sensors

MDPI

Article

Synthetic Source Universal Domain Adaptation through Contrastive Learning

Jungchan Cho

School of Computing, Gachon University, Seongnam 13120, Korea; thinkai@gachon.ac.kr; Tel.: +82-31-750-5328

Abstract: Universal domain adaptation (UDA) is a crucial research topic for efficient deep learning model training using data from various imaging sensors. However, its development is affected by unlabeled target data. Moreover, the nonexistence of prior knowledge of the source and target domain makes it more challenging for UDA to train models. I hypothesize that the degradation of trained models in the target domain is caused by the lack of direct training loss to improve the discriminative power of the target domain data. As a result, the target data adapted to the source representations is biased toward the source domain. I found that the degradation was more pronounced when I used synthetic data for the source domain and real data for the target domain. In this paper, I propose a UDA method with target domain contrastive learning. The proposed method enables models to leverage synthetic data for the source domain and train the discriminativeness of target features in an unsupervised manner. In addition, the target domain feature extraction network is shared with the source domain classification task, preventing unnecessary computational growth. Extensive experimental results on VisDa-2017 and MNIST to SVHN demonstrated that the proposed method significantly outperforms the baseline by 2.7% and 5.1%, respectively.

Keywords: universal domain adaptation; contrastive learning; classification; deep learning

Citation: Cho, J. Synthetic Source Universal Domain Adaptation through Contrastive Learning. *Sensors* **2021**, *21*, 7539. https://doi.org/10.3390/s21227539

Academic Editors: Ukho Le and Dae-Ki Kang

Received: 12 October 2021
Accepted: 11 November 2021
Published: 12 November 2021

1. Introduction

The application of deep learning has been rapidly improving performance in various fields of computer vision, such as object detection [1], human pose estimation [2,3], semantic segmentation [4,5], and image classification [6]. In addition, its application in real-world industries has also been promoted [7,8].

Achieving significant improvement in deep learning relies on the abundance of labeled samples [9] in a supervised manner. However, since images are collected from various sensors, test samples may be different from those used in the training phase. In this case, even if the test sample is semantically the same as the training samples, the performance is significantly reduced [10]. This is known as the domain shift problem [11]. As a result, deep learning models require new training for a new domain. Training data must be collected again, and the labeling process must be repeated continuously even for the same task in different domains. By contrast, humans are capable of robustly recognizing images and inferring their meanings across different domains. For example, a person who has seen and understood pictures of numerous cars (source domain) can also recognize them on artwork depicting cars (target domain). Hence, recent studies on deep learning have focused on increasing the efficiency of training models by leveraging information that has already been learned [12].

Domain adaptation (DA) [13,14] is the task of adapting algorithms trained on one or more source domains to other related target domains in the same task. Many studies transferred the knowledge learned from the source domain to the target domain successfully without target data labels. However, there is a common assumption that the label sets of the source and target domains are the same, i.e., closed-set domain adaptation (CDA) [15–18]. However, closed-set domain adaptation cannot effectively bridge label gaps in different

domains if the classes in the target domain are less than those in the source domain, i.e., partial domain adaptation (PDA) [19–21], or if the target domain includes "unknown" classes that are not visible in the source domain, i.e., open-set domain adaptation (ODA) [22–24].

Currently, research is underway on how to adapt learned knowledge to different domains with no restrictions to the source and target domain label sets. Universal domain adaptation (UDA) [25,26] includes CDA, PDA, and ODA. As a result, target domain samples should be correctly classified into "known" class labels in a source domain or "unknown" classes if the label is not in the source domain [25,26]. Saito et al. [26] achieved UDA by proposing neighbor clustering and entropy separation in a self-supervised manner. Neighbor clustering brings target samples closer to known source class prototypes or unknown target samples. At the same time, entropy separation separates unknown target samples from known class boundaries using confident target samples.

Even though [26] demonstrated a compelling performance for universal domain adaptation, I found that it is insufficient when synthesized data were used as a source domain. My hypothesis on this problem is that if the source and target domains are too different, self-supervision by two losses is erroneous, and the feature extraction network trained by [26] does not obtain discriminative feature representations for the target domains. This drawback stems from the fact that there is no direct training loss for the target domain. Ideally, universal domain adaptation should be able to adapt knowledge learned from synthesized source domain data to the real-world target domain, as depicted in Figure 1. It is a fundamental solution that can automatically generate labeled training data without human intervention, thereby reducing the cost of large-scale data labeling.

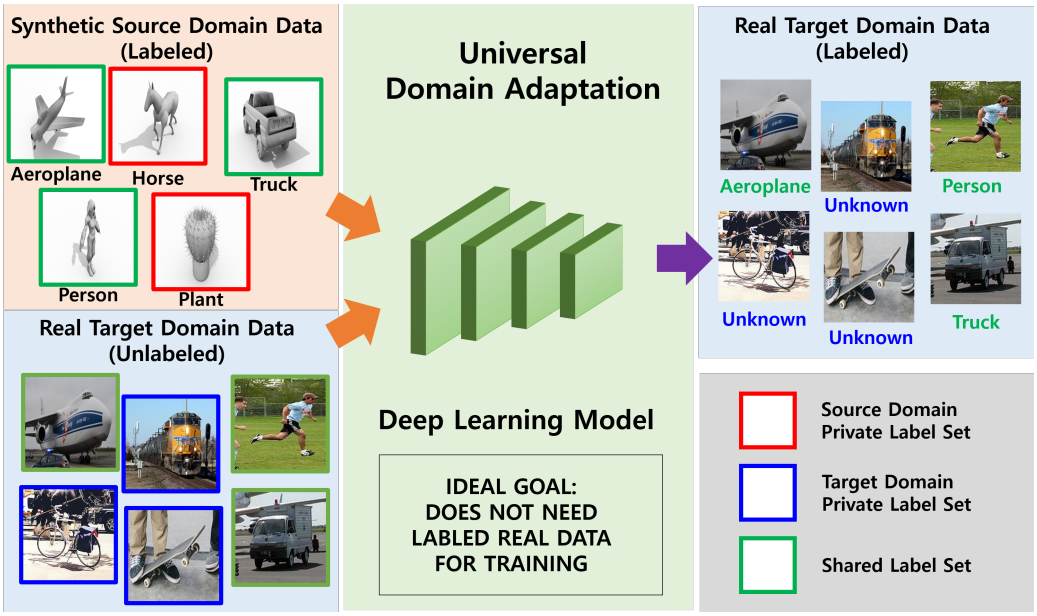

Figure 1. In ideal circumstances, universal domain adaptation should apply the knowledge learned from synthesized source domain data to the real target domain. Therefore, I focus on leveraging synthetic source domain data to reduce labeling costs by humans.

In this paper, I propose a novel universal domain adaptation method that improves the discriminative power of the target feature representation of [26]. Here, discriminative means that positive samples are pulled together, and multiple negative samples are pushed toward each other. However, because there are no labels available in the target domain, the proposed method utilizes contrastive learning [27] for target domain training, wherein I first generate a positive pair by augmenting the same target domain sample. Then, negative samples are generated by randomly selecting samples from the target minibatch except for the positive pair. By following this process, I can maximize the mutual information between different views on the same data in the target domain without requiring any labels

for positive pairs. Moreover, the improved discriminative power in the target domain helps neighborhood clustering and known/unknown separation. Thereby, the proposed method achieves better performance than the baseline method [26] when a synthetic source domain is adapted to a real target domain.

The contributions are summarized as follows:

- In contrast to the existing universal domain adaptation methods [25,26], I focused on using synthetic source domain data to reduce human labeling costs.
- In this case, owing to the limitations of data synthesis, the source domain has different characteristics from the real data in the target domain. Thus, the target domain information should be fully utilized to learn the feature representation. I used contrastive learning [27] to extract discriminative features from the target domain.
- For contrastive learning, the target domain feature extraction network is not separately constructed. The source and target domains share a common feature extraction network, thus avoiding unnecessary computation surges.
- The experiments conducted on the VisDa-2017 dataset and MNIST to SVHN dataset indicate that the proposed method significantly outperforms baselines by 2.7% and 5.1%, respectively.

The remainder of this paper is organized as follows: Section 2 introduces related work. In Section 3, the proposed method is described. Section 4 presents the implementation and experimental results. Finally, Section 5 concludes the paper.

2. Related Work

2.1. Domain Adaptation

Let L_s and L_t denote a set of class labels present in the source and target, respectively. Considering this, domain adaptions can be categorized into three main topics according to the label set constraints between domains:

- Closed-set domain adaptation, if $L_s = L_t$. The main challenge in closed-set domain adaptation is to mitigate the domain gap between the source and target domains.
- Partial domain adaptation, if $L_s \supset L_t$. This does not assume that label sets are identical across domains, but that the source label set subsumes the target label set. The mismatched label set in the source domain presents a new challenge for domain adaptation.
- Open-set domain adaptation, if $L_s \subset L_t$. To avoid the unrealistic assumptions of closed-set domain adaptation, open-set domain adaptation assumes that the target domain contains unknown labels in the source domain.

I here introduce recent domain adaptation methods.

2.1.1. Close-Set Domain Adaptation

Haeusser et al. [28] produced similar feature representations of the source and target domains by utilizing a bipartite graph. Tzeng et al. [29] first outlined a unified framework for adversarial domain adaptation by combining discriminative modeling, untied weight sharing, and generative adversarial loss. Long et al. [30] designed a conditional domain adversarial network with multilinear and entropy conditioning to improve discriminability and transferability.

2.1.2. Partial Domain Adaptation

Cao et al. [19] introduced partial domain adaptation as a new challenge and proposed a down-weighting solution to handle outlier source classes that do not appear in the target domain. Cao et al. [20] further improved PDA and proposed an example transfer network that was designed using a weighting scheme to quantify the transferability of examples in the source domain. Therefore, it alleviates negative transfers and promotes positive transfers. Zhang et al. [21] extended the adversarial nets-based domain adaptation that identifies the importance score of source samples based on a two-domain classifier strategy.

2.1.3. Open-Set Domain Adaptation

Busto et al. [22] introduced the concept of open sets to domain adaptation and proposed a method to fit in both closed and open-set scenarios by solving the assignment problem of targets that are potentially known classes in the source domain. Saito et al. [23] proposed a method for learning feature representations that separate unknown targets from known target samples based on adversarial training.

Recently, You et al. [25] introduced a universal domain adaptation and proposed a universal adaptation network that enables the shared label to be distinguished from the private label set for each domain by quantifying sample-level transferability. Satio et al. [26] improved UDA by proposing neighboring cluster and entropy separation losses, which are trained in a self-supervised manner. However, I found that this method [26] was insufficient when synthesized samples were used for the source domain. In this paper, I propose a contrastive-based UDA method for leveraging synthetic source data.

2.2. Contrastive Learning

Recent studies have shown that unsupervised feature representations for various downstream tasks can be learned in a contrastive manner. Oord et al. [31] captured useful feature representation by predicting the future in latent space based on autoregressive models and probabilistic contrastive loss. MoCo [32] is a seminal method for contrastive learning that maintains a dynamic dictionary (memory bank) for computing the contrastive loss and shows competitive results on ImageNet [9] classification. Chen et al. [27] conducted systematic studies to understand the factors that enable contrastive prediction tasks to learn useful representations. Grill et al. [33] achieved an improved accuracy of the ImageNet classification task without negative pairs. Instead, they relied on online target networks that interact and learn from each other.

Based on the abovementioned success of contrastive learning, I extracted the target domain information in a contrastive manner for universal domain adaptation.

3. Approach

3.1. Notation

Let the dataset for the source domain be $\mathcal{D}^s = \{\mathbf{x}_i^s, y_i^s\}_{i=1}^{N^s}$, where $\mathbf{x}_i^s$ and y_i^s indicate the i-th input sample and its corresponding true label, respectively. N^s is the number of samples in the source domain. The target domain dataset $\mathcal{D}^t = \{\mathbf{x}_i^t\}_{i=1}^{N^t}$ does not have true labels in which samples $\mathbf{x}^t$ correspond to the same classes or unknown samples in the source domain. Let $\tilde{\mathbf{x}}_i^s$ indicate augmented source domain samples by data augmentation μ, e.g., random cropping, random color distortion, etc. The feature representation $\tilde{\mathbf{f}}_i^s \in \mathbb{R}^d$ is calculated using a feature extraction network G, i.e., $\tilde{\mathbf{f}}_i^s = G(\tilde{\mathbf{x}}_i^s)$. For the target domain, two different data augmentations μ and μ' are applied to the target domain sample $\mathbf{x}_i^t$. The augmented samples are denoted by $\tilde{\mathbf{x}}_i^t$ and $\hat{\mathbf{x}}_i^t$, and their feature representations by G are denoted by $\tilde{\mathbf{f}}_i^t$ and $\hat{\mathbf{f}}_i^t$. The same feature extraction network G is shared for both the source and target domains. Let $C(\cdot; \mathbf{W})$ be a linear classification network, where the weights are represented as $\mathbf{W} = [\mathbf{w}_1, \mathbf{w}_2, \ldots, \mathbf{w}_K]$. The k-th weight vector $\mathbf{w}_k \in \mathbb{R}^d$ is normalized by the l_2 norm and used as a prototype representing the k-th class. The proposed method uses a memory bank $\tilde{\mathbf{F}}^t = [\tilde{\mathbf{f}}_1^t, \tilde{\mathbf{f}}_2^t, \ldots, \tilde{\mathbf{f}}_{N^t}^t] \in \mathbb{R}^{d \times N^t}$ that saves N^t feature representations of samples in the target domain. I also denote the total memory bank as $\mathbf{V} = [\tilde{\mathbf{F}}^t, \mathbf{W}] = [\tilde{\mathbf{f}}_1^t, \tilde{\mathbf{f}}_2^t, \ldots, \tilde{\mathbf{f}}_{N^t}^t, \mathbf{w}_1, \mathbf{w}_2, \ldots, \mathbf{w}_K] \in \mathbb{R}^{d \times (N^t + K)}$.

3.2. Architecture

In this study, domain adaptation aims to transfer the knowledge of labeled samples in the source domain to train a classification model for unlabeled target domain samples. However, the problem is made more challenging for universal domain adaptation by including unknown samples in the target domain. Furthermore, some classes in the source domain may not have samples in the target domain. Therefore, the feature extraction

network should move target domain samples closer to known class samples from the source domain, simultaneously making it easier to distinguish between unknown and known samples. To achieve this goal, researchers in a previous work (DANCE) [26] trained the feature extraction network through two losses: neighbor clustering and entropy separation. However, there is no direct loss function available to learn a target-domain-specific representation due to a lack of true labels for the target domain sample. My hypothesis is that the absence of the target domain-specific loss function causes the feature extraction network to create biased feature representations for source domain classification. This prevents useful feature representation for unknown target domain samples. To resolve this, I propose a method that learns feature representation for a source domain classification task in a supervised manner and a target domain instance-level classification task in an unsupervised manner.

To summarize, the proposed method consists of four loss functions as depicted in Figure 2. The first function is the loss function for source domain classification ($\mathcal{L}_{cls}$), the second function is the instance-level classification of the target domain ($\mathcal{L}_{ct}$), and the other two functions are the losses for neighbor clustering ($\mathcal{L}_{nc}$) and entropy separation ($\mathcal{L}_{es}$) proposed in [26]. I sequentially explain three loss functions, except for the source domain classification loss function.

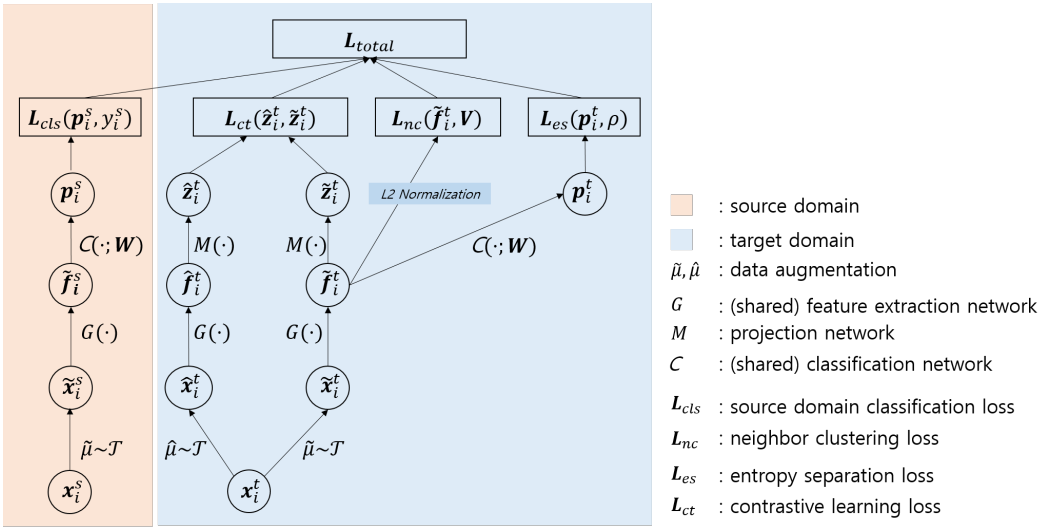

Figure 2. Overview. The proposed method learns a feature representation space considering the discriminative power in both the source ($\mathcal{L}_{cls}$) and target ($\mathcal{L}_{ct}$ in Section 3.2.1) domains. The target domain feature is enhanced using contrastive learning, which is an instance-level classification task to overcome the limitations of data synthesis in the source domain and unlabeled target domain data. The remaining losses ($\mathcal{L}_{nc}$ in Section 3.2.2 and $\mathcal{L}_{es}$ in Section 3.2.3) cluster class samples and separate known/unknown classes, respectively. Notably, the feature extraction network G is shared in the source and target domains. $\mathcal{T}$ denotes the pool of data augmentation.

3.2.1. Target Domain Contrastive Loss

Contrastive representation learning aims to learn a discriminative embedding space devoid of any true labels through self-supervised learning wherein similar pairs of samples are close to each other, and dissimilar pairs of samples are distant from each other. To achieve this, each image in a given dataset is considered its own class, i.e., instance-level classification [27].

I used the SimCLR-based contrastive learning method [27] to learn the target domain feature representation. The procedure followed is as given below.

- Generate different perspectives based on the same target sample:Randomly selected data augmentations μ and μ' are applied sequentially to a target domain sample $\mathbf{x}_i^t$, obtaining two augmented samples $\tilde{\mathbf{x}}_i^t$ and $\hat{\mathbf{x}}_i^t$. As a result, two sets of minibatches

are obtained: $\tilde{\mathcal{B}}^t = \{\tilde{\mathbf{x}}_k^t\}$ and $\hat{\mathcal{B}}^t = \{\hat{\mathbf{x}}_k^t\}$. The size of each minibatch is equal, i.e., $|\tilde{\mathcal{B}}^t| = |\hat{\mathcal{B}}^t|$.

- Extract features from augmented target samples: The augmented target samples are fed into a feature extraction network G to generate feature representations $\tilde{\mathbf{f}}_i^t$ and $\hat{\mathbf{f}}_i^t$. The feature extraction network is shared for the source-domain classification task. Although any network is freely available, I use *ResNet* [6,27], i.e.,

$$\begin{aligned} \tilde{\mathbf{f}}_i^t &= G(\tilde{\mathbf{x}}_i^t) = ResNet(\tilde{\mathbf{x}}_i^t), \\ \hat{\mathbf{f}}_i^t &= G(\hat{\mathbf{x}}_i^t) = ResNet(\hat{\mathbf{x}}_i^t). \end{aligned} \tag{1}$$

- Obtain target projection features for contrastive learning: Projection feature representations $\tilde{\mathbf{z}}_i^t$ and $\hat{\mathbf{z}}_i^t$ are obtained via a projection network M, and contrastive learning loss is applied to them. I use a shallow multilayer perceptron (*MLP*) for projection networks M, i.e.,

$$\begin{aligned} \tilde{\mathbf{z}}_i^t &= M(\tilde{\mathbf{f}}_i^t) = MLP(\tilde{\mathbf{f}}_i^t), \\ \hat{\mathbf{z}}_i^t &= M(\hat{\mathbf{f}}_i^t) = MLP(\hat{\mathbf{f}}_i^t). \end{aligned} \tag{2}$$

- Minimize contrastive learning loss in the target domain: Given two augmented minibatches $\tilde{\mathcal{B}}^t$ and $\hat{\mathcal{B}}^t$, contrastive learning aims to identify $\hat{\mathbf{x}}_i^t$ using $\tilde{\mathbf{x}}^t$ or vice versa. The loss function is defined using the projection representation of the target samples as follows:

$$\begin{aligned} \mathcal{L}_{ct} &= \frac{1}{2|\tilde{\mathcal{B}}^t|} \Sigma_{i=1}^{|\tilde{\mathcal{B}}|} [\mathcal{L}_{ict}(\tilde{\mathbf{z}}_i^t, \hat{\mathbf{z}}_i^t) + \mathcal{L}_{ict}(\hat{\mathbf{z}}_i^t, \tilde{\mathbf{z}}_i^t)], \\ \mathcal{L}_{ict}(\tilde{\mathbf{z}}_i^t, \hat{\mathbf{z}}_i^t) &= -\log \frac{\exp(s(\tilde{\mathbf{z}}_i^t, \hat{\mathbf{z}}_i^t)/\tau)}{Z}, \end{aligned} \tag{3}$$

  where τ is the temperature parameter [34].
  $Z = \Sigma_{k=1}^{N^t} \exp(s(\tilde{\mathbf{z}}_i^t, \hat{\mathbf{z}}_k^t)/\tau) + \Sigma_{k=1,k\neq i}^{N^t} \exp(s(\tilde{\mathbf{z}}_i^t, \tilde{\mathbf{z}}_k^t)/\tau)$, and $s(\tilde{\mathbf{z}}_i^t, \hat{\mathbf{z}}_i^t)$ is the similarity function. I used the cosine similarity for this as $s(\tilde{\mathbf{z}}_i^t, \hat{\mathbf{z}}_i^t) = \frac{\tilde{\mathbf{z}}_i^{t\,T} \hat{\mathbf{z}}_i^t}{||\tilde{\mathbf{z}}_i^t|| ||\hat{\mathbf{z}}_i^t||}$.

Minimizing the contrastive learning loss function causes the two projection feature representations from the same sample to be similar and the feature representations from different samples to be dissimilar. This results in learning powerful feature representations for instance-level classification, even without any labeled samples. Thus, the contrastive learning loss finds a feature space that represents better target domain samples by eliminating unnecessary information.

3.2.2. Neighbor Clustering (NC) Loss

The purpose of this loss function [26] is to bring target samples together in the unknown class prototypes in the source domain or locate them close to their neighborhood in unknown target domain samples. This can be achieved by minimizing the following entropy function:

$$\mathcal{L}_{nc} = -\frac{1}{|\tilde{\mathcal{B}}^t|} \Sigma_{i=1}^{|\tilde{\mathcal{B}}^t|} \Sigma_{j=1,j\neq i}^{N^t+K} p_{i,j} \log(p_{i,j}), \tag{4}$$

where $p_{i,j}$ is a probability based on the similarity between the i-th target feature representation $\tilde{\mathbf{f}}_i^t$ and the j-th total memory bank element $\mathbf{v}_j$ as

$$p_{i,j} = \frac{\exp(\mathbf{v}_j^T \tilde{\mathbf{f}}_i^t/\tau)}{\Sigma_{j=1,j\neq i}^{N^t+K} \exp(\mathbf{v}_j^T \tilde{\mathbf{f}}_i^t/\tau)}, \tag{5}$$

where τ is the temperature parameter to control the distribution concentration degree [34].

3.2.3. Entropy Separation Loss

Even with neighbor clustering loss, it is challenging to separate unknown samples from known classes. To enhance separation, the entropy separation loss function proposed in [26] is minimized.

$$\begin{aligned} \mathcal{L}_{es} &= \frac{1}{|\tilde{\mathcal{B}}^t|}\Sigma_{i=1}^{|\tilde{\mathcal{B}}|}\mathcal{L}_{ies}(\mathbf{p}_i), \\ \mathcal{L}_{ies} &= \begin{cases} -|H(\mathbf{p}_i) - \rho| & (|H(\mathbf{p}_i) - \rho| > m), \\ 0 & otherwise \end{cases} \end{aligned} \tag{6}$$

where $\mathbf{p}_i$ is a source class probability vector for the i-th target sample, which is calculated as $\mathbf{p}_i = \mathbf{W}^T\tilde{\mathbf{f}}_i^t$. $H(\cdot)$ is the entropy function. If the value $H(\mathbf{p}_i)$ is much lower, it becomes nearly indistinguishable to one particular class in the source domain (low entropy), whereas if it is much higher, there are no similar classes in the source class (high entropy). Thus, the extremes entropy value $H(\mathbf{p}_i)$ is obtained by minimizing the entropy separation loss. This creates a separation effect in which the unknown samples move away from the class of the source domain. ρ is a threshold boundary value used to separate unknown samples from known classes, and m is a tuning parameter for minimizing the loss function using only reliable samples.

3.2.4. Total Loss

Finally, I combine all four loss functions described above with two hyperparameters, λ_1 and λ_2, as follows:

$$\mathcal{L}_{total} = \mathcal{L}_{cls} + \lambda_1\mathcal{L}_{ct} + \lambda_2(\mathcal{L}_{nc} + \mathcal{L}_{es}). \tag{7}$$

Minimizing the total loss function results in learning the discriminative feature representation for both source and target domains, neighborhood clustering in the feature space, and maximizing the separation of unknown samples from known classes.

4. Experiments

4.1. Implementation Details

I conducted experiments in PyTorch [35] and on a single NVIDIA Titan RTX GPU and followed experimental settings [26]. However, the proposed method is not hardware dependent. The feature extractor G was set to *ResNet*50 [6] and pretrained on ImageNet [9] after removing the last linear layer in all experiments. In addition, I added a new source classification layer $\mathbf{W}$. For contrastive projection features, I used a two-layer perception with ReLU activation, i.e., Linear-ReLU-Linear. For baselines, I used the implementation of a previous work [26]. For the proposed method, the values of ρ, m, and λ_1 were set to $\log(K)/2$, 0.5, and 0.05, respectively, where K is the number of shared classes. The batch size was set to 36. I used the SGD optimizer, and the initial learning rate and weight decay were set to 0.01 and 0.0005, respectively.

Table 1 shows a comparison of the number of parameters and GFLOPs between the baseline [26] and the proposed method. Since the proposed method in the training phase requires contrastive projection features, additional parameters and GFLOPs of 9.4 and 5.3 M are required. However, for inference, the proposed method uses the same network as the baseline that consists of one feature extractor G and one classifier C. Notably, the parameters and computational costs for inference do not increase.

Table 1. Comparisons of the number of baseline [26] parameters and GFLOPs during training. For inference, the proposed method uses the same network as the baseline.

Training	Method	Feature Extraction (G)	Classification (C)	Projection (M)	Total
Params	Baseline	23.5 M	0.01 M	-	23.5 M
	Proposed	23.5 M	0.01 M	8.4 M	31.9 M
GFLOPs	Baseline	4.1×2	0.0004×2	-	8.2
	Proposed	4.1×3	0.0004×2	0.6×2	13.5

4.2. Evaluation and Data Augmentation

The goal of the experiments was to compare the proposed results with DANCE [26] across subcases of UDA, i.e., CDA, PDA, and ODA, under synthetic source domain and real target domain classification tasks. Following the evaluation metrics in a previously published study [26], I calculated the accuracy over all target samples in CDA and PDA. In ODA, I used the average per class, including "unknown". For example, VisDa-2017 ODA reported an average of over seven classes, i.e., six shared and one unknown class. I ran the experiment three times and reported the average results.

I denote the accuracies reported in [26] as DANCE in the tables in this paper. In addition, I present two additional results from [26] as DANCE-R and DANCE-A for a fair comparison. DANCE-R represents the reproduced result of [26] based on their codes (https://github.com/VisionLearningGroup/DANCE. Accessed time: 12 July 2021.) with the same random seeds. DANCE-A represents another trained version of DANCE by the same augmentation μ as that used for the proposed method. This validates my hypothesis that the data augmentation of synthetic source data is not sufficient to cover the real target domain. I set μ to random flip, Gaussian blur, color jitter, and grayscale. I also set μ' to random flip and a scale transform for adjusting the input size to feature extraction network G because contrastive learning needs different views generated augmentations μ and μ' based on the same source data.

4.3. Datasets and Results

I used VisDA-2017 [36], MNIST [37] and SVHN [38] datasets. VisDA-2017, MNIST, and SNHN were used to analyze the performance of the proposed method in synthetic-to-real environments.

4.3.1. VisDA-2017 Dataset

VisDA-2017 [36] is a large-scale dataset. It contains 12 category images of various sizes in two domains. One of these contains synthetic 2D renderings of 3D objects with 152,397 images, and the other contains photographs of real-world objects with 55,388 images. I resized the images to 256 × 256 for the feature extraction input. Figure 3 depicts examples of the VisDA-2017 dataset. The first and second rows in Figure 3 depict the source and target domain images, respectively. This dataset exhibits a significant domain shift. I followed [26] to construct closed-set, partial, and open-set domain adaptation tasks to validate the proposed method in large-scale synthetic-to-real domain adaptation. The values in parentheses correspond to the number of shared classes, source private classes, and target private classes, respectively. For instance, (6/6/0) indicates partial domain adaptation, and $|L_s \cap L_t|$, $|L_s - L_t|$, and $|L_t - L_s|$ are 6, 6, and 0, respectively.

As expected, the results in DANCE-* in Table 2 were unfavorable compared to the proposed version in all cases. In particular, the highest improvement was observed in the open-set case. The contrastive loss provides more help since the open-set has many unknown classes. I also checked the effect of hyperparameter λ_1. As shown in Table 2, the proposed method is insensitive to λ_1. In both cases, $\lambda_1 = 0.1$ and $\lambda_1 = 0.03$ were better than the baseline methods. Table 2 also shows a comparison of accuracy between the proposed method and existing other methods. The baseline DANCE-A achieved significantly better accuracy than traditional domain adaptation methods. Nevertheless, the proposed method

improved the accuracy of a baseline on VisDa-2017 dataset. Especially for the open-set domain adaptation, learning in the target domain is important; therefore, the proposed method achieved an accuracy improvement of 4.1% over baseline. The proposed method demonstrates almost the same performance as a baseline without significant performance degradation and outperforms other methods.

Figure 3. Examples of VisDa-2017. The first row shows synthetic images for the source domain, and the second row shows the real-photographed images. There are substantial variations between the two domains.

Table 2. Comparisons of baselines on the VisDA-2017 dataset.

Method (Split)	Closed-Set (12/0/0)	Partial (6/6/0)	Open-Set (6/0/6)	Avg.
SO [26]	46.3	46.3	43.3	45.3
DANN [17]	69.1	38.7	48.2	52.0
ETN [20]	64.1	59.8	51.7	58.5
STA [39]	48.1	48.2	51.7	49.3
UAN [25]	66.4	39.7	50.0	52.0
DANCE [26]	70.2	73.7	65.3	69.7
DANCE-R	71.3	76.6	64.2	70.7
DANCE-A	73.0	80.1	63.8	72.3
Proposed ($\lambda_1 = 0.1$)	73.7	82.6	66.7	74.3
Proposed ($\lambda_1 = 0.03$)	**74.0**	**83.0**	**67.9**	**75.0**

The source only (SO), DANCE, SO, DANN, ETN, STA, and UAN are the values reported in [26]. The other values are my experimental results.

4.3.2. Digit Datasets: MNIST → SVHN

For another synthetic-to-real domain adaptation scenario, I used two-digit datasets: MNIST [37] and SVHN [38].

MNIST consists of 28 × 28 sized gray images containing handwritten numbers ranging from zero to nine. The standard training and test splits included 60,000 and 10,000 images, respectively. I resized it to 32 × 32 for the experiments. Examples of the images are shown in the first row of Figure 4.

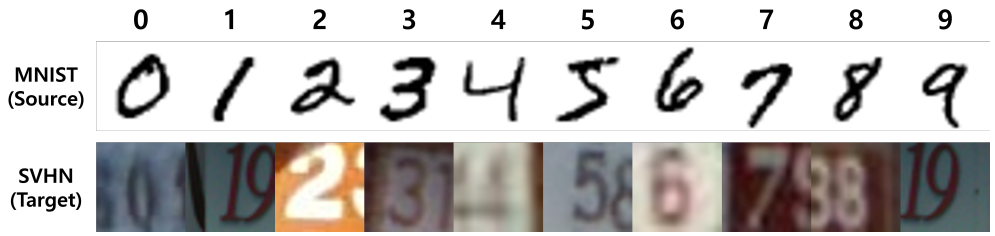

Figure 4. Examples of MNIST and SVHN. The first row shows the MNIST dataset for the source domain. The figures appear synthetic compared to the second-row images in the SVHN dataset acquired from Google Street View. The variations between the two domains are extensive.

SVHN, the Street View House Numbers dataset, consists of 32 × 32 sized natural scene images of numbers acquired from Google Street View. The training and test splits contain 73,257 and 26,032 images, respectively. The images in the SVHN dataset are aligned at the

center of the desired number, but the surroundings are cluttered with visual artifacts and distractors. Examples of the images are depicted in Figure 4.

Images in the MNIST dataset are not synthetic but appear synthetic because they contain only black-and-white image files, as depicted in Figure 4. By contrast, the SVHN images in Figure 4 are obtained from the vision cameras. According to their appearances, I set the training split of MNIST as the source domain and the test split of SVHN as the target domain. I did not apply the random crop, flip, and translation augmentations because the SVHN dataset images also included a second number around the centered number as well as a centered ground-truth number, as depicted in the second row of Figure 4.

For the experiments, I constructed closed-set, partial, and open-set domains as follows: Closed-set domain adaptation: $L_s \cap L_t = L_s = L_t = \{0, 1, 2, 3, 4, 5, 6, 7, 8, 9\}$; Partial domain adaptation: $L_s \cap L_t = \{0, 1, 2, 3, 4\}$ and $L_s - L_t = \{5, 6, 7, 8, 9\}$; Open-set domain adaptation: $L_s \cap L_t = \{0, 1, 2, 3, 4\}$ and $L_t - L_s = \{5, 6, 7, 8, 9\}$. The results presented in Table 3 are consistent with those of the experiment discussed in Section 4.3.1. DANCE-A shows significantly better results than DANCE-R. Therefore, data augmentation can be used to handle domain gaps. However, the proposed method outperforms DANCE-R and DANCE-A, regardless of the values of hyperparameter λ_1, i.e., for both cases $\lambda_1 = 0.1$ and $\lambda_1 = 0.03$. Even though the baseline DANCE-A uses the same data augmentation as that of the proposed method, the results are, on average, approximately five percent lower than those of the proposed method.

Table 3. Comparison of baselines on MNIST → SVHN.

Method (Split)	Closed Set (10/0/0)	Partial (5/5/0)	Open Set (5/0/5)	Avg.
SO-R	22.7	25.2	30.2	26.0
SO-A	39.5	36.5	46.5	40.8
DANCE-R	25.6	29.1	28.8	27.8
DANCE-A	43.1	41.9	46.0	43.7
Proposed ($\lambda_1 = 0.1$)	**53.0**	42.2	48.3	47.8
Proposed ($\lambda_1 = 0.03$)	52.8	**44.2**	**49.3**	**48.8**

SO-R and SO-A correspond to the values by a model trained only with source domain samples. DANCE-R and DANCE-A correspond to baselines set [26]. All values represent my experimental results.

4.3.3. Analysis of the Target Domain Contrastive Loss Function

The results of both experiments in Sections 4.3.1 and 4.3.2 validated that (1) augmentation of synthetic source domain data does not make sufficient generalization for the target domain even when used in universal domain adaptation algorithms, (2) the added contrastive loss helps generalize the feature space to cover the target domain, and (3) feature network G can be shared in target contrastive learning for efficiency.

Figure 5 depicts the effect of batch sizes on MNIST to SVHN datasets, where 'SO', 'DANCE-A', and Proposed ($\lambda = 0.3$) are the same those in Table 3. In all comparison methods, regardless of the baseline and proposed method, the accuracy decreased as the batch size increased in closed-set, partial, and open-set domain adaptations. For the domain adaptations, the larger the batch, the more significant the impact of the target domain in the feature space. This makes it difficult for supervised classification features to be transferred from the source domain to the target domain.

I also analyzed the effect of the target domain contrastive learning loss function on domain adaptation. To do this, I added a target domain contrastive learning loss to 'SO' that applied unsupervised classification loss function only to the source domain. This is marked 'SO+Target_Contrastive'. In Figure 5, the cyan triangles consistently demonstrate better accuracy than the red squares, regardless of the batch size. The accuracy improvement rates are represented by green bars. This means that the additional target domain contrastive loss helps the model adapt to the target domain. Thus, the proposed method, which adds a contrastive loss to the baseline DANCE-A, achieved the best result by efficiently learning the target domain. The accuracy improvement rates relative to the baselines are

represented by yellow bars. As the batch size increased, the baseline did not optimize the model parameters on close-set and partial domain adaptations; therefore, it remains blank.

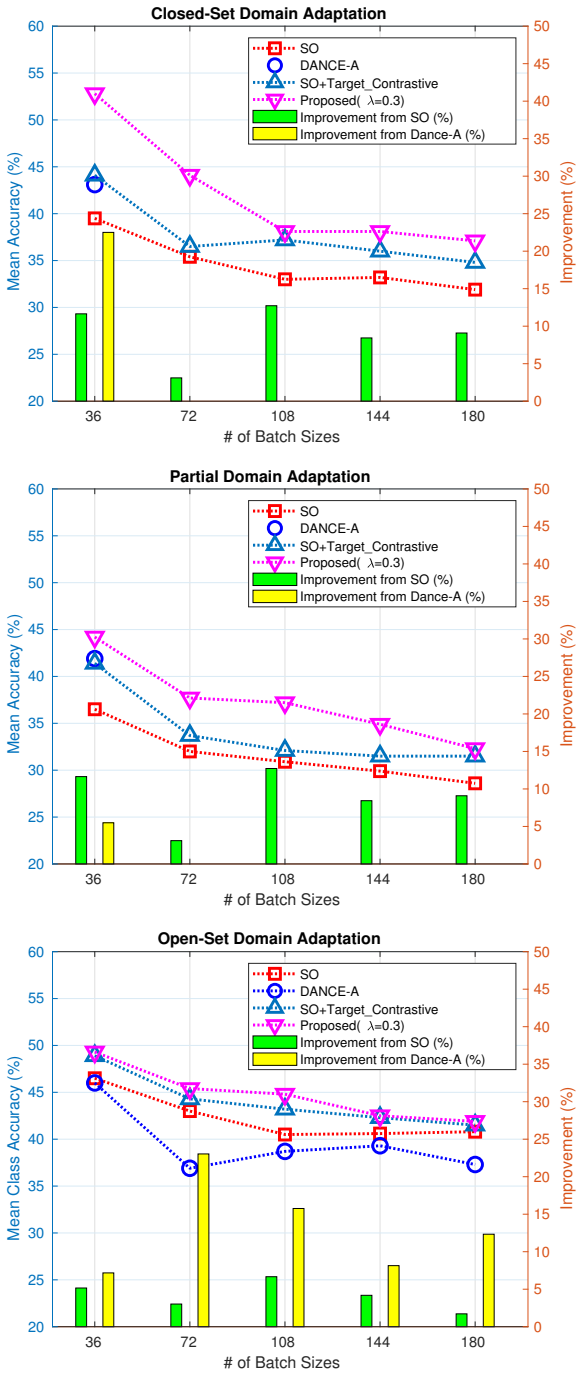

Figure 5. The effect of batch sizes on MNIST to SVHN datasets. 'SO', 'DANCE-A', and Proposed ($\lambda = 0.3$) are the same those in Table 3. 'SO+Target_Contrastive' means is the addition of target contrastive learning loss to 'SO'. The accuracy improvement rates are represented by green and yellow bars. In the case of optimization unstable, it remains blank.

5. Conclusions

Universal domain adaptation (UDA) is an important research topic for the efficient use of trained models in various image sensors. I found that the baseline method does not have a direct training loss for the target domain to improve the discriminative power. I hypothesized that if the source and target domains are too different, the feature extraction

network does not obtain discriminative feature representations for the target domains. To overcome the limitations of the synthetic data, the information about the target domain data should be fully utilized to learn feature representations. To do this, I used contrastive learning [27] in the target domain. The experimental results validated that the proposed method significantly contributed to improving the UDA task. In addition, the target domain feature extraction network was shared with the source domain classification task, avoiding unnecessary computation increases. The proposed method can be easily expanded to help efficient model training on various problems, such as imaging sensors of self-driving cars.

Author Contributions: J.C. conceived the idea, and he designed and performed the experiments; he also wrote the paper. All authors have read and agreed to the published version of the manuscript.

Funding: This work was supported by the National Research Foundation of Korea (NRF) grant funded by the Korea government (MSIT) (No. NRF-2019R1F1A1058666).

Institutional Review Board Statement: Not applicable.

Informed Consent Statement: Not applicable.

Data Availability Statement: The dataset information. VisDa-2017: https://github.com/VisionLearningGroup/taskcv-2017-public/tree/master/classification; MNIST: http://yann.lecun.com/exdb/mnist/; SVHN: http://ufldl.stanford.edu/housenumbers/.

Conflicts of Interest: The author declares no conflict of interest.

References

1. Ren, S.; He, K.; Girshick, R.; Sun, J. Faster R-CNN: Towards real-time object detection with region proposal networks. In Proceedings of the Advances in Neural Information Processing Systems, Montreal, QC, Canada, 7–12 December 2015.
2. Xiao, B.; Wu, H.; Wei, Y. Simple baselines for human pose estimation and tracking. In Proceedings of the European Conference on Computer Vision, Munich, Germany, 8–14 September 2018.
3. Yun, K.; Park, J.; Cho, J. Robust human pose estimation for rotation via self-supervised learning. *IEEE Access* **2020**, *8*, 32502–32517. [CrossRef]
4. Ronneberger, O.; Fischer, P.; Brox, T. U-net: Convolutional networks for biomedical image segmentation. In Proceedings of the International Conference on Medical Image Computing and Computer-Assisted Intervention, Munich, Germany, 5–9 October 2015.
5. Chen, L.C.; Papandreou, G.; Kokkinos, I.; Murphy, K.; Yuille, A.L. Deeplab: Semantic image segmentation with deep convolutional nets, atrous convolution, and fully connected crfs. *IEEE Trans. Pattern Anal. Mach. Intell.* **2017**, *40*, 834–848. [CrossRef] [PubMed]
6. He, K.; Zhang, X.; Ren, S.; Sun, J. Deep residual learning for image recognition. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, Las Vegas, NV, USA, 26 June–1 July 2016.
7. Lee, G.; Yun, K.; Cho, J. Improved Human-Object Interaction Detection Through On-the-Fly Stacked Generalization. *IEEE Access* **2021**, *9*, 34251–34263. [CrossRef]
8. Vo, D.M.; Nguyen, D.M.; Le, T.P.; Lee, S.W. HI-GAN: A hierarchical generative adversarial network for blind denoising of real photographs. *Inf. Sci.* **2021**, *570*, 225–240. [CrossRef]
9. Krizhevsky, A.; Sutskever, I.; Hinton, G.E. Imagenet classification with deep convolutional neural networks. In Proceedings of the Advances in Neural Information Processing Systems, Stateline, NV, USA, 3–8 December 2012.
10. Ben-David, S.; Blitzer, J.; Crammer, K.; Pereira, F. Analysis of representations for domain adaptation. In Proceedings of the Advances in Neural Information Processing Systems, Vancouver, BC, Canada, 3–6 December 2007.
11. Sun, B.; Feng, J.; Saenko, K. Return of frustratingly easy domain adaptation. In Proceedings of the AAAI Conference on Artificial Intelligence, Phoenix, AZ, USA, 12–17 February 2016.
12. Cho, J.; Lee, M. Building a compact convolutional neural network for embedded intelligent sensor systems using group sparsity and knowledge distillation. *Sensors* **2019**, *19*, 4307. [CrossRef] [PubMed]
13. Ben-David, S.; Blitzer, J.; Crammer, K.; Kulesza, A.; Pereira, F.; Vaughan, J.W. A theory of learning from different domains. *Mach. Learn.* **2010**, *79*, 151–175. [CrossRef]
14. Sun, S.; Shi, H.; Wu, Y. A survey of multi-source domain adaptation. *Inf. Fusion* **2015**, *24*, 84–92. [CrossRef]
15. Long, M.; Cao, Y.; Wang, J.; Jordan, M. Learning transferable features with deep adaptation networks. In Proceedings of the International Conference on Machine Learning, Lille, France, 6–11 July 2015.
16. Tzeng, E.; Hoffman, J.; Darrell, T.; Saenko, K. Simultaneous deep transfer across domains and tasks. In Proceedings of the IEEE International Conference on Computer Vision, Santiago, Chile, 11–18 December 2015; pp. 4068–4076.
17. Ganin, Y.; Ustinova, E.; Ajakan, H.; Germain, P.; Larochelle, H.; Laviolette, F.; Marchand, M.; Lempitsky, V. Domain-adversarial training of neural networks. *J. Mach. Learn. Res.* **2016**, *17*, 2096–2030.

18. Long, M.; Zhu, H.; Wang, J.; Jordan, M.I. Unsupervised Domain Adaptation with Residual Transfer Networks. In Proceedings of the Advances in Neural Information Processing Systems, Barcelona, Spain, 5–10 December 2016.
19. Cao, Z.; Ma, L.; Long, M.; Wang, J. Partial adversarial domain adaptation. In Proceedings of the European Conference on Computer Vision, Munich, Germany, 8–14 September 2018.
20. Cao, Z.; You, K.; Long, M.; Wang, J.; Yang, Q. Learning to transfer examples for partial domain adaptation. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, Long Beach, CA, USA, 16–20 June 2019.
21. Zhang, J.; Ding, Z.; Li, W.; Ogunbona, P. Importance weighted adversarial nets for partial domain adaptation. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, Salt Lake City, UT, USA, 18–22 June 2018.
22. Panareda Busto, P.; Gall, J. Open set domain adaptation. In Proceedings of the IEEE International Conference on Computer Vision, Venice, Italy, 22-29 October 2017.
23. Saito, K.; Yamamoto, S.; Ushiku, Y.; Harada, T. Open set domain adaptation by backpropagation. In Proceedings of the European Conference on Computer Vision, Munich, Germany, 8–14 September 2018.
24. Zhang, Y.; Liu, T.; Long, M.; Jordan, M. Bridging theory and algorithm for domain adaptation. In Proceedings of the International Conference on Machine Learning, Long, Beach, CA, USA, 9–15 June 2019.
25. You, K.; Long, M.; Cao, Z.; Wang, J.; Jordan, M.I. Universal domain adaptation. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, Long Beach, CA, USA, 16–20 June 2019.
26. Saito, K.; Kim, D.; Sclaroff, S.; Saenko, K. Universal Domain Adaptation through Self Supervision. In Proceedings of the Advances in Neural Information Processing Systems, Virtual-only, 6–12 December 2020.
27. Chen, T.; Kornblith, S.; Norouzi, M.; Hinton, G. A simple framework for contrastive learning of visual representations. In Proceedings of the International Conference on Machine Learning, Xi'an, China, 10 August 2020.
28. Haeusser, P.; Frerix, T.; Mordvintsev, A.; Cremers, D. Associative domain adaptation. In Proceedings of the IEEE International Conference on Computer Vision, Venice, Italy, 22–29 October 2017.
29. Tzeng, E.; Hoffman, J.; Saenko, K.; Darrell, T. Adversarial discriminative domain adaptation. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, Honolulu, HI, USA, 21–26 July 2017.
30. Long, M.; Cao, Z.; Wang, J.; Jordan, M.I. Conditional adversarial domain adaptation. *arXiv* **2018**, arXiv:1705.10667.
31. Oord, A.v.d.; Li, Y.; Vinyals, O. Representation learning with contrastive predictive coding. *arXiv* **2018**, arXiv:1807.03748.
32. He, K.; Fan, H.; Wu, Y.; Xie, S.; Girshick, R. Momentum contrast for unsupervised visual representation learning. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, Seattle, WA, USA, 13–19 June 2020.
33. Grill, J.B.; Strub, F.; Altché, F.; Tallec, C.; Richemond, P.H.; Buchatskaya, E.; Doersch, C.; Pires, B.A.; Guo, Z.D.; Azar, M.G.; et al. Bootstrap your own latent: A new approach to self-supervised learning. *arXiv* **2020**, arXiv:2006.07733.
34. Hinton, G.; Vinyals, O.; Dean, J. Distilling the knowledge in a neural network. *arXiv* **2015**, arXiv:1503.02531.
35. Paszke, A.; Gross, S.; Chintala, S.; Chanan, G.; Yang, E.; DeVito, Z.; Lin, Z.; Desmaison, A.; Antiga, L.; Lerer, A. Automatic Differentiation in Pytorch. In Proceedings of the NIPS 2017 Workshop Autodiff Submission, Long Beach, CA, USA, 28–29 October 2017.
36. Peng, X.; Usman, B.; Kaushik, N.; Hoffman, J.; Wang, D.; Saenko, K. Visda: The visual domain adaptation challenge. *arXiv* **2017**, arXiv:1710.06924.
37. LeCun, Y.; Bottou, L.; Bengio, Y.; Haffner, P. Gradient-based learning applied to document recognition. *Proc. IEEE* **1998**, *86*, 2278–2324. [CrossRef]
38. Netzer, Y.; Wang, T.; Coates, A.; Bissacco, A.; Wu, B.; Ng, A.Y. Reading digits in natural images with unsupervised feature learning. In Proceedings of the NIPS Workshop on Deep Learning and Unsupervised Feature Learning, Granada, Spain, 12–17 December 2011.
39. Liu, H.; Cao, Z.; Long, M.; Wang, J.; Yang, Q. Separate to adapt: Open set domain adaptation via progressive separation. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, Long Beach, CA, USA, 16–20 June 2019.