——— Supplementary Material ———
Unsupervised Continual Domain Shift Learning with Multi-Prototype Modeling

Haopeng Sun[1,2] * Yingwei Zhang[1,2] * Lumin Xu[3]
Sheng Jin[4,5] † Ping Luo[4,6] Chen Qian[5] Wentao Liu[5] Yiqiang Chen[1,2] †
[1] Beijing Key Lab. of Mobile Computing and Pervasive Device,
Institute of Computing Technology, Chinese Academy of Sciences
[2] University of Chinese Academy of Sciences
[3] The Chinese University of Hong Kong [4] The University of Hong Kong
[5] SenseTime Research and Tetras.AI [6] HKU Shanghai Intelligent Computing Research Center
sunhaopeng22s@ict.ac.cn, zhangyingwei@ict.ac.cn, jinsheng@tetras.ai, yqchen@ict.ac.cn

1. Comparison of Related Settings

In Table 1, we make comparisons between the task of Unsupervised Continual Domain Shift Learning (UCDSL) [14] with the related settings, including fine-tuning [9], continual learning [6], unsupervised domain adaptation [7], continual domain adaptation [25], source-free domain adaptation [10], and single source domain generalization [5]. UCDSL requires the model to simultaneously handle domain generalization, domain adaptation and forgetting alleviation with only unlabeled target data, making it extremely challenging.

Table 1. Comparison between UCDSL and related settings.

Topics	Source Data	Target Data	Target Label	Target Domain Generalization	Target Domain Adaptation	Forgetting Alleviation
Fine-Tuning	✗	✓	✓	✗	✓	✗
Continual Learning	✗	✓	✓	✗	✓	✓
Unsupervised DA	✓	✓	✗	✗	✓	✗
Continual DA	✓	✓	✗	✗	✓	✓
Source-Free DA	✗	✓	✗	✗	✓	✗
Single Source DG	✓	✗	✗	✓	✗	✗
UCDSL	✗	✓	✗	✓	✓	✓

Additional Related Work. Continual learning (CL), also known as incremental learning or life-long learning, aims to learn a series of new tasks without forgetting knowledge obtained from the preceding tasks. Recently, the idea of CL has been extended to *Continual Domain Adaptation (CDA)* [2, 15, 20, 21, 23] to tackle continual domain drifts in dynamic environments by avoiding catastrophic forgetting. [2] adopts domain adversarial training and uses sample replay techniques to retain performance on previous domains. [23] learns by using domain-specific memory buffer for each domain. AdaGraph [16] learns domain-specific BN parameters by encoding inter-domain relationships with domain metadata. These methods start from a labeled source domain and continually adapts to a series of target domains, while avoiding performance drop on previously seen domains.

2. Detail of Multi-Prototype Learning

Theorem 1: Error bound for single classifier (Theorem 4.1 in [27]). Let $\tilde{\mathcal{D}}_s$ and $\tilde{\mathcal{D}}_t$ denote the induced distribution over the feature space for each distribution $\mathcal{D}_s$ and $\mathcal{D}_t$ over the original input space. The following inequality holds for the risk $\varepsilon_t(\hat{h})$ with single classifier on the target domain $\mathcal{D}_t$:

$$\varepsilon_t(\hat{h}) \leq \min\{\varepsilon_s(h_s, h_t), \varepsilon_t(h_s, h_t)\} + \varepsilon_s(\hat{h}) + d_{\mathcal{H}}(\tilde{\mathcal{D}}_s, \tilde{\mathcal{D}}_t). \tag{1}$$

The first term measures the disagreement between the source and target labeling functions, the second term is the source error, and the third term quantifies the discrepancy between the marginal feature distributions. The bound identifies three key factors for successful domain adaptation: small source risk, close marginal distributions, and consistent labeling functions across domains. Note that this bound was first derived by Zhao et al. [27]. Please refer to Theorem 4.1 therein for additional details.

Theorem 2: Error bound for multi-prototype learning. Let $\tilde{\mathcal{D}}_s$ and $\tilde{\mathcal{D}}_t$ denote the induced distribution over the feature space for each distribution $\mathcal{D}_s$ and $\mathcal{D}_t$ over the original input space. The following inequality holds for the risk $\varepsilon_t(\hat{h})$ with multi-prototype learning (with mixed weights $\{w_s, w_t\}$, $w_s + w_t = 1, w_s \geq 0, w_t \geq 0, \hat{h} = w_s\hat{h}_s + w_t\hat{h}_t$) on the target domain $\mathcal{D}_t$. We derive the error bound of

*Equal contribution.
†Corresponding authors.

multi-prototype learning as follows:

$$\varepsilon_t(\hat{h}) \leq \varepsilon_t(\hat{h}, h_s) + \varepsilon_t(h_s, h_t). \tag{2}$$

Proof of Theorem 2.

$$\begin{aligned}
\varepsilon_t(\hat{h}) &= \varepsilon_t(\hat{h}, h_t) \\
&= \mathbb{E}_{X\sim\mathcal{D}_t}[|\hat{h}(X) - h_t(X)|] \\
&= \mathbb{E}_{X\sim\mathcal{D}_t}[|\hat{h}(X) - h_s(X) + h_s(X) - h_t(X)|] \\
&\leq \left[\mathbb{E}_{X\sim\mathcal{D}_t}[|\hat{h}(X) - h_s(X)|] + \mathbb{E}_{X\sim\mathcal{D}_t}[|h_s(X) - h_t(X)|]\right] \\
&= \varepsilon_t(\hat{h}, h_s) + \varepsilon_t(h_s, h_t).
\end{aligned} \tag{3}$$

Lemma 1. Let $\tilde{\mathcal{D}}_s$ and $\tilde{\mathcal{D}}_t$ denote the induced distribution over the feature space for each distribution $\mathcal{D}_s$ and $\mathcal{D}_t$ over the original input space. Then for any $h_s \in \mathcal{H}$, $h_t \in \mathcal{H}$, we have $|\varepsilon_s(h_s, h_t) - \varepsilon_t(h_s, h_t)| \leq d_{\mathcal{H}}(\tilde{\mathcal{D}}_s, \tilde{\mathcal{D}}_t)$.

Proof of Lemma 1. By definition, for any $h_s \in \mathcal{H}$, $h_t \in \mathcal{H}$, we have:

$$\begin{aligned}
&|\varepsilon_s(h_s, h_t) - \varepsilon_t(h_s, h_t)| = \sup_{h\in\mathcal{H}} |\varepsilon_s(h_s, h_t) - \varepsilon_t(h_s, h_t)| \\
&= \sup_{h\in\mathcal{H}} |\left(\mathbb{E}_{X\sim\mathcal{D}_s}[|h_s(X) - h_t(X)|] - \mathbb{E}_{X\sim\mathcal{D}_t}[|h_s(X) - h_t(X)|]\right)|.
\end{aligned} \tag{4}$$

Since $||h||_\infty \leq 1$ for all $h \in \mathcal{H}$, we have $0 \leq |h_s(X) - h_t(X)| \leq 1$ for all $X \in \mathcal{D}_s/\mathcal{D}_t$, where $h_s, h_t \in \mathcal{H}$.

Here we define a hypothesis space $\tilde{\mathcal{H}} := \{\operatorname{sgn}\left(|h_s(X) - h_t(X)| - z\right) | \, h_s, h_t \in \mathcal{H}, 0 \leq z \leq 1\}$. Then we use Fubini's theorem to bound:

$$\begin{aligned}
&|\mathbb{E}_{X\sim\mathcal{D}_s}[|h_s(X) - h_t(X)|] - \mathbb{E}_{X\sim\mathcal{D}_t}[|h_s(X) - h_t(X)|]| \\
&= \left|\int_0^1 \left(\Pr_s(|h_s(X) - h_t(X)| > z) - \Pr_t(|h_s(X) - h_t(X)| > z)\right) dz\right| \\
&\leq \int_0^1 |(\Pr_s(|h_s(X) - h_t(X)| > z) - \Pr_t(|h_s(X) - h_t(X)| > z))| \, dz \\
&= \sup_{z\in[0,1]} |(\Pr_s(|h_s(X) - h_t(X)| > z) - \Pr_t(|h_s(X) - h_t(X)| > z))|.
\end{aligned} \tag{5}$$

Combining Eq. 4 and Eq. 5, and in view of the definition of $\mathcal{H}$-divergence, we have:

$$\begin{aligned}
&\sup_{h_s,h_t\in\mathcal{H}} \sup_{z\in[0,1]} |(\Pr_s(|h_s(X) - h_t(X)| > z) - \Pr_t(|h_s(X) - h_t(X)| > z))| \\
&= 2 \sup_{\tilde{h}\in\tilde{\mathcal{H}}} |\Pr_s(\tilde{h}(X) = 1) - \Pr_t(\tilde{h}(X) = 1)| \\
&= 2 \sup_{\tilde{h}\in\tilde{\mathcal{H}}} |\Pr_s(\tilde{h}) - \Pr_t(\tilde{h})| \\
&= d_{\tilde{\mathcal{H}}}(\mathcal{D}_s, \mathcal{D}_t) \\
&= d_{\mathcal{H}}(\mathcal{D}_s, \mathcal{D}_t).
\end{aligned} \tag{6}$$

From Lemma 1 in Section 3.2 of the paper [1], we can deduce: $d_{\mathcal{H}}(\mathcal{D}_s, \mathcal{D}_t) = d_{\mathcal{H}}(\tilde{\mathcal{D}}_s, \tilde{\mathcal{D}}_t)$.

Combining all the above equations completes the proof. Note that, for notation simplicity, we opt to use $d_{\tilde{\mathcal{H}}}(\cdot)$ and $d_{\mathcal{H}}(\cdot)$ interchangeably in our paper.

Comparisons of two bounds. Furthermore, we will show that our error bound (Eq. 2) is tighter than that of the single classifier (Eq. 1).

Firstly, based on Lemma1, our bound Eq. 2 is bounded by:

$$\begin{aligned}
\varepsilon_t(\hat{h}, h_s) + \varepsilon_t(h_s, h_t) &\leq \varepsilon_s(\hat{h}, h_s) + d_{\mathcal{H}}(\tilde{\mathcal{D}}_s, \tilde{\mathcal{D}}_t) + \varepsilon_t(h_s, h_t) \\
&= \varepsilon_s(\hat{h}) + \varepsilon_t(h_s, h_t) + d_{\mathcal{H}}(\tilde{\mathcal{D}}_s, \tilde{\mathcal{D}}_t).
\end{aligned} \tag{7}$$

In addition, our bound can be rewritten as:

$$\begin{aligned}
\varepsilon_t(\hat{h}) &\leq \left[\mathbb{E}_{\mathcal{D}_t}[|\hat{h} - h_s|] + \mathbb{E}_{\mathcal{D}_t}[|h_s - h_t|]\right] \\
&= \left[\int |\hat{h} - h_s| \cdot \Pr_t(X) dX + \mathbb{E}_{\mathcal{D}_t}[|h_s - h_t|]\right] \\
&= \left[\int |\hat{h} - h_s| \cdot \Pr_s(X) \frac{\Pr_t(X)}{\Pr_s(X)} dX + \mathbb{E}_{\mathcal{D}_t}[|h_s - h_t|]\right] \\
&= \left[\mathbb{E}_{X\sim\mathcal{D}_s}\left[\frac{\Pr_t(X)}{\Pr_s(X)} \cdot |\hat{h} - h_s|\right] + \mathbb{E}_{\mathcal{D}_t}[|h_s - h_t|]\right] \\
&= \mathbb{E}_{X\sim\mathcal{D}_s}\left[\frac{\Pr_t(X)}{\Pr_s(X)} \cdot |\hat{h} - h_s|\right] + \varepsilon_t(h_s, h_t). \\
&= \frac{\Pr_t(X)}{\Pr_s(X)} \varepsilon_s(\hat{h}) + \varepsilon_t(h_s, h_t) \\
&= \varepsilon_s(\hat{h}) + \varepsilon_t(h_s, h_t) \\
&\leq \varepsilon_s(\hat{h}) + \varepsilon_s(h_s, h_t) + d_{\mathcal{H}}(\tilde{\mathcal{D}}_s, \tilde{\mathcal{D}}_t).
\end{aligned} \tag{8}$$

Note that since the density ratio $\frac{\Pr_t(X)}{\Pr_s(X)}$ is intractable and during implementation, this term is set to a constant and ignored in the above calculation. Combining Eq. 7 and Eq. 8, we can get the error bound of multi-prototype learning in UCDSL is tighter than that of single classifier, thereby completing the proof.

3. Detail of Multi-Prototype Modeling

3.1. Learning Process

Source Domain Training. To initially learn the target task, we train the multi-prototype model using the labeled source domain $\mathcal{S} = \mathcal{D}_0$. To enhance the model's generalization ability, we apply the RandMix [14] data augmentation technique. Following existing approaches [3], we minimize the cross-entropy loss over the labeled source domain data with RandMix augmentation:

$$\mathcal{L}_{\text{ST}}(\mathcal{D}_0) = \mathcal{L}^{\text{ce}}_{(x,y)\in\mathcal{D}_0}\left(f_{\text{MPM}}(R(x); \theta_{\text{MPM},0}), y\right), \tag{9}$$

where f_{MPM} is our multi-prototype model, $\theta_{\text{MPM},0}$ are the parameters trained on $\mathcal{D}_0$, $\mathcal{L}^{\text{ce}}$ is the cross-entropy loss for classification, and R represents the RandMix augmentation [14].

Pseudo-label Generation. Existing methods usually update the source model directly with unlabeled data from the target domain. However, in our framework, we adapt the domain model for the target domain to generate higher-quality pseudo-labels.

Our domain model f_{DM} for the target domain $\mathcal{D}_t$ is initialized using the parameters from the previous $\mathcal{D}_{t-1}$ multi-prototype model. This allows the domain model to leverage the generalization ability of the multi-prototype model, facilitating efficient adaptation even in the presence of a large domain gap. We train the domain model on $\mathcal{D}_t$ using information maximization and self-supervised pseudo-labeling. The adaptation loss is written as:

$$\mathcal{L}_{\text{DM}}(\mathcal{D}_t) = \mathcal{L}^{\text{shot}}_{x\in\mathcal{D}_t}\left(f_{\text{DM}}(x;\theta_{\text{DM},t})\right), \tag{10}$$

where f_{DM} is our domain model, $\theta_{\text{DM},t}$ are its parameters for domain $\mathcal{D}_t$, and $\mathcal{L}^{\text{shot}}$ is the cross-entropy loss with pseudo-labels and SHOT regularization [13].

Since the multi-prototype model contains a set of prototype classifiers, we adopt the Shannon entropy-based strategy to combine the inference results. The label is computed as:

$$y_{Label} = \frac{\sum_{i=0}^{t-1}\bar{y}_i e_i}{\sum_{i=0}^{t-1} e_i}, \tag{11}$$

where $\bar{y}_i$ is the prediction from the i-th prototype classifier, and e_i is the Shannon entropy [22] of the prediction:

$$e_i = -\sum_k \hat{p}_{ik}\log\hat{p}_{ik}, \tag{12}$$

where $\hat{p}_{ik}$ is the probability of class k for the i-th prototype classifier.

After self-training, the domain model learns the knowledge of the target domain $\mathcal{D}_t$, and we use it to generate a higher-quality pseudo-labeled dataset $\hat{\mathcal{D}}_t$. Specifically, for each unlabeled sample x, we compute its pseudo-label $\hat{y}$ as follows:

$$\hat{y} = \arg\max_k \delta_k(f_{\text{DM}}(x;\theta_{\text{DM},t})), \tag{13}$$

where $\delta_k(\cdot)$ is the k-th element of the softmax output.

Domain Adaptation Training. We create a new set of prototypes for domain $\mathcal{D}_t$. In this stage, the main goal is to adapt the newly generated prototypes to domain $\mathcal{D}_t$. We freeze the feature extractor and update the newly introduced prototypes with the following loss [13]:

$$\mathcal{L}_{\text{DA}}(\hat{\mathcal{D}}_t) = \mathcal{L}^{\text{ce}}_{(x,\hat{y})\in\hat{\mathcal{D}}_t}\left(f_{\text{MPM}}(x;\theta_{\text{MPM},t}),\hat{y}\right), \tag{14}$$

where f_{MPM} is our multi-prototype model with new prototypes, and $\theta_{\text{MPM},t}$ are the parameters of the multi-prototype model on the domain $\mathcal{D}_t$.

Domain Generation and Anti-forget Training. After completing the domain adaptation process, we perform domain generalization and anti-forget training on the model. To improve the model's generalization, we use data augmentation methods on the $\tilde{\mathcal{D}}_t$ and the replay buffer ($\mathcal{D}_r$) data, such as RandMix [14], Mixup [26], and color jittering [4]. We train multi-prototype model using empirical risk minimization (ERM) with the following domain generation loss:

$$\mathcal{L}_{\text{DG}}(\mathcal{D}_a) = \mathcal{L}^{\text{ce}}_{(x,\hat{y})\in\mathcal{D}_a}\left(f_{\text{MPM}}(R(x);\theta_{\text{MPM},t}),\hat{y}\right), \tag{15}$$

where $\mathcal{D}_a = \tilde{\mathcal{D}}_t \cup \mathcal{D}_r$ and R represents the data augmentation technique.

Mitigating forgetting is crucial in unsupervised continual domain shift learning, where limited access to previous domain data hampers performance on earlier tasks. Upon encountering a new target domain, the performance of the multi-prototype model on earlier domains degrades, a phenomenon known as catastrophic forgetting. To address this issue, we add a simple distillation loss term [24] to ensure that the model retains knowledge from previous domains, given by:

$$\mathcal{L}_{\text{AT}}(\mathcal{D}_r) = \mathcal{L}^{\text{kl}}_{(x,\hat{y})\in\mathcal{D}_r}(f_{\text{MPM}}(x;\theta_{\text{MPM},t-1})||f_{\text{MPM}}(x;\theta_{\text{MPM},t})), \tag{16}$$

where $\mathcal{L}^{\text{kl}}$ represents the KL divergence loss, and $f_{\text{MPM}}(x;\theta_{\text{MPM},t-1})$ and $f_{\text{MPM}}(x;\theta_{\text{MPM},t})$ are the predicted probabilities from the previous $t-1$ and current t multi-prototype models, respectively.

Our total loss to update the multi-prototype model via domain generation and anti-forget training is as follows:

$$\mathcal{L}_{\text{TOTAL}} = \mathcal{L}_{\text{DG}} + \mathcal{L}_{\text{AT}}. \tag{17}$$

Through these two training strategies, our multi-prototype model not only achieves good generalization on domains following $\mathcal{D}_t$, but also prevents forgetting of knowledge from $\mathcal{D}_t$ and earlier domains.

3.2. Detail of BiGE

Derivation of Equation for Label Propagation.

We rewrite:

$$\boldsymbol{Y}^* = \arg\min_{\boldsymbol{Y}} \frac{1}{2}\Big(\sum_{i,j=1}^{M+N} w_{ij}\|y_i - y_j\|_2^2 + \mu\sum_{i=1}^{M+N}\|y_i - s_i\|_2^2\Big), \tag{18}$$

as follows:

$$\mathcal{L}(\boldsymbol{Y}) = \frac{1}{2}\Big(\sum_{i,j=1}^{M+N} w_{ij}\|y_i - y_j\|_2^2 + \mu\sum_{i=1}^{M+N}\|y_i - s_i\|_2^2\Big). \tag{19}$$

This defines the cost function $\mathcal{L}$ associated with $\boldsymbol{Y}$.

$$\boldsymbol{Y}^* = \arg\min_{\boldsymbol{Y}} \mathcal{L}(\boldsymbol{Y}). \tag{20}$$

Differentiating $\mathcal{L}(\boldsymbol{Y})$ with respect to $\boldsymbol{Y}$, we have

$$\left.\frac{\partial\mathcal{L}}{\partial\boldsymbol{Y}}\right|_{\boldsymbol{Y}=\boldsymbol{Y}^*} = \boldsymbol{Y}^* - \boldsymbol{W}\boldsymbol{Y}^* + \mu(\boldsymbol{Y}^* - \boldsymbol{S}) = 0. \tag{21}$$

We expand the parentheses at the end to obtain the following expression:

$$\left.\frac{\partial \mathcal{L}}{\partial \boldsymbol{Y}}\right|_{\boldsymbol{Y}=\boldsymbol{Y}^*} = (1+\mu)\boldsymbol{Y}^* - \boldsymbol{W}\boldsymbol{Y}^* - \mu\boldsymbol{S} = 0. \quad (22)$$

Dividing both sides of the equation by (1 + μ), we get:

$$\boldsymbol{Y}^* - \frac{1}{1+\mu}\boldsymbol{W}\boldsymbol{Y}^* - \frac{\mu}{1+\mu}\boldsymbol{S} = 0. \quad (23)$$

Introducing a new variable $\lambda = \frac{1}{1+\mu}$, we get:

$$(\mathbb{I} - \lambda\boldsymbol{W})\boldsymbol{Y}^* = (1-\lambda)\boldsymbol{S}. \quad (24)$$

We define adjacency matrix $\boldsymbol{W}$ as:

$$\begin{aligned} w_{ij} &= \text{similarity}(\mathbf{x}_i, \mathbf{x}_j) \propto p(\mathbf{x}_i|\mathbf{x}_j) \\ &= \sum_{k=1}^{K} p(\mathbf{x}_i|\mathbf{c}_k)p(\mathbf{c}_k|\mathbf{x}_j) = \sum_{k=1}^{K} s_{ik} \cdot \frac{s_{jk}}{\sum_{j'=1}^{M+N} s_{j'k}}. \end{aligned} \quad (25)$$

We have $p(\mathbf{x}_i|\mathbf{c}_k) = s_{ik}$, where $1 \le i \le M+N$ and $1 \le k \le K$. $\boldsymbol{W}$ is a symmetric matrix and $w_{ij} = w_{ji}$ (w_{ij} is the element in the i-th row and j-th column of the matrix $\boldsymbol{W}$). Given the diagonal matrix $\boldsymbol{V} \in \mathbb{R}^{K\times K}$ and $v_{kk} = \sum_{i=1}^{M+N} s_{ik}$, the $\boldsymbol{W}$ can be formulated as:

$$\boldsymbol{W} = \boldsymbol{S}\boldsymbol{V}^{-1}\boldsymbol{S}^\top. \quad (26)$$

Since $\boldsymbol{W}$ is Symmetric Normalized Laplacian Matrix and $\lambda < 1$, $(\mathbb{I} - \lambda\boldsymbol{W})$ is invertible [17, 28]. We have:

$$\begin{aligned} \boldsymbol{Y}^* &= (1-\lambda)\left(\mathbb{I} - \lambda\boldsymbol{W}\right)^{-1}\boldsymbol{S} \\ &= (1-\lambda)\left(\mathbb{I} - \lambda\boldsymbol{S}\boldsymbol{V}^{-1}\boldsymbol{S}^\top\right)^{-1}\boldsymbol{S}, \end{aligned} \quad (27)$$

where $\mathbb{I}$ is the identical matrix, $\boldsymbol{V}$ is a diagonal matrix with row sums of $\boldsymbol{S}$ as its elements and $\lambda = \frac{1}{1+\mu}$.

DGF Fuse. Since the multi-prototype model contains a set of prototype classifiers, it is essential to select an appropriate fusion method. Existing approaches, such as averaging and entropy-based fusion, each have their own drawbacks. The averaging fusion method assumes equal importance for all prototypes, which is unrealistic. Prototypes and test samples from the same domain should be assigned higher weights. The entropy-based fusion method, while more adaptive, is sensitive to erroneous inference results, which can lead to amplified errors. To address these issues, we propose DGF, an iterative graph-based method for deriving more accurate weighting coefficients. We use the domain-level predictions produced by DGF to guide the fusion of prototypes from different domains. Our multi-prototype prediction is:

$$y_{Proto} = \frac{\sum_{i=0}^{t} \tilde{p}_i y_i^*}{\sum_{i=0}^{t} y_i^*}, \quad (28)$$

where $\tilde{p}_i$ is the predicted result from the i-th group of prototypes, and y_i^* is the corresponding domain-level prediction. Ablation experiments show that the DGF weighting method yields better results.

CGC Calibrate. We use the data from the replay buffer to calibrate the inference results and further improve the accuracy of the predictions. To obtain the final inference results, CGC employs a Shannon entropy-based [22] calibration strategy. The prototype-based inference results y_{Proto} and the CGC inference results y_{CGC} are combined using weighted averaging. The weights are determined by the entropies of their respective inference results. The final inference result y is computed as:

$$y = \frac{y_{Proto} \cdot e_{Proto} + y_{CGC} \cdot e_{CGC}}{e_{Proto} + e_{CGC}}, \quad (29)$$

where e_{Proto} and e_{CGC} are the entropies of y_{Proto} and y_{CGC}, respectively. The entropy [22] is calculated as: $e = -\sum_k \hat{p}_k \log \hat{p}_k$, where $\hat{p}_k$ is the probability of class k. This strategy facilitates the integration of both global and local semantic information, resulting in robust performance.

3.3. Algorithm for Multi-Prototype Modeling

The pipeline of our MPM is summarized in Algorithm 1. We train the initial multi-prototype model on the labeled source domain $\mathcal{D}_0$. The model transitions into the Unsupervised Continual Domain Shift Learning (UCDSL) phase. For each unlabeled domain $\mathcal{D}_t$, the domain model is initialized with the parameters $\theta_{\text{MPM},t-1}$ from the previous multi-prototype model and is self-training on the unlabeled dataset $\mathcal{D}_t$. The domain model generates pseudo-labels for $\mathcal{D}_t$, which are used to train the multi-prototype module $\theta_{\text{MPM},t}$. Next, we present the domain adaptation training process, along with the domain generation and anti-forgetting training process for the MPM. After training, the MPM demonstrates strong generalization on $\mathcal{D}_t$ and subsequent domains, while also preventing the forgetting of knowledge from both $\mathcal{D}_t$ and earlier domains. Finally, we employ DGF to fuse the inference results and use CGC to further calibrate the predictions.

3.4. Algorithm for Domain-aware Graph Fuser

The overall process of Domain-aware Graph Fuser (DGF) in BiGE is summarized in Algorithm 2. We initialize the prototypes using the mean vector of the labeled nodes belonging to this domain. We establish connections between nodes relying on the prototypes and use label propagation algorithm to obtain their labels. The prototypes are updated based on the inference labels with exponential running average. This procedure is repeated n_{step} times to obtain the domain labels of test samples from predicted $\boldsymbol{Y}^*$.

Algorithm 1 Multi-Prototype Modeling (MPM)

Input: Labeled source domain $\mathcal{S} = \mathcal{D}_0$; Unlabeled continuously changing target domains $\mathcal{T} = \{\mathcal{D}_1, \mathcal{D}_2, \ldots, \mathcal{D}_T\}$.
Output: Model output y.

Train the initial multi-prototype model on the labeled source domain $\mathcal{D}_0$ with Eq. 9.
for $t = 1$ to T **do**
  Initialize $\mathcal{D}_t$ domain model with $\mathcal{D}_{t-1}$ multi-prototype model.
  Train $\mathcal{D}_t$ domain model using unlabeled data $\mathcal{D}_t$ with Eq. 10.
  Generate pseudo-labeled dataset $\hat{\mathcal{D}}_t$ with Eq. 13.
  Domain adaption training $\mathcal{D}_t$ multi-prototype model using pseudo-labeled dataset $\hat{\mathcal{D}}_t$ with Eq. 14.
  Domain generation and anti-forget training $\mathcal{D}_t$ multi-prototype model using pseudo-labeled dataset $\hat{\mathcal{D}}_t$ and replay buffer $\mathcal{D}_r$ with Eq. 17.
  Obtain multi-prototype inference results y_{Proto} using DGF with Eq. 28.
  Calibrate and obtain the final results y using CGC with Eq. 29.
end for
Return: y

Algorithm 2 Domain-aware Graph Fuser (DGF)

Input: Node set $\boldsymbol{X} = [\mathbf{x_1}, \ldots, \mathbf{x}_M, \mathbf{x_{M+1}}, \ldots, \mathbf{x_{M+N}}]^\top \in \mathbb{R}^{(M+N)\times C}$ (M sample nodes $\mathbf{x_u}$ $(1 \le u \le M)$ from replay buffer with domain labels and N query nodes $\mathbf{x_v}$ $(M+1 \le v \le M+N)$ from test samples without domain labels).
Output: Test samples labels $\boldsymbol{Y}^*$.

Initialize prototypes: $\mathbf{c}_k = \frac{1}{|\boldsymbol{X}_k|} \sum_{(\mathbf{x}_i, l_i) \in \boldsymbol{X}_k} \mathbf{x}_i$.
for iter $= 1$ to n_{step} **do**
  Graph construction: $s_{vk} = \frac{\exp\left(-||\mathbf{x_v} - \mathbf{c_k}||_2^2\right)}{\sum_{k=1}^{K} \exp\left(-||\mathbf{x_v} - \mathbf{c_k}||_2^2\right)}$, $v_{kk} = \sum_{i=1}^{M+N} s_{ik}$ and $\boldsymbol{W} = \boldsymbol{S}\boldsymbol{V}^{-1}\boldsymbol{S}^\top$.
  Graph label propagation: $\boldsymbol{Y}^* = (1-\lambda)\left(\mathbb{I} - \lambda \boldsymbol{S}\boldsymbol{V}^{-1}\boldsymbol{S}^\top\right)^{-1}\boldsymbol{S}$.
  Update prototypes: $\mathbf{c}_k^* = \frac{1}{|\boldsymbol{X}|} \frac{\sum_{i=1}^{M+N} \mathbf{x}_i y_{ik}^*}{\sum_{j=1}^{M+N} y_{jk}^*}$, $\mathbf{C}_{new} = (1-\sigma)\mathbf{C} + \sigma \boldsymbol{C}^*$.
end for
Return: $\boldsymbol{Y}^*$

4. More Experimental Setup

4.1. Detail of Datasets

The PACS dataset consists data of 7 classes and 4 domains: Photo (P), Art painting (A), Cartoon (C), and Sketch (S). The Digits-5 dataset encompasses 5 domains with 10 classes (0 to 9): MNIST (mt), MNIST-M (mm), Synthetic Digits (syn), SVHN (sv), and USPS (up). For DomainNet, we follow [3] to utilize a subset composed of the top 10 classes with the highest number of images in the entire dataset. This selection is made to mitigate class imbalance. The dataset contains common objects in six different domain: Quickdraw (Qu), Clipart (Cl), Painting (Pa), Infograph (In), Sketch (Sk), and Real (Re).

Experiments are conducted using 10 distinct domain orderings from each dataset. The specific orders follow the previous work [3] and are presented in Table 2. The initial domain in each ordering serves as the source domain, while the remaining domains are treated as target domains. Three independent trials with distinct random seeds (2022, 2023, 2024) are performed per ordering.

4.2. Detail of Evaluation

We use TDA, TDG and FA as the evaluation metrics [3] to assess the model performance. To enable comparative analysis across models, we calculate the average value of each metric across all domains and introduce a composite score, termed "All", representing the overall average of all metrics combined. This composite score allows us to evaluate the overall performance of the models in a comprehensive manner. By considering both the individual metrics and the composite score, we gain a holistic understanding of the models' performance across various domains and their adaptation, generalization, and forgetting alleviation capabilities during the unsupervised continual domain shift learning process. It is worth noting that CoDAG [3] uses the DA model to evaluate TDA, while the DG model is used to evaluate TDG and FA. Although this setting will lead to increased performance, we believe it is not a practical setting as two separate models are required to be maintained. In this work, we simply use the final DG model to evaluate all metrics.

Table 2. The list of different domain orders from each dataset for the main experiments, which are referenced from [3].

Order	PACS	Digits-five	DomainNet
1	A→C→P→S	SN→MT→MM→SD→US	Re→Pa→In→Cl→Sk→Qu
2	A→C→S→P	SN→SD→MT→US→MM	Cl→In→Pa→Qu→Re→Sk
3	A→P→C→S	MM→US→MT→SD→SN	Cl→Re→In→Qu→Sk→Pa
4	C→A→S→P	MT→MM→SN→SD→US	In→Qu→Cl→Pa→Re→Sk
5	C→S→P→A	MT→MM→US→SN→SD	Pa→Sk→Qu→In→Re→Cl
6	P→A→C→S	SD→MM→SN→MT→US	Qu→Re→Cl→Pa→In→Sk
7	P→S→A→C	SD→SN→US→MM→MT	Qu→Sk→Cl→In→Pa→Re
8	P→S→C→A	SD→US→MM→SN→MT	Sk→In→Pa→Cl→Re→Qu
9	S→C→A→P	US→MT→SN→MM→SD	Sk→Re→Pa→Cl→Qu→In
10	S→P→C→A	US→SD→SN→MM→MT	Sk→Re→Qu→Pa→In→Cl

4.3. Detail of Implementation

Empirical evaluation is conducted on three benchmark datasets: PACS [12], Digits-five [7, 8, 11, 18], and DomainNet [19]. Consistent training procedures spanning 80 epochs per domain were maintained across all domains and orderings for each dataset. The cost balance coefficient hyper-parameter λ is set to 0.7 ,the exponential running average σ set to 0.4 and the number of iteration steps n_{step} is set to 25 for all datasets. For the optimization process, stochastic gradient descent with momentum 0.9, weight decay 0.0005, and a polynomial learning rate scheduler is utilized. The initial learning rates are 0.01 for PACS and Digits-five, and 0.005 for DomainNet. Mini-batch training is performed with a batch size of 64. Our data augmentation pipeline is aligned with prior studies [3, 14], which encompasses random cropping, horizontal flipping, color jittering, Mixup regularization, and grayscale.

5. More Experimental Results

Granular results across the 10 orderings on the PACS, Digits-five, and DomainNet datasets are reported in Tables 3, 4, and 5, respectively. The results of the baseline methods are referenced from [3]. Notably, the proposed Multi-Phase Model (MPM) consistently surpasses all comparative baselines across orderings, datasets, and evaluation metrics, substantiating its effectiveness and robustness.

Table 3. Comparison of the performance on the PACS dataset for different state-of-art methods in TDA, TDG, FA, and All. The results are presented for each domain order. The results of the baseline models are referenced from [3]. The best results are highlighted in bold.

Metric	Orders	SHOT	SHOT++	Tent	AdaCon	EATA	L2D	PDEN	RaTP	CoDAG	Ours
TDA	Order 1	86.7	89.4	84.0	85.8	86.7	84.5	83.7	85.5	88.3	**90.4**
	Order 2	87.8	89.4	82.0	81.6	85.3	83.6	83.3	87.5	87.9	**89.8**
	Order 3	88.7	88.8	82.5	82.8	85.6	82.9	79.9	85.6	89.0	**90.7**
	Order 4	89.2	91.2	88.2	88.7	89.2	84.6	83.0	87.6	89.9	**92.4**
	Order 5	85.2	85.4	88.6	86.4	88.2	80.1	78.2	85.6	89.8	**92.9**
	Order 6	83.1	85.3	75.7	78.6	79.2	75.5	74.0	83.2	86.0	**88.6**
	Order 7	66.9	69.9	74.4	74.0	73.0	71.3	71.6	75.9	80.6	**82.5**
	Order 8	64.0	68.8	72.5	73.9	72.3	68.5	69.8	74.9	80.0	**81.9**
	Order 9	91.5	92.2	69.6	77.8	73.0	83.0	82.5	89.6	92.6	**94.2**
	Order 10	75.9	83.2	69.8	69.7	70.4	74.4	72.1	91.3	91.7	**93.1**
	Avg.	81.9	84.4	78.7	79.9	80.3	78.8	77.8	84.7	87.6	**89.7**
TDG	Order 1	69.4	70.4	75.5	75.2	75.1	74.0	73.7	76.8	77.8	**79.5**
	Order 2	67.0	68.7	73.1	74.6	72.5	76.0	71.6	76.7	77.2	**79.2**
	Order 3	67.8	63.3	75.6	75.9	76.1	72.8	73.5	77.7	76.2	**78.6**
	Order 4	69.5	66.1	78.5	77.1	77.4	78.1	77.2	79.5	82.5	**84.5**
	Order 5	61.1	62.2	81.6	74.6	78.3	74.6	73.5	78.5	81.2	**82.8**
	Order 6	48.5	50.0	56.2	57.2	57.4	56.5	55.8	63.4	62.1	**64.7**
	Order 7	36.6	43.2	52.5	55.4	54.3	54.9	52.0	56.1	60.1	**61.8**
	Order 8	37.2	39.0	50.6	52.1	51.8	52.8	51.5	53.8	58.8	**60.9**
	Order 9	53.1	52.7	54.3	57.3	48.2	62.0	60.9	73.3	74.6	**77.0**
	Order 10	39.1	44.6	60.2	52.3	50.0	56.7	54.6	69.7	71.4	**73.0**
	Avg.	54.9	56.0	65.8	65.2	64.1	65.8	64.4	70.6	72.2	**74.2**
FA	Order 1	73.0	78.6	89.5	90.7	91.4	85.6	85.2	87.8	91.5	**93.1**
	Order 2	72.4	82.3	79.5	77.7	83.7	80.6	77.4	79.8	86.8	**89.2**
	Order 3	81.8	78.9	88.5	89.5	90.5	84.8	78.7	87.1	91.7	**93.6**
	Order 4	76.9	77.6	83.3	84.5	87.7	77.5	77.1	83.2	89.4	**91.3**
	Order 5	82.9	86.1	89.0	88.0	90.8	76.7	76.5	84.1	90.2	**92.1**
	Order 6	79.6	84.3	81.4	80.7	83.5	71.0	70.9	86.4	87.7	**89.6**
	Order 7	65.3	80.5	78.0	78.0	74.4	75.7	75.4	78.8	83.9	**85.8**
	Order 8	58.3	83.5	73.3	74.0	73.3	72.6	72.1	74.2	83.5	**85.5**
	Order 9	86.5	88.8	74.1	79.0	76.6	78.0	78.3	87.7	91.1	**92.7**
	Order 10	72.0	89.5	73.1	73.7	74.3	73.4	71.4	89.8	91.8	**94.1**
	Avg.	74.9	83.0	81.0	81.6	82.6	77.6	76.3	83.9	88.8	**90.7**
ALL	Order 1	76.4	79.5	83.0	83.9	84.4	81.4	80.9	83.4	85.9	**87.7**
	Order 2	75.7	80.1	78.2	78.0	80.5	80.1	77.4	81.3	84.0	**86.1**
	Order 3	79.4	77.0	82.2	82.7	84.1	80.2	77.4	83.5	85.6	**87.6**
	Order 4	78.5	78.3	83.3	83.4	84.8	80.1	79.1	83.4	87.3	**89.4**
	Order 5	76.4	77.9	86.4	83.0	85.8	77.1	76.1	82.7	87.1	**89.3**
	Order 6	70.4	73.2	71.1	72.2	73.4	67.7	66.9	77.7	78.6	**81.0**
	Order 7	56.3	64.5	68.3	69.1	67.2	67.3	66.3	70.3	74.9	**76.7**
	Order 8	53.2	63.8	65.5	66.7	65.8	64.6	64.5	67.6	74.1	**76.1**
	Order 9	77.0	77.9	66.0	71.4	65.9	74.3	73.9	83.5	86.1	**88.0**
	Order 10	62.3	72.4	67.7	65.2	64.9	68.2	66.0	83.6	85.0	**86.7**
	Avg.	70.6	74.5	75.2	75.6	75.7	74.1	72.9	79.7	82.9	**84.9**

Table 4. Comparison of the performance on the Digits-five dataset for different state-of-art methods in TDA, TDG, FA, and All. The results are presented for each domain order. The results of the baseline models are referenced from [3]. The best results are highlighted in bold.

Metric	Orders	SHOT	SHOT++	Tent	AdaCon	EATA	L2D	PDEN	RaTP	CoDAG	Ours
TDA	Order 1	84.0	87.5	71.5	77.4	76.8	85.9	81.9	89.7	95.5	**97.1**
	Order 2	91.6	94.8	77.5	76.0	76.9	91.3	89.5	90.7	95.7	**97.0**
	Order 3	81.2	79.9	70.7	75.8	76.4	85.9	86.2	87.8	91.8	**93.6**
	Order 4	73.8	79.6	59.9	64.9	65.0	77.6	75.3	86.8	90.9	**92.9**
	Order 5	79.7	84.9	59.5	65.3	65.8	79.3	78.3	87.5	91.5	**93.4**
	Order 6	87.0	92.1	80.2	80.5	81.1	89.7	89.0	90.0	93.6	**95.3**
	Order 7	89.9	91.2	80.9	82.1	83.2	87.6	85.2	91.6	92.6	**94.0**
	Order 8	89.0	91.5	80.5	80.2	82.2	88.6	85.9	89.7	92.6	**93.9**
	Order 9	48.4	48.8	48.7	55.7	55.4	74.2	70.9	85.9	91.2	**93.5**
	Order 10	61.2	62.9	57.3	58.3	57.1	82.9	80.3	87.1	91.1	**92.3**
	Avg.	78.6	81.3	68.7	71.6	72.0	84.3	82.3	88.7	92.7	**94.3**
TDG	Order 1	66.2	68.3	71.1	72.6	71.3	72.3	69.4	77.0	79.2	**81.3**
	Order 2	78.0	78.2	72.9	75.8	71.5	78.1	78.4	79.5	81.8	**84.0**
	Order 3	68.3	65.8	70.7	67.0	69.6	71.7	70.5	77.0	77.1	**78.5**
	Order 4	49.1	52.0	52.2	53.2	53.7	62.3	60.4	72.0	71.9	**73.6**
	Order 5	54.0	54.1	53.1	51.1	53.6	62.7	61.4	72.9	72.5	**73.8**
	Order 6	72.3	75.2	76.9	75.8	77.8	78.2	76.8	81.0	82.6	**84.1**
	Order 7	74.8	76.0	76.9	73.0	76.1	78.1	76.8	81.9	81.5	**82.5**
	Order 8	73.9	72.6	79.3	76.9	77.9	78.0	77.3	81.3	82.2	**84.7**
	Order 9	35.1	39.0	41.3	41.3	44.1	61.7	61.7	73.2	73.2	**74.8**
	Order 10	38.6	41.7	45.9	46.3	44.2	65.5	63.8	71.7	72.3	**75.5**
	Avg.	61.0	62.3	64.0	63.3	64.0	70.9	69.7	76.8	77.4	**79.3**
FA	Order 1	60.0	67.1	67.8	75.2	76.2	75.2	71.4	83.8	87.5	**88.9**
	Order 2	73.9	75.5	82.2	82.7	83.6	81.1	79.6	87.4	89.8	**90.7**
	Order 3	70.7	71.2	72.9	80.4	85.5	85.1	81.9	90.1	91.7	**93.4**
	Order 4	56.5	65.3	50.8	59.0	58.8	72.3	70.0	82.3	85.2	**86.4**
	Order 5	77.0	79.1	61.4	71.7	71.2	74.9	73.9	85.2	87.8	**89.2**
	Order 6	59.3	67.4	81.2	80.4	79.7	76.8	74.1	84.9	86.5	**88.5**
	Order 7	62.2	71.0	79.8	82.1	80.9	77.6	76.1	84.7	86.4	**89.5**
	Order 8	57.2	66.0	80.0	81.9	79.4	75.0	72.6	83.3	85.3	**87.6**
	Order 9	25.1	30.0	33.1	56.8	61.8	72.5	68.5	85.1	86.5	**88.1**
	Order 10	39.7	52.5	51.5	52.0	52.4	74.1	72.0	82.8	84.2	**85.6**
	Avg.	58.2	64.5	66.1	72.2	73.0	76.5	74.0	85.0	87.1	**88.8**
ALL	Order 1	70.1	74.3	70.1	75.1	74.8	77.8	74.2	83.5	87.4	**89.1**
	Order 2	81.2	82.8	77.5	78.2	77.3	83.5	82.5	85.9	89.1	**90.6**
	Order 3	73.4	72.3	71.4	74.4	77.2	80.9	79.5	85.0	86.9	**88.5**
	Order 4	59.8	65.6	54.3	59.0	59.2	70.7	68.6	80.4	82.7	**84.3**
	Order 5	70.2	72.7	58.0	62.7	63.5	72.3	71.2	81.9	83.9	**85.5**
	Order 6	72.9	78.2	79.4	78.9	79.5	81.6	80.0	85.3	87.6	**89.3**
	Order 7	75.6	79.4	79.2	79.1	80.1	81.1	79.4	86.1	86.8	**88.7**
	Order 8	73.4	76.7	79.9	79.7	79.8	80.5	78.6	84.8	86.7	**88.7**
	Order 9	36.2	39.3	41.0	51.3	53.8	69.5	67.0	81.4	83.6	**85.5**
	Order 10	46.5	52.4	51.6	52.2	51.2	74.2	72.0	80.5	82.5	**84.5**
	Avg.	65.9	69.4	66.2	69.1	69.6	77.2	75.3	83.5	85.7	**87.5**

Table 5. Comparison of the performance on the DomainNet dataset for different state-of-the-art methods in TDA, TDG, FA, and All. The results are presented for each domain order. The results of the baseline models are referenced from [3]. The best results are highlighted in bold.

Metric	Orders	SHOT	SHOT++	Tent	AdaCon	EATA	L2D	PDEN	RaTP	CoDAG	Ours
TDA	Order 1	68.4	70.5	59.0	60.4	60.0	59.9	60.8	68.6	70.3	**72.7**
	Order 2	69.7	66.2	28.9	66.2	65.8	56.5	54.2	72.0	74.7	**76.8**
	Order 3	72.6	73.4	65.6	68.6	69.4	54.8	52.7	66.1	74.3	**76.3**
	Order 4	51.3	53.5	54.6	52.2	52.9	57.3	55.2	61.7	67.0	**69.7**
	Order 5	68.5	70.9	60.3	60.3	58.7	56.9	55.7	64.4	74.4	**77.1**
	Order 6	63.1	65.3	51.8	52.4	55.0	49.3	50.2	56.3	63.2	**65.9**
	Order 7	47.7	48.1	50.0	50.8	51.7	41.7	40.4	58.0	59.2	**61.8**
	Order 8	72.8	73.2	67.6	71.6	70.7	64.0	63.4	67.4	76.4	**79.9**
	Order 9	74.2	75.9	67.6	71.3	69.3	61.9	62.2	72.5	76.4	**78.1**
	Order 10	71.9	72.1	31.0	67.9	71.2	59.9	61.0	66.9	74.2	**76.5**
	Avg.	66.0	66.9	53.6	62.2	62.5	56.2	55.6	65.4	71.0	**73.5**
TDG	Order 1	46.9	45.5	52.1	51.3	51.2	49.6	48.0	53.3	54.0	**56.9**
	Order 2	52.2	50.0	31.0	52.7	55.2	55.6	51.2	57.6	58.1	**60.0**
	Order 3	53.6	53.3	58.4	53.7	58.7	52.8	51.1	57.5	60.2	**63.1**
	Order 4	40.8	41.9	50.6	51.3	51.1	48.2	47.0	55.8	57.7	**60.2**
	Order 5	48.4	49.6	52.8	53.0	52.8	53.1	51.0	54.2	56.0	**60.0**
	Order 6	34.0	35.3	33.1	32.9	33.6	36.5	37.2	41.8	43.5	**45.9**
	Order 7	23.2	25.7	35.4	32.9	34.4	32.2	30.2	42.5	42.6	**45.5**
	Order 8	59.2	59.9	61.0	62.0	62.0	62.1	61.4	63.2	62.7	**64.7**
	Order 9	58.7	59.3	61.3	61.6	63.3	59.4	59.9	63.8	63.5	**66.0**
	Order 10	56.2	60.1	41.2	61.3	59.0	57.4	56.4	62.3	63.2	**66.1**
	Avg.	47.3	48.1	47.7	51.3	52.1	50.7	49.3	55.2	56.2	**58.8**
FA	Order 1	61.4	66.5	67.4	67.0	64.3	63.7	61.1	67.5	70.9	**73.5**
	Order 2	64.5	68.9	34.1	62.6	65.8	48.9	46.3	70.4	74.3	**76.6**
	Order 3	62.9	67.7	65.6	66.3	69.2	45.2	43.1	64.7	72.9	**75.3**
	Order 4	42.1	65.4	56.4	53.3	52.7	41.5	39.5	57.1	66.4	**69.6**
	Order 5	60.9	68.5	58.0	56.6	57.4	51.2	48.6	62.0	72.4	**74.7**
	Order 6	61.1	66.3	52.4	49.4	54.8	48.0	46.0	53.8	63.6	**67.9**
	Order 7	42.8	51.7	48.5	48.5	47.7	37.2	36.0	55.0	57.5	**60.1**
	Order 8	61.6	67.5	71.6	72.8	73.5	58.8	55.1	63.1	74.9	**75.6**
	Order 9	67.4	77.3	76.8	76.1	76.0	66.5	65.6	76.3	82.8	**84.9**
	Order 10	60.4	69.6	30.4	65.5	66.3	61.4	60.9	64.6	72.9	**76.0**
	Avg.	58.5	66.9	56.1	61.8	62.8	52.2	50.2	63.5	70.9	**73.4**
ALL	Order 1	58.9	60.8	59.5	59.6	58.5	57.7	56.6	63.1	65.1	**67.7**
	Order 2	62.1	61.7	31.3	60.5	62.3	53.7	50.6	66.7	69.0	**71.1**
	Order 3	63.0	64.8	63.2	62.9	65.8	50.9	49.0	62.8	69.1	**71.6**
	Order 4	44.7	53.6	53.9	52.3	52.2	49.0	47.2	58.2	63.7	**66.5**
	Order 5	59.3	63.0	57.0	56.6	56.3	53.7	51.8	60.2	67.6	**70.6**
	Order 6	52.7	55.6	45.8	44.9	47.8	44.6	44.5	50.6	56.8	**59.9**
	Order 7	37.9	41.8	44.6	44.1	44.6	37.0	35.5	51.8	53.1	**55.8**
	Order 8	64.5	66.9	66.7	68.8	68.7	61.6	60.0	64.6	71.3	**73.4**
	Order 9	66.8	70.8	68.6	69.7	69.5	62.6	62.6	70.9	74.2	**76.3**
	Order 10	62.8	67.3	34.2	64.9	65.5	59.6	59.4	64.6	70.1	**72.9**
	Avg.	57.3	60.6	52.5	58.4	59.1	53.0	51.7	61.4	66.0	**68.6**

References

[1] Isabela Albuquerque, João Monteiro, Mohammad Darvishi, Tiago H Falk, and Ioannis Mitliagkas. Generalizing to unseen domains via distribution matching. *arXiv preprint arXiv:1911.00804*, 2019.

[2] Andreea Bobu, Eric Tzeng, Judy Hoffman, and Trevor Darrell. Adapting to continuously shifting domains. *Int. Conf. Learn. Represent. Worksh.*, 2018.

[3] Wonguk Cho, Jinha Park, and Taesup Kim. Complementary domain adaptation and generalization for unsupervised continual domain shift learning. In *Int. Conf. Comput. Vis.*, pages 11442–11452, 2023.

[4] Ekin D Cubuk, Barret Zoph, Jonathon Shlens, and Quoc V Le. Randaugment: Practical automated data augmentation with a reduced search space. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition workshops*, pages 702–703, 2020.

[5] Ilke Cugu, Massimiliano Mancini, Yanbei Chen, and Zeynep Akata. Attention consistency on visual corruptions for single-source domain generalization. In *IEEE Conf. Comput. Vis. Pattern Recog.*, pages 4165–4174, 2022.

[6] Matthias De Lange, Rahaf Aljundi, Marc Masana, Sarah Parisot, Xu Jia, Aleš Leonardis, Gregory Slabaugh, and Tinne Tuytelaars. A continual learning survey: Defying forgetting in classification tasks. *IEEE Trans. Pattern Anal. Mach. Intell.*, 44(7):3366–3385, 2021.

[7] Yaroslav Ganin and Victor Lempitsky. Unsupervised domain adaptation by backpropagation. In *Int. Conf. Mach. Learn.*, pages 1180–1189. PMLR, 2015.

[8] Jonathan J. Hull. A database for handwritten text recognition research. *IEEE Trans. Pattern Anal. Mach. Intell.*, 16(5): 550–554, 1994.

[9] Ananya Kumar, Aditi Raghunathan, Robbie Jones, Tengyu Ma, and Percy Liang. Fine-tuning can distort pretrained features and underperform out-of-distribution. *arXiv preprint arXiv:2202.10054*, 2022.

[10] Jogendra Nath Kundu, Naveen Venkat, R Venkatesh Babu, et al. Universal source-free domain adaptation. In *IEEE Conf. Comput. Vis. Pattern Recog.*, pages 4544–4553, 2020.

[11] Yann LeCun, Léon Bottou, Yoshua Bengio, and Patrick Haffner. Gradient-based learning applied to document recognition. *Proceedings of the IEEE*, 86(11):2278–2324, 1998.

[12] Da Li, Yongxin Yang, Yi-Zhe Song, and Timothy M Hospedales. Deeper, broader and artier domain generalization. In *Int. Conf. Comput. Vis.*, pages 5542–5550, 2017.

[13] Jian Liang, Dapeng Hu, and Jiashi Feng. Do we really need to access the source data? source hypothesis transfer for unsupervised domain adaptation. *Int. Conf. Mach. Learn.*, pages 6028–6039, 2020.

[14] Chenxi Liu, Lixu Wang, Lingjuan Lyu, Chen Sun, Xiao Wang, and Qi Zhu. Deja vu: Continual model generalization for unseen domains. *arXiv preprint arXiv:2301.10418*, 2023.

[15] Hong Liu, Mingsheng Long, Jianmin Wang, and Yu Wang. Learning to adapt to evolving domains. *Adv. Neural Inform. Process. Syst.*, 33:22338–22348, 2020.

[16] Massimiliano Mancini, Samuel Rota Bulo, Barbara Caputo, and Elisa Ricci. Adagraph: Unifying predictive and continuous domain adaptation through graphs. In *IEEE Conf. Comput. Vis. Pattern Recog.*, pages 6568–6577, 2019.

[17] Russell Merris. Laplacian matrices of graphs: a survey. *Linear algebra and its applications*, 197:143–176, 1994.

[18] Yuval Netzer, Tao Wang, Adam Coates, Alessandro Bissacco, Baolin Wu, Andrew Y Ng, et al. Reading digits in natural images with unsupervised feature learning. In *NIPS workshop on deep learning and unsupervised feature learning*, page 4. Granada, 2011.

[19] Xingchao Peng, Qinxun Bai, Xide Xia, Zijun Huang, Kate Saenko, and Bo Wang. Moment matching for multi-source domain adaptation. In *Int. Conf. Comput. Vis.*, pages 1406–1415, 2019.

[20] Mohammad Rostami. Lifelong domain adaptation via consolidated internal distribution. *Adv. Neural Inform. Process. Syst.*, 34:11172–11183, 2021.

[21] Antoine Saporta, Arthur Douillard, Tuan-Hung Vu, Patrick Pérez, and Matthieu Cord. Multi-head distillation for continual unsupervised domain adaptation in semantic segmentation. In *IEEE Conf. Comput. Vis. Pattern Recog.*, pages 3751–3760, 2022.

[22] Claude Elwood Shannon. A mathematical theory of communication. *The Bell system technical journal*, 27(3):379–423, 1948.

[23] Shixiang Tang, Peng Su, Dapeng Chen, and Wanli Ouyang. Gradient regularized contrastive learning for continual domain adaptation. In *AAAI Conf. Artif. Intell.*, pages 2665–2673, 2021.

[24] Frederick Tung and Greg Mori. Similarity-preserving knowledge distillation. In *Proceedings of the IEEE/CVF international conference on computer vision*, pages 1365–1374, 2019.

[25] Shiqi Yang, Yaxing Wang, Joost Van De Weijer, Luis Herranz, and Shangling Jui. Generalized source-free domain adaptation. In *Int. Conf. Comput. Vis.*, pages 8978–8987, 2021.

[26] Hongyi Zhang. mixup: Beyond empirical risk minimization. *arXiv preprint arXiv:1710.09412*, 2017.

[27] Han Zhao, Remi Tachet Des Combes, Kun Zhang, and Geoffrey Gordon. On learning invariant representations for domain adaptation. In *Int. Conf. Mach. Learn.*, pages 7523–7532. PMLR, 2019.

[28] Dengyong Zhou, Olivier Bousquet, Thomas Lal, Jason Weston, and Bernhard Schölkopf. Learning with local and global consistency. *Adv. Neural Inform. Process. Syst.*, 16, 2003.