Class-Incremental Domain Adaptation

Jogendra Nath Kundu*, Rahul Mysore Venkatesh*, Naveen Venkat,
Ambareesh Revanur, and R. Venkatesh Babu

Video Analytics Lab, Indian Institute of Science, Bangalore

Abstract. We introduce a practical Domain Adaptation (DA) paradigm called Class-Incremental Domain Adaptation (CIDA). Existing DA methods tackle domain-shift but are unsuitable for learning novel target-domain classes. Meanwhile, class-incremental (CI) methods enable learning of new classes in absence of source training data, but fail under a domain-shift without labeled supervision. In this work, we effectively identify the limitations of these approaches in the CIDA paradigm. Motivated by theoretical and empirical observations, we propose an effective method, inspired by prototypical networks, that enables classification of target samples into both shared and novel (one-shot) target classes, even under a domain-shift. Our approach yields superior performance as compared to both DA and CI methods in the CIDA paradigm.

1 Introduction

Deep models have been shown to outperform human evaluators on image recognition tasks [15]. However, a common assumption in such evaluations is that the training and the test data distributions are alike. In the presence of a larger domain-shift [43] between the training and the test domains, the performance of deep models degrades drastically resulting from the domain-bias [18,50]. Moreover, the recognition capabilities of such models is limited to the set of learned categories, which further limits their generalizability. Thus, once a model is trained on a source training dataset (the *source* domain), it is essential to further upgrade the model to perform well in the test environment (the *target* domain).

For example, consider a *self-driving car* installed with an *object recognition model* trained on urban scenes. Such a model will underperform in rural landscapes (test environment) where objects differ in their visual appearance and the surrounding context. Moreover, the model will also misclassify objects from unseen categories (*a.k.a* target-private categories) into one of the learned classes. This is a direct result of the domain-shift between urban and rural environments. A naive approach to address this problem would be to fine-tune [28] the model on an annotated dataset drawn from the target environment. However, this is often not a practical solution as acquiring label-rich data is an expensive process. Moreover, for an efficient model upgrade, it is also imperative that the model supports adaptation to new domains and tasks, without re-training on the source

*Equal contribution.

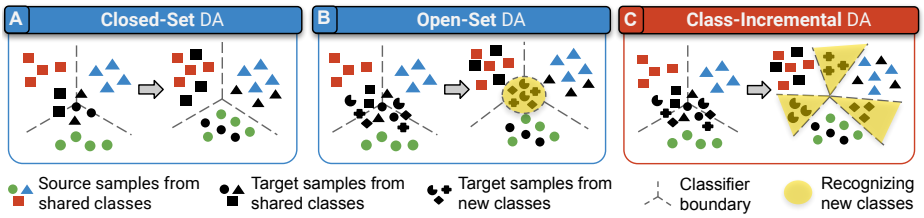

Fig. 1. Problem Setting. A) Closed-set DA assumes a shared label-set between the source and the target domains. **B)** Open-set DA rejects target samples from unseen categories into a single *unknown* class. **C)** In Class-Incremental DA, we aim to recognize both shared and new target classes by assigning a unique semantic label to each class.

training data [7,28] from scratch. Motivated by these challenges, in this paper we ask "how to effectively upgrade a trained model to the target domain?".

In the literature, this question has been long-standing. A line of work called Unsupervised Domain Adaptation (UDA) [2,3,5,21,25,31,22,51] has emerged that offers an elegant solution to the domain-shift problem. In UDA, the usual practice [10,37] is to obtain a labeled source dataset and unlabeled targets samples, to perform adaptation under the co-existence of samples from both the domains. However, most UDA methods [10,12,44,52] assume that the two domains share the same label space (as shown in Fig. 1A), making them impractical in real-world where a target domain potentially contains unseen categories (*in the self-driving car example, novel objects occur in the deployed environment*). To this end, open-set DA [1,36,24,45] and universal DA [23,54] have gained attention, where the target domain is allowed to have novel (target-private) classes not present in the source domain. These target-private samples are assigned an "*unknown*" label (see Fig. 1B). As a result, target-private samples with diverse semantic content get clustered together in a single "*unknown*" class in the latent space.

While UDA methods tackle the domain-shift problem, these require simultaneous access to both source and target domain samples, which makes them unsuitable in cases where the source training data is proprietary [24,32,34] (*e.g. in a self-driving car*), or simply unavailable during model upgrade [7,23,28]. Moreover, these methods can only detect new target categories as a single *unknown* class [36], and cannot assign individual semantic labels to such categories (Fig. 1C). Thus, these methods do not truly facilitate model upgrade (*e.g. adding new classes to the recognition model*) thereby having a limited practical use-case.

Another line of work consists of Class-Incremental (CI) learning methods [4,28,38,42,53] which aim at adding new classes to a trained model while preserving the performance on the previously learned classes. Certain methods [7] achieve this even without accessing the source training data (hereon, we call such methods as *source-free*). However, these methods are not tailored to address domain-shift (*thus, in our example, the object recognition model would still underperform in rural scenarios*). Moreover, many of these methods [4,7,41] require the target data to be labeled, which is impractical for real world applications.

To summarize, UDA and CI methods address different challenges under separate contexts and neither of them alone suffices practical scenarios. A characteristic comparison against prior arts is given in Table 1. Acknowledging this gap between the available solutions and their practical usability, in this work we introduce a new paradigm called Class-Incremental Domain Adaptation (CIDA) with the best of both worlds. While formalizing the paradigm, we draw motivation from both UDA and CI and address their limitations in CIDA.

Table 1. Characteristic comparison based on the support for *source-free* (**SF**), class-incremental (**CI**) model upgrade under domain-shift (**DA**).

Method	SF	CI	DA
DANN [11]	✗	✗	✔
OSBP [45]	✗	✗	✔
UAN [54]	✗	✗	✔
STA [29]	✗	✗	✔
LETR [33]	✗	✔	✔
E2E [4]	✗	✔	✗
iCaRL [41]	✗	✔	✗
LwF-MC [7]	✔	✔	✗
LwM [7]	✔	✔	✗
Ours	✔	✔	✔

In CIDA, we aim to adapt a source-trained model to the desired target domain in the presence of domain-shift as well as unseen classes using a minimal amount of labeled data. To this end, we propose a novel training strategy which enables a *source-free* upgrade to an unlabeled target domain by utilizing one-shot target-private samples. Our approach is motivated by prototypical networks [48] which exhibit a simpler inductive bias in the limited data regime. We now review the prior arts and identify their limitations to design a suitable approach for CIDA. Our contributions are as follows:

- We formalize a novel Domain Adaptation paradigm, Class-Incremental Domain Adaptation (CIDA), which enables the recognition of both shared and novel target categories under a domain-shift.
- We discuss the limitations of existing approaches and identify the challenges involved in CIDA to propose an effective training strategy for CIDA.
- The proposed solution is motivated by theoretical and empirical observations and outperforms both UDA and CI approaches in CIDA.

2 Background

Before formalizing the CIDA paradigm, we review the prior methods and study their limitations. In the UDA problem, we consider a labeled source domain with the label-set $\mathcal{C}_s$ and an unlabeled target domain with the label-set $\mathcal{C}_t$. The goal is to improve the task-performance on the target domain by transferring the task-relevant knowledge from the source to the target domain.

The most popular UDA approach [11,25,30,46,49,52] is to learn a predictor $h(x) = g \circ f(x)$ having a domain-agnostic feature extractor f that is common to both the domains, and a classifier g which can be learned using source supervision. These methods align the latent features $f(\cdot)$ of the two domains and use the classifier g to predict labels for the target samples. A theoretical upper bound [2] for the target-domain risk of such predictors is as follows,

$$\epsilon_t(g) \leq \epsilon_s(g) + \frac{1}{2} d_{\mathcal{H}\Delta\mathcal{H}}(s,t) + \lambda \tag{1}$$

where, given a hypothesis space $\mathcal{H}$, ϵ_s and ϵ_t denote the expected risk of the classifier $g \in \mathcal{H}$ in the source and the target domains respectively, and $d_{\mathcal{H}\Delta\mathcal{H}} = 2 \sup_{g,g' \in \mathcal{H}} |\epsilon_s(g,g') - \epsilon_t(g,g')|$ measures the distribution shift (or the domain discrepancy) between the two domains and $\lambda = \min_{g \in \mathcal{H}} \epsilon_s(g) + \epsilon_t(g)$ is a constant that measures the risk of the optimal joint classifier.

Notably, UDA methods aim to minimize the upper bound of the target risk (Eq. 1) by minimizing the distribution shift $d_{\mathcal{H}\Delta\mathcal{H}}$ in the latent space $f(\cdot)$, while preserving a low source risk ϵ_s. This works well under the closed-set assumption (*i.e.* $\mathcal{C}_s = \mathcal{C}_t$). However, in the presence of target-private samples (*i.e.* samples from $\mathcal{C}'_t = \mathcal{C}_t \setminus \mathcal{C}_s$), a direct enforcement of such constraints often degrades the performance of the model, even on the shared categories - a phenomenon known as negative transfer [35]. This is due to two factors. Firstly, a shared feature extractor (f), which is expected to generalize across two domains, acts as a bottleneck to the performance on the target domain. Secondly, a shared feature extractor enforces a common semantic granularity in the latent space ($f(\cdot)$) across both the domains. This is especially unfavorable in CIDA, where the semantic space must be modified to accommodate target-private categories (see Fig. 3).

Why are UDA methods insufficient? Certain UDA methods [29,54] tackle negative transfer by detecting the presence of target-private samples and discarding them during domain alignment. As a result, these samples (with diverse semantic content) get clustered into a single *unknown* category. While this improves the performance on the shared classes, it disturbs the semantic granularity of the latent space (*i.e.* $f(\cdot)$), making the model unsuitable for a class-incremental upgrade. This additional issue must be tackled in CIDA.

To demonstrate this effect, we employ the state-of-the-art open-set DA method STA [29] for image recognition on the Amazon $\rightarrow$ DSLR task of Office [43] dataset. A possible way to extend STA for CIDA would be to collect the target samples that are predicted as *unknown* (after adaptation) and obtain few-shot labeled samples from this set (by randomly labeling, say, 5% of the samples). One could then train a classifier using these labeled samples. We follow this approach and over 5 separate runs, we calculate the class-averaged accuracy. The model achieves an accuracy of $95.9 \pm 0.3\%$ on the shared classes, while only $17.7 \pm 3.5\%$ on the target-private classes. See Suppl. for experimental details. This clearly indicates that the adaptation disturbs the granularity of the semantic space [20], which is no more useful for discriminating among novel target categories.

Why are CI methods insufficient? Works such as [4,33,41] use an exemplary set to receive supervision for the source classes $\mathcal{C}_s$ along with labeled samples from target-private classes. [33] aims to address domain-shift using labeled samples. However, the requirement of the source data during model upgrade is a severe drawback for practical applications [28]. While [7] is *source-free*, it still assumes access to labeled target samples, which may not be viable in practical deployment scenarios. As we show in Sec. 4, these methods yield suboptimal results in the presence of limited labeled data. Nevertheless, most CI methods are not geared to tackle domain-shift. Thus, the assumption that the source-model is proficient in classifying samples in $\mathcal{C}_s$ [7], will not hold good for the target

domain. To the best of our knowledge, the most closely related CI work is [8] that uses a reinforcement-learning based framework to select source samples during one-shot learning. However, [8] assumes non-overlapping label sets ($\mathcal{C}_s \cap \mathcal{C}_t = \phi$), and does not consider the effect of negative transfer during model upgrade.

Why do we need CIDA? Prior arts independently address the problem of class-incremental learning and unsupervised adaptation in seperate contexts, by employing learning procedures specific to the problem at hand. As a result of this specificity, they are not equipped to address practical scenarios (*such as the self-driving car example* in Sec. 1). Acknowledging their limitations, we propose CIDA where the focus is to improve the performance on the target domain to achieve class-incremental recognition in the presence of domain-shift. This makes CIDA more practical and more challenging than the available DA paradigms.

What do we assume in CIDA? To realize a concrete solution, we make the following assumptions that are within the bounds of a practical DA setup. Firstly, considering that the labeled source dataset may not be readily available to perform a model upgrade, we consider the *adaptation step to be source-free*. Accordingly, we propose an effective source-model training strategy which allows *source-free* adaptation to be implemented in practice. Secondly, as the target domain may be label-deficient, we pose CIDA as an Unsupervised DA problem wherein the *target samples are unlabeled*. However, conceding that it may be impractical to discover semantics for unseen target classes in a completely unsupervised fashion, we assume that we can obtain a single labeled target sample for each target-private class $\mathcal{C}'_t$ (*one-shot target-private* samples). This can be perceived as the knowledge of new target classes that must be added during the model upgrade. Finally, the overarching objective in CIDA is to *improve the performance in the target domain* while the performance on the source domain remains secondary.

The assumptions stated above can be interpreted as follows. In CIDA, we first quantify the upgrade that is to be performed. We identify "what domain-shift is to be tackled?" by collecting unlabeled target domain samples, and determine "what new classes are to be added?" by obtaining one-shot target-private samples. This deterministic quantification makes CIDA different from UDA and CI methods, and enhances the reliability of a *source-free* adaptation algorithm. In the next section, we formalize CIDA and describe our approach to solve the problem.

3 Class-Incremental Domain Adaptation

Let $\mathcal{X}$ and $\mathcal{Y}$ be the input and the label spaces. The source and the target domains are characterized by the distributions p and q on $\mathcal{X} \times \mathcal{Y}$. We denote the set of labeled source samples as $\mathcal{D}_s = \{(\mathbf{x}_s, y_s) : (\mathbf{x}_s, y_s) \sim p\}$ with label set $\mathcal{C}_s$ and the set of unlabeled target samples as $\mathcal{D}_t = \{\mathbf{x}_t : \mathbf{x}_t \sim q_{\mathcal{X}}\}$ with label-set $\mathcal{C}_t$, where $q_{\mathcal{X}}$ denotes the marginal input distribution and $\mathcal{C}_s \subset \mathcal{C}_t$. The set of target-private classes is denoted as $\mathcal{C}'_t = \mathcal{C}_t \setminus \mathcal{C}_s$. See Suppl. for a notation table. To perform class-incremental upgrade, we are given one target sample from each target-private category $\{(\tilde{\mathbf{x}}^c_t, \tilde{y}^c_t)\}_{c \in \mathcal{C}'_t}$ (one-shot target-private samples). Further, we assume that source samples are unavailable during model upgrade [23,24,28].

Thus, the goal is to train a model on the source domain, and later, upgrade the model (address domain-shift and learn new classes) for the target domain. Accordingly, we formalize a two-stage approach as follows,

1. **Foresighted source-model training.** It is imperative that a source-trained model supports *source-free* adaptation. Thus, during source training, we aim to suppress the domain and category bias [18] that culminates from overconfident class-predictions. Specifically, we augment the model with the capability of out-of-distribution [27] detection. This step is inspired by prototypical networks that have a simpler inductive bias in the limited data regime [48]. Finally, the source-model is shipped along with prototypes as meta-data, for performing a future *source-free* upgrade.
2. **Class-Incremental DA.** During CIDA, we aim to align the target samples from shared classes with the high-source-density regions in the latent space, allowing the reuse of the source classifier. Further, we must accommodate new target classes in the latent space while preserving the semantic granularity. We achieve both these objectives by learning a target-specific latent space in which we obtain learnable centroids called *guides* that are used to gradually steer the target features into separate clusters. We theoretically argue and empirically verify that this enables a suitable ground for CIDA.

3.1 Foresighted source-model training

The architecture for the source model contains a feature extractor f_s and a $(|\mathcal{C}_s|+1)$-class classifier g_s (see Fig. 2A). We denote the latent-space by $\mathcal{U}$. A naive approach to train the source-model would be using the cross-entropy loss,

$$\mathcal{L}_{vanilla} : \mathop{\mathbb{E}}_{(\mathbf{x}_s, y_s)\sim p} l_{ce}\left(g_s \circ f_s(\mathbf{x}_s), y_s\right) \tag{2}$$

where, $\circ$ denotes composition. However, enforcing $\mathcal{L}_{vanilla}$ alone biases the model towards source domain characteristics. As a result, the model learns highly discriminative features and mis-classifies out-of-distribution samples into one of the learned categories ($\mathcal{C}_s$) with high confidence [24]. For *e.g.*, an MNIST image classifier is shown to yield a predicted class-probability of 91% on random input [16]. We argue that such effects are due to the domain and category bias culminating from the overconfident predictions. Thus, we aim to suppress this bias in the presence of the source samples for a reliable *source-free* upgrade.

We note two requirements for a source-trained model suitable for CIDA. First, we must penalize overconfident predictions [40] which is a crucial step to enable generalization over unseen target categories. This will aid in mitigating the effect of negative-transfer (discussed in Sec. 2). Second, we aim for *source-free* adaptation in CIDA, which calls for an alternative to source samples. We satisfy both these requirements using class-specific gaussian prototypes [9,48] as follows.

a) Gaussian Prototypes. We define a Gaussian Prototype for a class c as $\mathcal{P}_s^c = \mathcal{N}(\boldsymbol{\mu}_s^c, \boldsymbol{\Sigma}_s^c)$ where $\boldsymbol{\mu}_s^c$ and $\boldsymbol{\Sigma}_s^c$ are the mean and the covariance obtained over the features $f(\mathbf{x}_s)$ for samples $\mathbf{x}_s$ in class c. In other words, a Gaussian

Prototype is a multivariate Gaussian prior defined for each class in the latent space $\mathcal{U}$. Similarly, a global Gaussian Prototype is defined as $\mathcal{P}_s = \mathcal{N}(\boldsymbol{\mu}_s, \boldsymbol{\Sigma}_s)$, where $\boldsymbol{\mu}_s$ and $\boldsymbol{\Sigma}_s$ are calculated over the features $f_s(\mathbf{x}_s)$ for all source samples $\mathcal{D}_s$. We hypothesize that at the $\mathcal{U}$-space, we can approximate the class semantics using these Gaussian priors which can be leveraged for *source-free* adaptation.

To ensure that this Gaussian approximation is accurate, we explicitly enforce the source features to attain a higher affinity towards these class-specific Gaussian priors. We refer to this as the *class separability* objective defined as,

$$\mathcal{L}_{s1} : \mathop{\mathbb{E}}_{(\mathbf{x}_s, y_s)\sim p} -\log \Big(\exp\left(\mathcal{P}_s^{y_s}(\mathbf{u}_s)\right) / \sum_{c\in\mathcal{C}_s} \exp\left(\mathcal{P}_s^{c}(\mathbf{u}_s)\right) \Big) \tag{3}$$

where $\mathbf{u}_s = f(\mathbf{x}_s)$, and the term inside the logarithm is the posterior probability of a feature $\mathbf{u}_s$ corresponding to its class y_s (obtained as the softmax over likelihoods $\mathcal{P}_s^c(\mathbf{u}_s)$). In effect, $\mathcal{L}_{s1}$ drives the latent space to form well-separated, compact clusters for each class $c \in \mathcal{C}_s$. We verify in Sec. 4 that compact clusters enhance the reliability of a *source-free* model upgrade, where the clusters must rearrange to attain a semantic granularity suitable for the target domain.

b) Negative Training. While $\mathcal{L}_{s1}$ enforces well-separated feature clusters, it does not ensure tight decision boundaries, without which the classifier misclassifies out-of-distribution (OOD) samples [27] with high confidence. This overconfidence issue must be resolved to effectively learn new target categories. Certain prior works [55] suggest that a Gaussian Mixture Model based likelihood threshold could effectively detect OOD samples. We argue that additionally, the classifier g_s should also be capable of assigning a low confidence to OOD samples [27], forming tight decision boundaries around the source clusters (as in Fig. 2A).

We leverage the Gaussian Prototypes to generate *negative* feature samples to model the low-source-density (OOD) region. The *negative* samples are denoted as $\mathcal{D}_n = \{(\mathbf{u}_n, y_n) : (\mathbf{u}_n, y_n) \sim r\}$ where r is the distribution of the OOD regime. More specifically, we obtain the samples $\mathbf{u}_n$ from the global Gaussian Prototype $\mathcal{P}_s$ which are beyond 3-σ confidence interval of all class-specific Gaussian Prototypes $\mathcal{P}_s^c$ (see Suppl. for an algorithm). These *negative* samples correspond to the $(|\mathcal{C}_s|+1)^{\text{th}}$ category and the classifier g_s is trained to assign a low confidence to such samples (see Fig. 2A). Thus, the cross-entropy loss in Eq. 2 is modified as:

$$\mathcal{L}_{s2} : \mathop{\mathbb{E}}_{(\mathbf{x}_s, y_s)\sim p} l_{ce}(g_s \circ f_s(\mathbf{x}_s), y_s) \quad + \mathop{\mathbb{E}}_{(\mathbf{u}_n, y_n)\sim r} l_{ce}(g_s(\mathbf{u}_n), y_n) \tag{4}$$

By virtue of $\mathcal{L}_{s2}$, the classifier g_s assigns a high source-class confidence to samples in $\mathcal{D}_s$, and a low source-class confidence to samples in $\mathcal{D}_n$. Thus, g_s learns compact decision boundaries (as shown in Fig. 2A).

c) Optimization. We train $\{f_s, g_s\}$ via alternate minimization of $\mathcal{L}_{s1}$ and $\mathcal{L}_{s2}$ using Adam [19] optimizers (see Suppl.). Effectively, the total loss $\mathcal{L}_s = \mathcal{L}_{s1} + \mathcal{L}_{s2}$ enforces the Cluster Assumption at the $\mathcal{U}$-space (via $\mathcal{L}_{s1}$) that enhances the model's generalizability [6,14], and, mitigates the overconfidence issue (via $\mathcal{L}_{s2}$) thereby reducing the discriminative bias towards the source domain. We update the Gaussian Prototypes and the *negative* samples at the end of each epoch.

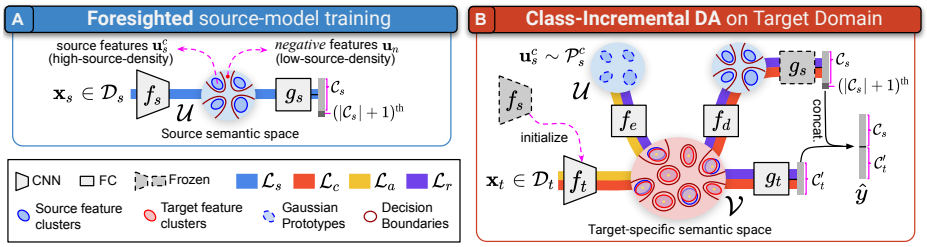

Fig. 2. Our Approach. A) The source-model is trained with an additional $(|\mathcal{C}_s|+1)^{\text{th}}$ class representing out-of-distribution (OOD) region. **B)** During CIDA, we learn a target-specific feature extractor f_t (to minimize domain-shift) and classifier g_t (to learn $\mathcal{C}_t'$). The adaptation process aligns the shared classes, and separates the target-private classes. Colored lines represent the gradient pathway for each loss.

Once trained, the source-model is ready to be shipped along with the Gaussian Prototypes as meta-data. Note, in contrast to source data, Gaussian Prototypes are cheap and can be readily shared (similar to BatchNorm [17] statistics).

3.2 Class-Incremental DA on the Target Domain

Following the CIDA paradigm during the model upgrade, we have access to a source model $\{f_s, g_s\}$ and its meta-data (Gaussian Prototypes $\mathcal{P}_s^c$), unlabeled target samples $\mathcal{D}_t$, and one-shot target-private samples $\{(\tilde{\mathbf{x}}_t^c, \tilde{y}_t^c)\}_{c\in\mathcal{C}_t'}$. We now formalize an approach that tightens the target risk bound (Eq. 1) exploiting a foresighted source-model trained using $\mathcal{D}_s \cup \mathcal{D}_n$. Recall that the bound comprises of three terms - source risk (ϵ_s), distribution shift ($d_{\mathcal{H}\Delta\mathcal{H}}$) and the constant λ.

a) Learning target features. A popular strategy for UDA is to learn domain-agnostic features [29,45,47]. However, as argued in Sec. 2, in CIDA we must learn a target-specific latent space (annotated as $\mathcal{V}$ in Fig. 2B) which attains a semantic granularity suitable for the target domain. To this end, we introduce a target-specific feature extractor f_t that is initialized from f_s. Informally, this process "initializes the $\mathcal{V}$-space from the $\mathcal{U}$-space". Thereafter, we gradually rearrange the feature clusters in the $\mathcal{V}$-space to learn suitable target semantics. To receive stable gradients, g_s is kept frozen throughout adaptation. Further, we introduce a classifier g_t to learn the target-private categories $\mathcal{C}_t'$ (see Fig. 2B).

b) Domain projection. The key to effectively learn target-specific semantics is to establish a transit mechanism between the $\mathcal{U}$-space (capturing the semantics of the learned classes $\mathcal{C}_s$) and the $\mathcal{V}$-space (where $\mathcal{C}_t$ must be learned). We address this using the domain projection networks $f_e : \mathcal{U} \to \mathcal{V}$ and $f_d : \mathcal{V} \to \mathcal{U}$. Specifically, we obtain feature samples from the Gaussian Prototypes $\mathbf{u}_s^c \sim \mathcal{P}_s^c$ for each class $c \in \mathcal{C}_s$ (called as proxy-source samples). Thereafter, we formalize the following losses to minimize the source risk (ϵ_s in Eq. 1) during adaptation,

$$\mathcal{L}_{r1} : \mathop{\mathbb{E}}_{\mathbf{u}_s^c\sim\mathcal{P}_s^c} l_{ce}(\hat{y}(\mathbf{u}_s^c), c) \quad ; \quad \mathcal{L}_{r2} : \mathop{\mathbb{E}}_{\mathbf{u}_s^c\sim\mathcal{P}_s^c} l_2(f_d \circ f_e(\mathbf{u}_s^c), \mathbf{u}_s^c)^2 \tag{5}$$

where l_2 is the euclidean distance and the output $\hat{y}(\cdot)$ is the concatenation (Fig. 2B) of logits pertaining to $\mathcal{C}_s$ ($g_s \circ f_d \circ f_e(\mathbf{u}_s^c)|_{c\in\mathcal{C}_s}$) and those of $\mathcal{C}_t$ ($g_t \circ f_e(\mathbf{u}_s^c)$). The total loss $\mathcal{L}_r = \mathcal{L}_{r1} + \mathcal{L}_{r2}$ acts as a regularizer, where $\mathcal{L}_{r1}$ preserves the semantics of the learned classes in the $\mathcal{V}$-space, while $\mathcal{L}_{r2}$ prevents degenerate solutions. In Sec. 4, we show that $\mathcal{L}_r$ mitigates catastrophic forgetting [13] (by minimizing ϵ_s in Eq. 1) that would otherwise occur in a *source-free* scenario.

c) Semantic alignment using guides. We aim to align target samples from shared classes $\mathcal{C}_s$ with the high source-density region (proxy-source samples) and disperse the target-private samples away into the low source-density region (*i.e.* the *negative* regime). Note, as the source model was trained on $\mathcal{D}_s$ augmented with $\mathcal{D}_n$, this process would entail the minimization of $d_{\mathcal{H}\Delta\mathcal{H}}$ (Eq. 1) measured between the target and the augmented source distributions in the $\mathcal{V}$-space.

To achieve this, we obtain a set of $|\mathcal{C}_t|$ *guides* ($\mathbf{v}_g^c$) that act as representative centers for each class $c \in \mathcal{C}_t$ in the $\mathcal{V}$-space. We model the euclidean distance to a *guide* as a measure of class confidence, using which we can assign a pseudo class-label [26] to the target samples. These pseudo-labels can be leveraged to rearrange the target features into separate compact clusters (Fig. 3B-F). Note that $\mathcal{L}_{s1}$ (*class separability* objective) enforced during the source training is crucial to improve the reliability of the *guides* during adaptation.

We consider the features of the one-shot target-private samples $f_t(\tilde{\mathbf{x}}_s^c)$ as the *guides* for the target-private classes. Further, since $\mathcal{V}$ is initialized from $\mathcal{U}$, one might consider the source class-means $\boldsymbol{\mu}_s^c$ as the *guides* for the shared classes. However, we found that a fixed *guide* representation (*e.g.* $\boldsymbol{\mu}_s^c$) hinders the placement of target-private classes. Thus, we obtain trainable *guides* for the shared classes as $f_e(\boldsymbol{\mu}_s^c)$, by allowing f_e to modify the placement of the *guides* in the $\mathcal{V}$-space (Fig. 3). This allows all the *guides* to rearrange and steer the target clusters in the $\mathcal{V}$-space as the training proceeds. To summarize, the *guides* are computed as $\mathbf{v}_g^c = f_e(\boldsymbol{\mu}_s^c)\ \forall c \in \mathcal{C}_s$, and, $\mathbf{v}_g^c = f_t(\tilde{\mathbf{x}}_t^c)\ \forall c \in \mathcal{C}_t'$.

To minimize $d_{\mathcal{H}\Delta\mathcal{H}}$ (Eq. 1), we must first detect the target-shared and target-private samples and then perform feature alignment. To this end, for a target feature $\mathbf{v}_t = f_t(\mathbf{x}_t)$, we obtain the euclidean distance d to its nearest *guide*, and assign a pseudo-label k corresponding to the class represented by the *guide* as, $d = \min_{c\in\mathcal{C}_t}\ l_2(\mathbf{v}_t, \mathbf{v}_g^c)$, and, $k = \arg\min_{c\in\mathcal{C}_t}\ l_2(\mathbf{v}_t, \mathbf{v}_g^c)$.

Using pseudo-labeled samples we obtain Gaussian Prototypes $\mathcal{P}_t^c = \mathcal{N}(\boldsymbol{\mu}_t^c, \boldsymbol{\Sigma}_t^c)$ $\forall c \in \mathcal{C}_t$ in the $\mathcal{V}$-space (as done in Sec. 3.1a), and enforce the *class separability* objective. Further, for each *guide* $\mathbf{v}_g^c$ ($c \in \mathcal{C}_t$), we define a set $\mathcal{B}_t^c$ of the closest n-percent target samples based on the distance d (see Suppl. for the algorithm). Notionally, $\mathcal{B}_t^c$ represents the confident target samples which are then pulled closer to $\mathbf{v}_g^c$. These two losses are defined as,

$$\mathcal{L}_{a1} : \mathop{\mathbb{E}}_{\mathbf{x}_t\sim q_{\mathcal{X}}} -\log\Big(\exp\left(\mathcal{P}_t^k(\mathbf{v}_t)\right)/\sum_{c\in\mathcal{C}_t}\exp\left(\mathcal{P}_t^c(\mathbf{v}_t)\right)\Big)\ ;\ \mathcal{L}_{a2} : \mathop{\mathbb{E}}_{\mathbf{x}_t\sim\mathcal{B}_t^c} l_2(\mathbf{v}_t, \mathbf{v}_g^c)^2 \quad (6)$$

The total adaptation loss is $\mathcal{L}_a = \mathcal{L}_{a1} + \mathcal{L}_{a2}$. Overall, $\mathcal{L}_a$ pulls the target-shared samples towards the high source-density region and separates the target-private

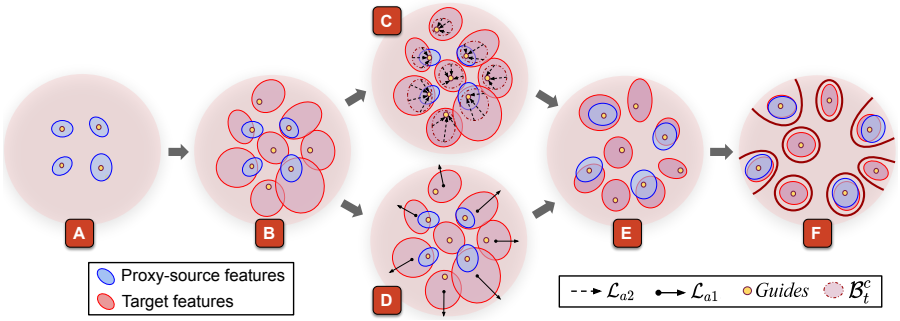

Fig. 3. Semantic Alignment using *guides* in $\mathcal{V}$-space. A) $\mathcal{V}$ is initialized from $\mathcal{U}$. **B)** Domain-shift between target and proxy-source features. **C)** $\mathcal{L}_{a2}$ steers the confident target samples ($\mathcal{B}_t^c$) towards the corresponding *guides* ($\mathbf{v}_g^c$). **D)** $\mathcal{L}_{a1}$ separates the clusters, making space for target-private classes. **E)** This rearrangement aligns the shared classes while separating the target-private classes. **F)** The classifiers $\{g_s, g_t\}$ recognize all target classes by assigning an individual semantic label to each class.

clusters away from the high source-density regions (Fig. 3B-E). This results in a superior alignment thereby minimizing $d_{\mathcal{H}\Delta\mathcal{H}}$. Particularly, the separation caused by $\mathcal{L}_{a1}$ minimizes the negative influence of target-private samples during adaptation, thereby preventing negative transfer [29]. $\mathcal{L}_{a2}$ ensures compact feature clusters which aids in preserving the semantic granularity across the target classes.

d) Learning target-private classes. Finally, to learn new target classes, we apply cross-entropy loss on the confident target samples $\mathcal{B}_t^c$ ($c \in \mathcal{C}_t$) as,

$$\mathcal{L}_c : \mathop{\mathbb{E}}_{\mathbf{x}_t \sim \mathcal{B}_t^c} l_{ce}(\hat{y}(\mathbf{v}_t), c) \tag{7}$$

where the output $\hat{y}(\cdot)$ is obtained similar to that in Eq. 5, by concatenating the logits $g_s \circ f_d(\mathbf{v}_t)|_{c \in \mathcal{C}_s}$ and $g_t(\mathbf{v}_t)$. We verify in Suppl. that the precision of pseudo-labels for target samples in $\mathcal{B}_t^c$ is high. Thus, the loss $\mathcal{L}_c$ along with $\mathcal{L}_{r1}$ can be viewed as conditioning the classifier $\{g_s, g_t\}$ to deliver a performance close to that of the optimal joint classifier (with the minimal risk λ).

e) Optimization. We pre-train $\{f_e, f_d\}$ to a near-identity function with the losses $l_2(\mathbf{u}_s^c, f_e(\mathbf{u}_s^c))^2$ and $l_2(\mathbf{u}_s^c, f_d(\mathbf{u}_s^c))^2$, where $\mathbf{u}_s \sim \mathcal{P}_s^c$ $\forall c \in \mathcal{C}_s$ and l_2 is the euclidean distance (similar to an auto-encoder). The total loss employed is $\mathcal{L}_t = \mathcal{L}_a + \mathcal{L}_c + \mathcal{L}_r$, which tightens the bound in Eq. 1 as argued above, yielding a superior adaptation guarantee. Instead of directly enforcing $\mathcal{L}_t$ at each iteration, we alternatively optimize each loss using separate Adam [19] optimizers in a round robin fashion (*i.e.* we cycle through the losses $\{\mathcal{L}_{a1}, \mathcal{L}_{a2}, \mathcal{L}_c, \mathcal{L}_{r1}, \mathcal{L}_{r2}\}$ and minimize a single loss at each iteration). Since each optimizer minimizes its corresponding loss function independently, the gradients pertaining to each loss are adaptively scaled via the higher order moments [19]. This allows us to avoid hyperparameter search for loss scaling. See Suppl. for the training algorithm.

4 Experiments

We conduct experiments on three datasets. **Office** [43] is the most popular benchmark containing 31 classes across 3 domains - Amazon (**A**), DSLR (**D**) and Webcam (**W**). **VisDA** [39] contains 12 classes with 2 domains - Synthetic (**Sy**) and Real (**Re**) with a large domain-shift. **Digits** dataset is composed of MNIST (**M**), SVHN (**S**) and USPS (**U**) domains. See Suppl. for label-set details.

a) Evaluation. We consider two setups for target-private samples - i) one-shot, and, ii) few-shot (5% labeled). In both cases, we report the mean target accuracy over $\mathcal{C}_t$ (ALL) and $\mathcal{C}'_t$ (PRIV), over 5 separate runs (with randomly chosen one-shot and few-shot samples). We compare against prior UDA methods DANN [11], OSBP [45], UAN [54], STA [29], and CI methods E2E [4], LETR [33], iCaRL [41], LwF-MC [7], LwM [7]. To evaluate UDA methods in CIDA, we collect the target samples predicted as *unknown* after adaptation. We annotate a few of these samples following the few-shot setting, and train a separate target-private classifier (TPC) similar in architecture to g_t. At test time, a target sample is first classified into $\mathcal{C}_s \cup \{unknown\}$, and if predicted as *unknown*, it is further classified by the target-private classifier. We evaluate the prior arts only in the few-shot setting since they require labeled samples for reliable model upgrade.

b) Implementation. See Suppl. for the architectural details and an overview of the training algorithms for each stage. A learning rate of 0.0001 is used for the Adam [19] optimizers. For the source-model training, we use equal number of source and *negative* samples per batch. For adaptation, we set $n = 20\%$ for confident samples. At test time, the prediction for a target sample $\mathbf{x}_t$ is obtained as arg max over the logits pertaining to $\mathcal{C}_s$ $(g_s \circ f_d \circ f_t(\mathbf{x}_t)|_{c\in\mathcal{C}_s})$ and $\mathcal{C}_t$ $(g_t \circ f_t(\mathbf{x}_t))$.

4.1 Discussion

a) Baseline Comparisons. To empirically verify the effectiveness of our approach, we implement the following baselines. See Suppl. for illustrations of the architectures. The results are summarized in Table 2.

i) *Ours-a*: To corroborate the need for a target-specific feature space, we remove $\{f_e, f_d\}$, and discard the loss $\mathcal{L}_{r2}$. Here, the $\mathcal{V}$-space is common to both the target and the proxy-source samples. Thus, the *guides* for the shared classes are the fixed class-means ($\boldsymbol{\mu}_s^c$), and the only trainable components are f_t and g_t. In doing so, we force the target classes to acquire the semantics of the source domain which hinders the placement of target-private classes and degrades the target-private accuracy. However, in our approach (*Ours*), trainable *guides* allow the rearrangement of features which effectively minimizes the $d_{\mathcal{H}\Delta\mathcal{H}}$ (in Eq. 1).

ii) *Ours-b*: To study the regularization of the sampled proxy-source features, we modify our approach by removing $\mathcal{L}_r$. We observe a consistent degradation in performance resulting from a lower target-shared accuracy. This verifies the role of $\mathcal{L}_r$ in mitigating catastrophic forgetting (*i.e.* by minimizing ϵ_s in Eq. 1).

iii) *Ours-c*: We modify our approach by removing $\mathcal{L}_{a2}$ that produces compact target clusters. We find that the target-private accuracy decreases, verifying the need for compact clusters to preserve the semantic granularity across the

Table 2. Baseline Comparisons. Results on **Office**, **Digits** and **VisDA** for CIDA using **one-shot** target-private samples.

Method	**Office** $\|\mathcal{C}_s\| = 20, \|\mathcal{C}_t\| = 31$													
	A→D		**A→W**		**D→A**		**D→W**		**W→A**		**W→D**		**Avg**	
	ALL	PRIV	ALL	PRIV	ALL	PRIV	ALL	PRIV	ALL	PRIV	ALL	PRIV	ALL	PRIV
Ours-a	69.1	66.7	58.2	55.9	60.6	58.1	70.2	68.9	59.4	57.6	80.4	80.0	66.4	64.5
Ours-b	69.5	71.9	58.2	60.4	60.9	61.1	73.3	75.6	61.1	62.3	81.7	82.8	67.5	69.0
Ours-c	70.4	70.1	60.4	58.7	61.7	60.4	75.4	73.8	61.7	61.2	85.9	84.8	69.3	68.1
Ours-d	73.7	73.6	64.8	64.6	64.4	63.9	80.9	80.7	63.6	61.8	90.1	89.6	72.9	72.4
Ours	73.3	73.1	63.6	62.6	64.1	64.3	80.3	79.4	63.7	62.8	89.5	88.4	72.4	71.8

Method	**Digits** $(\|\mathcal{C}_s\| = 5, \|\mathcal{C}_t\| = 10)$								**VisDA** $(\|\mathcal{C}_s\| = 6, \|\mathcal{C}_t\| = 12)$					
	S→M		**M→U**		**U→M**		**Avg**		**Sy→Re**		**Re→Sy**		**Avg**	
	ALL	PRIV	ALL	PRIV	ALL	PRIV	ALL	PRIV	ALL	PRIV	ALL	PRIV	ALL	PRIV
Ours-a	41.2	38.7	64.9	63.5	63.1	62.6	56.4	54.9	52.3	51.4	50.9	49.6	51.6	50.5
Ours-b	42.4	42.9	66.1	67.2	63.9	64.7	57.5	58.3	53.1	53.6	51.1	51.4	52.1	52.5
Ours-c	44.5	43.8	69.4	69.3	65.3	64.5	59.7	59.2	54.3	54.0	52.3	51.9	53.3	52.9
Ours-d	46.5	45.3	72.7	72.2	69.4	68.8	62.9	62.1	56.6	56.3	55.8	55.4	56.2	55.8
Ours	46.4	45.7	72.5	71.6	69.4	68.6	62.8	61.9	56.4	56.3	55.8	55.7	56.1	56.0

target classes. Note, *Ours-c* (having trainable *guides* for $\mathcal{C}_s$) outperforms *Ours-a* (having frozen *guides* for $\mathcal{C}_s$), even in the absence of $\mathcal{L}_{a2}$.

iv) *Ours-d:* To establish the reliability of the Gaussian Prototypes, we perform CIDA using the source dataset, *i.e.* using the features $f_s(\mathbf{x}_s)$ instead of the sampled proxy-source features. The performance is similar to *Ours*, confirming the efficacy of the Gaussian Prototypes in modelling the source distribution. This is owed to $\mathcal{L}_{s1}$ that enhances the reliability of the Gaussian approximation.

b) Comparison against prior arts. We compare against prior UDA and CI approaches in Table 3. Further, we run a variation of our approach with few-shot (5% labeled) target-private samples (*Ours**), where the *guides* for $\mathcal{C}'_t$ are obtained as the class-wise mean features of the few-shot samples.

UDA methods exploit unlabeled target samples but require access to labeled source samples during adaptation. They achieve a low target-private (PRIV) accuracy owing to the loss of semantic granularity. This effect is evident in open-set methods, where target-private samples are forced to be clustered into a single *unknown* class. However in DANN and UAN, such a criterion is not enforced, instead a target sample is detected as *unknown* using confidence thresholding. Thus, DANN and UAN achieve a higher PRIV accuracy than STA and OSBP.

The performance of most CI methods in CIDA is limited due to the inability to address domain-shift. LETR, E2E and iCaRL require labeled samples from both the domains during the model upgrade. E2E exploits these labeled samples to re-train the source-trained model with all classes ($\mathcal{C}_t$). However, the need to generalize across two domains degrades the performance on the target domain where target-shared samples are unlabeled. In contrast, LwM and LwF-MC learn a separate target model, by employing a distillation loss using the target samples. However, distillation is not suitable under a domain-shift since the source model is biased towards the source domain characteristics that cannot be generalized for the target domain. In LETR, the global domain statistics across the two

Table 3. Comparison against prior arts. Results on **Office** ($|\mathcal{C}_s| = 10, |\mathcal{C}_t| = 20$) for CIDA. Unsup. denotes the method is unsupervised (on target). SF denotes model upgrade is *source-free*. Note, non-*source-free* methods access labeled source data. Results are grouped based on access to i) few-shot and ii) one-shot target-private samples.

Method	SF	Unsup.	A→D		A→W		D→A		D→W		W→A		W→D		Avg	
			ALL	PRIV	ALL	PRIV	ALL	PRIV	ALL	PRIV	ALL	PRIV	ALL	PRIV	ALL	PRIV
Using few-shot target-private samples (5% labeled)																
DANN+TPC	✗	✔	54.3	21.4	52.4	16.7	48.2	19.8	61.4	24.3	57.9	21.1	56.5	38.0	55.1	23.6
OSBP+TPC	✗	✔	51.6	13.9	49.2	9.5	58.8	14.3	55.5	18.4	49.1	13.6	64.0	29.1	54.7	14.5
STA+TPC	✗	✔	56.6	17.7	51.2	10.2	54.8	16.5	59.6	21.8	54.7	15.9	67.4	35.4	57.4	19.6
UAN+TPC	✗	✔	56.2	24.4	54.8	21.2	57.3	24.7	62.6	29.5	59.2	28.9	68.4	42.8	59.8	28.6
iCaRL	✗	✗	63.6	63.2	54.3	53.8	56.9	56.1	65.4	65.2	57.5	56.8	76.8	77.5	62.4	62.1
E2E	✗	✗	64.2	61.9	55.6	53.2	58.8	58.4	66.3	66.5	57.9	56.6	76.5	73.2	63.2	60.8
LETR	✗	✔	71.3	68.5	58.4	57.6	58.2	58.4	70.3	69.8	62.0	60.7	84.2	82.9	67.4	66.3
LwM	✔	✗	66.5	66.2	56.3	55.9	57.6	56.8	68.4	68.3	59.8	59.4	78.4	78.1	64.5	61.9
LwF-MC	✔	✗	64.3	63.8	55.6	55.1	55.5	55.7	67.6	67.7	59.4	59.0	77.3	76.9	63.3	63.0
*Ours**	✔	✔	**78.8**	**74.3**	**70.1**	**69.8**	**66.9**	**67.1**	**85.0**	**84.6**	**67.2**	**65.3**	**90.4**	**90.8**	**76.4**	**75.3**
Using one-shot target-private samples																
Ours-a	✔	✔	67.4	64.1	56.2	53.4	60.1	57.8	69.2	68.3	57.1	55.6	77.9	76.5	65.0	62.3
Ours-b	✔	✔	68.4	70.2	57.5	59.6	60.6	60.8	70.9	72.4	58.4	58.7	79.8	80.2	65.9	67.0
Ours-c	✔	✔	70.0	69.3	59.5	57.4	61.5	60.2	73.1	71.4	61.8	60.1	82.3	81.1	68.0	66.6
Ours	✔	✔	**72.2**	**72.6**	**62.1**	**62.0**	**62.6**	**61.8**	**78.5**	**78.7**	**62.1**	**62.4**	**87.8**	**87.6**	**70.7**	**70.8**

domains are aligned. However, such a global alignment is prone to the negative influence of target-private samples which limits its performance.

Our method addresses these limitations and outperforms both UDA and CI methods. The foresighted source-model training suppresses domain and category bias by addressing the overconfidence issue. Then, a gradual rearrangement of features in a target-specific semantic space allows the learning of target-private classes while preserving the semantic granularity. Furthermore, the regularization from the proxy-source samples mitigates catastrophic forgetting. Thus our approach achieves a more stable performance in CIDA, even in the challenging *source-free* scenario. See Suppl. for a discussion from the theoretical perspective.

c) Effect of class separability objective. We run an ablation on the **A→D** task (Table 3) without enforcing $\mathcal{L}_{s1}$ during the source-model training. The accuracy post adaptation is 68.6% (PRIV = 70.4%) as compared to 72.2% (PRIV = 72.6%) in *Ours*. This suggests that the *class separability* objective (enforcing the Cluster Assumption) helps in generalization to the target domain.

d) Effect of negative training. On the **A→D** task (Table 3), a source-model trained with *negative* samples ($\mathcal{D}_s \cup \mathcal{D}_n$) achieves a source accuracy of 96.7%, while that trained without *negative* samples yields 96.9%. Thus, there is no significant drop on the source performance due to negative training. However, this aids in generalizing the model to novel target classes. Specifically, a source-model trained with *negative* samples (*Ours*) yields 72.2% (PRIV = 72.6%) after adaptation, while that without *negative* training achieves 67.4% (PRIV = 62.3%) after adaptation. The performance gain in *Ours* is attributed to the mitigation of the overconfidence issue thereby reliably classifying target-private samples.

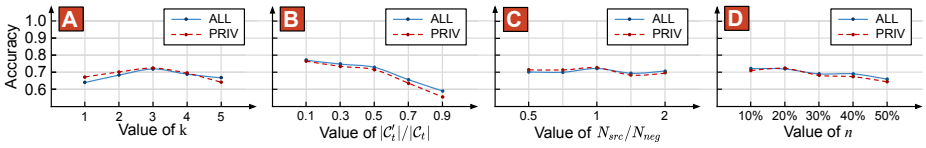

Fig. 4. Sensitivity for A→D task (Office). A) Confidence interval k-σ for *negative* sampling. **B)** Fraction of target-private classes $|\mathcal{C}_t'|/|\mathcal{C}_t|$ during CIDA. **C)** Batch size ratio of source (N_{src}) and *negative* (N_{neg}) samples during source-training. **D)** Percentage of confident target samples for $\mathcal{B}_t^c$ during CIDA. Note the scale of the axes.

e) Sensitivity to hyperparameters. In Fig. 4, we plot the target accuracy post adaptation for various hyperparameter values for the task **A**→**D**. Empirically, we found that a 3-σ confidence interval for *negative* sampling was most effective in capturing the source distribution (Fig. 4A). We choose an equal number of source (N_{src}) and *negative* (N_{neg}) samples in a batch during source training to avoid the bias caused by imbalanced data. Fig. 4C shows the sensitivity to the batch size ratio N_{src}/N_{neg}. Further, the hyperparameter n is marginally stable around $n = 20\%$ (Fig. 4D) which was used across all experiments. Finally, the trend in Fig. 4B is a result of the challenging one-shot setting.

f) Two-step model upgrade. We extend our approach to perform two-step model upgrade under CIDA on **Office** (See Suppl. for details). First a source model ($\{f_s, g_s\}$) is trained on the 10 classes of Amazon (**A**) which is upgraded to the 20 classes of DSLR (**D**) thereby learning $\{f_t, g_t, f_e, f_d\}$. We upgrade this DSLR-specific model to the Webcam (**W**) domain, having 20 classes shared with (**A+D**), and 11 new classes. This is done by learning feature extractor f_{t_2}, classifier g_{t_2}, and domain projection networks $\{f_{e_2}, f_{d_2}\}$ learned between the latent spaces of f_t and f_{t_2}. We observe an accuracy of 79.9% on **W**, which is close to that obtained by directly adapting from 20 classes of DSLR to 31 classes in Webcam (80.3%, Table 2). This corroborates the practical applicability of our approach to multi-step model upgrades. See Suppl. for a detailed discussion.

5 Conclusion

We proposed a novel Domain Adaptation paradigm (CIDA) addressing class-incremental learning in the presence of domain-shift. We studied the limitations of prior approaches in the CIDA paradigm and proposed a two-stage approach to address CIDA. We presented a foresighted source-model training that facilitates a *source-free* model upgrade. Further, we demonstrated the efficacy of a target-specific semantic space, learned using trainable *guides*, that preserves the semantic granularity across the target classes. Finally, our approach shows promising results on multi-step model upgrades. As a future work, the framework can be extended to a scenario where a series of domain-shifts and task-shifts are observed.

Acknowledgement. This work is supported by a Wipro PhD Fellowship and a grant from Uchhatar Avishkar Yojana (UAY, IISC_010), MHRD, Govt. of India.

References

1. Baktashmotlagh, M., Faraki, M., Drummond, T., Salzmann, M.: Learning factorized representations for open-set domain adaptation. In: ICLR (2019) 2
2. Ben-David, S., Blitzer, J., Crammer, K., Kulesza, A., Pereira, F., Vaughan, J.W.: A theory of learning from different domains. Machine learning **79**(1-2), 151–175 (2010) 2, 3
3. Ben-David, S., Blitzer, J., Crammer, K., Pereira, F.: Analysis of representations for domain adaptation. In: NeurIPS (2007) 2
4. Castro, F.M., Marin-Jimenez, M.J., Guil, N., Schmid, C., Alahari, K.: End-to-end incremental learning. In: ECCV (2018) 2, 3, 4, 11
5. Chang, W.G., You, T., Seo, S., Kwak, S., Han, B.: Domain-specific batch normalization for unsupervised domain adaptation. In: CVPR (2019) 2
6. Chapelle, O., Zien, A.: Semi-supervised classification by low density separation. In: AISTATS (2005) 7
7. Dhar, P., Singh, R.V., Peng, K.C., Wu, Z., Chellappa, R.: Learning without memorizing. In: CVPR (2019) 2, 3, 4, 11
8. Dong, N., Xing, E.P.: Domain adaption in one-shot learning. In: ECML-PKDD (2018) 5
9. Fort, S.: Gaussian prototypical networks for few-shot learning on omniglot. arXiv preprint arXiv:1708.02735 (2017) 6
10. Ganin, Y., Lempitsky, V.: Unsupervised domain adaptation by backpropagation. In: ICML (2015) 2
11. Ganin, Y., Ustinova, E., Ajakan, H., Germain, P., Larochelle, H., Laviolette, F., Marchand, M., Lempitsky, V.: Domain-adversarial training of neural networks. JMLR **17**(1), 2096–2030 (2016) 3, 11
12. Gong, B., Shi, Y., Sha, F., Grauman, K.: Geodesic flow kernel for unsupervised domain adaptation. In: CVPR (2012) 2
13. Goodfellow, I.J., Mirza, M., Xiao, D., Courville, A., Bengio, Y.: An empirical investigation of catastrophic forgetting in gradient-based neural networks. arXiv preprint arXiv:1312.6211 (2013) 9
14. Grandvalet, Y., Bengio, Y.: Semi-supervised learning by entropy minimization. In: NeurIPS (2005) 7
15. He, K., Zhang, X., Ren, S., Sun, J.: Delving deep into rectifiers: Surpassing human-level performance on imagenet classification. In: ICCV (2015) 1
16. Hendrycks, D., Gimpel, K.: A baseline for detecting misclassified and out-of-distribution examples in neural networks. In: ICLR (2017) 6
17. Ioffe, S., Szegedy, C.: Batch normalization: Accelerating deep network training by reducing internal covariate shift. In: ICML (2015) 8
18. Khosla, A., Zhou, T., Malisiewicz, T., Efros, A.A., Torralba, A.: Undoing the damage of dataset bias. In: ECCV (2012) 1, 6
19. Kingma, D.P., Ba, J.L.: Adam: A method for stochastic optimization. In: ICLR (2014) 7, 10, 11
20. Kundu, J.N., Gor, M., Agrawal, D., Babu, R.V.: GAN-Tree: An incrementally learned hierarchical generative framework for multi-modal data distributions. In: ICCV (2019) 4
21. Kundu, J.N., Lakkakula, N., Babu, R.V.: UM-Adapt: Unsupervised multi-task adaptation using adversarial cross-task distillation. In: ICCV (2019) 2
22. Kundu, J.N., Uppala, P.K., Pahuja, A., Babu, R.V.: Adadepth: Unsupervised content congruent adaptation for depth estimation. In: CVPR (2018) 2

23. Kundu, J.N., Venkat, N., M V, R., Babu, R.V.: Universal source-free domain adaptation. In: CVPR (2020) 2, 5
24. Kundu, J.N., Venkat, N., Revanur, A., M V, R., Babu, R.V.: Towards inheritable models for open-set domain adaptation. In: CVPR (2020) 2, 5, 6
25. Kuroki, S., Charoenphakdee, N., Bao, H., Honda, J., Sato, I., Sugiyama, M.: Unsupervised domain adaptation based on source-guided discrepancy. In: AAAI (2019) 2, 3
26. Lee, D.H.: Pseudo-label: The simple and efficient semi-supervised learning method for deep neural networks. In: Workshop on Challenges in Representation Learning at ICML (2013) 9
27. Lee, K., Lee, H., Lee, K., Shin, J.: Training confidence-calibrated classifiers for detecting out-of-distribution samples. In: ICLR (2018) 6, 7
28. Li, Z., Hoiem, D.: Learning without forgetting. TPAMI **40**(12), 2935–2947 (2017) 1, 2, 4, 5
29. Liu, H., Cao, Z., Long, M., Wang, J., Yang, Q.: Separate to adapt: Open set domain adaptation via progressive separation. In: CVPR (2019) 3, 4, 8, 10, 11
30. Long, M., Cao, Y., Wang, J., Jordan, M.I.: Learning transferable features with deep adaptation networks. ICML (2015) 3
31. Long, M., Zhu, H., Wang, J., Jordan, M.I.: Unsupervised domain adaptation with residual transfer networks. In: NeurIPS (2016) 2
32. Lopes, R.G., Fenu, S., Starner, T.: Data-free knowledge distillation for deep neural networks. In: LLD Workshop at NeurIPS (2017) 2
33. Luo, Z., Zou, Y., Hoffman, J., Fei-Fei, L.F.: Label efficient learning of transferable representations across domains and tasks. In: NeurIPS (2017) 3, 4, 11
34. Nayak, G.K., Mopuri, K.R., Shaj, V., Radhakrishnan, V.B., Chakraborty, A.: Zero-shot knowledge distillation in deep networks. In: ICML (2019) 2
35. Pan, S.J., Yang, Q.: A survey on transfer learning. IEEE Transactions on knowledge and data engineering (2009) 4
36. Panareda Busto, P., Gall, J.: Open set domain adaptation. In: ICCV (2017) 2
37. Pei, Z., Cao, Z., Long, M., Wang, J.: Multi-adversarial domain adaptation. In: AAAI (2018) 2
38. Peng, H., Li, J., Song, Y., Liu, Y.: Incrementally learning the hierarchical softmax function for neural language models. In: AAAI (2017) 2
39. Peng, X., Usman, B., Kaushik, N., Hoffman, J., Wang, D., Saenko, K.: Visda: The visual domain adaptation challenge. arXiv preprint arXiv:1710.06924 (2017) 11
40. Pereyra, G., Tucker, G., Chorowski, J., Kaiser, Ł., Hinton, G.: Regularizing neural networks by penalizing confident output distributions. In: ICLR (2017) 6
41. Rebuffi, S.A., Kolesnikov, A., Sperl, G., Lampert, C.H.: icarl: Incremental classifier and representation learning. In: CVPR (2017) 2, 3, 4, 11
42. Ruping, S.: Incremental learning with support vector machines. In: ICDM (2001) 2
43. Saenko, K., Kulis, B., Fritz, M., Darrell, T.: Adapting visual category models to new domains. In: ECCV (2010) 1, 4, 11
44. Saito, K., Watanabe, K., Ushiku, Y., Harada, T.: Maximum classifier discrepancy for unsupervised domain adaptation. In: CVPR (2018) 2
45. Saito, K., Yamamoto, S., Ushiku, Y., Harada, T.: Open set domain adaptation by backpropagation. In: ECCV (2018) 2, 3, 8, 11
46. Sankaranarayanan, S., Balaji, Y., Castillo, C.D., Chellappa, R.: Generate to adapt: Aligning domains using generative adversarial networks. In: CVPR (2018) 3
47. Shu, Y., Cao, Z., Long, M., Wang, J.: Transferable curriculum for weakly-supervised domain adaptation. In: AAAI (2019) 8

48. Snell, J., Swersky, K., Zemel, R.: Prototypical networks for few-shot learning. In: NeurIPS (2017) 3, 6
49. Sun, B., Saenko, K.: Deep coral: Correlation alignment for deep domain adaptation. In: ECCV Workshops (2016) 3
50. Torralba, A., Efros, A.A.: Unbiased look at dataset bias. In: CVPR (2011) 1
51. Tzeng, E., Hoffman, J., Saenko, K., Darrell, T.: Adversarial discriminative domain adaptation. In: CVPR (2017) 2
52. Tzeng, E., Hoffman, J., Zhang, N., Saenko, K., Darrell, T.: Deep domain confusion: Maximizing for domain invariance. arXiv preprint arXiv:1412.3474 (2014) 2, 3
53. Wu, Y., Chen, Y., Wang, L., Ye, Y., Liu, Z., Guo, Y., Fu, Y.: Large scale incremental learning. In: CVPR (2019) 2
54. You, K., Long, M., Cao, Z., Wang, J., Jordan, M.I.: Universal domain adaptation. In: CVPR (2019) 2, 3, 4, 11
55. Zheng, Z., Hong, P.: Robust detection of adversarial attacks by modeling the intrinsic properties of deep neural networks. In: NeurIPS (2018) 7