

Zero-Shot Learning on 3D Point Cloud Objects and Beyond

Ali Cheraghian · Shafin Rahman · Townim F. Chowdhury ·
Dylan Campbell · Lars Petersson

Received: date / Accepted: date

Abstract Zero-shot learning, the task of learning to recognize new classes not seen during training, has received considerable attention in the case of 2D image classification. However, despite the increasing ubiquity of 3D sensors, the corresponding 3D point cloud classification problem has not been meaningfully explored and introduces new challenges. In this paper, we identify some of the challenges and apply 2D Zero-Shot Learning (ZSL) methods in the 3D domain to analyze the performance of existing models. Then, we propose a novel approach to address the issues specific to 3D ZSL. We first present an inductive ZSL process and then extend it to the transductive ZSL and Generalized ZSL (GZSL) settings for 3D point cloud classification. To this end, a novel loss function is developed that simultaneously aligns seen semantics with point cloud features and takes advantage of unlabeled test data to address some known issues (e.g., the problems of domain adaptation, hubness, and data bias). While designed for the particularities of 3D point cloud clas-

sification, the method is shown to also be applicable to the more common use-case of 2D image classification. An extensive set of experiments is carried out, establishing state-of-the-art for ZSL and GZSL on synthetic (ModelNet40, ModelNet10, McGill) and real (ScanObjectNN) 3D point cloud datasets.

Keywords Zero-shot Learning · 3D Point Clouds · Transductive Learning · Hubness Problem

1 Introduction

Capturing 3D point cloud data from complex scenes has been facilitated by increasingly accessible and inexpensive 3D depth camera technology. This in turn has expanded the interest in, and need for, 3D object classification methods that can operate on such data. However, much if not most of the data collected will belong to classes for which a classification system may not have been explicitly trained. In order to recognize such previously “unseen” classes, it is necessary to develop Zero-Shot Learning (ZSL) methods in the domain of 3D point cloud classification. While such methods are typically trained on a set of so-called “seen” classes, they are capable of classifying certain “unseen” classes as well. Knowledge about unseen classes is introduced to the network via semantic feature vectors that can be derived from networks pre-trained on image attributes or on a very large corpus of texts [37, 3, 1, 74, 65].

Performing ZSL for the purpose of 3D object classification is a more challenging task than ZSL applied to 2D images [43, 74, 1, 3, 37, 25, 65]. We identify three particular challenges in this regard.

1. Availability of high quality pre-trained models: ZSL methods in the 2D domain commonly take advantage of pre-trained models, like ResNet [18], that

Ali Cheraghian
Data61, CSIRO, ACT 2601, AU
Australian National University, Canberra ACT 0200 AU
E-mail: ali.cheraghian@anu.edu.au

Shafin Rahman
North South University, Dhaka, Bangladesh
E-mail: shafin.rahman@northsouth.edu

Townim F. Chowdhury
North South University, Dhaka, Bangladesh
E-mail: townim.faisal@northsouth.edu

Dylan Campbell
University of Oxford, Oxford, United Kingdom
E-mail: dylan@robots.ox.ac.uk

Lars Petersson
Data61, CSIRO, ACT 2601, AU
E-mail: lars.petersson@data61.csiro.au

have been trained on millions of labeled images featuring thousands of classes. As a result, the extracted 2D features are very well clustered. By contrast, there is no parallel in the 3D point cloud domain; labeled 3D datasets tend to be small and have only limited sets of classes. For example, pre-trained models like PointNet [38] are trained on only a few thousand samples from a small number of classes. This leads to poor-quality 3D features with clusters that are not nearly as well separated as their visual counterparts.

2. The hubness problem: In high-dimensional data, some points—called hubs—occur frequently in the k -nearest neighbor sets of other points. This is a consequence of the curse of dimensionality associated with nearest neighbor (NN) search [41]. In ZSL, the hubness problem occurs for two reasons [51]. Firstly, both input and semantic features reside in a high dimensional space. Secondly, ridge regression, which is widely used in ZSL, is known to induce hubness. As a result, it causes a bias in the predictions, with only a few classes predicted most of the time regardless of the query. The hubness problem is exacerbated by the relatively poor quality of 3D features, making it more difficult to relate those features to their corresponding semantics [74].
3. The domain shift problem: The function learned from seen samples is biased to those samples and cannot generalize well to unseen classes. In the *inductive learning* approach, where only seen classes are used during training, projected semantic vectors tend to move towards the seen feature vectors, making the intra-class distance between corresponding unseen semantic and feature vectors large. Similar to hubness, the domain shift problem is intensified when training is done on seen synthetic 3D point cloud objects (ModelNet40 [63]), but testing on unseen real-world 3D scanned data (ScanObjectNN [56]).

Some intuition about these challenges can be attained by visualizing the respective pre-trained feature spaces, as shown in Figure 1 for the 3D datasets (a) ModelNet10 [63] and (b) ScanObjectNN [56], and the 2D datasets (c) AwA2 [24] and (d) CUB [59]. The quality of the image features is much higher than the point cloud features, with a much more separable cluster structure. When the clusters are not well-separated, the hubness and domain shift problem are worsened. In this paper, we address the following questions for ZSL on 3D point cloud data:

(a) *How do standard ZSL approaches perform on low quality 3D point cloud features?* We conduct a series of experiments utilising four popular structures tradi-

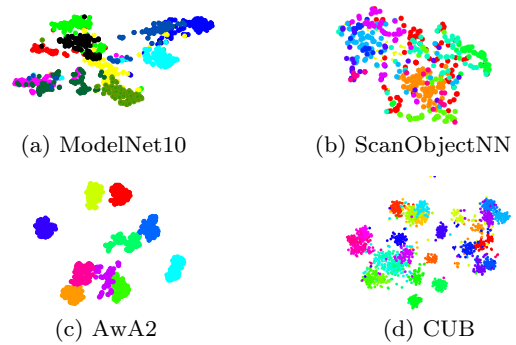


Fig. 1: tSNE [57] visualizations of unseen 3D point cloud features of the (a) ModelNet10 [63] and (b) ScanObjectNN [56] datasets, and unseen 2D image features of the (c) AwA2 [65] and (d) CUB [59] datasets. The cluster structure in the 2D feature space is much better defined, with tighter and more separated clusters than those in the 3D point cloud.

tionally used for feature extraction in 3D point clouds. These are PointNet [38], DGCNN [61], PointConv [62], and PointAugment [30]. With the help of these base architectures, we build a structure for ZSL that combines point cloud features with word vector semantic features thereby enabling the classification of previously unseen 3D classes. This combination process follows the standard approach of ZSL that maps the point cloud features to the space of semantic vectors. The performance obtained by this approach shows the complex nature of ZSL tasks on 3D data due to the poor feature quality, the hubness and domain-shift problems. However, it establishes suitable baselines for any ZSL models on 3D point cloud data.

(b) *How much can the domain shift be mitigated?*

In this paper, we attempt to address the domain shift problem using transductive learning. Our goal is to design a strategy that reduces the bias and encourages the projected semantic vectors to align with their true feature vector counterparts, minimizing the average intra-class distance. In 2D ZSL, the transductive setting has been shown to be effective [15, 76, 53], however, in the case of 3D point cloud data it is a more challenging task. Pre-trained 3D features are poorly clustered and exhibit large intra-class distances. In order to take advantage of the transductive learning approach for 3D point cloud zero-shot learning, we propose a transductive ZSL method using a novel triplet loss that is employed in an unsupervised manner. Unlike the traditional triplet formulation [50, 40], our proposed triplet loss works on unlabeled (test) data and can operate without the need of ground-truth supervision. This loss applies to unlabeled data such that intra-class distances are minimized while

also maximizing inter-class distances, reducing the bias problem. In addition to the triplet loss, we also employ a distance-based unbiased loss to balance seen and unseen prediction scores. As a result, a prediction function with greater generalization ability and effectiveness on unseen classes is learned.

(c) *How can we address the hubness problem for 3D data?* The hubness problem occurs when a model is biased to predict a small subset of labels for most of the test instances. Popular ZSL methods on 2D image data usually project the semantic features to the space of visual features to handle the hubness problem. In this paper, we first design our architecture by following the same trend and observe the performance gain of applying the reverse projecting trick. Secondly, to further improve the performance, we propose a new loss for the transductive setting to explicitly alleviate the hubness problem. We calculate this loss by evaluating each unlabeled test data element in an unsupervised manner, and counting the number of times each class gets predicted on the batch. This is used to estimate a measure of hubness: the skewness of the current prediction. We minimize the skewness of each batch to reduce the degree of hubness.

In addition to 3D point cloud data, our proposed method is also applicable in the case of 2D ZSL, which demonstrates the generalization strength of our method to other sensor modalities. Our main contributions are: (1) an evaluation of the zero-shot learning (ZSL) and generalized zero-shot learning (GZSL) tasks for 3D point cloud classification by adapting both inductive and transductive learning settings; (2) a novel triplet loss that takes advantage of unlabeled test data, applicable to both 3D point cloud data and 2D images; (3) an approach to address the hubness and bias problems of G/ZSL in transductive settings; (4) a new evaluation protocol for ZSL methods on 3D point clouds which consists of a seen and unseen split of data from the datasets ModelNet40 [63], ModelNet10 [63], McGill [52] and ScanObjectNN [56], and performing extensive experiments, establishing state-of-the-art on four 3D datasets.

Preliminary sections of this paper have been published previously [9, 8, 10]. Here, we encapsulate the contributions in a unified framework and extend the previous work as follows: (1) we address the hubness problem in the transductive settings and propose a new loss to balance seen and unseen scores; (2) we analyze the framework in detail, with new ablation studies, and situate it within the context of the related work; (3) we provide extensive evaluation of eight established ZSL and GZSL methods on 3D point cloud data; and (4) we propose a new seen/unseen split for a real-world

scanned 3D object dataset (ScanObjectNN) and evaluate on this dataset.

2 Related works

3D point cloud object recognition architecture:

The early methods utilizing deep learning for operating on 3D point clouds used volumetric [63] or multi-view [55] representations in order to work with 3D data. Recently, the trend in this area has shifted to instead using raw point clouds directly [39, 61, 28], without any preprocessing step. These methods do not suffer to the same degree from scalability issues as the volumetric representation does, and they do not make any *a priori* assumptions onto which 2D planes, and how many, that the point cloud should be projected on, like the view-based methods do. PointNet [38] was the first work that operated on raw point clouds directly at the input of the network. PointNet used a multi-layer perceptron (mlp) [46] to extract features from point sets, and max-pooling layers to remove the otherwise inherent issue of permutation from the point clouds. Later, many methods [39, 61, 28, 62, 30, 7] were proposed to overcome the limitations of PointNet, which does not utilize local features or a more advanced pooling operation than max-pooling. The traditional recognition where all the classes of interest have been seen at training time, have been considered in the case of 3D point cloud data. The current literature does not fully address the zero-shot version of the 3D recognition problem [9, 8, 10]. In this paper, we perform both transductive and inductive ZSL and GZSL on 3D point cloud objects.

Zero-Shot Learning: For the ZSL task, there has been significant progress, including on image recognition [43, 74, 1, 3, 37, 25, 65], multi-label ZSL [26, 42], and zero-shot detection [44]. Despite this progress, these methods solve the constrained problem where the test instances are restricted to only unseen classes, rather than being from either seen or unseen classes. This setting, where both seen and unseen classes are considered at test time, is called Generalized Zero-Shot Learning (GZSL). To address this problem, some methods decrease the scores that seen classes produce by a constant value [5], while others perform a separate training stage intended to balance the probabilities of the seen and unseen classes [43]. Also, some Generative Adversarial Networks (GAN) based approaches [66, 20, 49, 29, 36, 58, 31, 16] have been proposed to solve ZSL and GZSL problems in recent years. Schonfeld *et al.* [49] learned a shared latent space of image features and semantic representation based on a modality-specific VAE model. In our work, we propose novel loss functions (for both

inductive and transductive cases) to address the bias problem, leading to significantly better GZSL results.

Transductive Zero-shot Learning: The transductive learning approach takes advantage of unlabeled test samples, in addition to the labeled seen samples. For example, Rohrbach *et al.* [45] exploited the manifold structure of unseen classes using a graph-based learning algorithm to leverage the neighborhood structure within unseen classes. Fu *et al.* [15] proposed a multi-view transductive setting to address projection shift and to exploit various semantic representations of the visual feature. Yu *et al.* [71] proposed a transductive approach to predict class labels via an iterative refining process. Guo *et al.* [17] proposed a joint learning method that learns a shared model space to share knowledge between seen and unseen classes using semantic attributes jointly. All of these methods attempt to improve the accuracy of the unseen classes in transductive settings. More recently, transductive ZSL methods have started exploring how to improve the accuracy of both the seen and unseen classes in generalized ZSL tasks [76, 53]. Zhao *et al.* [76] proposed a domain invariant projection method that projects visual features to semantic space and reconstructs the same feature from the semantic representation in order to narrow the domain gap. In another approach, Song *et al.* [53] identified the model bias problem of inductive learning, that is, a trained model assigns higher prediction scores for seen classes than unseen. To address this, they proposed a quasi-fully supervised learning method to solve the GZSL task. Xian *et al.* [67] proposed f-VAEGAN-D2 which takes advantage of both VAEs and GANs to learn the feature distribution of unlabeled data. Narayan *et al.* [36] followed the same setting as proposed in the baseline f-VAEGAN-D2 [67]. Gao *et al.* [16] used K-Nearest Neighbors and classification probability to provide pseudo-labels for unlabeled unseen features. All of these approaches are designed for transductive ZSL tasks on 2D image data. In contrast, we explore to what extent a transductive ZSL setting helps to improve 3D point cloud recognition.

The Hubness Problem: The hubness problem in high dimensional nearest neighbor search spaces was first investigated in [41] where they illustrate that the hubness problem is related to the data distribution in the high dimensional space. In later studies [12, 51, 74], the hubness problem in ZSL is investigated. Dinu *et al.* [12] proposed an algorithm that corrects the hubness problem by using more unlabeled seen data in addition to test instances. Shigeto *et al.* [51] mentioned that the projection function used for least squares regularization affect the hubness problem negatively and instead introduces a reverse regularized function in order to weaken the

hubness problem. In contrast to the mentioned works, Zhang *et al.* [74] proposed to deal with the hubness problem by instead considering the feature space as the embedding space. In this paper, we address the hubness problem of ZSL on 3D point cloud classification.

Learning with a Triplet Loss: Triplet losses have been widely used in computer vision [50, 40, 14, 19, 13]. Schroff *et al.* [50] demonstrated how to select positive and negative anchor points from visual features within a batch. Qiao *et al.* [40] introduced using a triplet loss to train an inductive ZSL model. More recently, Do *et al.* [13] proposed a tight upper bound of the triplet loss by linearizing it using class centroids, Zakharov *et al.* [72] explored the triplet loss in manifold learning, Srivastava *et al.* [54] investigated weighting hard negative samples more than easy negatives, and Zhaoqun *et al.* [32] proposed the angular triplet-center loss, a variant that reduces the similarity distance between features. Triplet loss related methods typically work under inductive settings, where the ground-truth label of an anchor point remains available during training. In contrast, we describe a triplet formation technique in the transductive setting. Our method utilizes test data without knowing its true label. Moreover, we choose positive and negative samples of an anchor from word vectors instead of features.

3 Zero-Shot Learning for 3D Point Clouds

The comparative lack of large-scale 3D datasets with many object categories has meant that 3D features are not as robust and separable as 2D features. As a result, relating 3D features to their corresponding semantic vectors is more difficult than for the 2D case. Addressing the poor feature quality of typical 3D datasets, we investigate suitable 3D point cloud architectures and loss functions in both transductive and inductive settings. Our method specifically addresses the alignment of poor features (like those coming from 3D feature extractors) with semantic vectors. Therefore, while our method improves the results for both 2D and 3D modalities, the largest gain is observed in the 3D case.

3.1 Problem formulation

Let $\mathcal{X} = \{\mathbf{x}_i\}_{i=1}^n$ for $\mathbf{x}_i \in \mathbb{R}^3$ denote a 3D point cloud. Also let $\mathcal{Y}^s = \{y_i^s\}_{i=1}^S$ and $\mathcal{Y}^u = \{y_i^u\}_{i=1}^U$ denote disjoint ($\mathcal{Y}^s \cap \mathcal{Y}^u = \emptyset$) seen and unseen class label sets with sizes S and U respectively, and $\mathcal{E}^s = \{\phi(y_i^s)\}_{i=1}^S$ and $\mathcal{E}^u = \{\phi(y_i^u)\}_{i=1}^U$ denote the sets of associated semantic embedding vectors for the embedding function $\phi(\cdot)$, with $\phi(y) \in \mathbb{R}^d$. Then we define the set of n_s seen

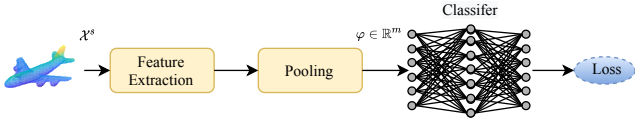


Fig. 2: General framework of a point cloud architecture. A traditional 3D point cloud recognition system consists of a feature extraction module, a pooling module, and a classifier. We design our backbone using such frameworks.

instances as $\mathcal{Z}^s = \{(\mathcal{X}_i^s, l_i^s, \mathbf{e}_i^s)\}_{i=1}^{n_s}$, where \mathcal{X}_i^s is the i^{th} point cloud of the seen set with label $l_i^s \in \mathcal{Y}^s$ and semantic vector $\mathbf{e}_i^s = \phi(l_i^s) \in \mathcal{E}^s$. The set of n_u unseen instances is defined similarly as $\mathcal{Z}^u = \{(\mathcal{X}_i^u, l_i^u, \mathbf{e}_i^u)\}_{i=1}^{n_u}$, where \mathcal{X}_i^u is the i^{th} point cloud of the unseen set with label $l_i^u \in \mathcal{Y}^u$ and semantic vector $\mathbf{e}_i^u = \phi(l_i^u) \in \mathcal{E}^u$.

We consider two learning problems in this work: zero-shot learning and its generalized variant. The goal of each problem is defined as follows.

- Zero-Shot Learning (ZSL): predict a class label $\hat{y}^u \in \mathcal{Y}^u$ from the unseen label set given an unseen point cloud \mathcal{X}^u .
- Generalized Zero-Shot Learning (GZSL): predict a class label $\hat{y} \in \mathcal{Y}^s \cup \mathcal{Y}^u$ from the seen or unseen label sets given a point cloud \mathcal{X} .

In this paper, we solve ZSL and GZSL problems in both the inductive and transductive setting. Transductive settings allow the use of unlabeled unseen point cloud instances \mathcal{X}^u during the training stage, whereas inductive settings do not allow access to this unlabeled information.

3.2 Point cloud feature extractors

Given an unordered point set representing an object from a seen class $\mathcal{X}^s = \{\mathbf{x}_1^s, \dots, \mathbf{x}_n^s\}$, a set function is defined such that any permutation of the point set is irrelevant,

$$f(\mathbf{x}_1^s, \mathbf{x}_2^s, \dots, \mathbf{x}_n^s) \approx g(h(\mathbf{x}_1^s, \beta), h(\mathbf{x}_2^s, \beta), \dots, h(\mathbf{x}_n^s, \beta))$$

where f is the set function, h is the feature extraction function, g is the pooling function with the ability to remove the effects of permutation of points in a set, and β represents a set of arguments associated with \mathbf{x}_i^s . The feature extraction function $h(\mathbf{x}_i^s, \beta)$ extracts a richer representation from the point cloud in a higher dimension. For instance, in PointNet [38], $h(\mathbf{x}_i^s, \beta) = h(\mathbf{x}_i^s) : \mathbb{R}^d \rightarrow \mathbb{R}^{d'}$, $\beta = \{\emptyset\}$, since each point is considered separately, the extracted feature vector contains global information. As another example, in DGCNN [61], which

extracts local features as well as global features, $h(\mathbf{x}_i^s, \beta) = h(\mathbf{x}_i^s, \mathbf{x}_j^s - \mathbf{x}_i^s) : \mathbb{R}^d \times \mathbb{R}^d \rightarrow \mathbb{R}^{d'}$, $\beta = \{\mathbf{x}_j^s - \mathbf{x}_i^s\}$. In this case, point sets are represented by a dynamic graph and edge features based on k -nearest neighbors are calculated. Since point sets are inherently unordered, a function which is invariant to permutation is necessary to pool point features into a feature vector. Here, g , is capable of removing the effects of permutation from point clouds. Finally, via a collection of $h(\mathbf{x}_i^s, \beta)$, corresponding values of f can be computed to form a vector $\varphi(\mathcal{X}^s) \in \mathbb{R}^m$. The obtained feature vector removes permutation from the point cloud. In the next step, a few fully-connected layers are applied to the feature vector $\varphi(\mathcal{X}^s)$ in order to transform the features into label space, where a cross-entropy loss is used to train the point cloud backbone. We illustrate the point cloud feature extractor architecture in Figure 2.

3.3 Inductive ZSL on point cloud data

Our model is trained in a fully-supervised manner with seen instances only from the set \mathcal{Z}^s . Let N be the number of instances in the batch and $\varphi(\mathcal{X}_i^s) \in \mathbb{R}^m$ be the point cloud feature vector associated with point cloud \mathcal{X}_i^s . For ZSL, both point cloud feature $\varphi(\mathcal{X})$ and semantic \mathcal{E} vectors need to embed into the same embedding space. In the ZSL literature, this is done in two ways, and we investigate both in the context of 3D point cloud objects.

Feature to Semantic (F2S): a point cloud feature $\varphi(\mathcal{X}_i^s)$ is projected into the semantic embedding space \mathcal{E} using a nonlinear projection function $\Theta_1(\cdot)$ with weights W_1 . The network calculates the following loss:

$$L_{F2S} = \frac{1}{N} \sum_{i=1}^N \|\Theta_1(\varphi(\mathcal{X}_i^s); W_1) - \mathbf{e}_i^s\|_2^2 + \lambda_1 \|W_1\|_2^2 \quad (1)$$

where the parameter λ_1 controls the amount of regularization.

Semantic to Feature (S2F): a semantic vector \mathbf{e}_i^s is projected into point cloud feature space using the nonlinear projection function $\Theta_2(\cdot)$ weights W_2 . The network calculates the following loss:

$$L_{S2F} = \frac{1}{N} \sum_{i=1}^N \|\varphi(\mathcal{X}_i^s) - \Theta_2(\mathbf{e}_i^s; W_2)\|_2^2 + \lambda_2 \|W_2\|_2^2 \quad (2)$$

where the parameter λ_2 controls the amount of regularization.

Zhang *et al.* [74] argue that ZSL models based on Semantic to Feature (S2F) projection exhibit less hubness than Feature to Semantic (F2S) projection models. In our experiments, we add evidence that this is also true

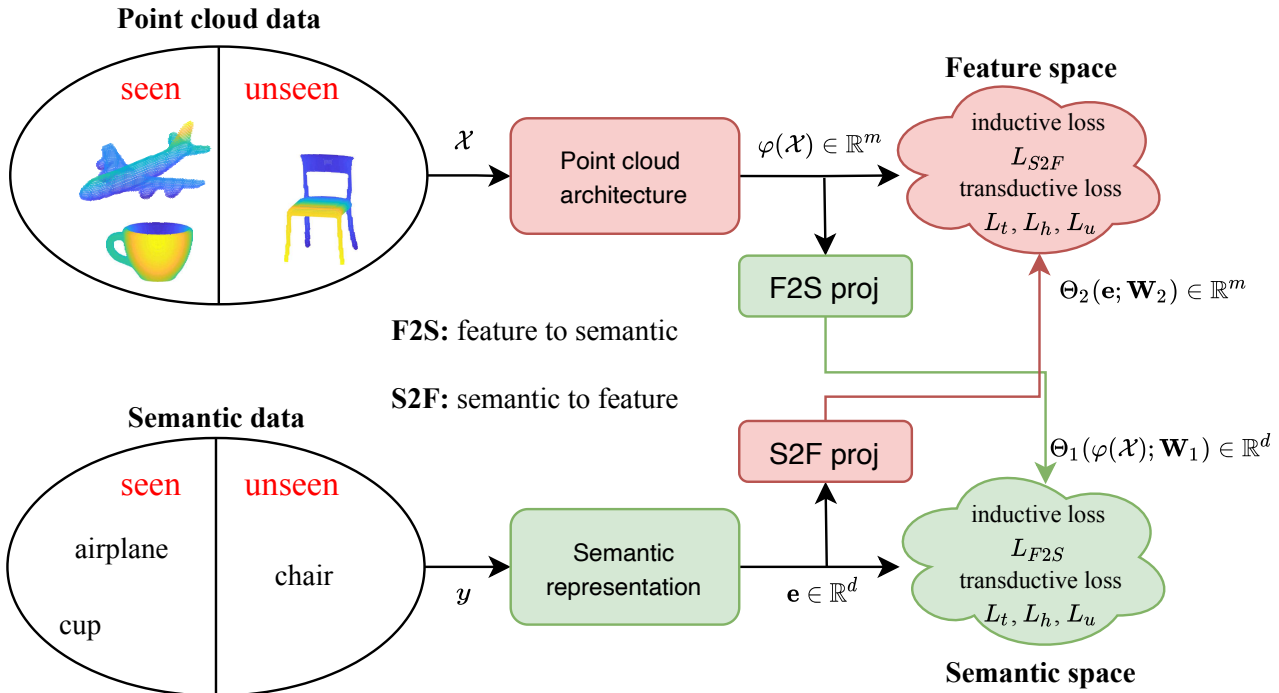


Fig. 3: The proposed architecture for ZSL and GZSL. For inductive learning, the input point cloud and semantic representation are $\mathcal{X} = \mathcal{X}^s$ and $\mathbf{e} = \phi(y) \in \mathcal{E}^s$, respectively. For transductive learning, the input point cloud and semantic representation are $\mathcal{X} = \mathcal{X}^s \cup \mathcal{X}^u$ and $\mathbf{e} \in \mathcal{E}^s \cup \mathcal{E}^u$ respectively. We project point cloud features to semantic space (F2S) or semantic vectors to feature space (S2F) and calculate distances between feature and semantics. Our proposed losses minimize those distances in both the inductive and transductive setting.

for 3D data. For the remainder of this treatment, we follow S2F embedding (that is, projection with $\Theta_2(\cdot)$) for ZSL.

3.4 Transductive ZSL on point cloud data

Transductive ZSL addresses the problem of the projection domain shift [15] inherent in inductive ZSL approaches. In ZSL, the seen and unseen classes are disjoint and often only very weakly related. Since the underlying distributions of the seen and unseen classes may be quite different, the ideal projection function between the semantic embedding space and point cloud feature space is also likely to be different for seen and unseen classes. As a result, using the projection function learned from only the seen classes without considering the unseen classes will cause an unknown bias. Transductive ZSL reduces the domain gap and the resulting bias by using unlabeled unseen class instances during training, improving the generalization performance. The effect of the domain shift in ZSL is shown in Figure 4. When inductive learning is used (a), the projected unseen semantic embedding vectors are far from the cluster centres of the associated point cloud

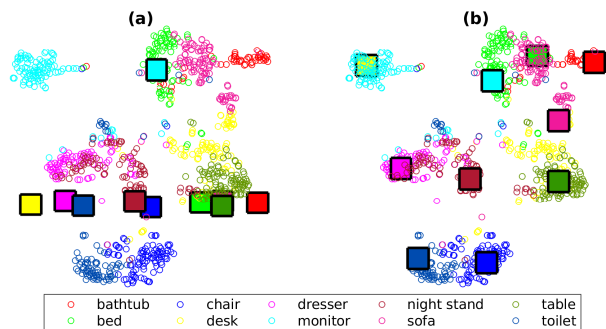


Fig. 4: 2D tSNE [57] visualization of unseen point cloud feature vectors (circles) and projected semantic feature vectors (squares) based on (a) inductive and (b) transductive learning on ModelNet10. The projected semantic feature vectors are much closer to the cluster centres of the point cloud feature vectors for transductive ZSL than for inductive ZSL, showing that the transductive approach is able to narrow the domain gap between seen and unseen classes.

feature vectors, however, when transductive learning is used (b), the vectors are much closer to the cluster centres.

Unsupervised triplet loss: We propose an unsupervised triplet loss that operates on the unlabeled test data. To compute a triplet loss, a positive and negative sample need to be found for each anchor sample [50]. In the fully-supervised setting, selecting positive and negative samples is not difficult, because all training samples have ground-truth labels. However, it is more challenging in the unsupervised setting, where ground-truth labels are not available. For transductive ZSL, we define a positive sample using a *pseudo-labeling* approach [27]. For each anchor \mathcal{X}^u , we assign a pseudo-label that chooses a positive sample \mathbf{e}^+ among the semantic embedding vectors which is the closest to the anchor feature vector $\varphi(\mathcal{X}^u)$ after projection $\Theta_2(\cdot)$, as follows

$$\mathbf{e}^+ = \arg \min_{\mathbf{e} \in \mathcal{E}^u} \|\varphi(\mathcal{X}^u) - \Theta_2(\mathbf{e}; W_2)\|_2^2. \quad (3)$$

Such pseudo-labeling is different from the usual practice [27] because it chooses a semantic vector as a positive sample in the triplet formation instead of a plausible ground-truth label. For GZSL, the unlabeled data \mathcal{X}^c for $c \in \{s, u\}$ can be from the seen or unseen classes during training. As a result, a pseudo-label must be found for both unlabeled seen and unlabeled unseen samples. Importantly, if the pseudo-label indicates that an unlabeled sample is from a seen class, then that sample is discarded. This reduces the impact of incorrect, noisy pseudo-labels on the model for seen classes. Samples from seen classes (with ground-truth labels) will instead influence the supervised loss function. Hence, we use true supervision where possible (seen classes), and only use pseudo-supervision where there is no alternative (unseen classes). The positive sample for GZSL is therefore chosen as follows

$$\mathbf{e}^+ = \arg \min_{\mathbf{e} \in \mathcal{E}^s \cup \mathcal{E}^u} \|\varphi(\mathcal{X}^c) - \Theta_2(\mathbf{e}; W_2)\|_2^2. \quad (4)$$

The negative sample is selected from the seen semantic embedding set \mathcal{E}^s for both ZSL and GZSL, since all elements of this set will have a different label from the unseen anchor. We choose the negative sample as the seen semantic embedding vector whose projection is closest to the anchor vector $\varphi(\mathcal{X}^u)$,

$$\mathbf{e}^- = \arg \min_{\mathbf{e} \in \mathcal{E}^s} \|\varphi(\mathcal{X}^s) - \Theta_2(\mathbf{e}; W_2)\|_2^2 \quad (5)$$

Finally, the unsupervised loss function L_t associated with the unlabeled instances for both ZSL and GZSL tasks is defined as follows:

$$L_t = \frac{1}{N'} \sum_{i=1}^{N'} \max \left\{ 0, \|\varphi(\mathcal{X}_i^u) - \Theta_2(\mathbf{e}^+; W_2)\|_2^2 + m - \|\varphi(\mathcal{X}_i^u) - \Theta_2(\mathbf{e}^-; W_2)\|_2^2 \right\} \quad (6)$$

where m is a margin that encourages separation between the clusters, and N' is the batch size of the unlabeled instances.

This proposed triplet loss is distinct from recent literature [50,40] in two ways. (1) Popular methods of triplet formation select a similar feature to the input feature as a positive sample, whereas we choose a semantic word vector for this purpose. This helps to better align the 3D point cloud features with the semantic vectors. (2) We employ a triplet loss in a transductive setting to utilize unlabeled (test) data, whereas established methods consider the triplet loss for inductive training only. This extends the role of the triplet loss beyond inductive learning.

Unsupervised hubness loss: Distance-based ZSL solutions often fall into the trap of the hubness problem. We observe that this issue is intensified for 3D ZSL. To calculate the degree of hubness in a nearest neighbor search problem, the skewness of the empirical distribution ρ_j can be used [51,41]. The distribution ρ_j counts the number of times ($\rho_j(i)$) the i^{th} point (known as the prototype) is in the top j nearest neighbors of the test samples. The skewness of this distribution is defined as:

$$\rho_j\text{-skewness} = \frac{\sum_{i=1}^n (\rho_j(i) - \mathbf{E}[\rho_j])^3}{n (\text{Var}[\rho_j])^{\frac{3}{2}}} \quad (7)$$

where n is the number of test prototypes. Large values of skewness indicate that the feature space is severely affected by the hubness problem. In this paper, we mitigate the hubness problem during both inductive and transductive training. As previously discussed, the S2F strategy is effective at reducing hubness during inductive training. We extend this to transductive training by designing a skewness loss based on Eq. 7.

The pseudo-label predicted for the i^{th} unlabeled instance of a batch with size N is defined as:

$$\hat{y}_i = \arg \min_{\substack{y \in \mathcal{Y}^s \cup \mathcal{Y}^u \\ \mathbf{e} \in \mathcal{E}^s \cup \mathcal{E}^u}} \|\varphi(\mathcal{X}^c) - \Theta_2(\mathbf{e}; W_2)\|_2^2 \quad (8)$$

Then, for all instances in the batch, we predict their pseudo-labels, and define a set $\hat{\mathcal{T}}^c = \{\hat{y}_1, \dots, \hat{y}_N\}$.

We calculate the frequency of each class from $\hat{\mathcal{T}}^c$ by using the histogram function $\mathcal{H}(\hat{y}_i)$, which uses counts of the number of times that a specific seen/unseen class is predicted. This function has the property that $\sum_{i=1}^{S+U} \mathcal{H}(\hat{y}_i^c) = N$. We use the predicted pseudo-labels to find the confidence score. We define the skewness loss as

$$L'_h = \frac{1}{N (\text{Var}[\mathcal{H}(\hat{\mathcal{T}}^c)])^{\frac{3}{2}}} \sum_{i=1}^N (\mathcal{H}(\hat{y}_i^c) - \mathbf{E}[\mathcal{H}(\hat{\mathcal{T}}^c)])^3 \quad (9)$$

where $\mathcal{H}(\hat{\mathcal{T}}^c)$ represents the statistics of prediction for all instances, that is, how many times each output is predicted regardless of being true or false. The loss L'_h tries to balance the number of times a particular class is predicted within a batch and helps the model predict a diverse set of classes. With a larger and randomized batch, the number of particular class instances does not dominate in that batch. As a result, this loss performs better with large batch sizes.

The loss L'_h may impact the correct predictions while balancing predicted class distribution. To counter this, inspired by focal loss [33] we weight each sample in the batch based on their confidence in the prediction. To be more specific, if an example in a batch is confident of predicting a pseudo-label, it should contribute less to the hubness loss and vice versa:

$$\pi = -\frac{1}{N} \sum_{i=1}^N \log \frac{e^{-\|\varphi(\mathcal{X}_i^c) - \Theta_2(\mathbf{e}_{y_i}; W_2)\|_2^2}}{\sum_{k \in \mathcal{Y}^s \cup \mathcal{Y}^u} e^{-\|\varphi(\mathcal{X}_i^c) - \Theta_2(\mathbf{e}_k; W_2)\|_2^2}}. \quad (10)$$

The final unsupervised hubness loss is given by

$$L_h = \pi L'_h. \quad (11)$$

Unsupervised unbiasing loss: The model has observed many labeled instances during inductive training, and seen class semantics align perfectly with 3D features. As a result, distances between seen semantics and features remain close during transductive learning, but unseen semantics and features reside far apart. This biases the model towards seen classes, confusing unseen instances as seen. Similar to previous work [53], we adopt an unsupervised unbiasing loss to minimize this effect.

Given unlabeled examples of the seen and unseen classes, this loss aids our model by increasing unseen probabilities calculated from distances. The model gradually learns to pull unseen semantics close to unseen instances.

$$L_u = -\frac{1}{N} \sum_{i=1}^N \log \sum_{j \in \mathcal{Y}^u} \frac{e^{-\|\varphi(\mathcal{X}_i^c) - \Theta_2(\mathbf{e}_j; W_2)\|_2^2}}{\sum_{k \in \mathcal{Y}^s \cup \mathcal{Y}^u} e^{-\|\varphi(\mathcal{X}_i^c) - \Theta_2(\mathbf{e}_k; W_2)\|_2^2}}. \quad (12)$$

This loss balances the average distances between semantic vectors and data features of both seen and unseen classes. Consequently, this process helps the model have less bias towards seen classes, resulting in better accuracy on unseen classes.

Overall transductive loss: The overall loss is given by the sum of the unsupervised triplet, hubness and unbiasing losses as follows:

$$L_T = \alpha_1 L_t + \alpha_2 L_h + \alpha_3 L_u \quad (13)$$

where hyper-parameters α_1 , α_2 , and α_3 control the importance of L_t , L_h , and L_u respectively.

4 Training

The proposed model architecture is shown in Figure 3, consisting of two branches: the point cloud network that extracts a feature vector $\varphi(\mathcal{X}) \in \mathbb{R}^m$ from a point cloud \mathcal{X} , and the semantic projection network that projects a semantic feature vector $\mathbf{e} \in \mathbb{R}^d$ into point cloud feature space. Any network that learns a feature space from 3D point sets and is invariant to permutations of points in the point cloud can be used in our method as the point cloud network [38, 39, 61, 28, 69, 60, 68]. The projection network $\Theta_2(\cdot)$ with trainable weights W consists of two fully-connected layers, with 512 and 1024 dimensions respectively, each followed by a tanh nonlinearity.

In contrast, transductive ZSL additionally uses the set of unlabeled, unseen instances $\{\mathcal{X}_i^u\}$ and the set of unseen semantic embedding vectors \mathcal{E}^u during training. To learn a transductive model in a semi-supervised manner, an objective function

$$L = L_{S2F} + L_T \quad (14)$$

is minimized.

We describe the overall training process in Algorithm 1. In the proposed algorithm, in the first stage, an inductive model W_{ind} is learned. Then the transductive model W_{tns} is initialized with the inductive model. Finally the transductive model is learned.

4.1 Inference

For the zero-shot learning task, given the learned optimal weights W_2 from training with labeled seen instances \mathcal{X}^s and unlabeled unseen instances \mathcal{X}^u , the label of the input point cloud \mathcal{X}^u is predicted as

$$\hat{y} = \arg \min_{y \in \mathcal{Y}^u} \|\varphi(\mathcal{X}^u) - \Theta_2(\phi(y); W_2)\|_2. \quad (15)$$

For the generalized zero-shot learning task, the label of the input point cloud \mathcal{X}^c for $c \in \{s, u\}$ is predicted as

$$\hat{y} = \arg \min_{y \in \mathcal{Y}^s \cup \mathcal{Y}^u} \|\varphi(\mathcal{X}^c) - \Theta_2(\phi(y); W_2)\|_2. \quad (16)$$

Algorithm 1 Transductive ZSL for 3D point cloud objects

Input: $\mathcal{X}^s, \mathcal{Y}^s, \mathcal{E}^s, n_s, \mathcal{X}^u, \mathcal{E}^u, n_u$
Output: A trained model W_{tns} to find \hat{y} for all \mathcal{X}^u

Inductive training stage

- 1: $W_{ind} \leftarrow$ train an inductive model using Eq 2 with only seen data: $\mathcal{X}^s, \mathcal{Y}^s, \mathcal{E}^s, n_s$

Transductive training stage

- 2: $W_{tns} \leftarrow W_{ind}$, initialize transductive model
- 3: **repeat**
- 4: **if** GZSL **then**
- 5: $\hat{y} \leftarrow$ use W_{tns} to assign positive and negative anchors to \mathcal{X}^u using Eq 4 and Eq 5 for triple formation
- 6: **else**
- 7: $\hat{y} \leftarrow$ use W_{tns} to assign positive and negative anchors to \mathcal{X}^u using Eq 3 and Eq 5 for triple formation
- 8: **for** $\forall I \in \mathcal{X}^s \cup \mathcal{X}^u$ **do**
- 9: Calculate triplet loss, L_t using Eq 6
- 10: Calculate hubness loss, L_h using Eq 9
- 11: Calculate unbiased loss, L_u using Eq 12
- 12: Calculate overall transductive loss, L_T using Eq 14
- 13: Backpropagate and update W_{tns}
- 14: **until** convergence

Return Class decision \hat{y} with W_{tns} using Eq 15 for ZSL or Eq 16 for GZSL

| | Dataset | Total classes | Seen/ Unseen | Train/ Valid/Test |
|-------------------|-------------------|---------------|-----------------|----------------------|
| 3D synt- hetic | ModelNet40 [63] | 40 | 30/- | 5852/1560/- |
| | ModelNet10 [63] | 10 | -/10 | -/-/908 |
| | McGill [52] | 19 | -/14 | -/-/115 |
| 3D real | ModelNet40 [63] | 40 | 26/- | 4999/1496/- |
| | ScanObjectNN [56] | 15 | /11 | -/-/495 |
| 2D | AwA2 SS [65] | 50 | 40/10 | 30337/-/6985 |
| | AwA2 PS [65] | 50 | 40/10 | 23527/5882/7913 |
| | CUB SS [59] | 200 | 150/50 | 8855/-/2933 |
| | CUB PS [59] | 200 | 150/50 | 7057/1764/2967 |

Table 1: Statistics of the 3D and 2D datasets. The total number of classes in the datasets are reported, alongside the actual splits used in this paper dividing the classes into seen or unseen and the elements into those used for training or testing. The 3D synthetic splits are from [10] and the 2D Standard Splits (SS) and Proposed Splits (PS) are from Xian *et al.* [65]. The 3D real split is newly proposed in this paper.

5 Experiments

5.1 Setup

Datasets: We evaluate our approach on four well-known 3D datasets, ModelNet10 [63], ModelNet40 [63], McGill [52], and ScanObjectNN [56], and two 2D datasets, AwA2 [65] and CUB [59]. The dataset statistics as used in this work are given in Table 1. We experiment on three different seen/unseen split settings. **(1)** For experiments with synthetic datasets (ModelNet10, ModelNet40, and

McGill), we follow the seen/unseen splits proposed by Cheraghian et al. [10], where the seen classes are those 30 in ModelNet40 that do not occur in ModelNet10, and the unseen classes are those from the test sets of ModelNet10 and McGill that are not in the set of seen classes. These splits allow us to test unseen classes from different distributions than that of the seen classes. **(2)** For experiments with the real 3D dataset (ScanObjectNN), we propose a new train-test setting. Unlike synthetic (CAD) modes of ModelNet40, ScanObjectNN contains real-world scanned objects. We train our method using 26 non-overlapped classes between ModelNet40 and ScanObjectNN as seen and test with 11 overlapped classes. This setup uses only ModelNet40 instances during training and ScanObjectNN instances during testing. This is a more realistic setup because we can get many synthetic examples of seen objects during training. However, the model may encounter many real-world 3D data instances of both seen and unseen classes at test time. **(3)** For the 2D datasets, we follow the Standard Splits (SS) and Proposed Splits (PS) of Xian et al. [65].

Semantic features: We use the 300-dimensional semantic feature vectors of word2vec [35] for the 3D dataset experiments, the 85-dimensional attribute vectors from Xian et al. [65] for the AwA2 experiments, and the 312-dimensional attribute vectors from Wah et al. [59] for the CUB experiments. Figure 5 visualizes word vectors of 3D datasets.

Evaluation: We report the top-1 accuracy as a measure of recognition performance, where the predicted label (the class with minimum distance from the test sample) must match the ground-truth label to be considered a successful prediction. For generalized ZSL, we also report the Harmonic Mean (HM) [65] of the accuracy of the seen and unseen classes, computed as

$$HM = \frac{2 \times Acc_s \times Acc_u}{Acc_s + Acc_u} \quad (17)$$

where Acc_s and Acc_u are seen and unseen class top-1 accuracies respectively. The harmonic mean is able to distinguish between methods that are biased towards seen classes and those that produce good results for both seen and unseen classes.

Cross-validation: We used cross-validation to find the best hyper-parameters, averaging over 10 repetitions. For ModelNet10 and McGill, 5 of the 30 seen classes were randomly selected as an unseen validation set, while 4 of the 26 seen classes were chosen randomly for the ScanObjectNN. Additionally, 20% of the seen classes were used as an unseen validation set for the AwA2 and CUB datasets. To find hyperparameters, we conducted a grid search within the range α_1, α_2 ,

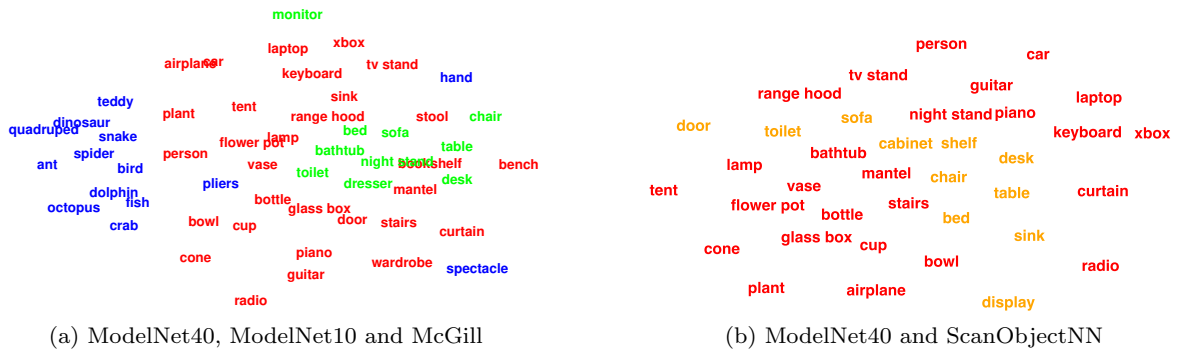


Fig. 5: 2D tSNE [57] visualization of word2vec vectors [34]. Red, green, blue and orange texts represent seen ModelNet40 [63], unseen ModelNet10 [63], unseen McGill [52] and unseen ScanObjectNN [56] classes respectively.

$\alpha_3 \in [0, 1]$. The selected hyper-parameters α_1 , α_2 , and α_3 were 0.4, 0.001, 0.001 for both ModelNet10 and McGill, 0.2, 0.2, 0.1 for ScanObjectNN, 0.12, 0.001, 0.01 for AWA, and 0.1, 0.001, 0.001 for CUB.

Implementation details¹: For the 3D data experiments, we used PointNet [38], DGCNN[61], PointConv [62], and PointAugment [30] as the point cloud feature extraction network. For synthetic 3D data, these networks were pre-trained on the 30 seen classes of ModelNet40. Also, for real 3D data, these networks were pre-trained on the 26 seen classes of ModelNet40. For the 2D data experiments, we used a 101-layered ResNet architecture [18], where the 2048-dimensional input feature embedding was obtained from the top-layer pooling unit. The network was pre-trained on ImageNet 1K [11]. For semantic projection layers, we used two fully connected (512,1024) with relu non-linearities for 3D experiments, and two fully connected (1024,2048) with relu non-linearities for 2D experiments. These parameters are fully-learnable. To train the network, we used the Adam optimizer [23] with an initial learning rate of 0.0001 for all experiments. We implemented the architecture using Pytorch and trained and tested it on a NVIDIA GTX Titan V GPU.

Compared approaches: We enlist different versions (baselines and our recommendation) of the proposed method below:

- *Baseline-I*: Our inductive baseline while projecting point cloud feature to semantic embedding space (F2S) using Eq. 1.
- *Ours (Inductive)*: Our recommended inductive approach while projecting semantic embedding space to point cloud feature (S2F) using Eq. 2

¹ Code and data are available at: https://github.com/ali-chr/Transductive_ZSL_3D_Point_Cloud

| Backbone | All-40 | Seen-30 | Seen-26 |
|-------------------|-------------|-------------|-------------|
| PointNet [38] | 89.2 | 85.7 | 87.1 |
| PointAugment [30] | 90.9 | 88.3 | 89.5 |
| DGCNN [61] | 92.2 | 91.2 | 92.5 |
| PointConv [62] | 92.2 | 92.6 | 93.1 |

Table 2: Results on seen classes of ModelNet40 for different feature extractor backbones.

- *Baseline-T*: Our transductive baseline using only triplet loss of Eq. 6 as transductive loss (without hubness and unbiased loss part), i.e., $L_T = L_t$.
- *Ours (Transductive)*: Our recommended transductive approach while using Eq. 14.

5.2 Comparing point cloud feature extractors

We evaluate four 3D point cloud recognition frameworks, namely, PointNet [38], DGCNN [61], PointConv [62], and PointAugment [30] as backbone to extract 3D point cloud features. In Figure 6, we visualize point cloud features for unseen ModelNet10 and ScanObjectNN instances using tSNE. We perform the inductive training (S2F) of all those frameworks on 30 and 26 seen classes of the synthetic ModelNet40 dataset. In Table 2, we report the performance of test seen classes. The values with 40 classes (All-40) of ModelNet40 are from the original published papers. The columns for Seen-30 and Seen-26 report the performance of 30 and 26 seen classes during training with synthetic and real-world scanned 3D datasets, respectively. We notice similar performance for All-40, Seen-30, and Seen-26 experiment setups, which tells that the backbone is well-trained for feature extraction.

In addition to test seen class performance, in Table 3 we show ZSL and GZSL results of the same inductive training using test samples from seen and un-

| Backbone | Method | ModelNet10 | | | | ScanObjectNN | | | |
|--------------|---------------------|--------------|------------------|------------------|--------------|--------------|------------------|------------------|--------------|
| | | ZSL | | GZSL | | ZSL | | GZSL | |
| | | Acc | Acc _s | Acc _u | HM | Acc | Acc _s | Acc _u | HM |
| PointNet | Ours (Inductive) | 21.26 | 79.37 | 3.74 | 7.15 | 18.95 | 75.13 | 3.58 | 6.83 |
| | Ours (Transductive) | 18.28 | 71.79 | 16.08 | 26.27 | 18.11 | 75.27 | 6.11 | 11.29 |
| PointAugment | Ours (Inductive) | 21.37 | 71.92 | 6.39 | 11.73 | 16.42 | 52.14 | 2.74 | 5.20 |
| | Ours (Transductive) | 23.68 | 66.86 | 12.67 | 21.30 | 18.95 | 40.64 | 14.32 | 21.17 |
| DGCNN | Ours (Inductive) | 38.33 | 69.87 | 8.26 | 14.77 | 22.95 | 79.28 | 1.89 | 3.70 |
| | Ours (Transductive) | 60.05 | 78.71 | 45.26 | 57.47 | 25.68 | 53.54 | 9.89 | 16.70 |
| PointConv | Ours (Inductive) | 32.49 | 89.42 | 6.83 | 12.69 | 21.89 | 89.37 | 5.68 | 10.69 |
| | Ours (Transductive) | 68.50 | 83.21 | 65.64 | 73.39 | 30.53 | 90.31 | 30.53 | 45.63 |

Table 3: Performance of our method using different backbones.

| Method (PointConv) | ModelNet10 | | | | McGill | | | | ScanObjectNN | | | |
|---------------------|--------------|------------------|------------------|--------------|--------------|------------------|------------------|--------------|--------------|------------------|------------------|--------------|
| | ZSL | | GZSL | | ZSL | | GZSL | | ZSL | | GZSL | |
| | Acc | Acc _s | Acc _u | HM | Acc | Acc _s | Acc _u | HM | Acc | Acc _s | Acc _u | HM |
| DEM [75] | 17.48 | 88.57 | 5.30 | 9.99 | 7.12 | 75.95 | 7.14 | 13.06 | 10.71 | 88.76 | 10.71 | 19.12 |
| LATEM [64] | 26.29 | - | - | - | 7.15 | - | - | - | 11.88 | - | - | - |
| SYNC [4] | 21.17 | - | - | - | 7.14 | - | - | - | 17.43 | - | - | - |
| GDAN [20] | - | 86.57 | 4.06 | 7.76 | - | 86.97 | 7.14 | 13.20 | - | 88.34 | 19.07 | 31.37 |
| I TF-VAEGAN [36] | 27.21 | 59.23 | 19.65 | 29.51 | 20.65 | 84.63 | 20.65 | 33.19 | 28.20 | 81.22 | 23.99 | 37.04 |
| f-CLSWGAN [66] | 13.73 | 67.13 | 15.57 | 25.27 | 17.21 | 85.13 | 13.57 | 23.41 | 18.35 | 85.60 | 11.61 | 20.44 |
| CADA-VAE [49] | 15.58 | 89.1 | 2.93 | 5.67 | 7.14 | 89.27 | 7.14 | 13.23 | 16.47 | 89.61 | 14.11 | 24.38 |
| Baseline-I | 24.45 | 27.12 | 8.81 | 13.30 | 13.04 | 62.69 | 0.00 | 0.00 | 21.68 | 37.10 | 1.05 | 2.05 |
| Ours (Inductive) | 32.49 | 89.42 | 6.83 | 12.69 | 13.91 | 90.51 | 13.91 | 14.39 | 24.12 | 89.37 | 5.68 | 10.69 |
| QFSL [53] | 38.80 | 58.10 | 21.80 | 31.70 | 9.56 | 86.08 | 9.56 | 17.21 | 18.71 | 81.88 | 18.53 | 30.21 |
| T Baseline-T | 43.17 | 85.58 | 42.96 | 57.20 | 5.22 | 90.71 | 5.22 | 9.87 | 25.05 | 88.50 | 25.05 | 39.05 |
| Ours (Transductive) | 68.50 | 83.21 | 65.64 | 73.39 | 15.71 | 71.08 | 8.69 | 15.49 | 30.53 | 90.31 | 30.53 | 45.63 |

Table 4: ZSL and GZSL results on the 3D ModelNet10 [63], McGill [52], and ScanObjectNN [56] datasets for PointConv [62]. We report the top-1 accuracy (%) on seen classes (Acc_s) and unseen classes (Acc_u) for each method, as well as the harmonic mean (HM) of both measures. ‘‘I’’ and ‘‘T’’ denote inductive and transductive learning respectively.

seen classes from both synthetic and real 3D datasets. From Table 2 and 3, we notice that DGCNN and PointConv point cloud backbone performs consistently better than other. The reason is that DGCNN and PointConv analyze the local and global information of point cloud data, while PointNet and PointAugment consider solely global information. We choose the best performing backbone, PointConv, for the remaining experiments in this paper.

5.3 3D point cloud experiments

For the experiments on 3D data, we compare different (baselines and recommended) versions of our method with eight 2D ZSL methods, DEM [75], SYNC [4], LATEM [64], GDAN [20], TF-VAEGAN [36], f-CLSWGAN [66], CADA-VAE [49], and QFSL [53] in Table 4. These state-of-the-art image-based methods were re-implemented and adapted to point cloud data to facilitate comparison. Our method significantly outperforms the other approaches on the ModelNet10 and ScanObjectNN datasets. Several observations can be made from the re-

sults. **(1)** Methods usually work better on the 3D synthetic dataset (ModelNet10) than real data (ScanObjectNN). This is likely due to domain shift from synthetic to real data and the presence of noise in real data. (see Figure 6). However, methods do not perform as well on the McGill dataset when compared to the ModelNet10 results, because the distributions of semantic feature vectors in the unseen McGill datasets are significantly different from the distribution in the seen ModelNet40 dataset, much more so than that of ModelNet10 (see Figure 5). **(2)** 2D ZSL methods can perform 3D ZSL using 3D features as input instead of 2D images. Generative methods (TF-VAEGAN, CADA-VAE) perform better than non-generative methods (DEM, SYNC) because generative models use unseen semantics during training to create fake features. **(3)** Transductive learning is much more effective than inductive learning for point cloud ZSL. This is likely due to inductive approaches being more biased towards seen classes, while transductive approaches alleviate the bias problem by using unlabeled, unseen instances during training. **(4)** Our proposed method performs better than

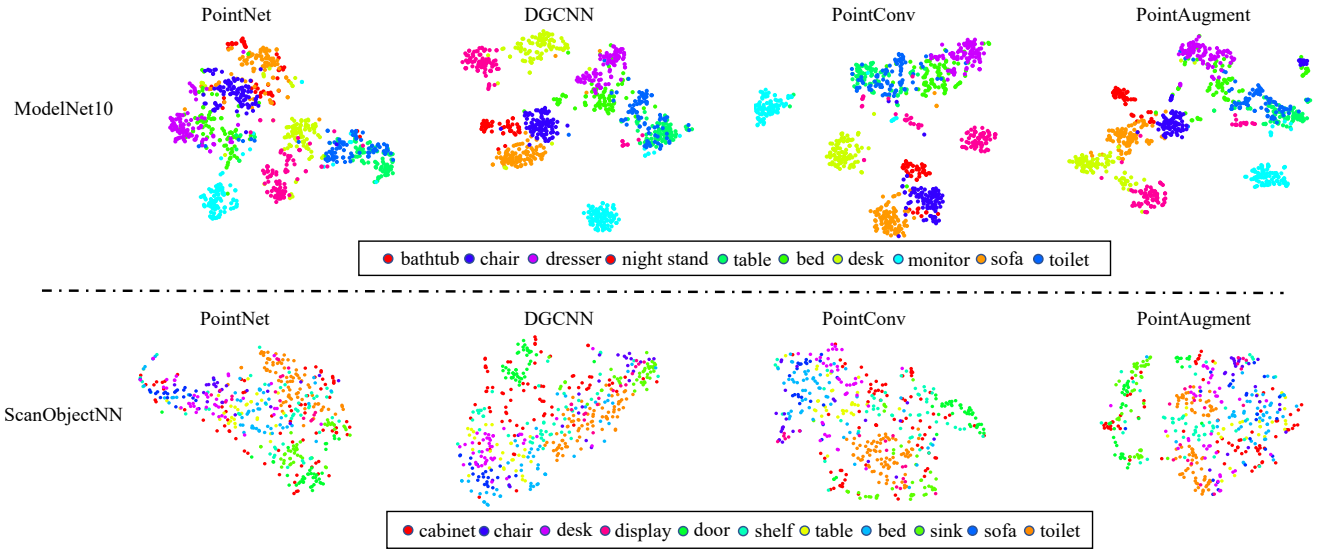


Fig. 6: 2D tSNE [57] visualization of unseen point cloud feature vectors (circles) with four backbone networks. Since the synthetic data has less noise, ModelNet40 features are clustered better than the real scanned 3D data (with noise) from ScanObjectNN. Moreover, for both datasets, the models are trained on synthetic instances belonging to a subset of ModelNet40 classes, and so we expect the ModelNet10 features to be better clustered than the ScanObjectNN features. We obtained the best overall performance using the PointConv backbone.

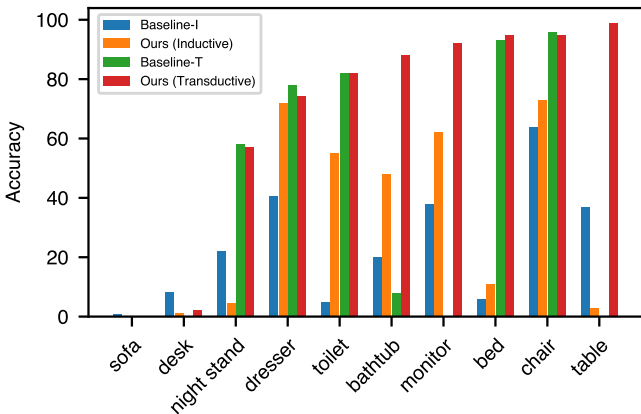


Fig. 7: ZSL per-class accuracy for ModelNet10 using PointConv backbone.

QFSL, which is likely due to our triplet loss formulation. While noisy, the positive and negative samples of unlabeled data provide useful supervision, unlike the unsupervised approach for only unlabeled data in QFSL. (5) There is a performance improvement from Baseline-T to Ours (Transductive) due to the use of the hubness and unbiasing losses, which mostly contribute to improving ZSL and GZSL performances, respectively. (6) Our method could not achieve the best performance on McGill because of fewer test instances (more specifically, only 115 instances) available for this dataset (see

Table 1). However, methods like TF-VAEGAN [36] and f-CLSWGAN [66]) are relatively successful because of generating pseudo-features with generative models, which balances the number of unseen instances similar to seen class instances. (7) Generalized ZSL, which is more realistic than standard ZSL, is more challenging than ZSL as there are both seen and unseen classes during inference. Our (Transductive) method obtained the best performance with respect to the harmonic mean (HM) on all datasets (not on McGill), and the best performance with respect to the unseen class accuracy Acc_u on most datasets, which demonstrates the utility of our method for GZSL as well as ZSL for 3D point cloud recognition.

Per-class results: We also show, in Figure 7, the performance of individual classes from ModelNet10. Baseline-I performs relatively well (above 30%) on only four classes (dresser, monitor, chair and table) of ModelNet40. Because of the hubness problem, Baseline-I mostly predicts those few classes regardless of the input. Our inductive and Baseline-T methods minimize this problem by confidently predicting more (five) classes than Baseline-I. Our final transductive method achieves the best accuracy in eight classes and outperforms its alternatives. This is likely due to minimizing the hubness and bias problem in transductive settings.

| Backbone | F2S | S2F | Triplet | Hubness | Unbiasing | ModelNet10 | | ScanObjectNN | |
|-----------|-----|-----|---------|---------|-----------|--------------|--------------|--------------|--------------|
| | | | | | | ZSL | GZSL (HM) | ZSL | GZSL (HM) |
| PointConv | ✓ | ✗ | ✗ | ✗ | ✗ | 24.45 | 13.30 | 21.68 | 2.05 |
| | ✗ | ✓ | ✗ | ✗ | ✗ | 32.49 | 12.69 | 21.89 | 10.69 |
| | ✗ | ✓ | ✓ | ✗ | ✗ | 43.17 | 34.83 | 25.05 | 20.65 |
| | ✗ | ✓ | ✓ | ✓ | ✗ | 61.45 | 57.33 | 30.74 | 33.59 |
| | ✗ | ✓ | ✓ | ✓ | ✓ | 68.50 | 73.39 | 30.53 | 45.63 |

Table 5: Ablation studies. Effect of adding different loss components incrementally.

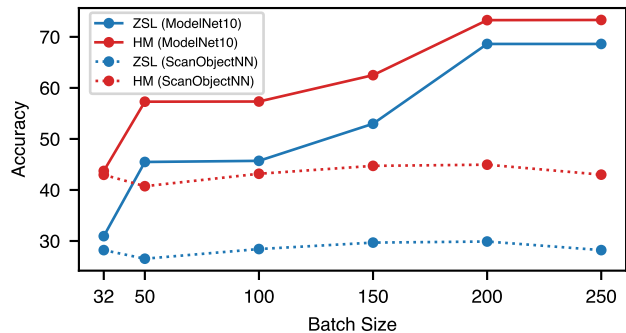


Fig. 8: Ours (Transductive) performance when varying batch sizes for ModelNet10 and ScanObjectNN.

| | | ModelNet10 | | ScanObjectNN | |
|------|------------------|------------|--------------------|--------------|--------------------|
| | | $\pi = 1$ | π as in Eq. 10 | $\pi = 1$ | π as in Eq. 10 |
| ZSL | Acc | 46.37 | 68.50 | 19.58 | 30.53 |
| | Acc _s | 80.06 | 83.21 | 76.87 | 90.31 |
| GZSL | Acc _u | 44.27 | 65.64 | 17.26 | 30.53 |
| | HM | 57.02 | 73.39 | 28.19 | 45.63 |

Table 6: The effect of weighting factor, π in Eq. 10 on Ours (Transductive) performance.

5.4 Ablation Studies

Impact of loss components: We ablate our proposed method with respect to the different loss components. The elements of the combined loss function incrementally bring robustness to our approach. Table 5 reports the ablation results with PointConv backbone. Our method performs poorly with only the F2S part (Eq. 1), largely because of the hubness problem. Replacing F2S with S2F (Eq. 2) improves performance since this mitigates the hubness problem for inductive learning. We perform transductive training based on unsupervised triplet loss (Eq. 6) on top of the S2F-based inductive weights. The utilization of unlabeled data raises the performance from inductive to transductive settings. Next, we add the hubness loss (Eq. 11) to minimize the hubness problem further in transductive settings. Finally, we include the unbiasing loss (Eq. 12) to balance seen and unseen class distances.

It mostly helps to achieve robust GZSL performance because GZSL considers both seen and unseen classes together.

Impact of batch size: Our proposed transductive loss has a noticeable impact on the batch size. With a larger batch size, the mistakes of pseudo-labeling while calculating the triplet loss (Eq. 6) become stabilized. Moreover, it also increases the chance of evenly distributing different class instances, which estimates the hubness loss of Eq. 11 better than small-batch cases. In Figure 8, we report ZSL and GZSL (HM) performance on ModelNet10 and ScanObjectNN using different batch sizes. As expected, increasing batch size improves the performance.

Impact of weighting factor of hubness loss: In Eq. 10, we design a weight factor for the hubness loss to penalize highly confident predictions less than low scores. In Table 6, we show the impact of using the weighting factor. We notice that the use of the weighting factor significantly improves the performance.

5.5 2D Image Experiments

While our method is designed to address ZSL and GZSL tasks for 3D point cloud recognition, we also adapt and evaluate our method for the case of 2D image recognition. The results for ZSL and GZSL are shown in Table 7. Our proposed method is evaluated on the AwA2 [65] and CUB [59] datasets using the SS and PS splits [65]. We achieve very competitive results on these datasets, indicating that the method can generalize to image data. Although we outperform many state-of-the-art methods in transductive ZSL settings, our results lag state-of-the-art in the GZSL problem. Fine-tuning the feature extraction network [53] or pseudo-features from generative models [66, 49, 67, 36] may go some way to closing this gap.

Another observation, the 2D image experiment results are better in general than 3D point cloud experiments in Table 4. The possible reasons could be the availability of large-scale 2D datasets, pre-trained models, and more accurate pseudo/fake features from gen-

| Method | AwA2 | | | | | CUB | | | | |
|------------------------|-------------|-------------|------------------|------------------|-------------|-------------|-------------|------------------|------------------|-------------|
| | ZSL | | GZSL | | | ZSL | | GZSL | | |
| | SS | PS | Acc _s | Acc _u | HM | SS | PS | Acc _s | Acc _u | HM |
| I SJE [2] | 69.5 | 61.9 | - | - | - | 55.3 | 53.9 | - | - | - |
| DEM[75] | - | 67.1 | 30.5 | 86.4 | 45.1 | 58.3 | 51.7 | 19.6 | 57.9 | 29.2 |
| CS[5] | - | 77.6 | 45.3 | 57.2 | - | 49.4 | 48.1 | 48.7 | - | - |
| TCN[22] | - | 71.2 | 65.8 | 61.2 | 63.4 | - | 59.5 | 52.0 | 52.6 | 52.3 |
| GDAN[20] | - | - | 67.5 | 32.1 | 43.5 | - | - | 75.0 | 30.4 | 43.4 |
| TF-VAEGAN[36] | - | 72.2 | 75.1 | 59.8 | 66.6 | - | 64.9 | 64.7 | 52.8 | 58.1 |
| TF-VAEGAN*[36] | - | 73.4 | 83.6 | 55.5 | 66.7 | - | 74.3 | 79.3 | 63.8 | 70.7 |
| f-CLSWGAN [66] | - | - | - | - | - | - | 57.3 | 43.7 | 57.7 | 49.7 |
| CADA-VAE [49] | - | - | 75.0 | 55.8 | 63.9 | - | - | 53.5 | 51.6 | 52.6 |
| f-VAEGAN-D2 [67] | - | 71.1 | 57.6 | 70.6 | 63.5 | - | 72.9 | 48.4 | 60.1 | 53.6 |
| f-VAEGAN-D2* [67] | - | 70.3 | 57.1 | 76.1 | 65.2 | - | 72.9 | 63.2 | 75.6 | 68.9 |
| Ours (Inductive) | 71.2 | 69.0 | 88.9 | 22.1 | 35.4 | 59.3 | 54.2 | 69.4 | 8.4 | 14.9 |
| T DIPL[76] | - | - | - | - | - | 68.2 | 65.4 | 44.8 | 41.7 | 43.2 |
| PREN [70] | - | 74.1 | 88.6 | 32.4 | 47.4 | - | 66.4 | 55.8 | 35.2 | 43.1 |
| EDE _{ex} [73] | - | 77.5 | 93.2 | 68.4 | 78.9 | - | 67.8 | 62.9 | 54.0 | 58.1 |
| QFSL*[53] | 84.8 | 79.7 | 93.1 | 66.2 | 77.4 | 69.7 | 72.1 | 74.9 | 71.5 | 73.2 |
| GMN [48] | - | - | - | - | - | - | 64.6 | 70.6 | 60.2 | 65.0 |
| T f-VAEGAN-D2 [67] | - | 89.8 | 84.8 | 88.6 | 86.7 | - | 71.1 | 61.4 | 65.4 | 63.2 |
| f-VAEGAN-D2* [67] | - | 89.3 | 86.3 | 88.7 | 87.5 | - | 82.6 | 73.8 | 81.4 | 77.3 |
| TF-VAEGAN[36] | - | 92.1 | 89.6 | 87.3 | 88.4 | - | 74.7 | 72.1 | 69.9 | 71.0 |
| TF-VAEGAN*[36] | - | 93.0 | 90.0 | 89.2 | 89.6 | - | 85.1 | 83.5 | 78.4 | 80.9 |
| Baseline-T | 83.3 | 75.6 | 88.0 | 67.2 | 76.2 | 70.6 | 58.3 | 51.4 | 40.2 | 45.1 |
| Ours (Transductive) | 91.2 | 90.2 | 84.7 | 81.9 | 83.3 | 72.0 | 71.5 | 60.2 | 58.7 | 59.8 |

Table 7: ZSL results on the Standard Splits (SS) and Proposed Splits (PS) and GZSL results on the 2D AwA2 and CUB datasets. We report the top-1 accuracy (%) on seen classes (Acc_s) and unseen classes (Acc_u) for each method, as well as the harmonic mean (HM) of both measures. “I” and “T” denote inductive and transductive learning respectively. *Image feature extraction model fine-tuned (we do not fine-tune our model).

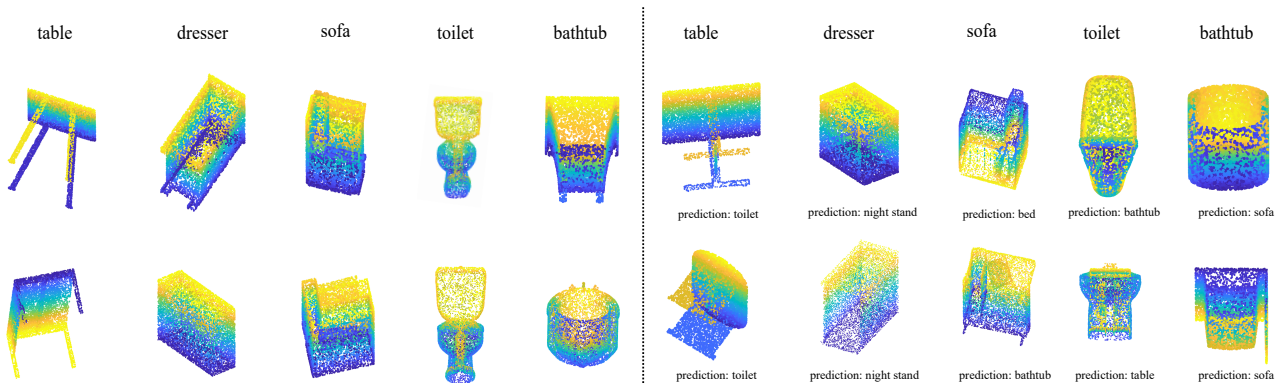


Fig. 9: Visualization of five classes from the ModelNet10 dataset with examples of (left) correctly and (right) incorrectly classified point clouds, respectively. The predicted classes are shown below each model for the incorrect cases.

erative models compared to that of 3D point cloud objects.

5.6 Qualitative Evaluation

We visualize five unseen classes from the ModelNet10 dataset with examples where our method correctly classified the point cloud, shown in Figure 9 (left), and ex-

amples where it incorrectly classified the point cloud, shown in Figure 9 (right). The network appears to be providing incorrect predictions for mostly hard examples, those that are quite different from standard examples in that class, or where the classes overlap in their geometry, such as dresser and night stand.

5.7 Discussion

Challenges with 3D data: Recent deep learning methods for classifying point cloud objects have achieved over 90% accuracy on several standard datasets, including ModelNet40 and ModelNet10. Moreover, due to significant progress in depth camera technology [6, 21], it is now possible to capture 3D point cloud objects at scale much more easily. It is therefore likely that many classes of 3D objects will not be present in the labeled training set. As a result, zero-shot classification systems will be needed to leverage other more easily-obtainable sources of information in order to classify unseen objects. However, we observe that the difference in accuracy between ZSL and supervised learning is still very large for 3D point cloud classification, e.g., 68.5% as compared to 95.7% [28] for ModelNet10. As such, there is significant potential for improvement for zero-shot 3D point cloud classification. While the performance is still quite low, this is also the case for 2D ZSL, with state-of-the-art being 31.1% top-5 accuracy on the ImageNet2010/12 [47] datasets, reflecting the challenging nature of the problem.

Visual features versus point cloud features: Moving from 2D visual features to 3D point cloud features for ZSL brings new challenges. Many deep learning models on 2D images rely on pre-trained deep features, which are obtained by considering thousands of classes and millions of images [65]. In contrast, 3D point cloud datasets of a similar scale are not yet available. Therefore, 3D point cloud features are less robust than image features. To illustrate this point, we visualize 6985 instances of 10 classes from the 2D dataset AwA2 [65] and 908 instances of 10 classes from the 3D dataset ModelNet10 [63] in Figure 1. It is apparent that although a larger number of instances were used in the 2D case, the cluster structure is more separable in 2D than in 3D. As 3D features are not as robust and separable as 2D features, relating those features to their corresponding semantic vectors is more difficult in 3D than 2D.

Hubness: Our approach, projecting semantic vectors to input feature (S2F) space, since it has been shown that this alleviates the hubness problem [51, 74], we validate this claim by measuring the skewness of the distribution N_k [51, 41] when projected in each direction, and the associated accuracy. We report these values in Table 8 for the ModelNet10 datasets. The degree of skewness is much lower when projecting the semantic feature space to the point cloud feature space, and achieves a significantly higher accuracy. This provides additional evidence that this projection direction is preferable for mitigating the problem of hubs and the consequent bias. In addition to 30 and 26 seen class

| # of seen | F2S (Inductive) | S2F (Inductive) | S2F (Transductive) |
|-----------|--------------------|--------------------|-----------------------|
| 20 | 1.43(9.91%) | 1.08(25.77%) | -0.06(27.42%) |
| 30 | 0.84(24.45%) | 0.76(32.49%) | -0.79(68.50%) |

Table 8: The skewness (and accuracy) on ModelNet10 with different projection directions in both inductive and transductive settings. The skewness is lower when projecting the semantic space to the input point cloud feature space, mitigating the hubness problem and leading to more accurate transductive ZSL.

settings while training with ModelNet40, we also train our proposed method using randomly selected less (20) number of seen classes, resulting in fewer data. We notice that performances decrease, but skewness scores increase while training with fewer data. It tells the overall impact of data scarcity which directly controls the generalization ability of seen features, contributes to the hubness problem and overall performance.

6 Conclusion

With the aid of better 3D capture systems, obtaining 3D point cloud data of objects at a very large scale has become more feasible than before. However, 3D point cloud recognition systems have not scaled up to handle this large scale scenario. We apply zero-shot learning approaches to facilitate the classification of previously unseen input to readjust such a system with newly available data that have not been observed during training. We identified and addressed issues that arise in the inductive and transductive settings of zero-shot learning and its generalized variant when applied to the domain of 3D point cloud classification. We observed that in the 2D domain, the embedding quality generated by the pre-trained feature space is of a significantly higher quality than that produced by its 3D counterpart due to the vast difference in the amount of labeled training data they have been exposed to. Moreover, like ZSL on 2D images, we notice that such classification of 3D point clouds suffers from the hubness problem. The hubness problem in 3D is more severe than that observed in the 2D case. One possible reason could be that the 3D features are not trained on millions of 3D instances in the same way that 2D convolutional networks can be. In this paper, we attempt to reduce the effect of the hubness problem while performing ZSL on 3D point cloud objects by proposing an unsupervised skewness loss. In addition, we report results on Generalized ZSL in conjunction with ZSL. Furthermore, we develop a novel triplet loss that makes use of unlabeled test data in a transductive setting. The utility of this method is

demonstrated via an extensive set of experiments that showed significant benefit in the 2D domain and established state-of-the-art results in the 3D domain (both real and synthetic data) for ZSL and GZSL tasks.

References

1. Akata, Z., Perronnin, F., Harchaoui, Z., Schmid, C.: Label-Embedding for Image Classification. *IEEE TPAMI* **38**(7), 1425–1438 (2016). DOI 10.1109/TPAMI.2015.2487986
2. Akata, Z., Reed, S., Walter, D., Lee, H., Schiele, B.: Evaluation of output embeddings for fine-grained image classification. In: *CVPR*, vol. 07-12-June-2015, pp. 2927–2936 (2015). DOI 10.1109/CVPR.2015.7298911
3. Changpinyo, S., Chao, W.L., Gong, B., Sha, F.: Synthesized classifiers for zero-shot learning. In: *CVPR*, vol. 2016-January, pp. 5327–5336 (2016)
4. Changpinyo, S., Chao, W.L., Gong, B., Sha, F.: Synthesized classifiers for zero-shot learning. In: *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)* (2016)
5. Chao, W.L., Changpinyo Soravitand Gong, B., Sha, F.: An Empirical Study and Analysis of Generalized Zero-Shot Learning for Object Recognition in the Wild, pp. 52–68. Springer International Publishing, Cham (2016)
6. Chen, C., Yang, B., Song, S., Tian, M., Li, J., Dai, W., Fang, L.: Calibrate multiple consumer rgb-d cameras for low-cost and efficient 3d indoor mapping. *Remote Sensing* **10**(2) (2018). DOI 10.3390/rs10020328. URL <http://www.mdpi.com/2072-4292/10/2/328>
7. Cheraghian, A., Petersson, L.: 3dcapsule: Extending the capsule architecture to classify 3d point clouds. In: *2019 IEEE Winter Conference on Applications of Computer Vision (WACV)*, pp. 1194–1202 (2019). DOI 10.1109/WACV.2019.00132
8. Cheraghian, A., Rahman, S., Campbell, D., Petersson, L.: Mitigating the hubness problem for zero-shot learning of 3d objects. In: *British Machine Vision Conference (BMVC'19)* (2019)
9. Cheraghian, A., Rahman, S., Campbell, D., Petersson, L.: Transductive zero-shot learning for 3d point cloud classification. In: *2020 IEEE Winter Conference on Applications of Computer Vision (WACV)*, pp. 912–922 (2020)
10. Cheraghian, A., Rahman, S., Petersson, L.: Zero-shot learning of 3d point cloud objects. In: *International Conference on Machine Vision Applications (MVA)* (2019)
11. Deng, J., Dong, W., Socher, R., Li, L.J., Li, K., Fei-Fei, L.: ImageNet: A Large-Scale Hierarchical Image Database. In: *CVPR09* (2009)
12. Dinu, G., Baroni, M.: Improving zero-shot learning by mitigating the hubness problem. in *ICLR workshop* (2014)
13. Do, T.T., Tran, T., Reid, I., Kumar, V., Hoang, T., Carneiro, G.: A theoretically sound upper bound on the triplet loss for improving the efficiency of deep distance metric learning. In: *The IEEE Conference on Computer Vision and Pattern Recognition (CVPR)* (2019)
14. Dong, X., Shen, J.: Triplet loss in siamese network for object tracking. In: *The European Conference on Computer Vision (ECCV)* (2018)
15. Fu, Y., Hospedales, T.M., Xiang, T., Gong, S.: Transductive multi-view zero-shot learning. *IEEE Trans. Pattern Anal. Mach. Intell.* **37**(11), 2332–2345 (2015)
16. Gao, R., Hou, X., Qin, J., Chen, J., Liu, L., Zhu, F., Zhang, Z., Shao, L.: Zero-vae-gan: Generating unseen features for generalized and transductive zero-shot learning. *IEEE Transactions on Image Processing* **29**, 3665–3680 (2020)
17. Guo, Y., Ding, G., Jin, X., Wang, J.: Transductive zero-shot recognition via shared model space learning. In: *Proceedings of the Thirtieth AAAI Conference on Artificial Intelligence, AAAI'16*, pp. 3494–3500. AAAI Press (2016)
18. He, K., Zhang, X., Ren, S., Sun, J.: Deep residual learning for image recognition. *2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)* pp. 770–778 (2016)
19. He, X., Zhou, Y., Zhou, Z., Bai, S., Bai, X.: Triplet-center loss for multi-view 3d object retrieval. In: *The IEEE Conference on Computer Vision and Pattern Recognition (CVPR)* (2018)
20. Huang, H., Wang, C., Yu, P.S., Wang, C.D.: Generative dual adversarial network for generalized zero-shot learning. In: *The IEEE Conference on Computer Vision and Pattern Recognition (CVPR)* (2019)
21. Izadi, S., Kim, D., Hilliges, O., Molyneaux, D., Newcombe, R., Kohli, P., Shotton, J., Hodges, S., Freeman, D., Davison, A., Fitzgibbon, A.: Kinectfusion: Real-time 3d reconstruction and interaction using a moving depth camera. In: *Proceedings of the 24th Annual ACM Symposium on User Interface Software and Technology, UIST '11*, pp. 559–568. ACM, New York, NY, USA (2011). DOI 10.1145/2047196.2047270. URL <http://doi.acm.org/10.1145/2047196.2047270>
22. Jiang, H., Wang, R., Shan, S., Chen, X.: Transferable contrastive network for generalized zero-shot learning. In: *Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV)* (2019)
23. Kingma, D.P., Ba, J.: Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980* (2014)
24. Lampert, C., Nickisch, H., Harmeling, S.: Learning to detect unseen object classes by between-class attribute transfer. In: *CVPR Workshops*, pp. 951–958 (2009). DOI 10.1109/CVPRW.2009.5206594
25. Lampert, C.H., Nickisch, H., Harmeling, S.: Attribute-based classification for zero-shot visual object categorization. *IEEE Transactions on Pattern Analysis and Machine Intelligence* **36**(3), 453–465 (2014). DOI 10.1109/TPAMI.2013.140
26. Lee, C.W., Fang, W., Yeh, C.K., Frank Wang, Y.C.: Multi-label zero-shot learning with structured knowledge graphs. In: *The IEEE Conference on Computer Vision and Pattern Recognition (CVPR)* (2018)
27. Lee, D.H.: Pseudo-label : The simple and efficient semi-supervised learning method for deep neural networks. *ICML 2013 Workshop : Challenges in Representation Learning (WREPL)* (2013)
28. Li, J., Chen, B.M., Lee, G.H.: So-net: Self-organizing network for point cloud analysis. In: *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pp. 9397–9406 (2018)
29. Li, J., Jing, M., Lu, K., Ding, Z., Zhu, L., Huang, Z.: Leveraging the invariant side of generative zero-shot learning. In: *IEEE Computer Vision and Pattern Recognition (CVPR)* (2019)
30. Li, R., Li, X., Heng, P.A., Fu, C.W.: Pointaugument: An auto-augmentation framework for point cloud classification. In: *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)* (2020)

31. Li, X., Zhang, D., Ye, M., Li, X., Dou, Q., Lv, Q.: Bidirectional generative transductive zero-shot learning. *Neural Computing and Applications* pp. 1–14 (2020)
32. Li, Z., Xu, C., Leng, B.: Angular triplet-center loss for multi-view 3d shape retrieval. In: *AAAI* (2019)
33. Lin, T., Goyal, P., Girshick, R., He, K., Dollár, P.: Focal loss for dense object detection. *IEEE Transactions on Pattern Analysis and Machine Intelligence* **42**(2), 318–327 (2020). DOI 10.1109/TPAMI.2018.2858826
34. Mikolov, T., Chen, K., Corrado, G., Dean, J.: Efficient estimation of word representations in vector space. *arXiv preprint arXiv:1301.3781* (2013)
35. Mikolov, T., Sutskever, I., Chen, K., Corrado, G.S., Dean, J.: Distributed representations of words and phrases and their compositionality. In: *NIPS*, pp. 3111–3119 (2013)
36. Narayan, S., Gupta, A., Khan, F.S., Snoek, C.G.M., Shao, L.: Latent embedding feedback and discriminative features for zero-shot classification. In: *European Conference on Computer Vision* (2020)
37. Palatucci, M., Pomerleau, D., Hinton, G.E., Mitchell, T.M.: Zero-shot learning with semantic output codes. In: Y. Bengio, D. Schuurmans, J.D. Lafferty, C.K.I. Williams, A. Culotta (eds.) *NIPS*, pp. 1410–1418 (2009)
38. Qi, C.R., Su, H., Mo, K., Guibas, L.J.: Pointnet: Deep learning on point sets for 3d classification and segmentation. *Proc. Computer Vision and Pattern Recognition (CVPR)*, *IEEE* **1**(2), 4 (2017)
39. Qi, C.R., Yi, L., Su, H., Guibas, L.J.: Pointnet++: Deep hierarchical feature learning on point sets in a metric space. In: *Advances in Neural Information Processing Systems*, pp. 5099–5108 (2017)
40. Qiao, R., Liu, L., Shen, C., van den Hengel, A.: Visually aligned word embeddings for improving zero-shot learning. In: *British Machine Vision Conference (BMVC'17)* (2017)
41. Radovanovic, M., Nanopoulos, A., Ivanovic, M.: Hubs in space: Popular nearest neighbors in high-dimensional data. *Journal of Machine Learning Research* **11**, 2487–2531 (2010)
42. Rahman, S., Khan, S.: Deep multiple instance learning for zero-shot image tagging. In: *Asian Conference on Computer Vision (ACCV)* (2018)
43. Rahman, S., Khan, S., Porikli, F.: A unified approach for conventional zero-shot, generalized zero-shot, and few-shot learning. *IEEE Transactions on Image Processing* **27**(11), 5652–5667 (2018). DOI 10.1109/TIP.2018.2861573
44. Rahman, S., Khan, S.H., Porikli, F.: Zero-shot object detection: Joint recognition and localization of novel concepts. *International Journal of Computer Vision* **128**(12), 2979–2999 (2020). DOI 10.1007/s11263-020-01355-6
45. Rohrbach, M., Ebert, S., Schiele, B.: Transfer learning in a transductive setting. In: C.J.C. Burges, L. Bottou, M. Welling, Z. Ghahramani, K.Q. Weinberger (eds.) *NIPS*, pp. 46–54. Curran Associates, Inc. (2013)
46. Rosenblatt, F.: *Principles of Neurodynamics: Perceptrons and the Theory of Brain Mechanisms*. Spartan Books, Washington (1962). it Early work on what would now be referred to as a “connectionist” model.
47. Russakovsky, O., Deng, J., Su, H., Krause, J., Satheesh, S., Ma, S., Huang, Z., Karpathy, A., Khosla, A., Bernstein, M., Berg, A.C., Fei-Fei, L.: *ImageNet Large Scale Visual Recognition Challenge*. *IJCV* **115**(3), 211–252 (2015). DOI 10.1007/s11263-015-0816-y
48. Sariyildiz, M.B., Cinbis, R.G.: Gradient matching generative networks for zero-shot learning. In: *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 2168–2178 (2019)
49. Schonfeld, E., Ebrahimi, S., Sinha, S., Darrell, T., Akata, Z.: Generalized zero- and few-shot learning via aligned variational autoencoders. In: *The IEEE Conference on Computer Vision and Pattern Recognition (CVPR)* (2019)
50. Schroff, F., Kalenichenko, D., Philbin, J.: Facenet: A unified embedding for face recognition and clustering. In: *2015 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 815–823 (2015). DOI 10.1109/CVPR.2015.7298682
51. Shigeto, Y., Suzuki, I., Hara, K., Shimbo, M., Matsumoto, Y.: Ridge regression, hubness, and zero-shot learning. In: *Joint European Conference on Machine Learning and Knowledge Discovery in Databases*, pp. 135–151. Springer (2015)
52. Siddiqi, K., Zhang, J., Macrini, D., Shokoufandeh, A., Bouix, S., Dickinson, S.: Retrieving articulated 3-d models using medial surfaces. *Mach. Vision Appl.* **19**(4), 261–275 (2008). DOI 10.1007/s00138-007-0097-8. URL <http://dx.doi.org/10.1007/s00138-007-0097-8>
53. Song, J., Shen, C., Yang, Y., Liu, Y.P., Song, M.: Transductive unbiased embedding for zero-shot learning. *2018 IEEE/CVF Conference on Computer Vision and Pattern Recognition* pp. 1024–1033 (2018)
54. Srivastava, S., Lall, B.: Deeppoint3d: Learning discriminative local descriptors using deep metric learning on 3d point clouds. *Pattern Recognition Letters* (2019). DOI 10.1016/j.patrec.2019.02.027
55. Su, H., Maji, S., Kalogerakis, E., Learned-Miller, E.G.: Multi-view convolutional neural networks for 3d shape recognition. In: *Proceedings of the IEEE international conference on computer vision*, pp. 945–953 (2015)
56. Uy, M.A., Pham, Q.H., Hua, B.S., Nguyen, D.T., Yeung, S.K.: Revisiting point cloud classification: A new benchmark dataset and classification model on real-world data. In: *International Conference on Computer Vision (ICCV)* (2019)
57. Van Der Maaten, L.: Accelerating t-sne using tree-based algorithms. *Journal of machine learning research* **15**(1), 3221–3245 (2014)
58. Vyas, M.R., Venkateswara, H., Panchanathan, S.: Leveraging seen and unseen semantic relationships for generative zero-shot learning. In: *Computer Vision – ECCV 2020*, pp. 70–86. Springer International Publishing, Cham (2020)
59. Wah, C., Branson, S., Welinder, P., Perona, P., Belongie, S.: *The Caltech-UCSD Birds-200-2011 Dataset*. Tech. Rep. CNS-TR-2011-001, California Institute of Technology (2011)
60. Wang, C., Samari, B., Siddiqi, K.: Local spectral graph convolution for point set feature learning. *arXiv preprint arXiv:1803.05827* (2018)
61. Wang, Y., Sun, Y., Liu, Z., Sarma, S.E., Bronstein, M.M., Solomon, J.M.: Dynamic graph cnn for learning on point clouds. *arXiv preprint arXiv:1801.07829* (2018)
62. Wu, W., Qi, Z., Fuxin, L.: Pointconv: Deep convolutional networks on 3d point clouds. In: *2019 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 9613–9622 (2019). DOI 10.1109/CVPR.2019.00985
63. 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*, pp. 1912–1920 (2015)

64. Xian, Y., Akata, Z., Sharma, G., Nguyen, Q., Hein, M., Schiele, B.: Latent embeddings for zero-shot classification. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR) (2016)
65. Xian, Y., Lampert, C.H., Schiele, B., Akata, Z.: Zero-shot learning - a comprehensive evaluation of the good, the bad and the ugly. *IEEE Transactions on Pattern Analysis and Machine Intelligence* pp. 1–1 (2018). DOI 10.1109/TPAMI.2018.2857768
66. Xian, Y., Lorenz, T., Schiele, B., Akata, Z.: Feature generating networks for zero-shot learning. In: The IEEE Conference on Computer Vision and Pattern Recognition (CVPR) (2018)
67. Xian, Y., Sharma, S., Schiele, B., Akata, Z.: F-vaegan-d2: A feature generating framework for any-shot learning. In: The IEEE Conference on Computer Vision and Pattern Recognition (CVPR) (2019)
68. Xie, S., Liu, S., Chen, Z., Tu, Z.: Attentional shapecontextnet for point cloud recognition. In: The IEEE Conference on Computer Vision and Pattern Recognition (CVPR) (2018)
69. Xu, Y., Fan, T., Xu, M., Zeng, L., Qiao, Y.: Spidercnn: Deep learning on point sets with parameterized convolutional filters. *arXiv preprint arXiv:1803.11527* (2018)
70. Ye, M., Guo, Y.: Progressive ensemble networks for zero-shot recognition. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, pp. 11728–11736 (2019)
71. Yu, Y., Ji, Z., Li, X., Guo, J., Zhang, Z., Ling, H., Wu, F.: Transductive zero-shot learning with a self-training dictionary approach. *IEEE Transactions on Cybernetics* **48**(10), 2908–2919 (2018)
72. Zakharov, S., Kehl, W., Planche, B., Hutter, A., Ilic, S.: 3d object instance recognition and pose estimation using triplet loss with dynamic margin. In: 2017 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS), pp. 552–559 (2017)
73. Zhang, L., Wang, P., Liu, L., Shen, C., Wei, W., Zhang, Y., Van Den Hengel, A.: Towards effective deep embedding for zero-shot learning. *IEEE Transactions on Circuits and Systems for Video Technology* **30**(9), 2843–2852 (2020)
74. Zhang, L., Xiang, T., Gong, S.: Learning a deep embedding model for zero-shot learning. In: CVPR (2017)
75. Zhang, L., Xiang, T., Gong, S.: Learning a deep embedding model for zero-shot learning. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (2017)
76. Zhao, A., Ding, M., Guan, J., Lu, Z., Xiang, T., Wen, J.R.: Domain-invariant projection learning for zero-shot recognition. In: Advances in neural information processing systems (NIPS) (2018)