. PCM is complementary to DefRec, and when combined they form a strong model with relatively simple architecture. We demonstrated the benefit of our method on several adaptation setups, reaching a new state of the art results.

Acknowledgments

IA was funded by the Israel innovation authority as part of the AVATAR consortium, and by a grant from the Israel Science Foundation (ISF 737/2018).
References


