Domain Adaptation with Dynamic Open-Set Targets

Jun Wu
University of Illinois at Urbana-Champaign
junwu3@illinois.edu

Jingrui He
University of Illinois at Urbana-Champaign
jingrui@illinois.edu

ABSTRACT

Open-set domain adaptation aims to improve the generalization performance of a learning algorithm on a target task of interest by leveraging the label information from a relevant source task with only a subset of classes. However, most existing works are designed for the static setting, and can be hardly extended to the dynamic setting commonly seen in many real-world applications. In this paper, we focus on the more realistic open-set domain adaptation setting with a static source task and a time evolving target task where novel unknown target classes appear over time. Specifically, we show that the classification error of the new target task can be tightly bounded in terms of positive-unlabeled classification errors for historical tasks and open-set domain discrepancy across tasks. By empirically minimizing the upper bound of the target error, we propose a novel positive-unlabeled learning based algorithm named `OuterAdapter` for dynamic open-set domain adaptation with time evolving unknown classes. Extensive experiments on various data sets demonstrate the effectiveness and efficiency of our proposed `OuterAdapter` algorithm over state-of-the-art domain adaptation baselines.

CCS CONCEPTS

• **Computing methodologies** → **Transfer learning**; • **Theory of computation** → **Sample complexity and generalization bounds**.

KEYWORDS

Domain Adaptation, Positive-Unlabeled Learning, Open-Set Targets

ACM Reference Format:
Jun Wu and Jingrui He. 2022. Domain Adaptation with Dynamic Open-Set Targets. In *Proceedings of the 28th ACM SIGKDD Conference on Knowledge Discovery and Data Mining (KDD '22), August 14–18, 2022, Washington, DC, USA.* ACM, New York, NY, USA, 11 pages. https://doi.org/10.1145/3534678.3539235

1 INTRODUCTION

Domain adaptation (DA) has achieved significant success across multiple high-impact applications, e.g., object recognition [25, 30, 40, 41] and video classification [10] in computer vision, sentiment classification [33] and conversation generation [35] in natural language processing, graph representation learning [32] in graph mining, to name a few. It improves the generalization performance of a learning algorithm on a target task of interest with scare labeled data, by using knowledge from a relevant source task with adequate label information publicly available on the web. Most previous adaptation algorithms [6, 36, 39] hold the close-set assumption that the source and target tasks share the same group of classes. This strong assumption is relaxed in the open-set domain adaptation setting [19, 22], where the groups of classes from the source and target tasks are partially overlapping. The goal of open-set domain adaptation is to classify the target examples from the shared classes correctly, and identify the target examples from all the unknown classes as "unknown".

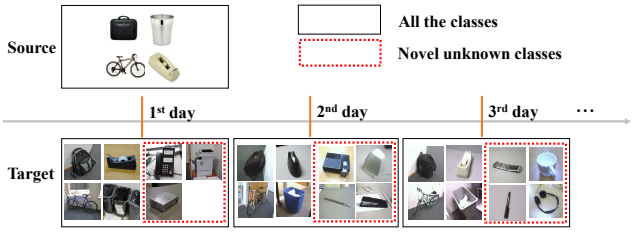


Figure 1: Illustration of dynamic open-set domain adaptation where novel unknown classes might appear every single day.

However, most existing open-set domain adaptation algorithms [4, 14] are designed only for the static scenarios. In other words, when applied to the dynamic scenarios commonly seen in many real-world applications, these algorithms would achieve sub-optimal performance. For example, eBay considers all the unpopular products (e.g., *Real Estate, Tickets & Travel*, etc.) as the "others" category[1], and new types of unpopular products frequently appear on eBay over time. These time evolving unknown classes might pose a significant challenge for predicting the class labels of eBay products when adapting from a relevant static source task (e.g., *Amazon*) without information about the new unknown classes. Another example is the wild animal recognition [25]. It leverages the annotated standard animal images (i.e., source data) to identify the categories of wild animal images (e.g., target data) captured by the cameras. However, in real scenarios, new species of animals are discovered every year[2]. In this case, previous static open-set domain adaptation algorithms [4, 14] might not be able to distinguish the known (task-shared) classes from the time evolving unknown (newly appeared) classes in the target task.

Therefore, in this paper, we study a more realistic open-set domain adaptation setting with a labeled static source task and an unlabeled time evolving target task where novel unknown classes might appear in every time stamp (see Figure 1). Compared with previous work [3, 14], this is a much more challenging problem

KDD '22, August 14–18, 2022, Washington, DC, USA

ACM ISBN 978-1-4503-9385-0/22/08…$15.00
https://doi.org/10.1145/3534678.3539235

[1] https://www.ebay.com/n/all-categories
[2] https://www.the-scientist.com/tag/new-species

due to the following unique properties: (1) the task relatedness between the static source task and the time evolving target task would change; (2) the class proportions of unknown classes appearing in the target task is varying with the increase of unknown target examples over time.

To bridge this gap, we first propose a novel open-set domain discrepancy ($\mathcal{OS}$-divergence) to measure the distribution shift in the presence of unknown target classes. The key idea of our $\mathcal{OS}$-divergence is to encourage the matching of joint distribution across tasks within the shared classes, and in the meanwhile, distinguish the marginal distribution between the shared and unknown classes. Compared to conventional domain discrepancy measures [1, 18, 36], $\mathcal{OS}$-divergence explicitly maximizes the distribution discrepancy of the target examples within the shared and unknown classes, thereby allowing us to identify the target examples within the unknown classes. Then based on $\mathcal{OS}$-divergence, we derive a novel generalization error bound in the context of dynamic open-set domain adaptation. To be more specific, we show that the classification error of the newest target task is bounded by the positive-unlabeled classification errors for historical tasks and the $\mathcal{OS}$-divergence across tasks. By empirically minimizing the error upper bound, we propose a novel domain adaptation algorithm named `OuterAdapter` to learn the predictive target function using knowledge from both the labeled source task and historical unlabeled target tasks.

Compared with previous works, the main contributions of this paper can be summarized as follows:

- **Problem:** We introduce a more realistic problem setting of dynamic open-set domain adaptation where novel unknown classes might appear over time.
- **Theory:** We provide theoretical analysis for this problem setting by deriving the generalization error bounds with our proposed $\mathcal{OS}$-divergence.
- **Algorithm:** A novel algorithm named `OuterAdapter` is proposed to minimize the error upper bound of dynamic open-set domain adaptation.
- **Experiment:** Extensive experiments confirm the effectiveness and efficiency of the `OuterAdapter` algorithm.

The rest of the paper is organized as follows. The related work is summarized in Section 2. We introduce our problem definition in Section 3. In Section 4, we present the error bounds for dynamic open-set domain adaptation, followed by a novel algorithm `OuterAdapter` in Section 5. Extensive experiments are provided in Section 6. Finally, we conclude the paper in Section 7.

2 RELATED WORK

In this section, we briefly introduce the related work on open-set domain adaptation and continual learning.

2.1 Open-Set Domain Adaptation

Open-set domain adaptation [19, 22] tackles the knowledge transfer problem where the source/target task involves unknown or task-specific classes. The goal is to identify every target example as a specific shared class or the "unknown" class. It has been theoretically shown [3, 17] that the target error is largely bounded in terms of the marginal distribution alignment between source and target examples within the shared classes as well as the misclassification rate of unknown target class. It revealed that the key challenge of open-set domain adaptation lied in the separation of known and unknown classes in the target domain. It motivated a variety of practical open-set domain adaptation algorithms [4, 14, 16] to solve the challenge. Nevertheless, all the aforementioned approaches assumed the stationary task distribution where the target domain associated with the unknown classes is static. Little effort has been devoted to studying the dynamic open-set domain adaptation where the target domain is evolving over time and novel unknown classes might appear in every time stamp. To the best of our knowledge, this is the first effort to provide both theoretical analysis and practical algorithm for the dynamic open-set domain adaptation scenarios.

2.2 Continual Learning

Continual lifelong learning aims to train a model on the new task with previously learned knowledge from a sequence of old tasks. It mitigates the catastrophic forgetting induced by the distribution shift of new task [13]. Moreover, in recent years, closed-set domain adaptation has also been studied in the dynamic scenarios [2, 8], where the knowledge can be transferred from labeled source task and historical target task to the newest target task. It is shown [12, 15, 26, 29, 31] that the historical target knowledge could be leveraged to bridge the gap between the source task and the newest target task. This results in the mitigation of negative transfer [27] in the context of dynamic domain adaptation. However, in real world scenarios, it is hard to guarantee that the target task always share the same group of classes over time. When novel unknown classes appear in the target domain, it is challenging for previous works to identify the category of the target examples. Therefore, different from previous works, we focus on the dynamic open-set domain adaptation scenarios where novel unknown classes might appear in every time stamp.

3 PRELIMINARIES

In this section, we introduce the problem definition of dynamic open-set domain adaptation and its unique challenges.

3.1 Notation

Let $\mathcal{X}$ and $\mathcal{Y}$ denote the input feature space and output class-label space. Specifically, we let $\mathcal{Y}_s, \mathcal{Y}_t \subset \mathcal{Y}$ denote the source and target class-label spaces respectively. Following the definition of open-set domain adaptation [22], in this paper, we assume $\mathcal{Y}_s \subset \mathcal{Y}_t$. That is, the target class-label space[3] $\mathcal{Y}_t = \{\mathcal{Y}_s, unknown\} = \{1, \cdots, C, C+1\}$, where the source task and target task share the first C classes, and the target task has an additional unknown class $C+1$. Let $\mathcal{D}_s$ and $\mathcal{D}_t$ denote the source and target tasks with the joint probability distribution $\mathbb{Q}^s$ and $\mathbb{P}^t$ over $\mathcal{X} \times \mathcal{Y}$, respectively. Let $\mathbb{Q}^s_X$ and $\mathbb{P}^t_X$ be the marginal distribution of the source and target tasks over $\mathcal{X}$, and $\mathbb{P}^t_{\leq C}$ be the target distribution limited to shared classes only. Let $\mathcal{H}$ be a hypothesis class on $\mathcal{X}$, where a hypothesis is a function $h : \mathcal{X} \to \mathcal{Y}$. When the target task is evolving over time, we use $\mathcal{D}_{t_j}$ to denote the target task at the time stamp j associated with class-label space $\mathcal{Y}_{t_j}$.

[3] In this paper, all classes not appearing in the source task are represented as "unknown".

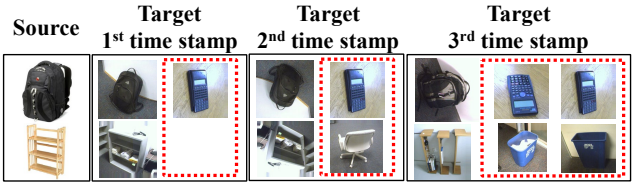


Figure 2: Challenges of dynamic open-set domain adaptation. (a) evolving distribution: data distribution of 'back pack' and 'bookcase' shifts over time; (b) varying class proportions: "unknown" class is minority at the 1st time stamp, but becomes the majority at the 3rd time stamp with the increase of unknown target examples.

3.2 Problem Definition

Formally, our dynamic open-set domain adaptation problem is defined as follows.

Definition 3.1. (**Dynamic Open-Set Domain Adaptation**) Given a labeled source task $\mathcal{D}_s$ and a time evolving unlabeled target task $\{\mathcal{D}_{t_j}\}_{j=1}^{N}$ with time stamp j, where $\mathcal{Y}_s \subset \mathcal{Y}_{t_j}$, *dynamic open-set domain adaptation* aims to learn an optimal prediction function for target task $\mathcal{D}_{t_{N+1}}$ using the knowledge from the source task $\mathcal{D}_s$ and the historical target tasks $\mathcal{D}_{t_j}(j = 1, \cdots, N)$, such that it can classify the examples from the shared classes in the target task $\mathcal{D}_{t_{N+1}}$ correctly, and identify the examples from all the unknown classes as "unknown".

Note that for dynamic domain adaptation involving N historical time stamps for the target task, for the sake of notation convenience, we use $\mathcal{D}_{t_0}$ to represent the source task $\mathcal{D}_s$ with n_{t_0} labeled source examples. In addition, we assume that the data distribution of the target task is smoothly evolving over time. In other words, the domain discrepancy $d(\cdot,\cdot)$ of the target task at adjacent time stamps is upper bounded by a small constant $\gamma > 0$, i.e., $d(\mathbb{P}^{t_j}_{\leq C}, \mathbb{P}^{t_{j+1}}_{\leq C}) \leq \gamma$, $j = 0, \ldots, N$ ($\mathbb{P}^{t_0}_{\leq C} = \mathbb{P}^{t_0} = \mathbb{Q}^s$ denotes the source distribution without unknown classes). As we will show in Theorem 4.4, this assumption ensures the successful knowledge transfer across tasks and time stamps, despite newly appearing examples from unknown classes in the target task.

3.3 Challenges

Compared to static open-set domain adaptation [19, 22], the unique challenges of dynamic open-set domain adaptation include (see Figure 2): (1) **Evolving distribution:** the target distribution evolves over time; (2) **Varying class proportions:** the ratio of target examples within the shared classes constantly changes with the increase of examples from the "unknown" class over time. In particular, with novel unknown classes appearing in the evolving target task, the openness [14] across tasks might increase over time. All these factors make it more difficult to address the influence of distribution shift across tasks and time stamps using existing domain discrepancy measures [1, 18]. They can even lead to negative transfer [27] with undesirable performance on the target task at the next time stamp in the dynamic open-set domain adaptation setting. Therefore, we propose to address these challenges by directly characterizing the open-set domain discrepancy in the presence of the "unknown" class, and minimizing the resulting error upper bound for the target task at the next time stamp.

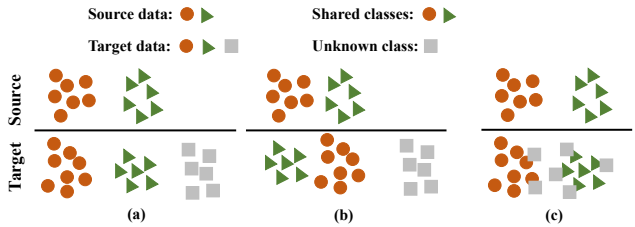


Figure 3: Distribution alignment of open-set domain adaptation: (a) label-informed distribution alignment after identifying "unknown" class; (b) marginal distribution alignment after identifying "unknown" class; (c) distribution alignment using $\mathcal{H}$-divergence without identifying the "unknown" class.

4 GENERALIZATION ERROR BOUNDS

In domain adaptation, a common practice is to learn a common latent feature space shared by source and target tasks, which minimizes a pre-defined domain discrepancy measure. In this section, we first demonstrate the limitation of existing domain discrepancy measures for learning the latent feature space in the context of open-set domain adaption. Then we formally introduce a novel open-set domain discrepancy measure ($\mathcal{OS}$-divergence), which characterizes both the shared classes and the "unknown" class. Finally, by formulating the problem as positive-unlabeled learning (PU-Learning) [11, 34], we derive the error bounds for open-set domain adaptation in the dynamic setting.

4.1 $\mathcal{OS}$-divergence

Following [1], we first consider a binary classification problem with $\mathcal{Y}_s \in \{0, 1\}$ for simplicity, although the analysis can be naturally generalized to multiple classes. A typical domain discrepancy measure between the source and target tasks is as follows.

Definition 4.1. (***$\mathcal{H}$-divergence***[1]) For $h, h' \in \mathcal{H}$, let B denote the subset of $\mathcal{X}$ such that $h(x) \neq h'(x)$ for any $x \in B$, the $\mathcal{H}$-divergence between source and target tasks over $\mathcal{X}$ is defined as:

$$d_{\mathcal{H}\Delta\mathcal{H}}(\mathbb{Q}^s_X, \mathbb{P}^t_X) = \sup_{h,h' \in \mathcal{H}} |\Pr_{\mathbb{Q}^s_X}[B] - \Pr_{\mathbb{P}^t_X}[B]| \quad (1)$$

However, it cannot be directly applied to measure the domain discrepancy for open-set domain adaptation since novel unknown classes exist in the target task. In particular, as shown in Figure 3(c), distribution alignment with $\mathcal{H}$-divergence in the presence of the "unknown" target class might lead to non-separable representation of target examples in the latent feature space. This is because in this case, it might enforce the alignment of the target examples of "unknown" class and the source examples of shared classes.

One solution proposed in previous work [3, 17] is to minimize the marginal domain discrepancy $d(\mathbb{Q}^s_X, \mathbb{P}^t_{X,\leq C})$ between source and target tasks within the shared classes, without using any label information. Nevertheless, it has two limitations: (i) it does not explicitly identify the "unknown" class, thus can suffer from the same problem of distribution alignment with $\mathcal{H}$-divergence (Figure 3(c)); (ii) marginal distribution alignment for source and target data within the shared classes cannot guarantee the success of knowledge transfer for domain adaptation (see also Theorem 4.7). As shown in Figure 3(b), exact marginal distribution alignment might lead to negative transfer [27], which is consistent with recent theoretical analysis for closed-set domain adaptation [37].

To solve these problems, we propose a novel discriminant domain discrepancy measure named $\mathcal{OS}$-divergence for open-set domain adaptation. In particular, we argue that a qualified open-set domain discrepancy measure should satisfy the following conditions (see Figure 3(a)): (i) identifying the "unknown" class and (ii) measuring the joint distribution shift (for both the features and the class labels) of source and target tasks within the shared classes. Inspired by the Fisher criterion [5] that encourages within-class examples to be close and between-class examples to be separable, we define $\mathcal{OS}$-divergence as the difference of the within-class discrepancy to the between-class discrepancy.

Definition 4.2. (***$\mathcal{OS}$-divergence***) For any $h \in \mathcal{H}$, let $I(h)$ denote the subset of $\mathcal{X}$ such that $\mathbf{x} \in I(h) \Leftrightarrow h(\mathbf{x}) = 1$ and $\overline{I(h)}$ be the complement of $I(h)$. The $\mathcal{OS}$-divergence between the source and target tasks is defined as

$$d_{\mathcal{OS}}(\mathbb{Q}^s, \mathbb{P}^t) = d_C(\mathbb{Q}^s, \mathbb{P}^t_{\leq C}) - \rho \cdot d_{\mathcal{H}\Delta\mathcal{H}}(\mathbb{Q}^s_X, \mathbb{P}^t_{X,>C}) \quad (2)$$

where $\rho > 0$ is a discrepancy coefficient and $d_C(\cdot,\cdot)$ is a label-informed domain discrepancy measure [29], i.e., $d_C(\mathbb{Q}^s, \mathbb{P}^t_{\leq C}) = \sup_{h\in\mathcal{H}} \left|\Pr_{\mathbb{Q}^s}[\{I(h), y = 1\} \cup \{\overline{I(h)}, y = 0\}] - \Pr_{\mathbb{P}^t_{\leq C}}[\{I(h), y = 1\} \cup \{\overline{I(h)}, y = 0\}]\right|$.

Remark. *The $\mathcal{OS}$-divergence is the first-order Taylor series of $d_C(\mathbb{Q}^s, \mathbb{P}^t_{\leq C}) + \frac{\rho}{d_{\mathcal{H}\Delta\mathcal{H}}(\mathbb{Q}^s_X, \mathbb{P}^t_{X,>C})} - 2\rho$, as $d_C(\mathbb{Q}^s, \mathbb{P}^t_{\leq C}) + \frac{\rho}{d_{\mathcal{H}\Delta\mathcal{H}}(\mathbb{Q}^s_X, \mathbb{P}^t_{X,>C})} - 2\rho \approx d_C(\mathbb{Q}^s, \mathbb{P}^t_{\leq C}) - \rho \cdot d_{\mathcal{H}\Delta\mathcal{H}}(\mathbb{Q}^s_X, \mathbb{P}^t_{X,>C}) + O\left((d^2_{\mathcal{H}\Delta\mathcal{H}}(\mathbb{Q}^s_X, \mathbb{P}^t_{X,>C})\right)$. One intuitive explanation of the $\mathcal{OS}$-divergence is that minimizing $\mathcal{OS}$-divergence is equivalent to encouraging the matching of joint distribution across tasks within the shared classes, and in the meanwhile, distinguishing the marginal distribution between the shared and "unknown" classes. In addition, its empirical estimate $\hat{d}_{\mathcal{OS}}(\cdot,\cdot)$ can be naturally defined as $\hat{d}_{\mathcal{OS}}(\mathbb{Q}^s, \mathbb{P}^t) = \hat{d}_C(\mathbb{Q}^s, \mathbb{P}^t_{\leq C}) - \rho \cdot \hat{d}_{\mathcal{H}\Delta\mathcal{H}}(\mathbb{Q}^s_X, \mathbb{P}^t_{X,>C})$ where $\hat{d}_C(\cdot,\cdot)$ and $\hat{d}_{\mathcal{H}\Delta\mathcal{H}}(\cdot,\cdot)$ denote the empirical estimate of C-divergence [29] and $\mathcal{H}$-divergence [1], respectively.*

4.2 Open-Set DA as PU-Learning

Open-set domain adaptation (DA) can be considered as multi-class positive-unlabeled learning (PU-Learning) [34], where all the C shared classes appearing in both source and target tasks are positive and the "unknown" class is negative. We first consider the static setting with one time stamp (i.e., $N = 0$) in the target task. The overall expected target classification error incurred by the hypothesis $h \in \mathcal{H}$ is $\epsilon_t(h) = \sum_{c=1}^{C+1} \pi^t_c \mathbb{E}_{x\sim\mathbb{P}^t(x|y=c)}[\mathcal{L}(h(x), y = c)]$, where π^t_c is the class-prior probability of class c. We have the following observation when there is no distribution shift across tasks.

Lemma 4.3. *Assume there is no distribution shift between source and target tasks, given labeled training data of C shared classes from source task and unlabeled training data from target task, the expected target error $\epsilon_t(h)$ incurred by the hypothesis $h \in \mathcal{H}$ is as follows.*

$$\epsilon_t(h) = (1 - \pi^t_{C+1})\epsilon_s(h) + \Delta_{PU}$$

where $\pi^t_{C+1} = \mathbb{P}^t(y = C + 1)$ is class-prior probability of "unknown" class in the target task and $\Delta_{PU} = \mathbb{E}_{x\sim\mathbb{P}^t_X}[\mathcal{L}(h(x), y = C + 1)] - (1 - \pi^t_{C+1})\mathbb{E}_{x\sim\mathbb{Q}^s_X}[\mathcal{L}(h(x), y = C + 1)]$ is a positive-unlabeled open-set risk. Furthermore, it has the following unbiased estimator:

$$O_{PU} = \frac{1 - \pi^t_{C+1}}{n_s} \sum_{i=1}^{n_s} \Big(\mathcal{L}(h(x^i_s), y^i_s) - \mathcal{L}(h(x^i_s), y = C + 1)\Big) + \frac{1}{m_t} \sum_{i=1}^{m_t} \mathcal{L}(h(x^i_t), y = C + 1)$$

where n_s and m_t are the numbers of training examples in the source and target tasks, respectively.

Remark. *It is notable that one recent work [16] also considered static open-set domain adaptation as a PU-Learning problem. However, it fundamentally differed from our results in that (1) little theoretical explanation was provided on the relationship between open-set domain adaptation and PU-Learning, and (2) it formulated the open-set domain adaptation as a two-stage PU-Learning problem separately recognizing the shared classes and identifying the "unknown" class. In contrast, we focus on deriving the unbiased estimator of target error in a unified manner (classifying all the classes by the hypothesis $h \in \mathcal{H}$) under mild conditions.*

As we will show in the next subsection, combined with the proposed open-set domain discrepancy measure $\mathcal{OS}$-divergence, the PU-Learning based unbiased estimator from this lemma can be directly used to derive the error bounds for dynamic open-set domain adaptation in the presence of distribution shift between tasks and time stamps.

4.3 Error Bounds

Based on the previous discussion, now we are ready to present the generalization error bound for dynamic open-set domain adaptation. Let $\pi_{C+1} = \mathbb{P}(y = C + 1)$ be the class-prior probability, where all the target classes are considered as the "unknown" class with the index $C + 1$ if they do not appear in the source task. We have,

Theorem 4.4. *(**Upper Bound**) Assume that the loss function $\mathcal{L}(\cdot,\cdot)$ is bounded, i.e., $|\mathcal{L}(\cdot,\cdot)| \leq M$. For any hypothesis $h \in \mathcal{H}$ and $\sum_{j=0}^{N} \alpha_j = 1$ where $\alpha_j \geq 0$ $(j = 0, \cdots, N)$, there exists $\rho > 0$ such that the expected error $\epsilon_{t_{N+1}}(h)$ of the target task at the $(N+1)^{th}$ time stamp is bounded as:*

$$\epsilon_{t_{N+1}}(h) \leq (1 - \pi^{t_{N+1}}_{C+1})\Big(\sum_{j=0}^{N} \alpha_j \mathbb{E}_{(x,y)\sim\mathbb{P}^{t_j}_{\leq C}}[\mathcal{L}(h(x), y)] + 4M \sum_{j=0}^{N} \alpha_j d_{\mathcal{OS}}(\mathbb{P}^{t_j}_{\leq C}, \mathbb{P}^{t_{N+1}})\Big) + \Delta_{PU} + CONST$$

where $\Delta_{PU} = \mathbb{E}_{x\sim\mathbb{P}^{t_{N+1}}_X}[\mathcal{L}(h(x), C + 1)] - (1 - \pi^{t_{N+1}}_{C+1}) \sum_{j=0}^{N} \alpha_j \cdot \mathbb{E}_{x\sim\mathbb{P}^{t_j}_{X|Y\leq C}}[\mathcal{L}(h(x), C + 1)]$ and $CONST$ is a constant.

Remark. *This theorem indicates that (1) the target error $\epsilon_{t_{N+1}}(h)$ could be upper bounded in terms of the classification error on historical tasks, the domain discrepancy across tasks, and PU-Learning based open-set risk Δ_{PU}; (2) if there exists a small constant $\gamma > 0$ such that $d_C(\mathbb{P}^{t_j}_{\leq C}, \mathbb{P}^{t_{j+1}}_{\leq C}) \leq \gamma$, the target error $\epsilon_{t_{N+1}}(h)$ could be upper bounded by the source error and PU-Learning based open-set risk Δ'_{PU} due to $d_{\mathcal{OS}}(\mathbb{P}^{t_j}_{\leq C}, \mathbb{P}^{t_{j+1}}) \leq d_C(\mathbb{P}^{t_j}_{\leq C}, \mathbb{P}^{t_{j+1}}_{\leq C}) \leq \gamma$; (3) the classification*

error O_{PU} in the error upper bound contains ordinary cost-sensitive learning on labeled data (i.e., $\mathbb{E}_{(x,y)\sim\mathbb{P}^{t_j}_{\leq C}}[\mathcal{L}(h(x),y)]$) and the open-set risk Δ_{PU} on unlabeled data.

Moreover, the following corollary states the generalization error bound with empirical Rademacher complexity [18] of a hypothesis class $\mathcal{H}$. This motivates us to design the practical algorithm for dynamic open-set domain adaptation by leveraging the knowledge from labeled source examples and historical unlabeled target examples.

Corollary 4.5. *Let n_{t_j} denote the number of examples within C shared classes and m_{t_j} be the number of all examples in the target task at the j^{th} time stamp. With the same assumption as in Theorem 4.4, for any $\delta > 0$ and $h \in \mathcal{H}$, with probability at least $1-\delta$, the expected target error $\epsilon_{t_{N+1}}(h)$ is bounded as follows.*

$$\epsilon_{t_{N+1}}(h) \leq (1-\pi^{t_{N+1}}_{C+1})\Big(\sum_{j=0}^{N}\alpha_j\frac{1}{n_{t_j}}\sum_{i=1}^{n_{t_j}}\mathcal{L}(h(x^i_{t_j}),y^i_{t_j}) + 4M\sum_{j=0}^{N}\alpha_j\hat{d}_{\mathcal{OS}}(\mathbb{P}^{t_j}_{\leq C},\mathbb{P}^{t_{N+1}})\Big)+\hat{\Delta}_{PU}+R_\delta$$

where $\hat{\Delta}_{PU} = \frac{1}{m_{t_{N+1}}}\sum_{i=1}^{m_{t_{N+1}}}\mathcal{L}(h(x^i_{t_{N+1}}),C{+}1) - (1-\pi^{t_{N+1}}_{C+1})\sum_{j=0}^{N}\frac{\alpha_j}{n_{t_j}}\cdot\sum_{i=1}^{n_{t_j}}\mathcal{L}(h(x^i_{t_j}),C+1)$ and R_δ is a Rademacher complexity term (see Appendix A.3).

Furthermore, the following corollary shows that static open-set domain adaptation [22] can be considered as a special case of Theorem 4.4 with $N=0$.

Corollary 4.6. *For static open-set domain adaptation with $N=0$, with the same assumption as in Theorem 4.4, the expected target error $\epsilon_{t_{N+1}}(h)$ is bounded:*

$$\epsilon_t(h) \leq (1-\pi^t_{C+1})\big(\epsilon_s(h)+4Md_{\mathcal{OS}}(\mathbb{Q}^s,\mathbb{P}^t)\big)+\Delta_{PU}+CONST$$

where $\Delta_{PU} = \mathbb{E}_{\mathbb{P}^t_X}[\mathcal{L}(h(x),C+1)] - (1-\pi^t_{C+1})\mathbb{E}_{\mathbb{Q}^s_X}[\mathcal{L}(h(x),C+1)]$.

Remark. *Compared to existing static error bounds of open-set domain adaptation [3, 17], the benefits of our PU-Learning based error bound are as follows. (1) Our open set risk Δ_{PU} is estimated from all the source and target examples with no need of ground truth labels of these examples; while existing works focus on designing the open-set risk with "unknown" target classification error, which is hard to estimate in real scenarios. (2) Our target error is bounded in terms of label-informed domain discrepancy measure by explicitly taking the conditional distribution shift [37] across tasks into consideration; while the discrepancy distance [18] used in previous works cannot guarantee the success of knowledge transfer between source and target tasks under the conditional shift.*

Theorem 4.7. ***(Lower Bound)** Given $h \in \mathcal{H}$, if $\mathcal{L}(h(x),y) = |h(x)-y|$, $\epsilon_{t_0}(h)=0$ and $\mathbb{P}^{t_i}_{X|Y\leq C} = \mathbb{P}^{t_j}_{X|Y\leq C}$ for $i,j=0,1,\cdots,N{+}1$, then the expected target error $\epsilon_{t_{N+1}}(h)$ is lower bounded.*

$$\epsilon_{t_{N+1}}(h) \geq \pi^{t_{N+1}}_{C+1}\epsilon^U_{t_{N+1}}(h) + \left|\sum_{c=1}^{C}c\left(\mathbb{P}^{t_{N+1}}(y=c)-(1-\pi^{t_{N+1}}_{C+1})\mathbb{P}^{t_0}(y=c)\right)\right|$$

where $\epsilon^U_{t_{N+1}}(h) = \mathbb{E}_{x\sim\mathbb{P}^{t_{N+1}}(x|y=C+1)}[\mathcal{L}(h(x),y=C+1)]$ is the "unknown" classification error.

Remark. *Theorem 4.7 shows that the target error $\epsilon_{t_{N+1}}(h)$ can be large even when source classification error is zero and marginal feature distribution within C shared classes across all tasks are exactly matched (i.e., the discrepancy distance [18] used in previous works [3, 17] is zero).*

5 PROPOSED ALGORITHM

In this section, we present a novel dynamic open-set domain adaptation algorithm `OuterAdapter`, followed by the model analysis.

5.1 `OuterAdapter`

The goal of dynamic open-set domain adaptation is to classify the target examples within the C shared classes correctly, and to identify all the "unknown" examples appearing in the new target task using knowledge from a related labeled source task and historical unlabeled target tasks. To this end, we propose a novel adaptation method `OuterAdapter`, which minimizes the generalization error upper bound derived in Section 4.3. Specifically, by empirically minimizing the upper bound of target error in Corollary 4.5, the overall objective function of `OuterAdapter` is formulated as follows.

$$\min_\theta O_{PU}(\theta)+\beta\sum_{j=0}^{N}\alpha_j\hat{d}_{\mathcal{OS}}(\mathbb{P}^{t_j}_{\leq C},\mathbb{P}^{t_{N+1}};\theta) \tag{3}$$

and

$$O_{PU}(\theta) = \sum_{j=0}^{N}\frac{\lambda\alpha_j}{n_{t_j}}\sum_{i=1}^{n_{t_j}}\Big(\mathcal{L}(h(x^i_{t_j}),\hat{y}^i_{t_j};\theta)-\mathcal{L}(h(x^i_{t_j}),C+1;\theta)\Big) + \frac{1}{m_{t_{N+1}}}\sum_{i=1}^{m_{t_{N+1}}}\mathcal{L}(h(x^i_{t_{N+1}}),C+1;\theta)$$

where $O_{PU}(\theta)$ is an unbiased positive-unlabeled learning (PU-Learning) based classification error (please see more details in next subsection) parameterized by θ, $\lambda = 1-\pi^{t_{N+1}}_{C+1} \geq 0$ is the class-prior probability, and $\beta \geq 0$ is a hyper-parameter to measure the trade-off of PU-Learning error and domain discrepancy across tasks and time stamps. In this case, since the class-prior probability in the target task is not known, we choose a fixed hyper-parameter λ to balance the classification terms in $O_{PU}(\theta)$. Here θ represents all the trainable parameters in our objective function. $\hat{y}^i_{t_j}$ is the pseudo-label of the example $x^i_{t_j}$ where $\hat{y}^i_{t_j} = y^i_{t_j}$ for labeled source examples and $\hat{y}^i_{t_j}$ would be empirically estimated for unlabeled historical target examples. The overall training process of `OuterAdapter` is illustrated in Algorithm 1. It is given the labeled source data and a sequence of unlabeled target data as input, and outputs the optimal prediction function h of the new target task $\mathcal{D}_{t_{N+1}}$. The hypothesis class $\mathcal{H}$ is assumed to the group of continuous functions, which could be universally approximated by neural networks [9] (convolutional neural network is used in our experiments for image classification). In particular, we use the following methods to estimate the parameters α_j and $\hat{y}^i_{t_j}$ as well as open-set discrepancy measure $\hat{d}_{\mathcal{OS}}$ involved in our objective function.

Estimation of pseudo-label $\hat{y}^i_{t_j}$: We sequentially learn the pseudo-label $\hat{y}^i_{t_j}$ for an example $x^i_{t_j}$ appearing in the target task $\mathcal{D}_{t_j}$ as follows. With labeled source data, the prediction function of the

first target task $\mathcal{D}_{t_1}$ can be learned by applying our OuterAdapter algorithm on the labeled source examples $(x^i_{t_0}, y^i_{t_0})$ and the unlabeled target examples $x^i_{t_1}$. Then the pseudo-label $\hat{y}^i_{t_1}$ of example $x^i_{t_1}$ from the first target task $\mathcal{D}_{t_1}$ can be obtained. After that, the prediction function of the target task $\mathcal{D}_{t_2}$ is learned by applying our OuterAdapter algorithm on the labeled examples $(x^i_{t_0}, y^i_{t_0})(x^i_{t_1}, \hat{y}^i_{t_1})$ and the unlabeled target examples $x^i_{t_2}$ for deriving the pseudo-label $\hat{y}^i_{t_2}$ of example $x^i_{t_2}$. In this way, the pseudo-label $\hat{y}^i_{t_j}$ of historical unlabeled target examples can be sequentially estimated.

Estimation of $\hat{d}_{\mathcal{OS}}$: Following [1, 29], the $\mathcal{H}$-divergence and C-divergence can be estimated from finite examples using the adversarial domain discriminator. However, it is hard to select the "unknown" target examples to estimate $d_{\mathcal{H}\Delta\mathcal{H}}(\mathbb{Q}^s_X, \mathbb{P}^t_{X,>C})$ because we have no prior information regarding the "unknown" class. Thus we choose the target examples which are classified as the "unknown" class with a high prediction probability (see Steps 11-12 in Algorithm 1). $d_C(\mathbb{Q}^s, \mathbb{P}^t_{\leq C})$ can be estimated in a similar way. In this case, the pseudo-label $\hat{y}^i_{t_j}$ of input $x^i_{t_j}$ is used to measure the distribution shift across tasks over $\mathcal{X} \times \mathcal{Y}$.

Estimation of α_j: In our algorithm, α_j indicates the importance of $\mathcal{D}_{t_j}$ on learning the prediction function of new target task $\mathcal{D}_{t_{N+1}}$. In this paper, we propose a simple but effective self-attention scheme to automatically learn the value of α_j: $\alpha_j = \exp\left(\text{LeakyReLU}(\sum_{c=1}^{C} a_c^T \bar{x}^c_{t_j})\right)$ and $\alpha_j = \alpha_j / \sum_{j=0}^{N} \alpha_j$ where $\cdot^T$ denotes the transpose. Here a_c is a class-specific weight parameter and $\bar{x}^c_{t_j} (j = 0, \cdots, N+1)$ is the class-specific average feature vector of examples $x^i_{t_j}$ within the class c in the task $\mathcal{D}_{t_j}$, by using either the real class-label $y^i_{t_j}$ (if available) or pseudo-label $\hat{y}^i_{t_j}$.

5.2 Discussion

Unbiasedness of OuterAdapter: The following theorem states that the empirical PU loss term O_{PU} of Eq. (3) is unbiased in the dynamic open-set domain adaptation setting.

Theorem 5.1. *If $d_{\mathcal{OS}}(\mathbb{P}^{t_j}_{\leq C}, \mathbb{P}^{t_{N+1}}) = 0$ for $j = 0, \cdots, N$, the empirical PU loss term O_{PU} of Eq. (3) is an unbiased estimator of target error $\epsilon_{t_{N+1}}(h)$. Furthermore, if the loss function $\mathcal{L}$ satisfies a symmetric condition, i.e., $\sum_{c=1}^{C+1} \mathcal{L}(h(x), c) = 1$ for any $x \in \mathcal{X}$ and $h \in \mathcal{H}$, the unbiased estimator involves only non-negative classification error terms as follows.*

$$
\begin{aligned}
O_{PU} = {} & 2(1 - \pi^{t_{N+1}}_{C+1}) \sum_{j=0}^{N} \alpha_j \frac{1}{n_{t_j}} \sum_{i=1}^{n_{t_j}} \mathcal{L}(h(x^i_{t_j}), \hat{y}^i_{t_j}) \\
& + \frac{1}{m_{t_{N+1}}} \sum_{i=1}^{m_{t_{N+1}}} \mathcal{L}(h(x^i_{t_{N+1}}), y = C+1) \\
& + (1 - \pi^{t_{N+1}}_{C+1}) \sum_{j=0}^{N} \alpha_j \frac{1}{n_{t_j}} \sum_{i=1}^{n_{t_j}} \sum_{c=1}^{C+1} \mathcal{L}(h(x^i_{t_j}), y = c) \\
& - (1 - \pi^{t_{N+1}}_{C+1})
\end{aligned}
$$

Computational Complexity: Compared to existing open-set domain adaptation methods [14, 17] separably classifying data of shared classes and identifying "unknown" data, OuterAdapter algorithm is trained using back propagation in an end-to-end manner.

Algorithm 1 OuterAdapter

1: **Input:** Labeled source data from $\mathcal{D}_{t_0}$, time evolving unlabeled target data from $\{\mathcal{D}_{t_j}\}_{j=1}^{N}$, a new target task $\mathcal{D}_{t_{N+1}}$, a hypothesis class $\mathcal{H}$, hyper-parameters ρ, λ, β.
2: **Output:** Prediction function on new target task $\mathcal{D}_{t_{N+1}}$.
3: Randomly initialize the model parameters θ;
4: **for** j **in** $[0, 1, \cdots, N+1]$ **do**
5: **while** Stopping criterion is not satisfied **do**
6: Sample labeled examples $(x^i_{t_k}, y^i_{t_k})$ from $\mathcal{D}_{t_0}$;
7: **for** $k \in \{1, \cdots, j-1\}$ **do**
8: Sample labeled examples $(x^i_{t_k}, \hat{y}^i_{t_k})$ from $\mathcal{D}_{t_k}$;
9: **end for**
10: Sample unlabeled examples $x^i_{t_j}$ from $\mathcal{D}_{t_j}$;
11: Estimate the probability of $x^i_{t_j}$ as "unknown";
12: Choose top-p unlabeled examples as "unknown" according to the estimated probability, and others as known ones;
13: Estimate $\hat{d}_{\mathcal{OS}}$ and $\alpha_k (k = 0, \cdots, j)$;
14: Update parameters θ and a_c using gradient descent;
15: **end while**
16: Estimate pseudo-label $\hat{y}^i_{t_j}$ for $x^i_{t_j}$ in $\mathcal{D}_{t_j}$;
17: **end for**

It has the computational complexity of $O(|\theta|)$ per iteration using gradient descent, where $|\theta|$ is the number of trainable parameters.

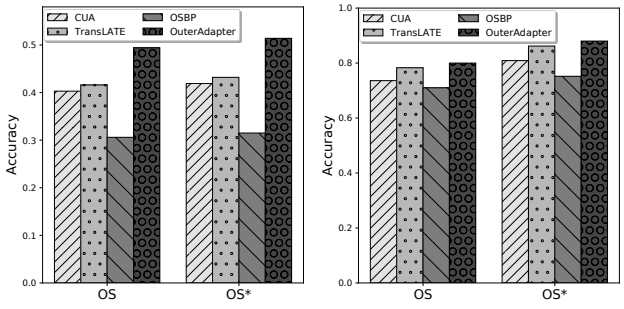
6 EXPERIMENTS

6.1 Experimental Setup

Data Sets: We use the following public data sets: Office-31 [21] with 3 domains (Amazon, DSLR, Webcam), Office-Home [24] with 4 domains (Art, Clipart, Product, Real World) and Syn2Real-O [20] with 2 domains (Synthetic, Real). Following [19, 29], we choose some classes as the "unknown" class in the target task, and generate the time evolving open-set target tasks by adding the random salt&pepper noise and rotation to the raw images (see Appendix A.6 for more details).

Baselines: In the experiments, we use the following baseline methods: (1) static closed-set adaptation methods: SourceOnly with basic ResNet-50 [7] and DANN [6]; (2) multi-source adaptation methods: MDAN [38] and DARN [28]; (3) dynamic closed-set adaptation methods: CUA [2] and TransLATE [29]; (4) static open-set adaptation methods: OSBP [22] and DAMC [23]; (5) dynamic open-set adaptation methods: it combines the open-set adaptation methods with CUA [2] to leverage the historical target knowledge, i.e., OSBP+CUA and DAMC+CUA. In addition to the proposed OuterAdapter algorithm, we also consider its simple variant OuterAdapter_μ where $\alpha_j = \frac{\mu^{N-j}}{\sum_{k=0}^{N} \mu^{N-k}}$ for $0 \leq \mu \leq 1$. Here we turn the hyperparameter $\mu \in \{0, 0.2, 0.4, 0.6, 0.8, 1.0\}$.

Implementation Details: In our experiments, we adopted ResNet-50 [7] pretrained on ImageNet as the base network for feature extraction. The overall model can be trained by back propagation, and we update the model parameters using stochastic gradient descent with mini-batch of size 16 where the number of selected "unknown" samples p is searched from {2,4,6,8,10} on estimating $\mathcal{OS}$-divergence. In addition, the hyper-parameters ρ, λ and β are empirically set as 0.25, 0.1 and 1.0, respectively. All the experiments



(a) Art→Clipart (b) DSLR→Webcam

Figure 4: Catastrophic forgetting mitigation of `OuterAdapter`

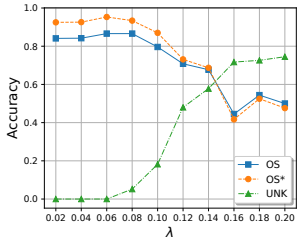


Figure 5: Hyper-parameter sensitivity

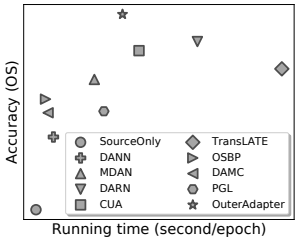


Figure 6: Comparison of computational efficiency

are performed on a Windows machine with four 3.80GHz Intel Cores, 64GB RAM and two NVIDIA Quadro RTX 5000 GPUs.

6.2 Results

Following [14, 22], we report the normalized accuracy for all classes (OS) and normalized accuracy for the known classes only (OS*) for dynamic open-set domain adaptation. The results on Office-Home, Office-31 and Syn2Real-O are shown in Table 1 (see also Table 5), Table 2 and Table 3 where the classification performance on the dynamic target task with multiple time stamps are reported (best results are indicated in bold) after running three times (the models are validated on the performance of OS*). It is observed that: (1) `OuterAdapter` algorithm consistently outperforms state-of-the-art domain adaptation baselines by a large margin; and (2) compared to $\texttt{OuterAdapter}_{\mu}$, the proposed self-attention scheme on automatically learning the weights α_j significantly improves the performance. Besides, Figure 4 shows the effect of `OuterAdapter` algorithm on mitigating the catastrophic forgetting, where the `OuterAdapter` model (as well as baselines) learned from new target task $\mathcal{D}_{t_6}$ are evaluated on all historical target tasks and then the average classification accuracy is reported. It indicates that the proposed `OuterAdapter` algorithm achieves better performance on mitigating catastrophic forgetting compared to dynamic adaptation baselines CUA and TransLATE as well as recent static open-set adaptation baseline OSBP.

6.3 Analysis

We analyze our proposed `OuterAdapter` algorithm from various aspects, including ablation study on each component of `OuterAdapter`, different evolution scenarios of open-set classes, hyper-parameter sensitivity and computational efficiency.

6.3.1 Ablation Study. We investigate the impact of the PU loss term O_{PU} and the $\mathcal{OS}$-divergence as well as the historical target knowledge in our `OuterAdapter` algorithm. It has the following variants. (i) `OuterAdapter` without PU-Learning: the loss term $O_{PU}(\theta)$ of the objective function Eq. (3) is ordinary cost-sensitive learning, i.e., $O_{PU}(\theta) = \sum_{j=0}^{N} \alpha_j \frac{1}{n_{t_j}} \sum_{i=1}^{n_{t_j}} \mathcal{L}(h(x_{t_j}^i), \hat{y}_{t_j}^i; \theta)$; (ii) `OuterAdapter` without historical data: it transfers the knowledge from labeled source data $\mathcal{D}_{t_0}$ (or $\mathcal{D}_s$) directly without using historical target knowledge; (iii) `OuterAdapter` without $\mathcal{OS}$-divergence: it uses only PU-Learning loss term without minimizing the domain discrepancy across tasks; (iv) `OuterAdapter` with $\mathcal{H}$-divergence (or $\mathcal{C}$-divergence): the $\mathcal{OS}$-divergence of the objective function in Eq. (3) is simply replaced with $\mathcal{H}$-divergence [1] (or $\mathcal{C}$-divergence [29]). Table 6 shows the results on new target task $\mathcal{D}_{t_6}$ from Office-31 (DSLR→Webcam) where "w" indicates "with" and "w/o" indicates "without". It is observed that the derived PU-Learning loss term positively affects `OuterAdapter`; historical target knowledge as well as our $\mathcal{OS}$-divergence could indeed improve the performance of `OuterAdapter` on dynamic open-set domain adaptation.

6.3.2 Evolution of Open-Set Classes. We evaluate our proposed `OuterAdapter` algorithm on various cases regarding the evolution of open-set classes. One common case is that the novel classes are appearing over time. Here we consider two additional scenarios with either constantly decreasing open-set classes, or randomly changing open-set classes at every time stamp (see Appendix A.6). Table 7 shows the performance of the `OuterAdapter` model on the new target task $\mathcal{D}_{t_6}$ (Final) from Office-31 (DSLR → Webcam) and on all historical target tasks (Average). It confirms that `OuterAdapter` consistently outperforms the baselines under different evolving conditions of open-set classes.

6.3.3 Hyper-parameter Sensitivity. We investigate the sensitivity of our `OuterAdapter` algorithm to hyper-parameter λ which approximates the class-prior probability $1-\pi_{C+1}^{t_{N+1}}$ in PU-Learning loss term. Figure 5 shows the normalized accuracy for all classes (OS), normalized accuracy for the known classes only (OS*), and accuracy for the unknown class only (UNK) on new target task $\mathcal{D}_{t_6}$ from Office-31 (DSLR→Webcam). It shows that the value of λ largely affects the trade-off of classifying the data as the shared classes or identifying the data as "unknown" target class. More specifically, Table 4 lists the classification accuracy on every class. It is observed that compared to static OSBP [22], continuous methods OSBP+CUA and `OuterAdapter` can better model the class membership of time evolving open-set target domain.

6.3.4 Efficiency. Besides, we compare the computational complexity of `OuterAdapter` with baselines. The running time (measured in seconds wall-clock time) per epoch and the normalized accuracy for all classes (OS) on new target task $\mathcal{D}_{t_5}$ from Syn2Real-O (Synthetic → Real) are reported in Figure 6. It is observed that `OuterAdapter` achieves the best model performance with less computational complexity compared to existing dynamic domain adaptation methods.

7 CONCLUSION

In this paper, we focus on a more realistic and challenging domain adaptation setting, where examples from unknown classes appear

Method	$\mathcal{D}_{t_1}$		$\mathcal{D}_{t_2}$		$\mathcal{D}_{t_3}$		$\mathcal{D}_{t_4}$		$\mathcal{D}_{t_5}$		$\mathcal{D}_{t_6}$	
	OS	OS*	OS	OS*	OS	OS*	OS	OS*	OS	OS*	OS	OS*
SourceOnly	$0.542_{\pm 0.011}$	$0.564_{\pm 0.012}$	$0.372_{\pm 0.011}$	$0.387_{\pm 0.012}$	$0.300_{\pm 0.012}$	$0.312_{\pm 0.012}$	$0.316_{\pm 0.008}$	$0.329_{\pm 0.008}$	$0.271_{\pm 0.014}$	$0.282_{\pm 0.014}$	$0.258_{\pm 0.011}$	$0.269_{\pm 0.011}$
DANN	$0.599_{\pm 0.014}$	$0.622_{\pm 0.014}$	$0.434_{\pm 0.016}$	$0.452_{\pm 0.016}$	$0.329_{\pm 0.012}$	$0.342_{\pm 0.013}$	$0.340_{\pm 0.009}$	$0.353_{\pm 0.009}$	$0.285_{\pm 0.008}$	$0.296_{\pm 0.008}$	$0.299_{\pm 0.016}$	$0.311_{\pm 0.017}$
MDAN	$0.597_{\pm 0.008}$	$0.621_{\pm 0.008}$	$0.490_{\pm 0.012}$	$0.510_{\pm 0.012}$	$0.415_{\pm 0.008}$	$0.432_{\pm 0.008}$	$0.405_{\pm 0.013}$	$0.421_{\pm 0.013}$	$0.392_{\pm 0.007}$	$0.408_{\pm 0.007}$	$0.398_{\pm 0.011}$	$0.415_{\pm 0.011}$
DARN	$0.601_{\pm 0.013}$	$0.625_{\pm 0.013}$	$0.480_{\pm 0.022}$	$0.499_{\pm 0.023}$	$0.412_{\pm 0.026}$	$0.429_{\pm 0.026}$	$0.403_{\pm 0.033}$	$0.419_{\pm 0.034}$	$0.389_{\pm 0.021}$	$0.404_{\pm 0.022}$	$0.382_{\pm 0.036}$	$0.397_{\pm 0.038}$
CUA	$0.602_{\pm 0.008}$	$0.626_{\pm 0.008}$	$0.488_{\pm 0.021}$	$0.508_{\pm 0.022}$	$0.408_{\pm 0.028}$	$0.425_{\pm 0.029}$	$0.402_{\pm 0.026}$	$0.418_{\pm 0.027}$	$0.382_{\pm 0.019}$	$0.397_{\pm 0.020}$	$0.384_{\pm 0.020}$	$0.399_{\pm 0.020}$
TransLATE	$0.590_{\pm 0.007}$	$0.614_{\pm 0.007}$	$0.473_{\pm 0.014}$	$0.492_{\pm 0.015}$	$0.408_{\pm 0.022}$	$0.424_{\pm 0.023}$	$0.382_{\pm 0.015}$	$0.398_{\pm 0.015}$	$0.380_{\pm 0.017}$	$0.396_{\pm 0.017}$	$0.380_{\pm 0.018}$	$0.395_{\pm 0.018}$
OSBP	$0.649_{\pm 0.009}$	$0.674_{\pm 0.008}$	$0.474_{\pm 0.015}$	$0.489_{\pm 0.016}$	$0.355_{\pm 0.012}$	$0.365_{\pm 0.012}$	$0.370_{\pm 0.002}$	$0.382_{\pm 0.004}$	$0.303_{\pm 0.002}$	$0.310_{\pm 0.003}$	$0.313_{\pm 0.006}$	$0.319_{\pm 0.007}$
DAMC	$0.610_{\pm 0.010}$	$0.623_{\pm 0.009}$	$0.431_{\pm 0.008}$	$0.438_{\pm 0.011}$	$0.316_{\pm 0.006}$	$0.321_{\pm 0.010}$	$0.338_{\pm 0.003}$	$0.345_{\pm 0.001}$	$0.278_{\pm 0.001}$	$0.286_{\pm 0.002}$	$0.299_{\pm 0.011}$	$0.306_{\pm 0.013}$
OSBP+CUA	$0.649_{\pm 0.009}$	$0.674_{\pm 0.008}$	$0.557_{\pm 0.009}$	$0.580_{\pm 0.010}$	$0.475_{\pm 0.006}$	$0.493_{\pm 0.007}$	$0.435_{\pm 0.007}$	$0.453_{\pm 0.008}$	$0.429_{\pm 0.008}$	$0.446_{\pm 0.009}$	$0.417_{\pm 0.009}$	$0.434_{\pm 0.009}$
DAMC+CUA	$0.610_{\pm 0.010}$	$0.623_{\pm 0.009}$	$0.487_{\pm 0.007}$	$0.503_{\pm 0.008}$	$0.413_{\pm 0.008}$	$0.428_{\pm 0.009}$	$0.404_{\pm 0.008}$	$0.419_{\pm 0.008}$	$0.390_{\pm 0.010}$	$0.405_{\pm 0.010}$	$0.386_{\pm 0.016}$	$0.401_{\pm 0.017}$
OuterAdapter$_\mu$	$\mathbf{0.652_{\pm 0.004}}$	$\mathbf{0.678_{\pm 0.004}}$	$0.590_{\pm 0.012}$	$0.614_{\pm 0.012}$	$0.507_{\pm 0.006}$	$0.527_{\pm 0.006}$	$0.487_{\pm 0.015}$	$0.506_{\pm 0.016}$	$0.440_{\pm 0.025}$	$0.457_{\pm 0.026}$	$0.425_{\pm 0.025}$	$0.442_{\pm 0.027}$
OuterAdapter	$\mathbf{0.652_{\pm 0.004}}$	$\mathbf{0.678_{\pm 0.004}}$	$\mathbf{0.598_{\pm 0.005}}$	$\mathbf{0.622_{\pm 0.005}}$	$\mathbf{0.541_{\pm 0.005}}$	$\mathbf{0.562_{\pm 0.004}}$	$\mathbf{0.525_{\pm 0.004}}$	$\mathbf{0.546_{\pm 0.004}}$	$\mathbf{0.501_{\pm 0.006}}$	$\mathbf{0.521_{\pm 0.007}}$	$\mathbf{0.494_{\pm 0.008}}$	$\mathbf{0.513_{\pm 0.008}}$

Table 1: Accuracy of dynamic open-set domain adaptation on Office-Home (Art → Clipart)

Method	$\mathcal{D}_{t_1}$		$\mathcal{D}_{t_2}$		$\mathcal{D}_{t_3}$		$\mathcal{D}_{t_4}$		$\mathcal{D}_{t_5}$		$\mathcal{D}_{t_6}$	
	OS	OS*	OS	OS*	OS	OS*	OS	OS*	OS	OS*	OS	OS*
SourceOnly	$0.871_{\pm 0.009}$	$0.958_{\pm 0.009}$	$0.822_{\pm 0.007}$	$0.905_{\pm 0.008}$	$0.690_{\pm 0.003}$	$0.759_{\pm 0.003}$	$0.733_{\pm 0.027}$	$0.806_{\pm 0.031}$	$0.649_{\pm 0.009}$	$0.714_{\pm 0.011}$	$0.542_{\pm 0.008}$	$0.596_{\pm 0.009}$
DANN	$0.891_{\pm 0.006}$	$0.980_{\pm 0.007}$	$0.821_{\pm 0.003}$	$0.903_{\pm 0.003}$	$0.775_{\pm 0.009}$	$0.853_{\pm 0.010}$	$0.804_{\pm 0.020}$	$0.884_{\pm 0.023}$	$0.767_{\pm 0.019}$	$0.844_{\pm 0.020}$	$0.742_{\pm 0.014}$	$0.815_{\pm 0.016}$
MDAN	$0.900_{\pm 0.005}$	$0.990_{\pm 0.005}$	$0.866_{\pm 0.017}$	$0.953_{\pm 0.019}$	$0.819_{\pm 0.013}$	$0.901_{\pm 0.019}$	$0.834_{\pm 0.011}$	$0.918_{\pm 0.012}$	$0.797_{\pm 0.016}$	$0.877_{\pm 0.017}$	$0.792_{\pm 0.014}$	$0.872_{\pm 0.016}$
DARN	$0.884_{\pm 0.008}$	$0.972_{\pm 0.008}$	$0.831_{\pm 0.022}$	$0.914_{\pm 0.025}$	$0.815_{\pm 0.020}$	$0.897_{\pm 0.021}$	$0.815_{\pm 0.025}$	$0.896_{\pm 0.027}$	$0.797_{\pm 0.022}$	$0.877_{\pm 0.024}$	$0.803_{\pm 0.024}$	$0.883_{\pm 0.027}$
CUA	$0.879_{\pm 0.002}$	$0.967_{\pm 0.003}$	$0.850_{\pm 0.013}$	$0.935_{\pm 0.015}$	$0.832_{\pm 0.015}$	$0.915_{\pm 0.017}$	$0.834_{\pm 0.007}$	$0.918_{\pm 0.008}$	$0.836_{\pm 0.006}$	$0.919_{\pm 0.007}$	$0.834_{\pm 0.010}$	$0.917_{\pm 0.011}$
TransLATE	$0.897_{\pm 0.006}$	$0.987_{\pm 0.007}$	$0.883_{\pm 0.014}$	$0.971_{\pm 0.015}$	$0.849_{\pm 0.026}$	$0.934_{\pm 0.029}$	$0.862_{\pm 0.015}$	$0.948_{\pm 0.017}$	$0.856_{\pm 0.026}$	$0.942_{\pm 0.028}$	$0.846_{\pm 0.021}$	$0.930_{\pm 0.023}$
OSBP	$\mathbf{0.907_{\pm 0.003}}$	$\mathbf{0.993_{\pm 0.001}}$	$0.848_{\pm 0.013}$	$0.929_{\pm 0.010}$	$0.792_{\pm 0.033}$	$0.868_{\pm 0.034}$	$0.788_{\pm 0.003}$	$0.862_{\pm 0.001}$	$0.813_{\pm 0.007}$	$0.892_{\pm 0.005}$	$0.777_{\pm 0.033}$	$0.853_{\pm 0.035}$
DAMC	$0.894_{\pm 0.002}$	$0.980_{\pm 0.000}$	$0.878_{\pm 0.011}$	$0.962_{\pm 0.010}$	$0.828_{\pm 0.006}$	$0.901_{\pm 0.007}$	$0.792_{\pm 0.025}$	$0.858_{\pm 0.025}$	$0.770_{\pm 0.028}$	$0.838_{\pm 0.031}$	$0.749_{\pm 0.017}$	$0.814_{\pm 0.026}$
OSBP+CUA	$\mathbf{0.907_{\pm 0.003}}$	$\mathbf{0.993_{\pm 0.001}}$	$0.855_{\pm 0.001}$	$0.940_{\pm 0.000}$	$0.853_{\pm 0.003}$	$0.938_{\pm 0.004}$	$0.852_{\pm 0.003}$	$0.936_{\pm 0.004}$	$0.855_{\pm 0.001}$	$0.940_{\pm 0.001}$	$0.846_{\pm 0.003}$	$0.931_{\pm 0.003}$
DAMC+CUA	$0.894_{\pm 0.002}$	$0.980_{\pm 0.000}$	$0.868_{\pm 0.015}$	$0.953_{\pm 0.017}$	$0.847_{\pm 0.018}$	$0.931_{\pm 0.019}$	$0.839_{\pm 0.010}$	$0.923_{\pm 0.011}$	$0.840_{\pm 0.031}$	$0.924_{\pm 0.034}$	$0.823_{\pm 0.016}$	$0.905_{\pm 0.018}$
OuterAdapter$_\mu$	$0.901_{\pm 0.007}$	$0.991_{\pm 0.008}$	$0.883_{\pm 0.007}$	$0.971_{\pm 0.008}$	$\mathbf{0.877_{\pm 0.019}}$	$\mathbf{0.964_{\pm 0.021}}$	$\mathbf{0.872_{\pm 0.008}}$	$\mathbf{0.959_{\pm 0.009}}$	$0.862_{\pm 0.021}$	$0.948_{\pm 0.023}$	$0.865_{\pm 0.007}$	$0.951_{\pm 0.008}$
OuterAdapter	$0.901_{\pm 0.007}$	$0.991_{\pm 0.008}$	$\mathbf{0.895_{\pm 0.003}}$	$\mathbf{0.985_{\pm 0.004}}$	$0.875_{\pm 0.009}$	$0.962_{\pm 0.010}$	$0.868_{\pm 0.012}$	$0.954_{\pm 0.013}$	$\mathbf{0.869_{\pm 0.010}}$	$\mathbf{0.955_{\pm 0.011}}$	$\mathbf{0.874_{\pm 0.010}}$	$\mathbf{0.962_{\pm 0.010}}$

Table 2: Accuracy of dynamic open-set domain adaptation on Office-31 (Amazon → DSLR)

Method	$\mathcal{D}_{t_1}$		$\mathcal{D}_{t_2}$		$\mathcal{D}_{t_3}$		$\mathcal{D}_{t_4}$		$\mathcal{D}_{t_5}$		$\mathcal{D}_{t_6}$	
	OS	OS*	OS	OS*	OS	OS*	OS	OS*	OS	OS*	OS	OS*
SourceOnly	$0.451_{\pm 0.005}$	$0.488_{\pm 0.005}$	$0.215_{\pm 0.017}$	$0.233_{\pm 0.019}$	$0.167_{\pm 0.011}$	$0.181_{\pm 0.011}$	$0.146_{\pm 0.015}$	$0.158_{\pm 0.017}$	$0.131_{\pm 0.007}$	$0.142_{\pm 0.008}$	$0.123_{\pm 0.004}$	$0.133_{\pm 0.004}$
DANN	$0.544_{\pm 0.019}$	$0.589_{\pm 0.021}$	$0.363_{\pm 0.001}$	$0.394_{\pm 0.001}$	$0.261_{\pm 0.011}$	$0.283_{\pm 0.012}$	$0.244_{\pm 0.017}$	$0.264_{\pm 0.019}$	$0.188_{\pm 0.010}$	$0.204_{\pm 0.011}$	$0.196_{\pm 0.003}$	$0.212_{\pm 0.003}$
MDAN	$0.534_{\pm 0.010}$	$0.578_{\pm 0.011}$	$0.358_{\pm 0.009}$	$0.388_{\pm 0.010}$	$0.298_{\pm 0.004}$	$0.323_{\pm 0.004}$	$0.277_{\pm 0.014}$	$0.300_{\pm 0.015}$	$0.225_{\pm 0.009}$	$0.244_{\pm 0.010}$	$0.244_{\pm 0.010}$	$0.265_{\pm 0.011}$
DARN	$0.541_{\pm 0.004}$	$0.586_{\pm 0.004}$	$0.376_{\pm 0.022}$	$0.408_{\pm 0.024}$	$0.310_{\pm 0.022}$	$0.336_{\pm 0.024}$	$0.269_{\pm 0.023}$	$0.291_{\pm 0.025}$	$0.220_{\pm 0.028}$	$0.238_{\pm 0.031}$	$0.230_{\pm 0.017}$	$0.250_{\pm 0.019}$
CUA	$0.550_{\pm 0.012}$	$0.596_{\pm 0.013}$	$0.384_{\pm 0.013}$	$0.417_{\pm 0.015}$	$0.301_{\pm 0.010}$	$0.326_{\pm 0.011}$	$0.263_{\pm 0.012}$	$0.286_{\pm 0.013}$	$0.202_{\pm 0.033}$	$0.219_{\pm 0.036}$	$0.171_{\pm 0.020}$	$0.186_{\pm 0.022}$
TransLATE	$0.551_{\pm 0.024}$	$0.597_{\pm 0.027}$	$0.374_{\pm 0.020}$	$0.405_{\pm 0.021}$	$0.297_{\pm 0.025}$	$0.322_{\pm 0.028}$	$0.262_{\pm 0.023}$	$0.283_{\pm 0.024}$	$0.210_{\pm 0.020}$	$0.228_{\pm 0.021}$	$0.194_{\pm 0.028}$	$0.210_{\pm 0.030}$
OSBP	$0.608_{\pm 0.006}$	$0.658_{\pm 0.007}$	$0.392_{\pm 0.017}$	$0.423_{\pm 0.018}$	$0.273_{\pm 0.007}$	$0.291_{\pm 0.010}$	$0.237_{\pm 0.009}$	$0.253_{\pm 0.011}$	$0.198_{\pm 0.016}$	$0.211_{\pm 0.017}$	$0.202_{\pm 0.004}$	$0.217_{\pm 0.002}$
DAMC	$0.563_{\pm 0.011}$	$0.605_{\pm 0.013}$	$0.345_{\pm 0.007}$	$0.371_{\pm 0.005}$	$0.251_{\pm 0.020}$	$0.272_{\pm 0.021}$	$0.231_{\pm 0.009}$	$0.249_{\pm 0.008}$	$0.198_{\pm 0.008}$	$0.213_{\pm 0.005}$	$0.190_{\pm 0.013}$	$0.205_{\pm 0.016}$
OSBP+CUA	$0.608_{\pm 0.006}$	$0.658_{\pm 0.007}$	$0.412_{\pm 0.006}$	$0.445_{\pm 0.006}$	$0.312_{\pm 0.014}$	$0.338_{\pm 0.015}$	$0.267_{\pm 0.014}$	$0.289_{\pm 0.015}$	$0.201_{\pm 0.021}$	$0.218_{\pm 0.024}$	$0.200_{\pm 0.016}$	$0.216_{\pm 0.017}$
DAMC+CUA	$0.563_{\pm 0.011}$	$0.605_{\pm 0.013}$	$0.366_{\pm 0.021}$	$0.396_{\pm 0.023}$	$0.262_{\pm 0.019}$	$0.284_{\pm 0.020}$	$0.244_{\pm 0.023}$	$0.265_{\pm 0.024}$	$0.182_{\pm 0.023}$	$0.197_{\pm 0.025}$	$0.181_{\pm 0.017}$	$0.196_{\pm 0.019}$
OuterAdapter$_\mu$	$\mathbf{0.615_{\pm 0.010}}$	$\mathbf{0.666_{\pm 0.011}}$	$0.469_{\pm 0.011}$	$0.508_{\pm 0.012}$	$0.361_{\pm 0.012}$	$0.391_{\pm 0.013}$	$0.284_{\pm 0.010}$	$0.308_{\pm 0.011}$	$0.230_{\pm 0.006}$	$0.249_{\pm 0.006}$	$0.208_{\pm 0.010}$	$0.225_{\pm 0.015}$
OuterAdapter	$\mathbf{0.615_{\pm 0.010}}$	$\mathbf{0.666_{\pm 0.011}}$	$\mathbf{0.490_{\pm 0.014}}$	$\mathbf{0.530_{\pm 0.016}}$	$\mathbf{0.379_{\pm 0.025}}$	$\mathbf{0.410_{\pm 0.027}}$	$\mathbf{0.328_{\pm 0.021}}$	$\mathbf{0.356_{\pm 0.024}}$	$\mathbf{0.271_{\pm 0.013}}$	$\mathbf{0.293_{\pm 0.013}}$	$\mathbf{0.275_{\pm 0.010}}$	$\mathbf{0.299_{\pm 0.011}}$

Table 3: Accuracy of dynamic open-set domain adaptation on Syn2Real-O (Synthetic → Real)

Method	Backpack	Bike	Helmet	Bookcase	Bottle	Calculator	Chair	Lamp	Computer	Cabinet	UNK	OS	OS*
OSBP	0.310	0.952	1.000	0.833	1.000	0.903	0.550	1.000	0.048	0.158	0.310	0.642	0.675
OSBP+CUA	0.931	0.952	0.857	1.000	0.938	0.839	0.875	0.889	0.095	0.947	0.368	0.790	0.832
OuterAdapter	0.966	0.952	1.000	1.000	1.000	1.000	0.725	0.889	0.714	0.895	0.246	0.853	0.914

Table 4: Performance comparisons Office-31 (DSLR → Webcam)

in the target task over time. To this end, we propose a novel dynamic open-set domain adaptation algorithm OuterAdapter by explicitly minimizing the generalization error bound. Experiments confirm the effectiveness and efficiency of our proposed OuterAdapter algorithm against state-of-the-art baselines.

ACKNOWLEDGMENTS

This work is supported by National Science Foundation under Award No. IIS-1947203, IIS-2117902, IIS-2137468, and Agriculture and Food Research Initiative (AFRI) grant no. 2020-67021-32799/ project accession no.1024178 from the USDA National Institute of

Method	Product → Real World		DSLR → Webcam	
	OS	OS*	OS	OS*
SourceOnly	$0.414_{\pm 0.034}$	$0.431_{\pm 0.036}$	$0.486_{\pm 0.008}$	$0.535_{\pm 0.011}$
DANN	$0.488_{\pm 0.006}$	$0.508_{\pm 0.006}$	$0.612_{\pm 0.002}$	$0.673_{\pm 0.004}$
MDAN	$0.596_{\pm 0.023}$	$0.620_{\pm 0.024}$	$0.773_{\pm 0.008}$	$0.815_{\pm 0.008}$
DARN	$0.590_{\pm 0.010}$	$0.614_{\pm 0.010}$	$0.766_{\pm 0.021}$	$0.842_{\pm 0.023}$
CUA	$0.587_{\pm 0.027}$	$0.610_{\pm 0.029}$	$0.805_{\pm 0.022}$	$0.885_{\pm 0.024}$
TransLATE	$0.627_{\pm 0.019}$	$0.652_{\pm 0.020}$	$0.787_{\pm 0.018}$	$0.866_{\pm 0.021}$
OSBP	$0.544_{\pm 0.011}$	$0.562_{\pm 0.013}$	$0.669_{\pm 0.023}$	$0.702_{\pm 0.030}$
DAMC	$0.473_{\pm 0.011}$	$0.482_{\pm 0.011}$	$0.552_{\pm 0.015}$	$0.571_{\pm 0.029}$
OSBP+CUA	$0.677_{\pm 0.021}$	$0.704_{\pm 0.021}$	$0.797_{\pm 0.004}$	$0.877_{\pm 0.005}$
DAMC+CUA	$0.646_{\pm 0.002}$	$0.671_{\pm 0.003}$	$0.809_{\pm 0.004}$	$0.890_{\pm 0.004}$
OuterAdapter$_{\mu}$	$0.626_{\pm 0.021}$	$0.651_{\pm 0.022}$	$0.791_{\pm 0.005}$	$0.870_{\pm 0.005}$
OuterAdapter	$\mathbf{0.698_{\pm 0.005}}$	$\mathbf{0.727_{\pm 0.005}}$	$\mathbf{0.852_{\pm 0.006}}$	$\mathbf{0.937_{\pm 0.007}}$

Table 5: Accuracy of dynamic open-set domain adaptation on Office-Home (Product → Real World) and Office-31 (DSLR → Webcam) where the classification performance on the final target task are reported

Method	OS	OS*
OuterAdapter w/o PU-Learning	0.545	0.600
OuterAdapter w/o historical data	0.730	0.803
OuterAdapter w/o $\mathcal{OS}$-divergence	0.782	0.860
OuterAdapter w $\mathcal{H}$-divergence	0.829	0.912
OuterAdapter w $\mathcal{C}$-divergence	0.839	0.922
OuterAdapter	**0.848**	**0.933**

Table 6: Ablation performance

Method	Decreasing				Random			
	Final		Average		Final		Average	
	OS	OS*	OS	OS*	OS	OS*	OS	OS*
CUA	0.640	0.704	0.755	0.831	0.773	0.851	0.844	0.928
OSBP	0.697	0.731	0.801	0.848	0.730	0.801	0.811	0.888
OSBP_CUA	0.798	0.877	0.788	0.867	0.797	0.877	0.827	0.910
OuterAdapter	**0.812**	**0.894**	**0.858**	**0.944**	**0.837**	**0.921**	**0.866**	**0.948**

Table 7: Performance on different types of evolution of open-set classes

Food and Agriculture. The views and conclusions are those of the authors and should not be interpreted as representing the official policies of the funding agencies or the government.

REFERENCES

[1] Shai Ben-David, John Blitzer, Koby Crammer, Alex Kulesza, Fernando Pereira, and Jennifer Wortman Vaughan. 2010. A theory of learning from different domains. *Machine Learning* 79, 1-2 (2010), 151–175.

[2] Andreea Bobu, Eric Tzeng, Judy Hoffman, and Trevor Darrell. 2018. Adapting to continuously shifting domains. (2018).

[3] Zhen Fang, Jie Lu, Feng Liu, Junyu Xuan, and Guangquan Zhang. 2020. Open set domain adaptation: Theoretical bound and algorithm. *TNNLS* (2020).

[4] Qianyu Feng, Guoliang Kang, Hehe Fan, and Yi Yang. 2019. Attract or distract: Exploit the margin of open set. In *ICCV*. 7990–7999.

[5] Ronald A Fisher. 1936. The use of multiple measurements in taxonomic problems. *Annals of eugenics* 7, 2 (1936), 179–188.

[6] Yaroslav Ganin, Evgeniya Ustinova, Hana Ajakan, Pascal Germain, Hugo Larochelle, François Laviolette, Mario Marchand, and Victor Lempitsky. 2016. Domain-adversarial training of neural networks. *JMLR* 17, 1 (2016), 2096–2030.

[7] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. 2016. Deep residual learning for image recognition. In *CVPR*. 770–778.

[8] Judy Hoffman, Trevor Darrell, and Kate Saenko. 2014. Continuous manifold based adaptation for evolving visual domains. In *CVPR*. 867–874.

[9] Kurt Hornik, Maxwell Stinchcombe, and Halbert White. 1989. Multilayer feedforward networks are universal approximators. *Neural Networks* 2, 5 (1989), 359–366.

[10] Andrew Kae and Yale Song. 2020. Image to Video Domain Adaptation Using Web Supervision. In *WACV*. 567–575.

[11] Ryuichi Kiryo, Gang Niu, Marthinus C Du Plessis, and Masashi Sugiyama. 2017. Positive-unlabeled learning with non-negative risk estimator. *NeurIPS* 30 (2017).

[12] Ananya Kumar, Tengyu Ma, and Percy Liang. 2020. Understanding self-training for gradual domain adaptation. In *ICML*. 5468–5479.

[13] Zhizhong Li and Derek Hoiem. 2017. Learning without forgetting. *TPAMI* 40, 12 (2017), 2935–2947.

[14] Hong Liu, Zhangjie Cao, Mingsheng Long, Jianmin Wang, and Qiang Yang. 2019. Separate to adapt: Open set domain adaptation via progressive separation. In *CVPR*. 2927–2936.

[15] Hong Liu, Mingsheng Long, Jianmin Wang, and Yu Wang. 2020. Learning to Adapt to Evolving Domains. *NeurIPS* 33 (2020).

[16] Mohammad Reza Loghmani, Markus Vincze, and Tatiana Tommasi. 2020. Positive-Unlabeled Learning for Open Set Domain Adaptation. *Pattern Recognition Letters* (2020).

[17] Yadan Luo, Zijian Wang, Zi Huang, and Mahsa Baktashmotlagh. 2020. Progressive graph learning for open-set domain adaptation. In *ICML*. 6468–6478.

[18] Yishay Mansour, Mehryar Mohri, and Afshin Rostamizadeh. 2009. Domain adaptation: Learning bounds and algorithms. In *COLT*.

[19] Pau Panareda Busto and Juergen Gall. 2017. Open set domain adaptation. In *ICCV*. 754–763.

[20] Xingchao Peng, Ben Usman, Kuniaki Saito, Neela Kaushik, Judy Hoffman, and Kate Saenko. 2018. Syn2real: A new benchmark forsynthetic-to-real visual domain adaptation. *arXiv preprint arXiv:1806.09755* (2018).

[21] Kate Saenko, Brian Kulis, Mario Fritz, and Trevor Darrell. 2010. Adapting visual category models to new domains. In *ECCV*. 213–226.

[22] Kuniaki Saito, Shohei Yamamoto, Yoshitaka Ushiku, and Tatsuya Harada. 2018. Open set domain adaptation by backpropagation. In *ECCV*. 153–168.

[23] Tasfia Shermin, Guojun Lu, Shyh Wei Teng, Manzur Murshed, and Ferdous Sohel. 2020. Adversarial network with multiple classifiers for open set domain adaptation. *IEEE Transactions on Multimedia* (2020).

[24] Hemanth Venkateswara, Jose Eusebio, Shayok Chakraborty, and Sethuraman Panchanathan. 2017. Deep hashing network for unsupervised domain adaptation. In *CVPR*. 5018–5027.

[25] Aparna Nurani Venkitasubramanian, Tinne Tuytelaars, and Marie-Francine Moens. 2016. Wildlife recognition in nature documentaries with weak supervision from subtitles and external data. *Pattern Recognition Letters* (2016).

[26] Hao Wang, Hao He, and Dina Katabi. 2020. Continuously Indexed Domain Adaptation. In *ICML*. 9898–9907.

[27] Zirui Wang, Zihang Dai, Barnabás Póczos, and Jaime Carbonell. 2019. Characterizing and avoiding negative transfer. In *CVPR*. 11293–11302.

[28] Junfeng Wen, Russell Greiner, and Dale Schuurmans. 2020. Domain aggregation networks for multi-source domain adaptation. In *ICML*. 10214–10224.

[29] Jun Wu and Jingrui He. 2020. Continuous Transfer Learning with Label-informed Distribution Alignment. *arXiv preprint arXiv:2006.03230* (2020).

[30] Jun Wu and Jingrui He. 2021. Indirect Invisible Poisoning Attacks on Domain Adaptation. In *KDD*. 1852–1862.

[31] Jun Wu and Jingrui He. 2022. A Unified Meta-Learning Framework for Dynamic Transfer Learning. In *IJCAI*.

[32] Man Wu, Shirui Pan, Chuan Zhou, Xiaojun Chang, and Xingquan Zhu. 2020. Unsupervised domain adaptive graph convolutional networks. In *Proceedings of The Web Conference 2020*. 1457–1467.

[33] Dongbo Xi, Fuzhen Zhuang, Ganbin Zhou, Xiaohu Cheng, Fen Lin, and Qing He. 2020. Domain adaptation with category attention network for deep sentiment analysis. In *WWW*. 3133–3139.

[34] Yixing Xu, Chang Xu, Chao Xu, and Dacheng Tao. 2017. Multi-positive and unlabeled learning. In *IJCAI*. 3182–3188.

[35] Wei-Nan Zhang, Qingfu Zhu, Yifa Wang, Yanyan Zhao, and Ting Liu. 2019. Neural personalized response generation as domain adaptation. *WWW* (2019).

[36] Yuchen Zhang, Tianle Liu, Mingsheng Long, and Michael Jordan. 2019. Bridging Theory and Algorithm for Domain Adaptation. In *ICML*. 7404–7413.

[37] Han Zhao, Remi Tachet Des Combes, Kun Zhang, and Geoffrey Gordon. 2019. On Learning Invariant Representations for Domain Adaptation. In *ICML*. 7523–7532.

[38] Han Zhao, Shanghang Zhang, Guanhang Wu, José MF Moura, Joao P Costeira, and Geoffrey J Gordon. 2018. Adversarial multiple source domain adaptation. *NeurIPS* (2018), 8559–8570.

[39] Dawei Zhou, Lecheng Zheng, Yada Zhu, Jianbo Li, and Jingrui He. 2020. Domain adaptive multi-modality neural attention network for financial forecasting. In *Proceedings of The Web Conference 2020*. 2230–2240.

[40] Yao Zhou, Lei Ying, and Jingrui He. 2017. MultiC^2: an Optimization Framework for Learning from Task and Worker Dual Heterogeneity. In *SDM*. 579–587.

[41] Yao Zhou, Lei Ying, and Jingrui He. 2019. Multi-task Crowdsourcing via an Optimization Framework. *TKDD* 13, 3 (2019), 27:1–27:26.

A APPENDIX

A.1 Proof of Lemma 4.3

Proof. We have $\pi^t_{C+1}\cdot\mathbb{P}^t(x|y=C+1) = \mathbb{P}^t(x) - \sum_{c=1}^C \pi^t_c \mathbb{P}^t(x|y=c) = \mathbb{P}^t(x) - (1-\pi^t_{C+1})\sum_{c=1}^C \pi^s_c \mathbb{Q}^s(x|y=c)$. Then, we have

$$\begin{aligned}
\epsilon_t(h) &= \pi^t_{C+1}\cdot \mathbb{E}_{x\sim\mathbb{P}^t(x|y=C+1)}\left[\mathcal{L}(h(x), y=C+1)\right] \\
&\quad + \sum_{c=1}^{C}\pi^t_c \cdot \mathbb{E}_{x\sim\mathbb{P}^t(x|y=c)}\left[\mathcal{L}(h(x), c)\right] = \Delta_{PU} + (1-\pi^t_{C+1})\epsilon_s(h) \\
&\approx O_{PU} = (1-\pi^t_{C+1})\frac{1}{n_s}\sum_{i=1}^{n_s}\mathcal{L}(h(x^i_s), y^i_s) \\
&\quad + \frac{1}{m_t}\sum_{i=1}^{m_t}\mathcal{L}(h(x^i_t), y=C+1) - \frac{1-\pi^t_{C+1}}{n_s}\sum_{i=1}^{n_s}\mathcal{L}(h(x^i_s), y=C+1)
\end{aligned}$$

Following [11], such a PU estimator is unbiased w.r.t. all popular loss functions. □

A.2 Proof of Theorem 4.4

Proof. We know that $\mathbb{P}^t(x) = \pi^t_{C+1}\cdot\mathbb{P}^t(x|y=C+1) + \sum_{c=1}^C \pi^t_c \cdot \mathbb{P}^t(x|y=c)$ where $\pi_c = \mathbb{P}^t(y=c)$. Based on the theory of positive-unlabeled learning (PU-Learning), the target error can be given by: $\epsilon_{t_{N+1}}(h) = \sum_{c=1}^C \pi^{t_{N+1}}_c \cdot \mathbb{E}_{x\sim\mathbb{P}^{t_{N+1}}(x|y=c)}\left[\mathcal{L}(h(x), y=c)\right] + \pi^{t_{N+1}}_{C+1}\cdot \mathbb{E}_{x\sim\mathbb{P}^{t_{N+1}}(x|y=C+1)}\left[\mathcal{L}(h(x), y=C+1)\right]$ and $d_{\mathcal{H}\Delta\mathcal{H}}((\mathbb{P}^{t_{N+1}}(x|y=C+1), \mathbb{P}^{t_{N+1}}(x|y\neq C+1))) + \mathbb{E}_{x\sim\mathbb{P}^{t_{N+1}}(x|y\neq C+1)}\left[h(x)=C+1\right] = 1$.

$$\begin{aligned}
&\pi^{t_{N+1}}_{C+1}\mathbb{E}_{x\sim\mathbb{P}^{t_{N+1}}(x|y=C+1)}\left[\mathcal{L}(h(x), y=C+1)\right] \\
&\le (1-\pi^{t_{N+1}}_{C+1})M\left(1 - d_{\mathcal{H}\Delta\mathcal{H}}\left(\mathbb{P}^{t_{N+1}}(x|y=C+1), \mathbb{P}^{t_{N+1}}(x|y\neq C+1)\right)\right) \\
&\quad + \pi^{t_{N+1}}_{C+1}\mathbb{E}_{x\sim\mathbb{P}^{t_{N+1}}(x|y=C+1)}\left[\mathcal{L}(h(x), y=C+1)\right] \\
&\quad - (1-\pi^{t_{N+1}}_{C+1})\mathbb{E}_{x\sim\mathbb{P}^{t_{N+1}}(x|y\neq C+1)}\left[\mathcal{L}(h(x), y=C+1)\right] \\
&\le (1-\pi^{t_{N+1}}_{C+1})M\sum_{j=0}^{N}\alpha_j\Big(1 + d_{\mathcal{H}\Delta\mathcal{H}}\left(\mathbb{P}^{t_j}(x|y\neq C+1), \mathbb{P}^{t_{N+1}}(x|y\neq C+1)\right) \\
&\quad - d_{\mathcal{H}\Delta\mathcal{H}}\left(\mathbb{P}^{t_{N+1}}(x|y=C+1), \mathbb{P}^{t_j}(x|y\neq C+1)\right)\Big) \\
&\quad + \mathbb{E}_{x\sim\mathbb{P}^{t_{N+1}}(x)}\left[\mathcal{L}(h(x), y=C+1)\right] \\
&\quad - (1-\pi^{t_{N+1}}_{C+1})\cdot\mathbb{E}_{x\sim\mathbb{P}^{t_{N+1}}(x|y\neq C+1)}\left[\mathcal{L}(h(x), y=C+1)\right] \\
&\le (1-\pi^{t_{N+1}}_{C+1})M\cdot\sum_{j=0}^{N}\alpha_j\Big(1 + d_C(\mathbb{P}^{t_j}_{\le C}, \mathbb{P}^{t_{N+1}}_{\le C}) - d_{\mathcal{H}\Delta\mathcal{H}}(\mathbb{P}^{t_j}_{\le C}, \mathbb{P}^{t_{N+1}}_{>C})\Big) \\
&\quad + \mathbb{E}_{x\sim\mathbb{P}^{t_{N+1}}(x)}\left[\mathcal{L}(h(x), y=C+1)\right] \\
&\quad - (1-\pi^{t_{N+1}}_{C+1})\cdot\mathbb{E}_{x\sim\mathbb{P}^{t_{N+1}}(x|y\neq C+1)}\left[\mathcal{L}(h(x), y=C+1)\right]
\end{aligned}$$

Here it is easy to show $d_{\mathcal{H}\Delta\mathcal{H}}\left(\mathbb{P}^{t_j}(x|y\neq C+1), \mathbb{P}^{t_{N+1}}(x|y\neq C+1)\right) = d_{\mathcal{H}\Delta\mathcal{H}}(\mathbb{P}^{t_j}_{\le C}, \mathbb{P}^{t_{N+1}}_{\le C}) \le d_C(\mathbb{P}^{t_j}_{\le C}, \mathbb{P}^{t_{N+1}}_{\le C})$.

$$\begin{aligned}
\epsilon_{t_{N+1}}(h) &\le \sum_{c=1}^{C}\pi^{t_{N+1}}_c\cdot\mathbb{E}_{x\sim\mathbb{P}^{t_{N+1}}(x|y=c)}\left[\mathcal{L}(h(x), y=c)\right] \\
&\quad + (1-\pi^{t_{N+1}}_{C+1})M\cdot\sum_{j=0}^{N}\alpha_j\Big(1 + d_C(\mathbb{P}^{t_j}_{\le C}, \mathbb{P}^{t_{N+1}}_{\le C}) - d_{\mathcal{H}\Delta\mathcal{H}}(\mathbb{P}^{t_j}_{\le C}, \mathbb{P}^{t_{N+1}}_{>C})\Big) \\
&\quad + \mathbb{E}_{x\sim\mathbb{P}^{t_{N+1}}(x)}\left[\mathcal{L}(h(x), y=C+1)\right] \\
&\quad - \sum_{c=1}^{C}\pi^{t_{N+1}}_c\cdot\mathbb{E}_{x\sim\mathbb{P}^{t_{N+1}}(x|y=c)}\left[\mathcal{L}(h(x), y=C+1)\right] \qquad (4)
\end{aligned}$$

For the first term of Eq. (4), it is a typical target classification error within the shared classes.

$$\begin{aligned}
&\sum_{c=1}^{C}\pi^{t_{N+1}}_c\mathbb{E}_{x\sim\mathbb{P}^{t_{N+1}}(x|y=c)}\left[\mathcal{L}(h(x), y=c)\right] \\
&\le (1-\pi^{t_{N+1}}_{C+1})\Big(\sum_{j=0}^{N}\alpha_j\mathbb{E}_{(x,y)\sim\mathbb{P}^{t_j}_{\le C}}\left[\mathcal{L}(h(x), y)\right] \\
&\quad + \sum_{j=0}^{N}\alpha_j\left|\Pr_{\mathbb{P}^{t_{N+1}}_{\le C}}\left[\mathcal{L}(h(x), y)\right] - \Pr_{\mathbb{P}^{t_j}_{\le C}}\left[\mathcal{L}(h(x), y)\right]\right|\Big) \\
&\le (1-\pi^{t_{N+1}}_{C+1})\Big(\sum_{j=0}^{N}\alpha_j\mathbb{E}_{(x,y)\sim\mathbb{P}^{t_j}_{\le C}}\left[\mathcal{L}(h(x), y)\right] \\
&\quad + M\sum_{j=0}^{N}\alpha_j\left|\Pr_{\mathbb{P}^{t_{N+1}}_{\le C}}\left[h(x)\neq y\right] - \Pr_{\mathbb{P}^{t_j}_{\le C}}\left[h(x)\neq y\right]\right|\Big) \\
&\le (1-\pi^{t_{N+1}}_{C+1})\Big(\sum_{j=0}^{N}\alpha_j\mathbb{E}_{(x,y)\sim\mathbb{P}^{t_j}_{\le C}}\left[\mathcal{L}(h(x), y)\right] + M\sum_{j=0}^{N}\alpha_j d_C(\mathbb{P}^{t_j}_{\le C}, \mathbb{P}^{t_{N+1}}_{\le C})\Big)
\end{aligned}$$

where $d_C(\mathbb{P}^{t_j}_{\le C}, \mathbb{P}^{t_{N+1}}_{\le C}) \le d_{\mathcal{OS}}(\mathbb{P}^{t_j}_{\le C}, \mathbb{P}^{t_{N+1}})$. In this case, we use the observation $d_{\mathcal{H}\Delta\mathcal{H}}(\mathbb{P}^{t_j}_{\le C}, \mathbb{P}^{t_{N+1}}_{>C}) \le 1$. For the last term of Eq. (4),

$$\begin{aligned}
&-\sum_{c=1}^{C}\pi^{t_{N+1}}_c\cdot\mathbb{E}_{x\sim\mathbb{P}^{t_{N+1}}(x|y=c)}\left[\mathcal{L}(h(x), y=C+1)\right] \\
&\le (1-\pi^{t_{N+1}}_{C+1})\cdot\Big(-\sum_{j=0}^{N}\alpha_j\mathbb{E}_{x\sim\mathbb{P}^{t_j}_{X|Y\le C}}\left[\mathcal{L}(h(x), y=C+1)\right] \\
&\quad + M\sum_{j=0}^{N}\alpha_j\cdot\left|\Pr_{\mathbb{P}^{t_{N+1}}_{\le C}}\left[h(x)\neq y\right] - \Pr_{\mathbb{P}^{t_j}_{\le C}}\left[h(x)\neq y\right]\right|\Big) \\
&\le (1-\pi^{t_{N+1}}_{C+1})\Big(-\sum_{j=0}^{N}\alpha_j\mathbb{E}_{x\sim\mathbb{P}^{t_j}_{X|Y\le C}}\left[\mathcal{L}(h(x), y=C+1)\right] + M\sum_{j=0}^{N}\alpha_j d_C(\mathbb{P}^{t_j}_{\le C}, \mathbb{P}^{t_{N+1}}_{\le C})\Big)
\end{aligned}$$

Therefore, the following holds:

$$\begin{aligned}
\epsilon_{t_{N+1}}(h) &\le \mathbb{E}_{\mathbb{P}^{t_{N+1}}_X}\left[\mathcal{L}(h(x), y=C+1)\right] \\
&\quad + (1-\pi^{t_{N+1}}_{C+1})\Big(\sum_{j=0}^{N}\alpha_j\mathbb{E}_{\mathbb{P}^{t_j}_{\le C}}\left[\mathcal{L}(h(x), y)\right] + 4M\sum_{j=0}^{N}\alpha_j d_{\mathcal{OS}}(\mathbb{P}^{t_j}_{\le C}, \mathbb{P}^{t_{N+1}})\Big) \\
&\quad - (1-\pi^{t_{N+1}}_{C+1})\sum_{j=0}^{N}\alpha_j\mathbb{E}_{\mathbb{P}^{t_j}_{X|Y\le C}}\left[\mathcal{L}(h(x), y=C+1)\right] + (1-\pi^{t_{N+1}}_{C+1})M
\end{aligned}$$

where $d_{\mathcal{OS}}(\mathbb{P}^{t_j}_{\le C}, \mathbb{P}^{t_{N+1}}) = d_C(\mathbb{P}^{t_j}_{\le C}, \mathbb{P}^{t_{N+1}}_{\le C}) - \frac{1}{4}d_{\mathcal{H}\Delta\mathcal{H}}(\mathbb{P}^{t_j}_{\le C}, \mathbb{P}^{t_{N+1}}_{>C})$. □

A.3 Proof of Corollary 4.5

Proof. With probability at least $1-\delta/4(N+1)$, we have

$$\mathbb{E}_{(x,y)\sim\mathbb{P}^{t_j}_{\le C}}\left[\mathcal{L}(h(x), y)\right] \le \frac{1}{n_{t_j}}\sum_{i=1}^{n_{t_j}}\mathcal{L}(h(x^i_{t_j}), y^i_{t_j}) + M\sqrt{\frac{\log\frac{8(N+1)}{\delta}}{2n_{t_j}}}$$

We have similar results for both terms in Δ_{PU}, i.e., with probability at least $1-\delta/2$,

$$\Delta_{PU} \le \hat{\Delta}_{PU} + M\sqrt{\frac{\log\frac{8}{\delta}}{2m_{t_{N+1}}}} + (1-\pi^{t_{N+1}}_{C+1})M\sum_{j=0}^{N}\alpha_j\sqrt{\frac{\log\frac{8(N+1)}{\delta}}{2n_{t_j}}}$$

For the empirical estimate of $\mathcal{OS}$-divergence, based on the definition, it can be estimated from empirical $\mathcal{C}$-divergence and $\mathcal{H}$-divergence. Thus, with probability at least $1-\delta/4(N+1)$, we have

$$d_{O\mathcal{S}}(\mathbb{P}^{t_j}_{\leq C}, \mathbb{P}^{t_{N+1}}) = d_C(\mathbb{P}^{t_j}_{\leq C}, \mathbb{P}^{t_{N+1}}_{\leq C}) - \frac{1}{4} d_{\mathcal{H}\Delta\mathcal{H}}(\mathbb{P}^{t_j}_{\leq C}, \mathbb{P}^{t_{N+1}}_{>C})$$
$$\leq \hat{d}_{O\mathcal{S}}(\mathbb{P}^{t_j}_{\leq C}, \mathbb{P}^{t_{N+1}}) + \frac{5}{4}\Big(\hat{\mathfrak{R}}_{\mathcal{B}_{t_j}}(L_H) + \hat{\mathfrak{R}}_{\mathcal{B}_{t_{N+1}}}(L_H)\Big)$$
$$+ \frac{15}{4}\sqrt{\frac{\log\frac{16(N+1)}{\delta}}{2n_{t_j}}} + 3\sqrt{\frac{\log\frac{16(N+1)}{\delta}}{2n_{t_{N+1}}}} + \frac{3}{4}\sqrt{\frac{\log\frac{16(N+1)}{\delta}}{2(m_{t_{N+1}} - n_{t_{N+1}})}}$$

Then, we have, with probability at least $1-\delta$,

$$\epsilon_{t_{N+1}}(h) \leq (1-\pi^{t_{N+1}}_{C+1})\Big(\sum_{j=0}^{N}\alpha_j \frac{1}{n_{t_j}}\sum_{i=1}^{n_{t_j}}\mathcal{L}(h(x^i_{t_j}), y^i_{t_j})$$
$$+ 4M\sum_{j=0}^{N}\alpha_j \hat{d}_{O\mathcal{S}}(\mathbb{P}^{t_j}_{\leq C}, \mathbb{P}^{t_{N+1}})\Big) + \hat{\Delta}_{PU} + R_\delta$$

where (since $1-\pi^{t_{N+1}}_{C+1} \leq 1$)

$$R_\delta = 5M\hat{\mathfrak{R}}_{\mathcal{B}_{t_{N+1}}}(L_H) + 3M\sqrt{\frac{\log\frac{16(N+1)}{\delta}}{2(m_{t_{N+1}} - n_{t_{N+1}})}} + 13M\sqrt{\frac{\log\frac{16(N+1)}{\delta}}{2n_{t_{N+1}}}}$$
$$+ M\sum_{j=0}^{N}\alpha_j\Big(\hat{\mathfrak{R}}_{\mathcal{B}_{t_j}}(L_H) + 17\sqrt{\frac{\log\frac{16(N+1)}{\delta}}{2n_{t_j}}}\Big) + CONST$$

which completes the proof. □

A.4 Proof of Theorem 4.7

Proof. Based on the definition of classification error, we have

$$\epsilon_{t_{N+1}}(h) = \mathbb{E}_{(x,y)\sim\mathbb{P}^{t_{N+1}}}[\mathcal{L}(h(x), y)] = \sum_{x,y}\mathbb{P}^{t_{N+1}}(x,y)\mathcal{L}(h(x),y)$$
$$\geq \left|\sum_{c=1}^{C}\sum_{x}\mathbb{P}^{t_{N+1}}(x, y=c)\,(h(x)-c)\right| + \pi^{t_{N+1}}_{C+1}\epsilon^{U}_{t_{N+1}}(h)$$
$$= \left|(1-\pi^{t_{N+1}}_{C+1})\sum_{x}\mathbb{P}^{t_{N+1}}_{X|Y\leq C}(x)h(x) - \sum_{c=1}^{C} c\mathbb{P}^{t_{N+1}}(y=c)\right| + \pi^{t_{N+1}}_{C+1}\epsilon^{U}_{t_{N+1}}(h)$$

Since $\epsilon_{t_0}(h) = 0$ and $\mathbb{P}^{t_0} = \mathbb{P}^{t_0}_{\leq C}$, we have

$$\epsilon_{t_0}(h) = \sum_{x,y}\mathbb{P}^{t_0}(x,y)\,\mathcal{L}(h(x),y) = \sum_{c=1}^{C}\sum_{x}\mathbb{P}^{t_0}(x, y=c)\,\mathcal{L}(h(x), y=c)$$
$$\geq \left|\sum_{c=1}^{C}\sum_{x}\mathbb{P}^{t_0}(x, y=c)(h(x)-c)\right| = \left|\sum_{x}\mathbb{P}^{t_0}_{X|Y\leq C}(x)h(x) - \sum_{c=1}^{C} c\mathbb{P}^{t_0}(y=c)\right|$$

Thus, it holds that $\sum_x \mathbb{P}^{t_0}_{X|Y\leq C}(x)\cdot h(x) = \sum_{c=1}^{C} c\cdot\mathbb{P}^{t_0}(y=c)$ for the source the task. Combining with $\mathbb{P}^{t_i}_{X|Y\leq C} = \mathbb{P}^{t_j}_{X|Y\leq C}$ for $i, j = 0, 1, \cdots, N+1$, we can derive the lower bound as in Theorem 4.7. □

A.5 Proof of Theorem 5.1

Proof. If $d_{O\mathcal{S}}(\mathbb{P}^{t_j}_{\leq C}, \mathbb{P}^{t_{N+1}}) = 0$ for $j = 0, \cdots, N$, then for any $\alpha_j \geq 0$ and $\sum_{j=0}^{N}\alpha_j = 1$, it holds $\mathbb{P}^{t_{N+1}}_{\leq C} = \sum_{j=0}^{N}\alpha_j\mathbb{P}^{t_j}_{\leq C}$. Thus we have,

$$\pi^{t_{N+1}}_{C+1}\cdot\mathbb{P}^{t_{N+1}}(x|y=C+1) = \mathbb{P}^{t_{N+1}}(x) - \sum_{c=1}^{C}\pi^{t_{N+1}}_{c}\mathbb{P}^{t_{N+1}}(x|y=c)$$
$$= \mathbb{P}^{t_{N+1}}(x) - (1-\pi^{t_{N+1}}_{C+1})\sum_{c=1}^{C}\sum_{j=0}^{N}\alpha_j\mathbb{P}^{t_j}_{\leq C}(x, y=c)$$

Then it can be proven as the Lemma 4.3, thus we omit it here for brevity. □

A.6 Detailed Data Description

Office-Home [24] is a challenging domain adaptation benchmark. It consists of 15,500 images of 65 categories from four domains: Art, Clipart, Product, and Real-World. In this data set, we select the first 25 classes in alphabetical order as the known classes and two other classes as "unknown" for the first target task. In real scenarios, the new unknown classes might appear in the target domain over time. Thus, we select two more novel classes as "unknown" in the target domain for every time stamp. Besides, the data distribution of target task would be time evolving in real scenarios due to the changing visual environment. To this end, we generate a set of time evolving task by adding the random salt&pepper noise and rotation to the sampled raw images in every time stamp. Specifically, we generate the data of $\mathcal{D}_j$ by rotating the original images with degree O_d and adding the random salt&pepper noise with magnitude O_n, i.e., $O_d = 30\cdot(j-1)$ and $O_n = 0.05\cdot(j-1)$.

Office-31 [21] contains 4,652 images of 31 categories from three domains: Amazon, Webcam, and DSLR. In this data set, we select the first 10 classes in alphabetical order as the known classes and two other classes as "unknown" for the first target task. Thus, we select two more novel classes as "unknown" in the target domain for every time stamp. We generate a set of time evolving task by rotating the raw images with degree O_d and adding the random salt&pepper noise with magnitude O_n, i.e., $O_d = 30\cdot(j-1)$ and $O_n = 0.05\cdot(j-1)$.

Syn2Real-O [20] is a large open-set domain adaptation benchmark with over 200K images of 12 object categories from two distinct domains: Synthetic and Real. In this case, the source domain (Synthetic) uses the training data from VisDA-17 as the known set and the target domain (Real) contains the test data from VisDA-17 (known set) as well as 50k images from irrelevant categories of COCO data set (unknown set). For the target domain, we choose all the known set at every time stamp and randomly choose $j\%$ of the entire unknown set at time stamp j. Besides, we simulate the distribution sift of target domain over time by rotating the raw images with degree O_d and adding the random salt&pepper noise with magnitude O_n, i.e., $O_d = 30\cdot(j-1)$ and $O_n = 0.05\cdot(j-1)$.

In addition, we study other two evolving scenarios in Subsection 6.3 for evaluating the `OuterAdapter` algorithm (see Table 7). On Office-31 (DSLR → Webcam), we select the first 10 classes in alphabetical order as the known classes. One scenario is associated with continuously decreasing open-set classes. In this case, we select twelve novel classes as "unknown" in the initial target domain and then remove two of them at every time stamp. The other one is associated with randomly generated open-set classes. That is, we randomly choose p novel classes at every time stamp where p is a random integer choosing from 0 to 10. For both scenarios, besides choosing the open-set classes, we also generate the time evolving task by rotating all the raw images with degree O_d and adding the random salt&pepper noise with magnitude O_n, i.e., $O_d = 30\cdot(j-1)$ and $O_n = 0.05\cdot(j-1)$.