MEnsA: Mix-up Ensemble Average for Unsupervised Multi Target Domain Adaptation on 3D Point Clouds

Ashish Sinha
Simon Fraser University
Burnaby, Canada
ashish_sinha@sfu.ca

Jonghyun Choi
Yonsei University
Seoul, South Korea
jc@yonsei.ac.kr

Abstract

Unsupervised domain adaptation (UDA) addresses the problem of distribution shift between the unlabeled target domain and labelled source domain. While the single target domain adaptation (STDA) is well studied in both 2D and 3D vision literature, multi-target domain adaptation (MTDA) is barely explored for 3D data despite its wide real-world applications such as autonomous driving systems for various geographical and climatic conditions. We establish an MTDA baseline for 3D point cloud data by proposing to mix the feature representations from all domains together to achieve better domain adaptation performance by an ensemble average, which we call **M**ixup **Ens**emble **A**verage *or* **MEnsA**. *With the mixed representation, we use a domain classifier to improve at distinguishing the feature representations of source domain from those of target domains in a shared latent space. In extensive empirical validations on the challenging PointDA-10 dataset, we showcase a clear benefit of our simple method over previous unsupervised STDA and MTDA methods by large margins (up to* 17.10% *and* 4.76% *on averaged over all domain shifts). We make the code publicly available here*[1].

1. Introduction

For real-world applications ranging from a surveillance system to self-driving cars, deep learning (DL) for 3D data has made significant progress in a wide variety of tasks including classification, segmentation, and detection [10, 16, 33, 49, 55]. Despite the impressive success of DL on 2D vision tasks, its success in 3D data regime involving point cloud data is yet limited by several factors as follows. First, as the point clouds usually do not come with color or textural information, it is not trivial to encode the visual appearances of the structure. Second, annotation cost for 3D is more expensive than that in 2D; the annotation of 3D point clouds may require several rotations, which sometimes is non-trivial due to partial occlusions. Third, the domain gap that arises from the difference in distribution between the original training data (source domain) and the deploying environment (target domain) is larger than that of 2D data owing to the characteristic of 3D geometry [19].

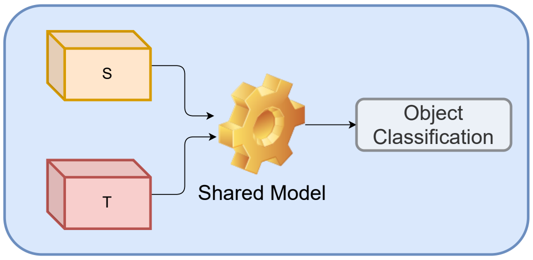
In this work, we address the challenge of reducing the domain gaps for 3D point cloud data, which alleviates the need for extensive annotation across all domains. Specifically, we focus on unsupervised domain adaptation (UDA), that involves transferring knowledge from a label-rich domain, *i.e*., source domain to a label-scarce domain, *i.e*., target domain to reduce the discrepancy between source and target data distributions, typically by exploiting the domain-invariant features [11,14,22,47]. Unfortunately, most of the existing literature on UDA primarily focuses on 2D data.

The mortality risk and associated costs of conducting real-world experiments for autonomous driving and robotics systems have led to the increasing prevalence of synthetic data, particularly 3D data, in the research community [43]. This necessitates the need to develop effective domain adaptation methods for 3D data across different domains, including real-to-sim or sim-to-real adaptation, to ensure successful deployment in real-world scenarios.

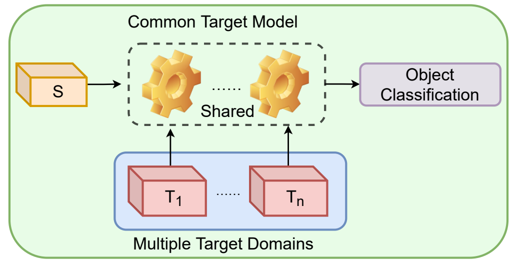
There are numerous works addressing the single-target domain adaptation (STDA) for 3D point clouds [1, 19, 36]. However, when 3D point cloud data of objects is collected under different environmental conditions using various depth cameras or LIDAR sensors for autonomous driving cars, it results in differences in statistical properties such as point cloud density, noise, and orientation. As a result, there is a pressing need for developing Multi-target Domain Adaptation (MTDA) methods, specifically for 3D point cloud data. Despite the well-studied 2D data regime [6,13,30], MTDA in 3D point cloud domain remains an unexplored area in the literature.

In the context of both STDA and MTDA, if the category configurations are identical across all domains, one straightforward solution could be to extend STDA to MTDA by

[1] https://github.com/sinAshish/MEnsA_mtda



(a) Single Target Domain Adaptation setup



(b) Multi Target Domain Adaptation setup

Figure 1. Illustrative comparison of Single Target Domain Adaptation (STDA) and Multi Target Domain Adaptation (MTDA) setup. S is the labelled source dataset, while T_i are the unlabelled target datasets for $i = 1, 2, ..., n$. STDA is a set-up where a single model is adapted to perform accurately on the target domain given a labelled source and unlabelled target data. MTDA is a set-up where a single model is adapted across all unlabelled target domains by learning on the labelled source domain.

using one model per target domain. However, at inference time, it becomes challenging to determine the appropriate model to use when information about the target domain is not available. Moreover, as the number of target domains increases, the computational complexity increases accordingly. Additionally, the model may experience catastrophic forgetting [21,39,44], that involves a neural network trained on a particular task forgetting the previously learned information when trained on a new task. As a result, the network's performance on the initial task deteriorates. This can be a significant challenge when adapting models to multiple target domains, as the model must be able to generalize well across all domains without forgetting the previously learned information. Therefore, we argue that it is preferable to have *a single model* that can adapt to multiple targets. Hence, we propose to learn a single MTDA model for 3D point cloud. We illustrate the differences between STDA and MTDA in Figure 1.

To learn *a single MTDA model*, we first model the multiple N targets as a random variable. We then generate shared information between source and N target domains as N realizations of the shared representations by mixing them. Then, we propose to take an ensemble average of the shared (*i.e*., mixed) representation for training a model that is invariant to multiple domains, calling it **M***ixup* **Ens***emble* **A***verage* or **MEnsA**. The shared representations are learned in a latent space for its low domain gaps [53] in a min-max manner; maximizing the mutual information (MI) in the embedding space between the domains and domain-specific information while minimizing the MI between the domains and the domain-invariant information [13]. We show that our proposed method outperforms several STDA and MTDA approaches proposed for both 2D and 3D regimes on the multiple target domains evaluated on challenging PointDA-10 benchmark dataset [36] by large margins. In summary, we present the following contributions:

- We show that a straightforward extension of domain adaptation methods designed for STDA, in particular 2D data, is non-trivial and does not transfer well to MTDA, specifically in the case of 3D data.
- We propose a simple and novel ensemble-average based mixup approach, named MEnsA, to address the challenging yet unaddressed task of adapting a *single* model across multiple target domains by learning on a single source domain, on point cloud data.
- Extensive validations on PointDA-10 dataset demonstrates a significant benefit of our simple approach over previous unsupervised STDA and MTDA methods by large margins (up to 17.10% and 4.76% on averaged over all domain shifts).
- To the best of our knowledge, this is the first work that benchmarks and addresses the task of MTDA on 3D data, specifically 3D point clouds.

2. Related Work

2.1. 3D Point Clouds

3D visual data is represented in various ways; 3D mesh, voxel grid, implicit surfaces and point clouds. Deep neural networks (DNNs) have been employed to encode the different modalities of 3D data [9, 27, 29, 40, 48]. Among them, point clouds, represented by a set of $\{x, y, z\}$ coordinates, is the most straightforward modality to represent 3D spatial information. PointNet [33] was the pioneering model to encode point clouds, taking advantage of a symmetric function to obtain the invariance of point permutation. But it ignores the local geometric information, which may be vital for describing the objects in 3D space. PointNet++ [34] proposed to stack PointNets hierarchically to model neighborhood information and increase model capacity. PointCNN [24] proposed $\mathcal{X}$-Conv to aggregate features in local patches and apply a bottom-up network structure like typical CNNs. Recent works [16, 52] propose to attend to point-point interactions using self-attention layers and achieve state-of-the-art accuracy on "supervised" classification and segmentation tasks.

Despite the wide usage, point cloud data suffers from labelling efficiency. In real-world scenario, some parts of an object may be occluded or lost (*e.g*., chairs lose legs) while scanning from acquisition devices, *e.g*., LIDAR, making annotation difficult. To alleviate the annotation cost, unsupervised domain adaption (UDA) method for point clouds could be a remedy.

2.2. Single Target Domain Adaptation (STDA)

STDA is an unsupervised transfer learning approach which focuses on adapting a model to perform accurately on unlabeled target data while using labelled source data. Most of the prior works are proposed for 2D data [12, 26, 41, 42]. They are categorized as (1) adversarial, (2) discrepancy, and (3) reconstruction-based approaches. The adversarial approach refers to a model with a discriminator and a generator, where the generator aims to fool the discriminator until the discriminator is unable to distinguish the generated features between the two domains [8, 12, 35, 42]. These approaches have been proposed using either gradient reversal [12] or a combination of feature extractor and domain classifier to encourage domain confusion. The discrepancy based approaches [26] rely on measures between source and target distributions that can be minimized to generalize on the target domain. The reconstruction-based approaches focus on the mapping of the source domain to the target domain data or vice versa [3, 18]. They often rely on the use of GAN [15] in order to find a mapping between source and target.

The STDA methods for 3D point clouds include a self-adaptive module for aligning local features [36], deformation reconstruction as a pretext task [1] or generating synthetic data from source domain to closely match data from target domain [19]. Recent works [1, 19, 38, 46, 54] have been proposed which either use an augmentation method as a self-supervised task or generate synthetic data from source domain to mimic the target domain for UDA on point-clouds in a STDA setting. Nevertheless, extending these approaches in MTDA scenario is not straightforward.

2.3. Multiple Target Domain Adaptation (MTDA)

MTDA requires adapting a model to perform accurately across multiple unlabeled target domains using labelled data from a single source domain. However, the existing MTDA literature has primarily focused on 2D data [6, 13, 30], where they either use target domain labels [13] or not [6, 25, 30, 32]. Gholami *et al*. [13] proposed an approach to adapt to multiple target domains by maximizing the mutual information between domain labels and domain-specific features while minimizing the mutual information between the shared features. Chen *et al*. [6] proposed to blend multiple target domains together and minimize the discrepancy between the source and the blended targets. Liu *et al*. [25] proposed to use a curriculum learning based domain adaptation strategy combined with an augmentation of feature representation from a source domain to handle multiple target domains. Nguyen *et al*. [30] proposed to perform UDA by exploiting the feature representation learned from different target domains using multiple teacher models and then transferring the knowledge to a common student model to generalize over all target domains using knowledge distillation. Although effective on 2D vision tasks, these methods often fail to generalize well on 3D vision tasks due to their design that focuses on images, and disregards local geometric information, and the problem of catastrophic forgetting that can occur during alternate optimization [30]. Consequently, MTDA for 3D vision tasks remains an underexplored research area despite its numerous real-world applications. Thus, we propose the first MTDA method for 3D point cloud.

3. Approach

3.1. Overview

Ganin *et al*. [12] argues that representations that are indistinguishable between the source and target domains are crucial for domain invariant inference. In the context of image classification [50, 51], a common data augmentation technique known as "mixing" or linear interpolation of two images has been employed to make two samples indistinguishable from each other. However, when considering domain-invariance of point clouds, directly mixing the input point clouds presents a challenge, as not all points are *equally* important in describing the object, and it is not trivial to determine which points to mix and which points to exclude. Instead, we encode the point clouds using a DNN, which implicitly weighs the important points and their point-point interactions, and use the embeddings for mixing. As argued in [50, 51], mixing can act as an effective regularizer for guiding the model to be discriminant of source domain from the target domains for point clouds, while remaining indiscriminant of the domain shifts across multiple domains. This enables a model to generalize across multiple domains.

We illustrate the overview of our proposed MTDA approach MEnsA in Figure 2. We employ an adversarial training strategy [12] to reduce the distribution shifts across multiple domains, using gradient reversal for the *domain confusion loss*. Specifically, we first encode the point clouds by the feature extractor module F using a variant of the node attention module proposed in PointDAN [36]. This module F preserves both local geometric structures and the

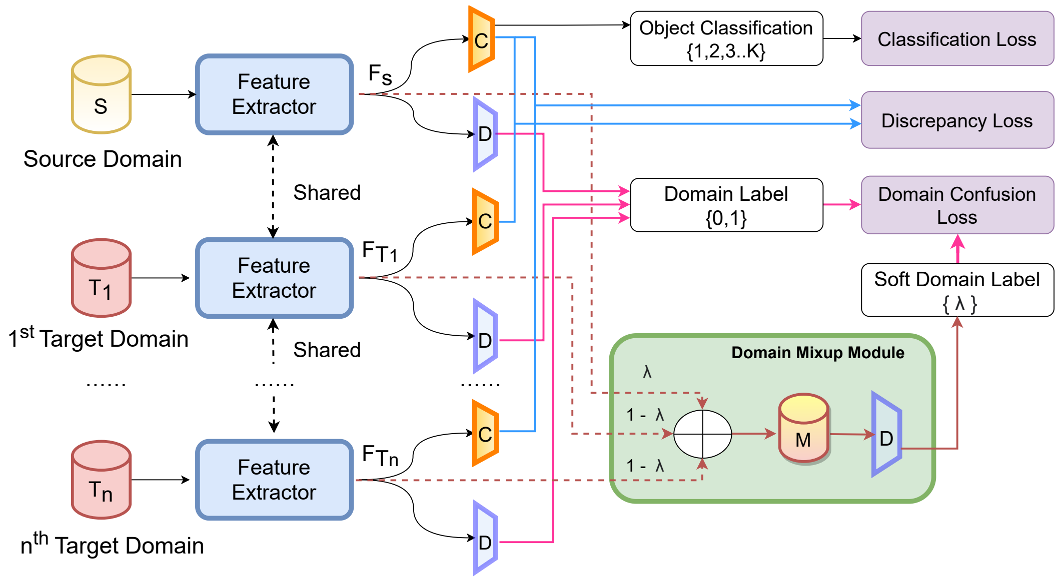


Figure 2. **Overview of our MTDA model.** The labeled source data S, and the unlabeled target data T_i from multiple domains $i = 1..n$, are taken as input by the feature extractor. The source feature F_s is used by object classifier C and domain classifier D to predict the category label, and domain of the input resp. The target feature F_{T_i} is used by C to calculate the discrepancy loss between the source and target features. F_{T_i} is also used by D to differentiate between source and target domain. F_s and F_{T_i} are fed to the domain mixup module to get mixed domain features M to predict the soft scores for source/ target. The model is optimized using a combination of object classification loss, domain confusion loss and discrepancy loss.

global relations between the local features, resulting in a tensor F_T that is split into two branches. The first branch, a *domain classifier* D, is composed of a Gradient Reversal Layer (GRL) [12] and a fully connected layer. The GRL helps in building a feature representation of the raw input $\mathcal{X}$ that is good enough to predict the correct object label $\mathcal{Y}$, but such that the domain label of $\mathcal{X}$ cannot be easily deduced from the feature representation. This promotes domain confusion, where the feature extractor F attempts to confuse the domain classifier D by bridging the two distributions closer. The second branch is an object classifier C consisting of a fully connected layer and a SoftMax activation function. D uses F_T to classify the feature representations into source or target domain, while C classifies them into K classes. Thus, F is adversarially trained by minimizing the object classifier's classification and maximizing the domain classifier's classification loss. Our model's core is the *domain mix-up module*, which is explained in detail in the following section.

3.2. Domain Mixup Module

Inspired by the mixup [51] approach for 2D data, we propose to mix the feature embeddings obtained by F, *but* from multiple domains in the latent space. Unlike the methods for 2D data where the input images are blended by an alpha factor [5, 50], we propose mixing the feature embeddings, since the feature embeddings from the deeper layers of the network contains information about the global shape of the point cloud and local point-point interaction, as demonstrated in [46] applied to STDA set-up. Specifically, we linearly interpolate the source (F_s) and target feature (F_{T_i}) embeddings to obtain F_i^m and the corresponding mixed *soft* domain labels L_i^m as:

$$F_i^m = \lambda F_s + (1 - \lambda) F_{T_i}, \tag{1}$$

$$L_i^m = \lambda L_s + (1 - \lambda) L_{T_i}, \tag{2}$$

where L_s and L_{T_i} denote the domain labels of source and target domain which are set to 1 and 0, respectively. The use of soft labels is essential in creating a continuous probability distribution that indicates the likelihood of a sample belonging to a particular domain. Unlike hard domain labels that limit the classification of samples to just one domain, soft labels promote the learning of domain-invariant features that are useful for both domains and not biased towards one or the other.

The linear interpolation of feature embeddings serves two purposes. Firstly, it helps create a continuous domain-invariant latent space, enabling the mixed features to be mapped to a location in-between the latent space of source and target domain [2]. This continuous latent space is crucial for domain-invariant inference across multiple domains. Secondly, it acts as an effective regularizer, helping the domain classifier D improve in predicting the soft scores for domains (source or target) of the mixed feature embeddings F_i^m, similar to [50, 51]. Since our approach involves multiple target domains, we model domain invariant representation obtained by the mixup F_i^m as a random variable. By using multiple realizations of the 'mixup' representation for different domains, we learn domain-invariant information that is robust to domain shifts.

Baseline mixup (Sep). The standard approach for utilizing the stochastic realization of mixed embeddings, involves mixing the feature embeddings of the source domain S and each of the target domains T_i from a set of target domains $\mathcal{T}$ to train a model. Specifically, each mixup feature is fed into the domain classifier D separately for each of the target domains $\mathcal{T}$, which predicts a soft score, *i.e.*, the mixup ratio for source S and target domain T_i. Then, the cross-entropy loss is calculated and back-propagated over the Gradient Reversal Layer (GRL). We call this approach as the 'Sep.' method and is illustrated in Figure 3 (a).

Mixup Ensemble Average (MEnsA). The sequential training approach employed in the Sep. method may not allow the model to effectively learn the interaction between the source and multiple target domains due to catastrophic forgetting [28, 39], as the method performs a pair-wise mixup between the source and target domains. This results in the model forgetting previously learned domain-invariant features when exposed to a new target domain. To alleviate this problem, we propose a simple method of taking an *ensemble average* of the mixed feature embeddings from the multiple targets F_i^m as:

$$F_m^M = \frac{1}{n}\sum_{i=1}^{n} F_i^m. \tag{3}$$

We call it **M***ixup* **Ens***emble* **A***verage* or **MEnsA**, illustrated in Figure 3 (b). The soft scores for the source and target domains are obtained by feeding the mixed feature F_i^m to the domain classifier D, and the mapping between the source and each target domain is optimized by reproducing kernel Hilbert space (RKHS) *i.e*. MMD. We posit that the ensemble average effectively captures shared information across *all* domains while mitigating conflicting information among them. Consequently, the model trained on this averaged representation, captures differences between the source domain and multiple target domains in a consolidated manner, resulting in improved generalization over domain shifts across multiple target domains.

Our method differs from [46] in that they propose a pairwise mixup at the input and intermediate stage followed by the reconstruction of image samples. In contrast, we explore mixing in a 3D MTDA setup by mixing the latent features from all domains into one, rather than pairwise mixing. Our approach is designed to capture shared domain-invariant features across multiple domains, whereas pairwise mixup only focuses on learning domain-invariant features between the source and one target domain, ignoring the shared features across multiple domains, thereby suffering from catastrophic forgetting.

3.3. Objective Function

The complete architecture is trained end-to-end by minimizing $\mathcal{L}$, which is a weighted combination of supervised classification loss on the source domain ($\mathcal{L}_{cls}$), domain confusion loss ($\mathcal{L}_{dc}$), mixup loss ($\mathcal{L}_{mixup}$) and MMD loss ($\mathcal{L}_{mmd}$), defined as:

$$\mathcal{L} = \log\left(\sum(e^{\gamma(\mathcal{L}_{cls}+\eta\mathcal{L}_{dc}+\zeta\mathcal{L}_{adv})})\right)/\gamma, \tag{4}$$

Here, η, ζ and γ are balancing hyperparameters. The classification, domain confusion and adversarial loss are cross-entropy losses, defined as:

$$\begin{aligned}
\mathcal{L}_{cls} &= \mathcal{L}_{CE}(C(F_s), y_s), \\
\mathcal{L}_{dc} &= \mathcal{L}_{CE}(D(F_s), L_s) + \mathcal{L}_{CE}(D(F_{T_i}, L_{T_i})), \\
\mathcal{L}_{adv} &= \lambda_1\mathcal{L}_{mmd} + \lambda_2\mathcal{L}_{dc} + \lambda_3\mathcal{L}_{mixup},
\end{aligned} \tag{5}$$

where C is the object classifier, D is the domain classifier, y_s is the ground truth object label, L_s is the domain label for source and L_{T_i} is the target domain label set as 1 and 0. λ_1, λ_2 and λ_3 are balancing hyperparameters with constant values of 5.0, 5.0 and 1.2 respectively, and are chosen empirically.

The MMD loss and mixup loss are defined as:

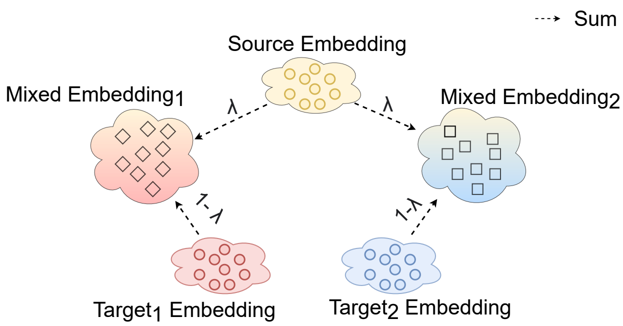
$$\begin{aligned}
\mathcal{L}_{mmd} &= \mathcal{L}_{rbf}(C(F_s), F_{T_i}, \sigma), \\
\mathcal{L}_{mixup} &= \mathcal{L}_{CE}(D(F_m^M), L_i^m),
\end{aligned} \tag{6}$$

where $\mathcal{L}_{rbf}$ is a radial basis function.

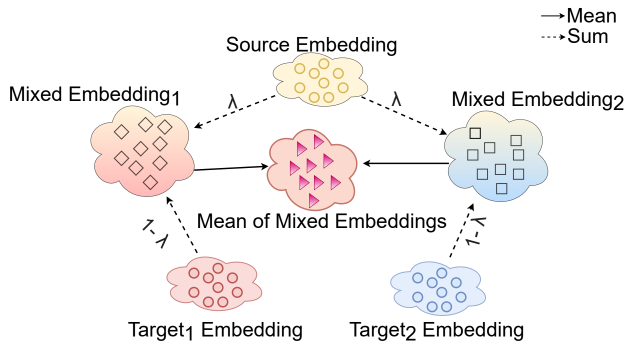
4. Experiments

4.1. Experimental Set-up

Dataset. We evaluate our method on PointDA-10, a benchmark dataset proposed by [36] for the task of point cloud classification. PointDA-10 consists of three subsets of three widely used datasets: ShapeNet [4], ScanNet [7] and ModelNet [45], each containing 10 common classes (chair, table, monitor, etc.). **ModelNet-10 (M)**, called ModelNet hereafter, contains samples of clean 3D CAD models. **ShapeNet-10 (S)**, called ShapeNet hereafter, contains samples of 3D CAD models collected from online repositories. **ScanNet-10 (S*)**, called ScanNet hereafter, contains samples of scanned and reconstructed real-world indoor scenes. Samples from this dataset are significantly harder to classify because (1) many objects have missing parts due to occlusion, and (2) some objects are sampled sparsely. For more details, we refer the readers to the supplementary material.



(a) Baseline Mixup Method (**Sep.**)



(b) Proposed (**MEnsA**) Method

Figure 3. **Comparative illustration of the mixup methods of proposed 'MEnsA' to a baseline method ('Sep')**. They mix feature embeddings of source and N target domains (here, we use $N = 2$ for visualization clarity). The **Sep.** mixup method mixes source domain embeddings to each of the target domain embeddings to create n mixed embeddings, F_i^m. Each of which is passed to domain classifier D to predict soft scores (mixup ratio between source and target domain) instead of hard labels for the domains.

Implementation Details. The proposed approach is implemented on PyTorch [31] framework with Adam [20] as the optimizer for training. The learning rate is assigned as 10^{-3} under the weight decay of 5^{-4} with β_1 and β_2 kept as 0.9 and 0.999. All models were trained for 100 epochs with a batch size of 64. We set λ_1, λ_2 and λ_3 used in Equation 5 to 5.0, 5.0 and 1.2 respectively. For Equation 1 and 2, $\lambda \in [0, 1]$ is a mixup ratio and $\lambda \sim \beta(\alpha, \alpha)$, where β is a beta function and α is set to 2.0 for all experiments. We sample λ from a beta distribution, $\beta(\alpha_1, \alpha_2)$ such that $\alpha_1 = \alpha_2$, as it enables sampling values from a non-skewed distribution.

Motivated by [30], we use scheduled tuning for η in Equation 4 as:

$$\eta = s \cdot e^{\left(\frac{\log \frac{f}{s}}{N_e} \cdot e\right)}. \tag{7}$$

where s is the starting value of 0.1, f is the final value of 0.9, N_e is the total number of epochs and e is the current epoch. This helps in measuring the importance of domain confusion loss over time to adversarially raise the error rate of the domain classifier, thereby forcing it to improve at distinguishing the domains over time.

Baselines. We compare the proposed approach with general purpose UDA methods including maximum mean discrepancy (MMD) [26], adversarial discriminative domain adaptation (ADDA) [12], domain adversarial neural network (DANN) [42] and maximum classifier discrepancy (MCD) [37]. It is also compared with STDA method on point clouds [36].

We also compared our approach to MTDA approaches for 2D vision tasks involving blending targets [6], exploiting shared and private domain spaces [13] and knowledge distillation from multiple teachers to a common student model [30] with minor modification to use 3D point cloud data. In adapting these methods in MTDA scenario, we follow the authors' implementations and the hyperparameters are kept the same as proposed in the respective papers. Since [30] was proposed for MTDA on 2D vision, the authors used ResNet50 [17] as the teacher model and AlexNet [23] as the student model for knowledge distillation. For modifying the approach to 3D MTDA, we used PointNet [33] as a compact student model and PCT [16] as a large teacher model. 'No adaptation' refers to the model trained only by source samples as a naïve baseline, and 'Supervised' refers to the training performed with labelled target samples.

Evaluation Metric. We compare the MTDA performance of the proposed method to the previous works and summarize them in Table 1. We use the same pre-processing steps for all methods. In all the experiments, we report the top-1 classification accuracy on the test set, averaged over 3-folds, for each target domain.

4.2. Results and Discussion

We summarize comparative results for classification on PointDA-10 in Table 1. The proposed approach outperforms UDA methods, STDA method for point clouds and MTDA approaches designed for 2D vision modified for 3D point clouds. Despite the large domain gap rising due to sim-to-real or real-to-sim adaptation on $M \rightarrow S^*$ and $S^* \rightarrow M$, respectively, the proposed approach significantly improves the overall performance.

MCD and DANN outperform most of the other methods, but performs worse than our approach. It is partly because they disentangle the domain-shared features from the domain-specific features, thus achieve better domain generalization. Moreover, we observe that a simple extension of STDA methods to MTDA does not adapt well on multiple

Table 1. Quantitative classification results (%) on PointDA-10 dataset in MTDA setting. For every source domain, we report performance for each target domain. **bold** and second best in <u>underline</u>. 'No adaptation' refers to the model trained only by source samples and 'Supervised' denotes the model when trained with labelled target data

.

Source Domain	ModelNet (M)		ScanNet (S*)		ShapeNet (S)		
Src → Tgt	M → S*	M → S	S* → M	S* → S	S→M	S→S*	Average
No adaptation (Baseline)	35.07	11.75	52.61	29.45	33.65	11.05	28.93
MMD [26]	57.16	22.68	55.40	28.24	36.77	24.88	37.52
DANN [12]	55.03	21.64	54.79	37.37	**42.54**	<u>33.78</u>	40.86
ADDA [42]	29.39	38.46	46.89	20.79	35.33	24.94	32.63
MCD [37]	**57.56**	27.37	54.11	<u>41.71</u>	<u>42.30</u>	22.39	<u>40.94</u>
PointDAN [36]	30.19	<u>44.26</u>	43.17	14.30	26.44	28.92	31.21
AMEAN [6]	<u>55.73</u>	33.53	51.50	30.89	34.73	22.21	38.10
MTDA-ITA [13]	55.23	20.96	<u>56.12</u>	33.71	32.33	25.62	37.33
MT-MTDA [30]	45.43	25.72	28.25	19.51	24.65	**35.27**	29.81
MEnsA (Ours)	45.31	**61.36**	**56.67**	**46.63**	37.02	27.19	**45.70**
↪ w/o mixup	28.48	40.05	33.89	12.14	27.83	24.48	27.81
Supervised in each domain	77.99	67.18	79.83	66.27	63.41	53.02	67.95

Table 2. Quantitative classification results (%) on PointDA-10 dataset in MTDA setting in different mixup scenarios. For every source domain, we report performance for each target domain. Best result in **bold** and second best in <u>underline</u>.

Source Domain	ModelNet (M)		ScanNet (S*)		ShapeNet (S)		
Src → Tgt	M → S*	M →S	S* → M	S*→ S	S→M	S→S*	Average
MEnsA (Ours)	45.31	**61.36**	**56.67**	**46.63**	**37.02**	27.19	**45.70**
Mixup Sep	41.32	<u>47.98</u>	<u>56.18</u>	<u>42.19</u>	28.85	<u>36.69</u>	<u>42.20</u>
Factor-Mixup	41.31	41.49	50.77	38.82	30.77	36.81	40.00
Concat-Mixup	<u>49.20</u>	29.57	50.47	37.5	<u>33.05</u>	25.64	37.57
Inter-Mixup	**50.95**	28.65	51.71	34.38	32.21	**40.80**	39.78
Best of all methods	50.95	61.36	56.67	46.63	37.02	40.80	48.91

target domains. For instance, MMD and DANN achieve an average accuracy of around 42 % in the STDA setup while they barely reach an accuracy of 40 % in the MTDA setup. Interestingly, MCD still performs better than most other methods. Furthermore, UDA methods designed for point clouds also do not perform well when applied to multiple targets, possibly due to catastrophic forgetting during sequential training on multiple target domains. We discuss the performance of methods in STDA setup in more detail in Table 4 of the supplementary due to space sake.

The MTDA methods designed for 2D vision tasks, such as AMEAN, MT-MTDA, and MTDA-ITA, do not perform well on 3D data due to their failure in capturing the local and global geometry of the data while aligning the features across domains. While methods designed for 2D tasks focus on aligning the global image features, local geometry plays a crucial role in achieving good performance for 3D data [36]. This suggests that modality difference can cause a performance drop due to the inherent property differences of each modality, such as brightness or texture in 2D data and geometry, point density, or orientation in 3D data. By incorporating local and global geometry information, our approach is able to align features across domains while preserving the intrinsic structures of 3D data, leading to better domain adaptation performance. Furthermore, the node attention module helps in focussing on important regions of the point cloud, which is critical for accurate classification. These design choices allow our model to effectively capture the modality-specific properties of 3D data, resulting in superior performance compared to existing MTDA methods. For MT-MTDA that uses knowledge distillation, a larger teacher model and a compact student model is desired. However, if the teacher model fails to align local structures to the global structure, it becomes challenging to transfer accurate knowledge to the student model, leading to relatively disappointing results.

AMEAN and MTDA-ITA perform better than other MTDA baselines. MTDA-ITA finds a strong link between the shared latent space common to all domains, while simultaneously accounting for the remaining private, domain-specific factors. Whereas AMEAN mixes the target domains and creates sub-targets implicitly blended with each other, resulting in better performance. Nonetheless, our approach outperforms AMEAN, as we takes features focus on learning domain-invariant features that are hard to distinguish from their originating domain. This forces the model to improve its classification performance independent of the domain, resulting in better overall performance.

Additionally, in Table 5 of the supplementary, we highlight the importance of each module used in the pipeline by conducting an ablation study on each loss term of $\mathcal{L}_{adv}$ in Equ. 5. It can be clearly observed that the mixup module significantly improves performance. Moreover, we show how adversely the class-imbalance in PointDA-10 affects class-wise classification accuracy in Table Tab. 6 of the supplementary due to space sake. Most classes show satisfactory improvements with our proposed approach except for *Bed*, *Bookshelf* and *Sofa*, which highlights the weakness of our model that neglects the scale information, and when different classes share very similar local structures, the model possibly aligns similar structures across these classes (*e.g.*, large columns contained both by *Lamps* and round *Tables*, small legs in *Beds* and *Sofas* or large cuboidal spaces present in *Beds* and *Bookshelves*).

4.3. Variants of the Mix-up Methods

To further investigate the effect of equal weight averaging that is proposed in the MEnsA, we vary scaling schemes in the averaging of the mixup representations. Here, we evaluate three different formulations for mixing, and name it as *Factor*-Mixup, *Concat*-Mixup and *Inter*-Mixup.

Factor-Mixup We mix the feature embeddings from multiple domains together and observe the effect of scaling factor in averaging in Equ. 3 as:

$$F_m^{factor} = \lambda F_s + \sum_{i=1}^{n} \frac{1-\lambda}{n} F_{T_i}. \tag{8}$$

Concat-Mixup Instead of summing the feature embeddings of the domains, we consider concatenation of the mixups with the intuition of learning the proper weights for each mixup embedding for downstream tasks. We use a scaling factor λ and $\frac{1-\lambda}{n}$ for balancing between source and targets both in feature and label as:

$$F_m^{concat} = [\lambda F_s, \frac{1-\lambda}{n} F_{T_1}, ..., \frac{1-\lambda}{n} F_{T_n}], \tag{9}$$

$$L_m^{concat} = [\lambda, 2\frac{1-\lambda}{n}, .., N\frac{1-\lambda}{n})], \tag{10}$$

where $[\cdot, \cdots, \cdot]$ denotes concatenation operation.

Inter-Mixup In addition to aggregating all the domains together in MEnsA, we also consider a linear interpolation of the target domains excluding the F_s for both feature and label as:

$$F_m^T = \lambda F_{T_1} + (1-\lambda) F_{T_2}. \tag{11}$$

$$L_m^T = \lambda L_{T_1} + (1-\lambda) L_{T_2}. \tag{12}$$

We devised Inter-Mixup, with the intuition that regularizing the target domains alone should help the model to learn a mapping where it is able to learn the target domain-invariant features promoting better MTDA, thus learning a good separation between the source and target domains in the latent space.

We compare the performance of the variants with MEnsA and Sep., and summarize the results in Table 2. As Scaler-Mixup is a linear interpolation of all the domains together, the mixed feature representation obtained by Scaler-Mixup has large values in each dimension, which may lead to gradients with large magnitude. It may hurt the accuracy. Unlike MEnsA and other mixup variants, Concat-Mixup concatenates the feature embeddings from multiple domains. As the number of domains increases, the shared latent space between the domains mixed becomes smaller. Therefore, it becomes difficult for the model to learn domain-invariant features across all domains, leading to poor performance among all other variants of mixup. Interestingly, we observe that mixing the target domains together with the source domain in Inter-Mixup performs better on ScanNet which has real-world samples. We believe it is because the model is able to learn better domain-invariant features between the real and synthetic domains, as the samples from ScanNet are more sparse and occluded as compared to other domains. Moreover, we show the visualization of the feature embeddings using t-SNE plots in the supplementary.

5. Conclusion

We model the multi target domains as a random variable and propose to mix latent space embeddings of all domains in an ensemble average to encode domain invariant information for the 3D point cloud for the first time in literature. The mixed representation helps the domain classifier to learn better domain-invariant features and improve the domain adaptation performance in multi-target domain adaptation set-up. We demonstrated the efficacy of our approach on the point cloud DA benchmark dataset of PointDA-10 by showing that our approach significantly outperforms UDA, STDA and MTDA methods proposed for 2D data.

References

[1] Idan Achituve, Haggai Maron, and Gal Chechik. Self-supervised learning for domain adaptation on point clouds. In *Proceedings of the IEEE/CVF Winter Conference on Applications of Computer Vision*, pages 123–133, 2021.

[2] David Berthelot, Colin Raffel, Aurko Roy, and Ian Goodfellow. Understanding and improving interpolation in autoencoders via an adversarial regularizer. *arXiv preprint arXiv:1807.07543*, 2018.

[3] Konstantinos Bousmalis, Nathan Silberman, David Dohan, Dumitru Erhan, and Dilip Krishnan. Unsupervised pixel-level domain adaptation with generative adversarial networks. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 3722–3731, 2017.

[4] Angel X Chang, Thomas Funkhouser, Leonidas Guibas, Pat Hanrahan, Qixing Huang, Zimo Li, Silvio Savarese, Manolis Savva, Shuran Song, Hao Su, et al. Shapenet: An information-rich 3d model repository. *arXiv preprint arXiv:1512.03012*, 2015.

[5] Yunlu Chen, Vincent Tao Hu, Efstratios Gavves, Thomas Mensink, Pascal Mettes, Pengwan Yang, and Cees GM Snoek. Pointmixup: Augmentation for point clouds. *arXiv preprint arXiv:2008.06374*, 2020.

[6] Ziliang Chen, Jingyu Zhuang, Xiaodan Liang, and Liang Lin. Blending-target domain adaptation by adversarial meta-adaptation networks. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 2248–2257, 2019.

[7] Angela Dai, Angel X Chang, Manolis Savva, Maciej Halber, Thomas Funkhouser, and Matthias Nießner. Scannet: Richly-annotated 3d reconstructions of indoor scenes. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 5828–5839, 2017.

[8] Jiahua Dong, Yang Cong, Gan Sun, and Dongdong Hou. Semantic-transferable weakly-supervised endoscopic lesions segmentation. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 10712–10721, 2019.

[9] Yutong Feng, Yifan Feng, Haoxuan You, Xibin Zhao, and Yue Gao. Meshnet: Mesh neural network for 3d shape representation. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 33, pages 8279–8286, 2019.

[10] Yifan Feng, Zizhao Zhang, Xibin Zhao, Rongrong Ji, and Yue Gao. Gvcnn: Group-view convolutional neural networks for 3d shape recognition. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 264–272, 2018.

[11] Basura Fernando, Amaury Habrard, Marc Sebban, and Tinne Tuytelaars. Unsupervised visual domain adaptation using subspace alignment. In *Proceedings of the IEEE international conference on computer vision*, pages 2960–2967, 2013.

[12] Yaroslav Ganin and Victor Lempitsky. Unsupervised domain adaptation by backpropagation. In *International conference on machine learning*, pages 1180–1189. PMLR, 2015.

[13] Behnam Gholami, Pritish Sahu, Ognjen Rudovic, Konstantinos Bousmalis, and Vladimir Pavlovic. Unsupervised multi-target domain adaptation: An information theoretic approach. *IEEE Transactions on Image Processing*, 29:3993–4002, 2020.

[14] Boqing Gong, Yuan Shi, Fei Sha, and Kristen Grauman. Geodesic flow kernel for unsupervised domain adaptation. In *2012 IEEE conference on computer vision and pattern recognition*, pages 2066–2073. IEEE, 2012.

[15] Ian J Goodfellow, Jean Pouget-Abadie, Mehdi Mirza, Bing Xu, David Warde-Farley, Sherjil Ozair, Aaron C Courville, and Yoshua Bengio. Generative adversarial nets. In *NIPS*, 2014.

[16] Meng-Hao Guo, Jun-Xiong Cai, Zheng-Ning Liu, Tai-Jiang Mu, Ralph R Martin, and Shi-Min Hu. Pct: Point cloud transformer. *arXiv preprint arXiv:2012.09688*, 2020.

[17] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 770–778, 2016.

[18] Weixiang Hong, Zhenzhen Wang, Ming Yang, and Junsong Yuan. Conditional generative adversarial network for structured domain adaptation. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 1335–1344, 2018.

[19] Junxuan Huang and Chunming Qiao. Generation for adaption: a gan-based approach for 3d domain adaption inpoint cloud. *arXiv preprint arXiv:2102.07373*, 2021.

[20] Diederik P Kingma and Jimmy Ba. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*, 2014.

[21] James Kirkpatrick, Razvan Pascanu, Neil C. Rabinowitz, Joel Veness, Guillaume Desjardins, Andrei A. Rusu, Kieran Milan, John Quan, Tiago Ramalho, Agnieszka Grabska-Barwinska, Demis Hassabis, Claudia Clopath, Dharshan Kumaran, and Raia Hadsell. Overcoming catastrophic forgetting in neural networks. *PNAS*, 2017.

[22] Elyor Kodirov, Tao Xiang, Zhenyong Fu, and Shaogang Gong. Unsupervised domain adaptation for zero-shot learning. In *Proceedings of the IEEE international conference on computer vision*, pages 2452–2460, 2015.

[23] Alex Krizhevsky, Ilya Sutskever, and Geoffrey E Hinton. Imagenet classification with deep convolutional neural networks. *Advances in neural information processing systems*, 25:1097–1105, 2012.

[24] Yangyan Li, Rui Bu, Mingchao Sun, Wei Wu, Xinhan Di, and Baoquan Chen. Pointcnn: Convolution on x-transformed points. *Advances in neural information processing systems*, 31:820–830, 2018.

[25] Ziwei Liu, Zhongqi Miao, Xingang Pan, Xiaohang Zhan, Dahua Lin, Stella X Yu, and Boqing Gong. Open compound domain adaptation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 12406–12415, 2020.

[26] Mingsheng Long, Jianmin Wang, Guiguang Ding, Jiaguang Sun, and Philip S Yu. Transfer feature learning with joint distribution adaptation. In *Proceedings of the IEEE international conference on computer vision*, pages 2200–2207, 2013.

[27] Daniel Maturana and Sebastian Scherer. Voxnet: A 3d convolutional neural network for real-time object recognition. In *2015 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pages 922–928. IEEE, 2015.

[28] M. McCloskey and Neal. Catastrophic interference in connectionist networks: The sequential learning problem. *Psychology of Learning and Motivation*, 24:109–165, 1989.

[29] Mateusz Michalkiewicz, Jhony K Pontes, Dominic Jack, Mahsa Baktashmotlagh, and Anders Eriksson. Implicit surface representations as layers in neural networks. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 4743–4752, 2019.

[30] Le Thanh Nguyen-Meidine, Atif Belal, Madhu Kiran, Jose Dolz, Louis-Antoine Blais-Morin, and Eric Granger. Unsupervised multi-target domain adaptation through knowledge distillation. In *Proceedings of the IEEE/CVF Winter Conference on Applications of Computer Vision*, pages 1339–1347, 2021.

[31] Adam Paszke, Sam Gross, Francisco Massa, Adam Lerer, James Bradbury, Gregory Chanan, Trevor Killeen, Zeming Lin, Natalia Gimelshein, Luca Antiga, et al. Pytorch: An imperative style, high-performance deep learning library. *arXiv preprint arXiv:1912.01703*, 2019.

[32] Xingchao Peng, Zijun Huang, Ximeng Sun, and Kate Saenko. Domain agnostic learning with disentangled representations. In *International Conference on Machine Learning*, pages 5102–5112. PMLR, 2019.

[33] Charles R Qi, Hao Su, Kaichun Mo, and Leonidas J Guibas. Pointnet: Deep learning on point sets for 3d classification and segmentation. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 652–660, 2017.

[34] Charles R Qi, Li Yi, Hao Su, and Leonidas J Guibas. Pointnet++: Deep hierarchical feature learning on point sets in a metric space. *arXiv preprint arXiv:1706.02413*, 2017.

[35] Can Qin, Lichen Wang, Yulun Zhang, and Yun Fu. Generatively inferential co-training for unsupervised domain adaptation. In *Proceedings of the IEEE/CVF International Conference on Computer Vision Workshops*, pages 0–0, 2019.

[36] Can Qin, Haoxuan You, Lichen Wang, C-C Jay Kuo, and Yun Fu. Pointdan: A multi-scale 3d domain adaption network for point cloud representation. *arXiv preprint arXiv:1911.02744*, 2019.

[37] Kuniaki Saito, Kohei Watanabe, Yoshitaka Ushiku, and Tatsuya Harada. Maximum classifier discrepancy for unsupervised domain adaptation. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 3723–3732, 2018.

[38] Khaled Saleh, Ahmed Abobakr, Mohammed Attia, Julie Iskander, Darius Nahavandi, Mohammed Hossny, and Saeid Nahvandi. Domain adaptation for vehicle detection from bird's eye view lidar point cloud data. In *Proceedings of the IEEE/CVF International Conference on Computer Vision Workshops*, pages 0–0, 2019.

[39] Joan Serra, Didac Suris, Marius Miron, and Alexandros Karatzoglou. Overcoming catastrophic forgetting with hard attention to the task. In *International Conference on Machine Learning*, pages 4548–4557. PMLR, 2018.

[40] Hang Su, Subhransu Maji, Evangelos Kalogerakis, and Erik Learned-Miller. Multi-view convolutional neural networks for 3d shape recognition. In *Proceedings of the IEEE international conference on computer vision*, pages 945–953, 2015.

[41] Baochen Sun, Jiashi Feng, and Kate Saenko. Correlation alignment for unsupervised domain adaptation. In *Domain Adaptation in Computer Vision Applications*, pages 153–171. Springer, 2017.

[42] Eric Tzeng, Judy Hoffman, Kate Saenko, and Trevor Darrell. Adversarial discriminative domain adaptation. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 7167–7176, 2017.

[43] Gul Varol, Javier Romero, Xavier Martin, Naureen Mahmood, Michael J Black, Ivan Laptev, and Cordelia Schmid. Learning from synthetic humans. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 109–117, 2017.

[44] Shixian Wen and Laurent Itti. Overcoming catastrophic forgetting problem by weight consolidation and long-term memory. *arXiv*, abs/1805.07441, 2018.

[45] Zhirong Wu, Shuran Song, Aditya Khosla, Fisher Yu, Linguang Zhang, Xiaoou Tang, and Jianxiong Xiao. 3d shapenets: A deep representation for volumetric shapes. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 1912–1920, 2015.

[46] Minghao Xu, Jian Zhang, Bingbing Ni, Teng Li, Chengjie Wang, Qi Tian, and Wenjun Zhang. Adversarial domain adaptation with domain mixup. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 34, pages 6502–6509, 2020.

[47] Donggeun Yoo, Namil Kim, Sunggyun Park, Anthony S Paek, and In So Kweon. Pixel-level domain transfer. In *European Conference on Computer Vision*, pages 517–532. Springer, 2016.

[48] Haoxuan You, Yifan Feng, Rongrong Ji, and Yue Gao. Pvnet: A joint convolutional network of point cloud and multi-view for 3d shape recognition. In *Proceedings of the 26th ACM international conference on Multimedia*, pages 1310–1318, 2018.

[49] Haoxuan You, Yifan Feng, Xibin Zhao, Changqing Zou, Rongrong Ji, and Yue Gao. Pvrnet: Point-view relation neural network for 3d shape recognition. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 33, pages 9119–9126, 2019.

[50] Sangdoo Yun, Dongyoon Han, Seong Joon Oh, Sanghyuk Chun, Junsuk Choe, and Youngjoon Yoo. Cutmix: Regularization strategy to train strong classifiers with localizable features. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 6023–6032, 2019.

[51] Hongyi Zhang, Moustapha Cisse, Yann N Dauphin, and David Lopez-Paz. mixup: Beyond empirical risk minimization. *arXiv preprint arXiv:1710.09412*, 2017.

[52] Hengshuang Zhao, Li Jiang, Jiaya Jia, Philip Torr, and Vladlen Koltun. Point transformer. *arXiv preprint arXiv:2012.09164*, 2020.

[53] Han Zhao, Shanghang Zhang, Guanhang Wu, Joao P Costeira, José MF Moura, and Geoffrey J Gordon. Multiple

source domain adaptation with adversarial training of neural networks. *arXiv preprint arXiv:1705.09684*, 2017.

[54] Xingyi Zhou, Arjun Karpur, Chuang Gan, Linjie Luo, and Qixing Huang. Unsupervised domain adaptation for 3d keypoint estimation via view consistency. In *Proceedings of the European conference on computer vision (ECCV)*, pages 137–153, 2018.

[55] Yin Zhou and Oncel Tuzel. Voxelnet: End-to-end learning for point cloud based 3d object detection. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 4490–4499, 2018.