Unmixing Mean Embeddings for Domain Adaptation with Target Label Proportion

Alain Rakotomamonjy
Criteo AI Lab
Paris

Maxime Bérar
Université Rouen Normandie
LITIS UR 4108
Rouen

Mokhtar Z. Alaya
Université de Technologie de Compiègne
LMAC
Compiègne

Abstract

We introduce a novel approach to domain adaptation within the context of Learning from Label Proportions (LLP). We address the challenging scenario where labeled samples are available in the source domain, but only bags of unlabeled samples with their corresponding label proportions are accessible in the target domain. Our proposed method, bagMME (Bag Matching Mean Embeddings), tackles the distributional shift between domains by focusing on matching class-conditional distributions. A key contribution of bagMME is a simple yet effective unmixing strategy that leverages the target label proportions to estimate the target class-conditional mean embeddings. These estimated target means are then aligned with their corresponding source class-conditional means, thereby reducing the domain discrepancy. We theoretically demonstrate the soundness of our approach and its effectiveness in mitigating distributional shifts. Extensive experiments on various computer vision datasets showcase the superior performance of bagMME compared to state-of-the-art baselines. Our results highlight the critical role of incorporating target label proportions into the learning process for improved generalization on the target domain.

1 INTRODUCTION

Learning from Label Proportions (LLP) has attracted increasing interest as a weakly supervised learning framework, particularly in scenarios where individual labeling is prohibitively expensive or raises privacy concerns (Musicant et al., 2007; Yu et al., 2013; Quadrianto et al., 2008; Fan et al., 2014). In the LLP setting, the learner is provided with bags of samples, each associated with label proportions rather than individual labels, and must train a classifier capable of generalizing to unseen instances. Traditional approaches address this challenge by redesigning loss functions to account for label proportions, such as minimizing the total variation distance between predicted and true proportions (Ardehaly and Culotta, 2017) or reformulating the loss to estimate individual labels within bags (Dulac-Arnold et al., 2019; Busa-Fekete et al., 2023).

While LLP has been extensively studied in privacy-preserving contexts, real-world applications often involve data with varying privacy levels. For instance, in computational advertising, practitioners may have access to both instance-level labels, from users who consent to share their data, and bag-level proportions, from users who do not. This mixed-data scenario introduces a distributional shift between the two types of data, complicating the learning process.

In this work, we focus on domain adaptation in the LLP setting, where the source domain provides fully labeled samples, and the target domain consists of bags with label proportions. We assume a generalized target shift (Zhang et al., 2013; Tachet des Combes et al., 2020; Rakotomamonjy et al., 2022; Kirchmeyer et al., 2022; Nguyen et al., 2025) meaning that the class-conditional distributions and label proportions differ between the source and target domains. Unlike previous studies, which primarily address unsupervised target domains (Ardehaly and Culotta, 2017; Li and Culotta, 2023) or regression tasks (Singh et al., 2024), we tackle a challenging and realistic scenario: multiclass classification with instance-level labels in the source domain and varying label proportions across bags in the target domain. This variability in label proportions adds complexity, as the model must adapt to shifting class distributions at the bag level.

To address this, we propose `bagMME`, a novel domain adaptation method tailored for the LLP setting. Our approach

Proceedings of the 29th International Conference on Artificial Intelligence and Statistics (AISTATS) 2026, Tangier, Morocco. PMLR: Volume 300.

is grounded in the idea of matching mean embeddings of class-conditional distributions between the source and target domains. In the target domain, these embeddings are estimated by unmixing the bag means using the provided label proportions. A central theoretical contribution is our demonstration that, under the square loss, the optimal way to reduce the gap between source and target domain losses leads to an approximated unmixing strategy for the target bag means. This strategy is derived from analyzing the relationship between instance-level source loss and bag-level target loss, and it provides a principled discrepancy measure between the two domains. By aligning the estimated target class-conditional means with their source counterparts, we effectively mitigate the distributional shift between domains. Through extensive experiments on multiple computer vision datasets, we show that our method significantly improves classifier performance on the target domain compared to baselines that ignore label proportions in the target domain, and unsupervised importance-weighted domain adaptation methods in which true label proportions are used for computing importance weights. Additionally, our empirical results confirm the theoretical advantage of using the square loss over cross-entropy in the source domain, further validating our analysis.

In summary our contributions are the following:

- **Methodology.** we introduce a domain adaptation approach for the LLP setting, focusing on aligning class-conditional distributions between source and target domains.
- **Theoretical strategy.** we develop an approximated unmixing method to estimate target class-conditional mean embeddings using label proportions and bag means, grounded in a source-target loss alignment analysis.
- **Empirical validation.** we demonstrate the method's effectiveness through experiments on multiple computer vision datasets, outperforming state-of-the-art baselines in unsupervised domain adaptation, LLP domain adaptation, and pseudo-label-based approaches.

2 RELATED WORK

Our work is the first to address the problem of multiclass domain adaptation in the LLP setting with instance-level labels in the source domain and label proportions in the target domain. It is at the intersection of the fields of domain adaptation and LLP and is of strong interest in practical applications like computational advertising or medical data analysis.

Learning from label proportions. Literature on Learning with Label Proportions (LLP) has started focusing on estimating label proportions within bags of instances, rather than predicting individual labels. This shift emphasizes bag-level estimations, as seen in the methods proposed by Musicant et al. (2007) and Yu et al. (2013), which modify instance-level loss functions to accommodate aggregate outputs. While these approaches yield promising results in certain contexts (Quadrianto et al., 2008; Fan et al., 2014; Patrini et al., 2014), they do not necessarily guarantee accurate predictions for individual labels (Scott and Zhang, 2019). The interest in predicting instance-level labels has surged recently due to growing privacy concerns. Recent contributions, including works by Ardehaly and Culotta (2017), Dulac-Arnold et al. (2019), Liu et al. (2019) and Busa-Fekete et al. (2023), have tackled the LLP challenge by reformulating loss functions to estimate labels within bags. Notably, Busa-Fekete et al. (2023) focused on estimating expected loss at the instance level.

Domain adaptation. Works on domain adaptation (DA) has primarily concentrated on unsupervised domain adaptation (UDA) (Ben-David et al., 2010). Covariate shift (Sugiyama et al., 2008) is a key concept in DA and it is addressed by reducing the divergence in feature distributions across domains by using different techniques such as adversarial training (Ganin et al., 2016) or divergence minimization (Courty et al., 2017; Long et al., 2018; Tachet des Combes et al., 2020; Kirchmeyer et al., 2022; Rakotomamonjy et al., 2022). In the context of LLP, Ardehaly and Culotta (2016) and Li and Culotta (2023) have proposed methods for UDA with LLP, focusing on the domain adaptation from source label proportions. Singh et al. (2024) proposed a method for domain adaptation in our LLP setting of interest but it is limited to regression tasks.

In this work, we propose a new method for classification domain adaptation in a setting where target domain is provided to us with label proportions. While UDA methods can address this problem, they do not leverage this information In contrast, our method is designed to exploit this key feature by estimating the class-conditional distribution means of the source and target domains and by incorporating in the learning process a bag-level loss function that takes into account the label proportions in the target domain.

3 ADAPTING BY UNMIXING MEAN EMBEDDINGS

Setting. We are interested in a classification domain adaptation problem where we have source and target distributions p_S and p_T. Both distributions are supported on the product space $\mathcal{X} \times \mathcal{Y}$, where $\mathcal{X}$ is the feature space and $\mathcal{Y} = \{1, \cdots, C\}$ is the label space. We assume that the class-conditional distributions $p_S(\mathbf{X}|\mathbf{Y})$ and $p_T(\mathbf{X}|\mathbf{Y})$ are different, as well as the label distributions $p_S(\mathbf{Y})$ and $p_T(\mathbf{Y})$. In practice, we have access to labeled samples from the source distribution, $\boldsymbol{D}_S = \{(\mathbf{x}_i, y_i)\}_{i=1}^{n_S}$, where $\boldsymbol{x}_i \in \mathcal{X}$

and $y_i \in \mathcal{Y}$ and for the target distribution we have access to bag of samples and their corresponding label proportion $\boldsymbol{D}_T = \{(\boldsymbol{B}_i, \pi_i)\}_{i=1}^{n_T}$, where $\mathbf{x}_i \in \mathcal{X}$ and $\pi_i \in \Delta(\mathcal{Y})$. Here $\Delta(\mathcal{Y})$ is the probability simplex over $\mathcal{Y}$. Note that we do not assume any specific structure on the bags $\boldsymbol{B}_i$, hence their label proportions π_i can be any vector in $\Delta(\mathcal{Y})$.

Our goal is to learn an embedding function $g : \mathcal{X} \to \mathcal{Z}$ that maps the input space to a new space $\mathcal{Z}$ such that the distribution of the embedded source samples $g(\boldsymbol{X}_S)$ is close to the distribution of the embedded target samples $g(\boldsymbol{X}_T)$. and a classifier $f : \mathcal{Z} \to \mathcal{Y}$ that maps the embedded samples to the label space. Note that unless otherwise specified, $\mathcal{X} \in \mathbb{R}^d$ and $\mathcal{Z} \in \mathbb{R}^K$ for some d and K.

Learning Framework. The problem we are solving is a weakly supervised domain adaptation problem where we have access to labeled samples from the source distribution and only aggregated label statistics from the target distribution. Technically, this problem can be addressed through unsupervised domain adaptation methods, but these methods do not take advantage of the label statistics available in the target domain. In this work, we propose a new method that leverages those label statistics available in the target domain to learn a better embedding function and a better classifier.

Recent works on unsupervised domain adaptation have shown that, without label shift, matching the class-conditional distributions of the source and target domains is sufficient to obtain better generalization bounds (Long et al., 2018; Tachet des Combes et al., 2020; Nguyen et al., 2025). Exploiting this statement, our method jointly learns to match the class-conditional distributions of the source and target domains in the embedding space, learn a classifier trained on source data at instance level and on target data using a bag loss. We denote as ℓ_C the instance loss function, and g and f are the embedding function and the classifier, respectively. The bag loss function ℓ_B that measures the discrepancy between the predicted target label distribution and true the target label distribution is defined as

$$\ell_B(f(g(\boldsymbol{B}_i)), \pi_i) = \sum_{c=1}^{C} \left| \frac{1}{n_i} \sum_{k'=1}^{n_i} \mathbb{I}(f(g(\mathbf{x}_{k'})) = c) - \pi_i(c) \right|^2 \tag{1}$$

where n_i is the number of sample in bag i and $\mathbb{I}$ is the indicator function. Here, the bag loss function is defined as the square norm between the predicted target label distribution and the true target label distribution, but any other loss function can be used. For matching class-conditional distributions, we propose to minimize a discrepancy $\mathcal{D}_{S,T}$ between the source and target class-conditional distributions that we detail in the sequel. Hence, all together, the final optimization problem is given by:

$$\min_{g,f} \sum_{i=1}^{n_S} \ell_C(f(g(\mathbf{x}_i)), y_i) + \lambda_B \sum_{i=1}^{n_T} \ell_B(f(g(\boldsymbol{B}_i)), \pi_i) + \lambda_D \mathcal{D}_{S,T} \tag{2}$$

where n_T is the number of target bags, n_S is the number of source samples.

Estimating the class-conditional distributions through unmixing. Several works on unsupervised domain adaptation (Tachet des Combes et al., 2020; Nguyen et al., 2025) have shown that matching class-conditional distributions of the source and target domains is sufficient to obtain good generalization bounds on the target domain. In the unsupervised domain adaptation (UDA) setting, we lack direct access to the class-conditional distributions of the target domain. However, in our case, we do have access to two critical pieces of information: label proportions for each bag and mean embeddings of the bags. Hence, we can leverage these two pieces of information to recover the class-conditional distributions. This leads, for each bag, to an unmixing task: $\min_{\mathbf{m}_c} \|\mathbf{m} - \sum_{c=1}^{C} \pi(c)\mathbf{m}_c\|_2^2$, with $\mathbf{m}$ being the mean embedding of a bag, $\pi(c)$ the label proportion of class c in the bag, and $\mathbf{m}_c$ the class-conditional mean embedding of class c. This problem is inherently ill-posed. To make it better-posed, we can leverage multiple bags and assume that the class-conditional means are identical across them (note that this assumption is necessary to make the problem well-posed only for bags of finite size, as bags of infinite size would provide exact estimates of the class-conditional distributions, $p_T(\mathbf{X}|\mathbf{Y} = k)$). Then, let's $\mathbf{M} \in \mathbb{R}^{K \times n_T}$ be the matrix of the mean embeddings of the bags (stacked in columns), $\mathbf{M}_c \in \mathbb{R}^{K \times C}$ the matrix of the class-conditional means (stacked in columns and fixed across bags), and $\mathbf{P} \in \mathbb{R}^{C \times n_T}$ the matrix of the label proportions (stacked in columns). One can recover the class-conditional means by solving the following optimization problem:

$$\min_{\mathbf{M}_c} \|\mathbf{M} - \mathbf{M}_c\mathbf{P}\|_F^2 \tag{3}$$

where $\|\cdot\|_F$ is the Frobenius norm. Equation (3) capitalizes on the linear relationship between the mean embedding of a bag and the class-conditional means, weighted by the label proportions, as in a mixture model. Equation (3) is a linear unmixing problem $\mathbf{M} \approx \mathbf{M}_c\mathbf{P}$, that can be solved using the pseudo-inverse of the label proportions matrix $\mathbf{P}$ and its closed-form solution is given by:

$$\mathbf{M}_c = \mathbf{M}\mathbf{P}^T \left(\mathbf{P}\mathbf{P}^T\right)^{-1} \tag{4}$$

This solution is valid under the assumption that the label proportions matrix $\mathbf{P}$ has full row rank. i.e., the label proportions are non-degenerate across bags. Given such an estimate of the class-conditional distributions, we can derive the class-conditional discrepancy between the source

and target domains by minimizing the distance between the class-conditional means as follows:

$$\mathcal{D}_{S,T} = \sum_{c=1}^{C} \|\mathbf{m}_{S,c} - \mathbf{m}_{T,c}\|_2^2 \tag{5}$$

where $\mathbf{m}_{S,c}$ is the source class-conditional mean embedding of class c and $\mathbf{m}_{T,c}$ is the target class-conditional mean embedding of class c (as columns of the matrix $\mathbf{M}_c$) in Equation (4).

While the full unmixing solution in Equation (4) is theoretically sound, it relies on a strong assumption: that the class-conditional means are identical across all bags. In practice, this assumption does not hold, and the pseudo-inverse operation can be computationally expensive and sensitive to noise in the label proportions. To address this, we derive an approximated unmixing strategy that is both computationally efficient and empirically robust. By analyzing the relationship between the bag-level loss in the target domain and the instance-level loss in the source domain (under the squared Euclidean loss), we show that their difference is bounded by a class-conditional discrepancy between the source and target domains of the form:

$$\mathcal{D}_{S,T} = \sum_{c=1}^{C} \left\| \frac{1}{n_T} \sum_{j=1}^{n_T} \frac{\pi_j(c)}{|\boldsymbol{B}_j|} \sum_{\mathbf{x}_i \in \boldsymbol{B}_j} g(\mathbf{x}_i) - \frac{1}{n_S} \sum_{i:y_i==c} g(\mathbf{x}_i) \right\|_2 \tag{6}$$

$$= \sum_{c=1}^{C} \left\| \frac{1}{n_T} \sum_{j=1}^{n_T} \pi_j(c) \mathbf{m}_{\boldsymbol{B}_j} - \frac{n_{S,c}}{n_S} \mathbf{m}_{S,c} \right\|_2$$

where $\mathbf{m}_{S,c} = \frac{1}{n_{S,c}} \sum_{i:y_i=c} g(\mathbf{x}_i)$ is the source class-conditional mean embedding of class c, $\mathbf{m}_{\boldsymbol{B}_j} = \frac{1}{|\boldsymbol{B}_j|} \sum_{\mathbf{x}_i \in \boldsymbol{B}_j} g(\mathbf{x}_i)$ is the mean embedding of the bag $\boldsymbol{B}_j$, and $n_{S,c}$ is the number of source samples of class c.

It is important to note that the term $\pi_j(c)\mathbf{m}_{\boldsymbol{B}_j}$ corresponds to the approximated unmixing of the bag mean, derived under the simplifying assumption that $\mathbf{P}\mathbf{P}^\top \approx \mathbf{I}$. While this assumption is a rough approximation, it offers two major advantages. It is computationally efficient, avoiding the need for matrix inversion, and it is robust, as it averages class-conditional means across bags, making the method less sensitive to local variations. Crucially, this approximation does not assume that class-conditional means are identical across bags, making it more flexible and adaptable to real-world scenarios. Despite its simplicity, our empirical results demonstrate that this approximated unmixing strategy performs remarkably well in practice, exceeding the performance of the full unmixing solution while being more efficient.

Algorithm. For learning the embedding function and the classifier, the algorithm is rather standard and follows the same principles as in unsupervised domain adaptation methods. In practice, for the update of g and f, we use a stochastic gradient descent with batched source data and a random bag from the target data. The instance loss ℓ_C is computed on the source mini-batch, while the bag loss ℓ_B is computed on the target bag. The class-conditional discrepancy $\mathcal{D}_{S,T}$ is computed using the source class-conditional means from the mini-batch and the target class-conditional mean embeddings estimated from the target bag. The update of the model parameters is done by minimizing the sum of the instance loss, the bag loss, and the class-conditional discrepancy. The algorithm is summarized in Algorithm 1. Note that the definition of the class-conditional discrepancy in Eq. 6 requires the mean embeddings of all target bags. However, in practice, we have found that using the mean embedding of a single bag is sufficient to obtain good results.

Algorithm 1: `bagMME` Training Algorithm

Require: Source data $\boldsymbol{D}_S$, target data $\boldsymbol{D}_T$, batch size B, learning rate η, regularization parameter λ

Initialize g and f
while not converged **do**
  Sample a mini-batch $\{(\mathbf{x}_i, y_i)\}_{i=1}^{B}$ from $\boldsymbol{D}_S$
  Sample a random bag $(\boldsymbol{B}, \pi)$ from $\boldsymbol{D}_T$
  Compute the instance loss ℓ_C on the source mini-batch
  Compute the bag loss ℓ_B on the target bag
  Compute the output of the model on the target bag $f(g(\boldsymbol{B}))$
  Compute the class-conditional means $\mathbf{m}_{S,c}$ from the source mini-batch
  Estimate the mean of target distribution
  Compute the class-conditional discrepancy $\mathcal{D}_{S,T}$ using Eq. 6
  Compute the total loss in Equation (2)
  Update g and f using stochastic gradient descent

4 THEORY

For the purpose of our theoretical analysis, we assume that $\mathcal{X} = \mathbb{R}^d$, that $\mathcal{Y}$ is a vector set , i.e., $\mathbf{y} \in \mathbb{R}^C$, and label $\mathbf{y}_i$ associated to a sample $\mathbf{x}_i$ is a one-hot encoded vector, i.e., $\mathbf{y}_i \in \{0,1\}^C$, where C is the number of classes. Remember that our goal is to learn a function $h : \mathbb{R}^d \to \mathbb{R}^C$ that maps an input sample $\mathbf{x}$ to a vector $\mathbf{y} = h(\mathbf{x})$, with h being the composition of a feature extractor $g : \mathbb{R}^d \to \mathbb{R}^K$ and a classifier $f : \mathbb{R}^K \to \mathbb{R}^C$, i.e., $h(\mathbf{x}) = f(g(\mathbf{x}))$. For our theoretical analysis, we consider the instance-based classification problem as a vector-valued regression problem with respect to the the one-hot encoded vector $\mathbf{y}$ label.

Instance and Bag-level losses. As a vector-valued regression problem, we define the loss using *mean squared-error (mse)*. For any function $h : \mathbb{R}^d \to \mathbb{R}^C$, the loss w.r.t. to a

distribution $\boldsymbol{D}$ over $\mathbb{R}^d \times \mathbb{R}^C$ is

$$\xi(\boldsymbol{D}, h) := \mathbb{E}_{(\mathbf{x},y)\leftarrow\boldsymbol{D}}\left[\|h(\mathbf{x}) - \mathbf{y}\|^2\right],$$

where we shall let $\boldsymbol{D}$ be $\boldsymbol{D}_S$ or $\boldsymbol{D}_T$ for our purpose. The loss over a finite sample $\mathcal{U}$ of labeled points is:

$$\hat{\xi}(\mathcal{U}, h) := \frac{1}{|\mathcal{U}|}\sum_{(\mathbf{x},\mathbf{y})\in\mathcal{U}}\left[\|h(\mathbf{x}) - \mathbf{y}\|^2\right]$$

And the loss on sampled bags given their label proportions is:

$$\bar{\xi}(\mathcal{B}, h) := \frac{1}{|\mathcal{B}|}\sum_{(\boldsymbol{B},y_B)\in\mathcal{B}}\left\|\left(\frac{1}{|B|}\sum_{\mathbf{x}\in\boldsymbol{B}} h(\mathbf{x})\right) - \mathbf{y}_B\right\|^2$$

where $\mathcal{B}$ is a set of bags $\boldsymbol{B}$ with their corresponding label proportions $\mathbf{y}_B = \frac{1}{|B|}\sum_{\mathbf{x}\in\boldsymbol{B}}\mathbf{y}$. In our specific case, the source and target domains are defined by two distributions $\boldsymbol{D}_S$ and $\boldsymbol{D}_T$ over $\mathbb{R}^d \times \mathbb{R}^C$ and we assume that we have access to finite samples n_S from the source distribution $\boldsymbol{D}_S$ and n_T bags from the target distribution $\boldsymbol{D}_T$. In what follows, we assume that the number of samples in each bag is k, and we set $m = n_T$ as the number of target bags and $n_S = mk$ as the number of source samples. Hence, for source, we have the samples and labels $\mathcal{S} = \{(\mathbf{z}_i, \mathbf{l}_i)\}_{i=1}^{mk}$, and the bags are $\mathcal{B} = \{(\boldsymbol{B}_j, \mathbf{y}_{B_j})\}_{j=1}^{m}$ be the bags constructed from $\mathcal{T}$ with k samples per bag. We also assume that f is a linear function, i.e., $f(\mathbf{z}) = \mathbf{R}_h^\top\mathbf{z}$, $\mathbf{R}_h \in \mathbb{R}^{K\times C}$ is the weight matrix of the classifier. Given these assumptions, the instance-level loss for the source domain becomes:

$$\begin{aligned}\hat{\xi}(\mathcal{S}, h) &= \frac{1}{mk}\sum_{i=1}^{mk}\left[\|h(\mathbf{z}_i) - \mathbf{l}_i\|^2\right] \qquad (7)\\ &= \frac{1}{mk}\sum_{i=1}^{mk}\left[\|h(\mathbf{z}_i)\|^2 + \|\mathbf{l}_i\|^2 - 2(\mathbf{R}_h^\top g(\mathbf{z}_i))^\top\mathbf{l}_i\right]\end{aligned}$$

Similarly, the bag level loss for the target domain is:

$$\begin{aligned}\bar{\xi}(\mathcal{B}, h) &= \frac{1}{m}\sum_{j=1}^{m}\left\|\frac{1}{k}\sum_{\mathbf{x}\in\boldsymbol{B}_j} h(\mathbf{x}) - \mathbf{y}_{B_j}\right\|^2\\ &= \frac{1}{m}\sum_{j=1}^{m}\left[\left(\frac{1}{k}\sum_{\mathbf{x}\in B_j} h(\mathbf{x})\right)^T\left(\frac{1}{k}\sum_{\mathbf{x}\in B_j} h(\mathbf{x})\right)\right.\\ &\qquad \left.+\mathbf{y}_{B_j}^T\mathbf{y}_{B_j} - 2\left(\frac{1}{k}\sum_{\mathbf{x}\in B_j} h(\mathbf{x})\right)^T\mathbf{y}_{B_j}\right]\\ &\le \frac{1}{m}\sum_{j=1}^{m}\left[\frac{1}{k}\sum_{\mathbf{x}_i\in B_j}(\|h(\mathbf{x_i})\|^2 + \|\mathbf{y}_i\|^2)\right.\\ &\qquad \left.-2\left(\frac{1}{k}\sum_{\mathbf{x}\in B_j} h(\mathbf{x})\right)^T\mathbf{y}_{B_j}\right]\end{aligned}$$

where the last inequality holds because the square of the mean is less than the mean of the squares (Jensen's inequality) and the last term is the inner product between the mean of the outputs and the label vector. We also remind that the label vector $\mathbf{y}_{B_j}$ is the mean of all vector labels in the bag $\boldsymbol{B}_j$.

Now, based on the above definitions of the losses , we can claim the following lemma that provides an upper bound on the difference between the bag-level loss and the instance-level loss.

Lemma 1. *For any $h \in \mathcal{F}$,*

$$\begin{aligned}\bar{\xi}(\mathcal{B}, h) - \hat{\xi}(\mathcal{S}, h) \le 2\xi(\mathcal{S}, \mathcal{T})\|\mathbf{R}_h\|_F + \lambda'(\mathcal{S}, \mathcal{T})\\ + R(h, \mathcal{S}, \mathcal{T})\end{aligned} \qquad (8)$$

where $\lambda'(\mathcal{S}, \mathcal{T}) = \frac{1}{mk}\sum_i(\|\mathbf{y}_i\|^2 - \|\mathbf{l}_i\|^2)$ is independent of h and $R(h, \mathcal{S}, \mathcal{T}) = \frac{1}{mk}\sum_i(\|h(\mathbf{x}_i)\|^2 - \|h(\mathbf{z}_i)\|^2)$ is a label-independent regularization on $\mathcal{S}$ and $\mathcal{T}$ where

$$\xi(\mathcal{S}, \mathcal{T}) = \left\|\left(\frac{1}{mk}\sum_i \mathbf{l}_i g(\mathbf{z}_i)^T - \frac{1}{m}\sum_j \mathbf{y}_{B_j}\mathbf{u}_j^T\right)\right\|_F$$

and $\mathbf{u}_j = \frac{1}{k}\sum_{\mathbf{x}\in B_j} g(\mathbf{x})$ is the mean of the features in bag $\boldsymbol{B}_j$.

Proof. Proof is given in detail in the appendix. □

Note that the regularization term $R(h, \mathcal{S}, \mathcal{T})$ is a label-independent regularization in the sense that it does not depend on the labels in the source and target domains. However, it enforces the model to have average output norm across the source and target domains. In practice, this regularization term is small when the model is well-trained and the outputs are close to one-hot encoded vectors for any inputs or when the models outputs uniformly distributed vectors. For our purpose, this lemma says that the bag loss can be minimized by jointly minimizing the instance source loss and the class-conditional discrepancy $\xi(\mathcal{S}, \mathcal{T})$. Since, this lemma controls the empirical proportion risk minimization in the target domain, we can apply the theory of empirical proportion risk minimization to derive generalization bounds. For instance, for two-class classification problems, we can apply the theory discussed in Yu et al. (2014) for bounding the generalization error of proportion estimation and then for bounding the instance-level error in the target domain by controlling its bag loss. Finally, in order to derive the class-conditional discrepancy measure in Equation (6), we simply expand the definition of $\xi(\mathcal{S}, \mathcal{T})$ class-wise.

5 NUMERICAL EXPERIMENTS

We present in this section the empirical evaluation of our method `bagMME`, using the approximated unmixing discrepancy defined in Equation (6) and the ℓ_2 loss for the instance-

Table 1: Comparison of bagMME with state-of-the-art methods on the domain adaptation datasets. For each dataset and each adaptation task (source-target), we report the average accuracy and standard deviation over 5 runs. The best results are in bold.

Data	Problem	bagBase	daWD	bagWD	bagCasual	bagCDAN	bagIWCDAN	bagMME
Office31	A → D	85.92 ± 1.5	79.38 ± 3.0	**87.74 ± 1.3**	87.00 ± 2.1	86.53 ± 4.6	87.45 ± 2.7	87.49 ± 2.0
	A → W	84.36 ± 3.0	68.35 ± 2.8	85.66 ± 4.3	85.25 ± 2.5	87.87 ± 1.9	87.15 ± 2.2	**90.85 ± 2.1**
	D → A	70.46 ± 0.9	61.04 ± 2.0	**76.19 ± 1.6**	75.49 ± 1.4	68.96 ± 3.7	66.70 ± 1.1	74.32 ± 1.8
	D → W	97.29 ± 0.8	94.57 ± 1.9	**97.87 ± 0.7**	96.97 ± 1.7	96.85 ± 0.6	95.91 ± 0.5	97.46 ± 0.8
	W → A	73.51 ± 1.4	56.12 ± 2.4	78.29 ± 1.3	78.06 ± 1.6	75.08 ± 1.6	71.67 ± 1.6	**80.72 ± 1.6**
	W → D	99.17 ± 0.8	98.45 ± 1.2	98.78 ± 0.9	**99.20 ± 0.5**	99.20 ± 0.6	99.09 ± 0.8	98.32 ± 0.6
OfficeHome	A → C	52.94 ± 2.3	44.29 ± 0.9	45.35 ± 1.6	45.75 ± 2.4	58.98 ± 2.1	54.60 ± 1.7	**66.42 ± 1.3**
	A → P	70.64 ± 1.5	62.16 ± 0.9	64.25 ± 1.1	63.41 ± 1.5	75.98 ± 2.6	74.29 ± 0.8	**84.45 ± 1.7**
	A → RW	72.97 ± 0.3	66.55 ± 1.2	68.42 ± 0.7	68.62 ± 0.9	74.89 ± 1.7	73.29 ± 1.7	**79.12 ± 0.8**
	C → A	46.05 ± 1.5	39.04 ± 1.8	38.71 ± 2.5	38.50 ± 2.6	48.63 ± 3.8	47.86 ± 3.0	**52.31 ± 1.4**
	C → P	65.77 ± 1.8	55.54 ± 1.4	56.71 ± 1.1	57.29 ± 0.9	71.54 ± 1.7	70.56 ± 1.7	**82.73 ± 0.4**
	C → RW	65.43 ± 0.6	57.28 ± 2.7	58.24 ± 1.7	58.54 ± 1.5	66.70 ± 1.1	67.01 ± 1.6	**74.63 ± 1.6**
	P → A	50.75 ± 3.3	41.02 ± 1.2	43.06 ± 1.4	42.01 ± 1.2	53.15 ± 1.8	52.18 ± 2.8	**57.27 ± 1.3**
	P → C	48.86 ± 1.5	35.58 ± 1.3	37.20 ± 0.4	37.99 ± 1.8	53.29 ± 1.8	52.47 ± 2.2	**64.92 ± 1.8**
	P → RW	71.49 ± 1.4	64.25 ± 0.6	66.19 ± 1.0	66.60 ± 0.3	73.02 ± 1.3	72.26 ± 1.9	**77.30 ± 1.5**
	RW → A	59.81 ± 2.4	56.46 ± 2.2	56.44 ± 1.5	55.60 ± 0.8	61.92 ± 1.4	58.67 ± 1.3	**63.39 ± 1.1**
	RW → C	52.91 ± 1.7	42.64 ± 0.8	43.86 ± 2.0	44.19 ± 1.4	57.46 ± 1.1	56.57 ± 2.6	**66.26 ± 1.2**
	RW → P	76.10 ± 1.2	72.00 ± 0.8	72.40 ± 0.8	72.02 ± 0.6	81.02 ± 1.1	80.20 ± 0.8	**87.43 ± 1.3**
VisDA	12-class	77.90 ± 0.6	39.01 ± 0.9	78.12 ± 0.3	77.40 ± 0.5	77.72 ± 0.5	79.26 ± 0.5	**79.76 ± 0.7**
	3-class	95.39 ± 0.6	82.73 ± 2.7	95.81 ± 0.2	95.93 ± 0.2	95.72 ± 0.5	96.01 ± 0.1	**96.01 ± 0.5**
MNIST USPS	-	97.93 ± 0.1	66.57 ± 4.4	97.51 ± 0.3	97.68 ± 0.2	98.05 ± 0.2	**98.55 ± 0.0**	97.98 ± 0.2
USPS MNIST	-	96.74 ± 0.7	67.71 ± 2.6	96.59 ± 0.9	97.20 ± 0.5	97.06 ± 0.4	97.39 ± 0.4	**97.52 ± 0.3**
MNIST SVHN	-	82.85 ± 0.4	23.26 ± 2.4	**83.84 ± 0.5**	81.53 ± 0.1	83.64 ± 0.9	39.12 ± 36	81.14 ± 1.2
SVHN MNIST	-	96.65 ± 0.6	62.63 ± 2.6	97.06 ± 0.4	97.34 ± 0.0	96.41 ± 0.4	**97.95 ± 0.2**	97.82 ± 0.4
STL CIFAR	-	44.98 ± 1.6	40.09 ± 1.6	46.07 ± 1.1	46.09 ± 1.1	45.97 ± 1.5	48.16 ± 1.3	**48.18 ± 1.0**
CIFAR STL	-	41.59 ± 0.9	34.33 ± 2.3	39.83 ± 2.3	40.91 ± 1.3	41.50 ± 1.2	44.42 ± 0.8	**44.60 ± 1.5**

level loss and bag loss as supported by our theoretical analysis. We have conducted experiments on classical domain adaptation probblems that we adapted to bag-based learning with proportion setting. We compare our method with state-of-the-art methods on the following datasets: Office-31, Office-Home, VisDA-2017, digit datasets as well as CIFAR-10 and STL-10. We also provide ablation studies to analyze the impact of each component of our method and its sensitivity to parameters of the learning problem.

Domain Adaptation LLP methods. We have compared our method bagMME with the several baselines. bagBase is a baseline method that uses the source loss and bag loss target without any adaptation. We have also considered an unsupervised domain adaptation method, named here daWD. It is a method tailored for domain adaptation problem with both covariate and label shift (Tachet des Combes et al., 2020). It learns from the labeled source domain with an adaptation strategy that aligns the marginal distributions of the source and target using a importance-weighted domain divergence loss (different divergences are possible but here we used the Wasserstein distance) (Tachet des Combes et al., 2020; Rakotomamonjy et al., 2022; Kirchmeyer et al., 2022). While original approaches estimate label proportion in the target domain, in our setting, we can directly used the exact label proportion provided with each bag. bagWD is the daWD method combined with a bag loss on the target domain. Two variants of popular adversarial domain adaptation methods CDAN (Long et al., 2018) and importance-weighted CDAN (Zhang et al., 2019; Tachet des Combes et al., 2020) adapted to the LLP setting by adding a bag loss on the target domain are also considered. They are named here bagCDAN and bagIWCDAN. For bagIWCDAN, we have used the true label proportions of each bag for computing the importance weights. bagCasual is a state-of-the-art method unsupervised domain adaptation method that seeks at aligning class-conditional distribution support using an adversarial method (Nguyen et al., 2025). For a sake of fair comparison, we have added a bag loss to their learning problem. Note that our baselines cover the different combination of the three loss functions we used for bagMME and analyze different domain adaptation strategies. For all these methods, we have used the same bag loss on the target domain as in bagMME and use the same neural network architecture and hyperparameters for the feature extraction g and the classifier f.

Datasets, protocols and metrics. We used for comparisons the following computer vision datasets: Office-31, Office-Home, VisDA-2017, digit datasets (MNIST-USPS, USPS-MNIST, MNIST-SVHN and SVHN-MNIST) and computer vision datasets CIFAR-10 and STL-10. For the Office datasets and VisDA-2017, we have used Imagenet pre-trained features using a ResNet-50 backbone. For all

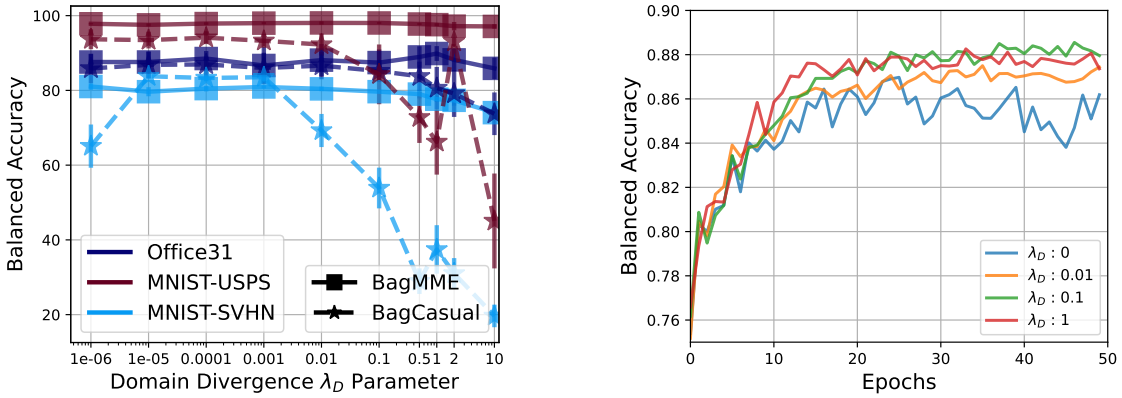

Figure 1: Analysing the effect of domain divergence regularization strength λ_D (left) for different algorithms. (right) for bagMME but across epochs during training on the Office31 A $\rightarrow$ D. Presented results are the averaged accuracy over 10 different initializations of the model.

the datasets, we have source datasets that are instance-based while for target datasets we built bags based from target samples. Regarding bag construction, we considered the following protocol. Given a fixed number of classes in a bag, we first randomly select which classes are present in the bag then we sample a random number of instances for each class to include in the bag, then randomly sample this number of instance in the classes. By doing so, we ensure that bags propose a certain level of diversity in label proportions. Bags are constructed so as to approximately reach, on average, a target bag size. For learning, validation and testing purposes, we have splitted the target bags into a train, validation and test bags. Respectively, 50%, 10% and 40% of the target domain bags are used for training, validation and testing. Note that the validation metric is the total variation distance between the true and estimated label proportions. The test metric is the instance-based balanced accuracy of the classifier on the test bags. For all methods, we have validated the hyperparameters λ_B and λ_D on the validation bags and report the test balanced accuracy for the best hyperparameters. By default, unless specified, we have used a bag size of 50. More details are provided in the appendix.

Comparison with state-of-the-art methods. Results of this comparison are presented in Table 1. We can see that bagMME performs better than competitors on most problems (19 wins over 26 problems). Among competitors, it is interesting to note that bagWD is the best performing method. This method is an augmented version of daWD that uses a bag loss on the target domain and the exact label proportion in the bag for importance-weighting This is in contrast to bagCasual that does not use those extra information for matching class-conditional distributions. Leveraging the extra information provided by the target label proportions is crucial for improved performance. Adapted versions of CDAN and IWCDAN to the LLP setting do not perform as well as bagWD in terms of wins but they achieve strong performances when the number of classes is large (such as in OfficeHome). Our conjecture on why bagWD or IWCDAN perform worse than our method is that they struggle to align the class-conditional distributions as label proportions strongly vary across bags making the importance weights harder to estimate.

Table 2: Ablating the effect of the bag loss and domain divergence loss on the performance of bagMME and bagCasual on the OfficeHome A $\rightarrow$ RW, MNIST-USPS, USPS-MNIST, MNIST-SVHN and SVHN-MNIST problems.

Data	Bag Loss	DD Loss	bagCasual	bagMME
A $\rightarrow$ RW	✓	×	65.65 $\pm$ 3.1	77.49 $\pm$ 1.6
	×	✓	65.52 $\pm$ 1.5	61.26 $\pm$ 1.3
	✓	✓	**68.62 $\pm$ 0.9**	**79.12 $\pm$ 0.8**
MNIST USPS	✓	×	97.75 $\pm$ 0.3	87.23 $\pm$ 2.8
	×	✓	90.72 $\pm$ 1.3	54.34 $\pm$ 7.2
	✓	✓	**97.68 $\pm$ 0.2**	**97.98 $\pm$ 0.2**
USPS MNIST	✓	×	97.20 $\pm$ 0.5	96.92 $\pm$ 0.5
	×	✓	96.93 $\pm$ 0.3	42.18 $\pm$ 2.8
	✓	✓	97.20 $\pm$ 0.5	**97.52 $\pm$ 0.3**
MNIST SVHN	✓	×	**82.98 $\pm$ 0.5**	24.63 $\pm$ 4.7
	×	✓	24.91 $\pm$ 1.8	14.57 $\pm$ 2.0
	✓	✓	81.53 $\pm$ 0.1	**81.14 $\pm$ 1.2**
SVHN MNIST	✓	×	96.72 $\pm$ 0.3	96.42 $\pm$ 1.3
	×	✓	60.10 $\pm$ 3.1	32.02 $\pm$ 2.2
	✓	✓	**97.34 $\pm$ 0.0**	**97.82 $\pm$ 0.4**

Comparing bagMME with variants We have also compared bagMME with a variant that uses the exact unmixing discrepancy defined in Equation (5) instead of the approximated one. We also checked the effect of using the cross-entropy loss instead of the ℓ_2 loss for the instance-level loss. Results are presented in Table 3 in appendix. Summary is that the approximated unmixing discrepancy performs better than the exact one. This is likely due to the matrix inversion used in the exact unmixing discrepancy that may be ill-conditioned. Using the cross-entropy loss instead of

the ℓ_2 loss for the instance-level loss does not lead to better performance. This is in line with our theoretical analysis that supports the use of the ℓ_2 loss for instance-level loss.

Ablating loss components. We have conducted an ablation study to analyze the impact of two loss components of on the performances of `bagMME` and `bagCasual`: the bag loss and the domain divergence loss. We have run this ablation on the OfficeHome A $\rightarrow$ RW, MNIST-USPS, USPS-MNIST, MNIST-SVHN and SVHN-MNIST problems. The protocol is similar to the one used for the comparison with state-of-the-art methods except that either λ_B or λ_D is set to 0 when the corresponding loss is not used. Results are presented in Table 2. We can see that removing the bag loss leads to a drastic drop in performance for both `bagMME` and `bagCasual`. This confirms the importance of the extra-information provided by the label proportions in the target domain and the necessity of integrating them in the learning process through a bag loss. Removing the domain divergence loss leads to a drop in performance for both methods although the drop is modest. This also suggests that the bag loss compensates the effect of the domain divergence loss and that the minimizing source loss and bag loss considerably help for aligning the class-conditional distributions. Note that for most problems, using these two losses together lead to best performances.

A qualitative illustration of the effect of the bag loss and domain divergence loss on the learned features is provided in the appendix. Figure 4, in appendix, shows the t-SNE visualization of the learned features for the source and target domains weak and strong regularization on the bag loss and domain divergence loss. We can clearly see how having both losses helps in better aligning the class-conditional distributions.

Effect of the domain divergence regularization λ_D. In Figure 1, the left plot presents the balanced accuracy of `bagMME` and competitor `bagCasual` for three problems, across different values of the domain divergence parameter λ_D. `bagMME` consistently demonstrates superior stability, maintaining high balanced accuracy across a wide range of λ_D values. Unlike `bagCasual` that shows significant drops in accuracy with increasing λ_D, `bagMME` remains robust and reliable despite some fluctuations. The right plot illustrates the training dynamics of `bagMME` over 50 epochs for different λ_D values on the Office31 A $\rightarrow$ D problem (with $\lambda_B = 2$ and all runs start from the same initial). While for all values of λ_D, we observe some instability (see the peaks) during training, we also see that there is a range of λ_D values that helps in enhancing performance, compared to $\lambda_D = 0$, confirming the effectiveness of the domain divergence loss of `bagMME`.

Effect of the bag size. We have analyzed the effect of the bag size on the performance of `bagMME` and `bagCasual` on three datasets: USPS-MNIST, MNIST-USPS and VisDA in Figure 2. For this experiment, we have varied the bag size from 20 to 200 and used the same setting as in the "comparison" experiments, and we have validated the choice of hyperparameters λ_B and λ_D for all algorithms. We can see in Figure 2 that the performance of all algorithms are stable across bag sizes, especially when compared to pure learning from bags methods (Ardehaly and Culotta, 2017; Dulac-Arnold et al., 2019; Busa-Fekete et al., 2023). In our case, the largest drop in performance is observed for `bagMME` and reaches 2.5% drop in performance when bag size is 200 compared to bag size of 50. This is clearly negligible when compared to results shown in (Dulac-Arnold et al., 2019; Busa-Fekete et al., 2023) This finding is particularly interesting and suggests that domain adaptation compensates the effect of large bag size.

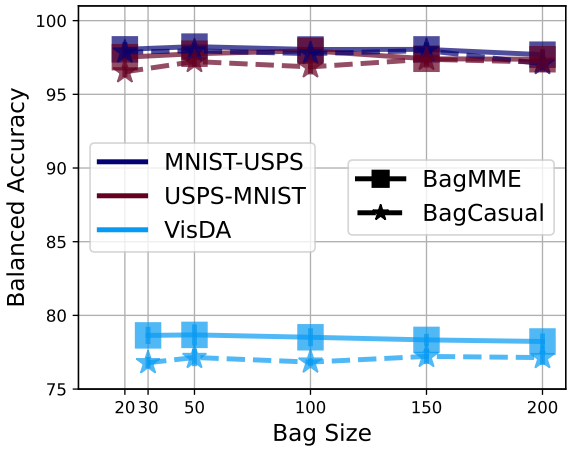

Figure 2: Effect of the bag size on the performances of `bagMME` and `bagCasual`. We can see that the performance of `bagMME` and `bagCasual` is stable across bag sizes. Compared to pure learning from bags methods, the drop in performance is negligible and it suggests that domain adaptation compensates the effect of the bag size.

Comparing with pseudo-labeling class-conditionals matching. Our results in Table 2 show that the bag loss on the target domain is crucial for the performance of all methods integrating it. We want to understand here how `bagMME` compares to a model that does class-conditional matchings using pseudo-labels obtained by applying the current model on the target bag. We have compared `bagMME` with such a method that we call `bagPL`. `bagPL` uses the objective function of `bagMME` but considers the pseudo-labels for estimating class-conditional distributions on the target domain. In practice, we use the domain divergence loss defined as $\mathcal{D}_{S,T} = \sum_{c=1}^{C} \mathcal{W}(\mu_{S_c}, \mu_{\hat{T}_c})$ where μ_{S_c} is the empirical distributions of the source domain for class c and $\mu_{\hat{T}_c}$ is the empirical distribution of the target bag based on the pseudo-labels obtained by applying the current model

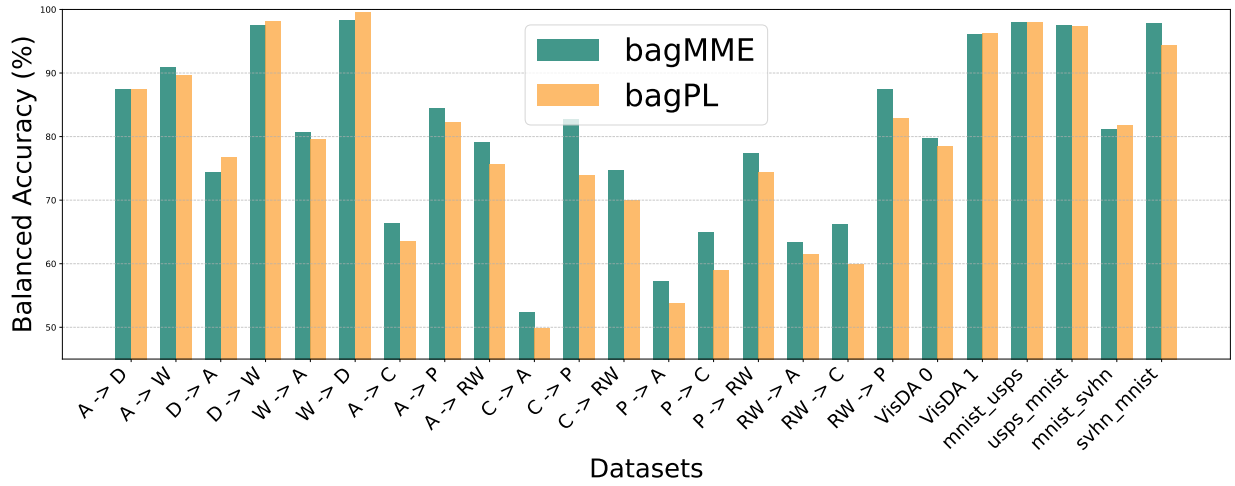

Figure 3: Comparing accuracy of `bagMME` and `bagPL` for the different datasets. `bagPL` is a method that uses pseudo-labels to estimate the target class-conditional distributions. We can see that `bagMME` performs on par with `bagPL` while being more efficient.

on the target bag and $\mathcal{W}$ is a divergence measure such as the Wasserstein distance (Peyré et al., 2019; Flamary and Courty, 2017). Results of this comparison (with same experimental setting as above) are presented in Figure 3. We can see that `bagMME` still performs better than `bagPL` with the 19 wins. In addition, `bagMME` is computationally more efficient than `bagPL` as it does not require to compute a Wasserstein distance between the class-conditional distributions and it is theorically sound.

6 CONCLUSION

In this paper, we have introduced a novel approach for domain adaptation in the context of Learning from Label Proportions (LLP), where we have access to labeled samples from the source domain and bags of samples with their corresponding label proportions from the target domain. Our method, `bagMME` builds on the core idea of matching class-conditional distributions of the source and target domains by leveraging the label proportions in the target domain. We have shown that our method is theoretically sound and that matching class-conditional distributions is an effective way to reduce the distributional shift between the source and target domains. Our results show that `bagMME` outperforms state-of-the-art methods in most cases, highlighting the importance of leveraging label proportions in the target domain.

ACKNOWLEDGMENTS

This project was provided with computing AI and storage resources by GENCI at IDRIS thanks to the grant 2024-A0171015707 on the supercomputer Jean Zay's V100 partition .

References

Ardehaly, E. M. and Culotta, A. (2016). Domain adaptation for learning from label proportions using self-training. In *IJCAI*, pages 3670–3676.

Ardehaly, E. M. and Culotta, A. (2017). Co-training for demographic classification using deep learning from label proportions. In *2017 IEEE International Conference on Data Mining Workshops (ICDMW)*, pages 1017–1024. Ieee.

Ben-David, S., Blitzer, J., Crammer, K., Kulesza, A., Pereira, F., and Vaughan, J. W. (2010). A theory of learning from different domains. *Machine learning*, 79:151–175.

Busa-Fekete, R., Choi, H., Dick, T., Gentile, C., and Munoz Medina, A. (2023). Easy learning from label proportions. *Advances in Neural Information Processing Systems*, 36:14957–14968.

Coates, A., Ng, A., and Lee, H. (2011). An analysis of single-layer networks in unsupervised feature learning. In *Proceedings of the fourteenth international conference on artificial intelligence and statistics*, pages 215–223. JMLR Workshop and Conference Proceedings.

Courty, N., Flamary, R., Tuia, D., and Rakotomamonjy, A. (2017). Optimal transport for domain adaptation. *IEEE transactions on pattern analysis and machine intelligence*, 39(9):1853–1865.

Dulac-Arnold, G., Zeghidour, N., Cuturi, M., Beyer, L., and Vert, J.-P. (2019). Deep multi-class learning from label proportions. *arXiv preprint arXiv:1905.12909*.

Fan, K., Zhang, H., Yan, S., Wang, L., Zhang, W., and Feng, J. (2014). Learning a generative classifier from label proportions. *Neurocomputing*, 139:47–55.

Flamary, R. and Courty, N. (2017). Pot python optimal transport library.

Ganin, Y., Ustinova, E., Ajakan, H., Germain, P., Larochelle, H., Laviolette, F., Marchand, M., and Lempitsky, V. (2016). Domain-adversarial training of neural networks. *The Journal of Machine Learning Research*, 17(1):2096–2030.

Hull, J. J. (1994). A database for handwritten text recognition research. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 16(5):550–554.

Kirchmeyer, M., Rakotomamonjy, A., de Bezenac, E., and patrick gallinari (2022). Mapping conditional distributions for domain adaptation under generalized target shift. In *International Conference on Learning Representations*.

Krizhevsky, A., Hinton, G., et al. (2009). Learning multiple layers of features from tiny images.

LeCun, Y., Cortes, C., and Burges, C. J. (1998). The mnist database of handwritten digits. *http://yann.lecun.com/exdb/mnist/*.

Li, X. and Culotta, A. (2023). Domain adaptation for learning from label proportions using domain-adversarial neural network. *SN Computer Science*, 4(5):615.

Liu, J., Wang, B., Qi, Z., Tian, Y., and Shi, Y. (2019). Learning from label proportions with generative adversarial networks. *Advances in neural information processing systems*, 32.

Long, M., Cao, Z., Wang, J., and Jordan, M. I. (2018). Conditional adversarial domain adaptation. *Advances in neural information processing systems*, 31.

Musicant, D. R., Christensen, J. M., and Olson, J. F. (2007). Supervised learning by training on aggregate outputs. In *Seventh IEEE International Conference on Data Mining (ICDM 2007)*, pages 252–261. IEEE.

Nguyen, A. T., Tran, L., Tong, A., Nguyen, T.-D. H., and Tran, T. (2025). Casual: Conditional support alignment for domain adaptation with label shift. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 39, pages 19668–19676.

Patrini, G., Nock, R., Rivera, P., and Caetano, T. (2014). (almost) no label no cry. *Advances in Neural Information Processing Systems*, 27.

Peng, X., Bai, Q., Xia, X., Huang, B., Saenko, K., and Wang, Z. (2017). Visda: The visual domain adaptation dataset. *arXiv preprint arXiv:1710.06924*.

Peyré, G., Cuturi, M., et al. (2019). Computational optimal transport: With applications to data science. *Foundations and Trends® in Machine Learning*, 11(5-6):355–607.

Quadrianto, N., Smola, A. J., Caetano, T. S., and Le, Q. V. (2008). Estimating labels from label proportions. In *Proceedings of the 25th international conference on Machine learning*, pages 776–783.

Rakotomamonjy, A., Flamary, R., Gasso, G., Alaya, M. E., Berar, M., and Courty, N. (2022). Optimal transport for conditional domain matching and label shift. *Machine Learning*, pages 1–20.

Saenko, K., Kulis, B., Fritz, M., and Darrell, T. (2010). Adapting visual category models to new domains. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 33(5):839–854.

Scott, C. and Zhang, J. (2019). Learning from multiple corrupted sources, with application to learning from label proportions. *arXiv preprint arXiv:1910.04665*.

Singh, S., Sharma, N., Havaldar, S., Saket, R., and Raghuveer, A. (2024). Learning from label proportions and covariate-shifted instances. *arXiv preprint arXiv:2411.12334*.

Sugiyama, M., Suzuki, T., Nakajima, S., Kashima, H., Von Bünau, P., and Kawanabe, M. (2008). Direct importance estimation for covariate shift adaptation. *Annals of the Institute of Statistical Mathematics*, 60:699–746.

Tachet des Combes, R., Zhao, H., Wang, Y.-X., and Gordon, G. J. (2020). Domain adaptation with conditional distribution matching and generalized label shift. *Advances in Neural Information Processing Systems*, 33:19276–19289.

Yu, F. X., Choromanski, K., Kumar, S., Jebara, T., and Chang, S.-F. (2014). On learning from label proportions. *arXiv preprint arXiv:1402.5902*.

Yu, F. X., Liu, D., Kumar, S., Jebara, T., and Chang, S.-F. (2013). inf-svm for learning with label proportions. In *Proceedings of the 30th International Conference on International Conference on Machine Learning-Volume 28*, pages III–504.

Zhang, K., Schölkopf, B., Muandet, K., and Wang, Z. (2013). Domain adaptation under target and conditional shift. In *International conference on machine learning*, pages 819–827. Pmlr.

Zhang, Y., Liu, T., Long, M., and Jordan, M. (2019). Bridging theory and algorithm for domain adaptation. In Chaudhuri, K. and Salakhutdinov, R., editors, *Proceedings of the 36th International Conference on Machine Learning*, volume 97 of *Proceedings of Machine Learning Research*, pages 7404–7413. PMLR.

Checklist

1. For all models and algorithms presented, check if you include:
   (a) A clear description of the mathematical setting, assumptions, algorithm, and/or model. [Yes]
   (b) An analysis of the properties and complexity (time, space, sample size) of any algorithm. [Yes]

(c) (Optional) Anonymized source code, with specification of all dependencies, including external libraries. [Not Applicable]

2. For any theoretical claim, check if you include:
   (a) Statements of the full set of assumptions of all theoretical results. [Yes/]
   (b) Complete proofs of all theoretical results. [Yes]
   (c) Clear explanations of any assumptions. [Yes]

3. For all figures and tables that present empirical results, check if you include:
   (a) The code, data, and instructions needed to reproduce the main experimental results (either in the supplemental material or as a URL). [Not Applicable]
   (b) All the training details (e.g., data splits, hyperparameters, how they were chosen). [Yes]
   (c) A clear definition of the specific measure or statistics and error bars (e.g., with respect to the random seed after running experiments multiple times). [Yes]
   (d) A description of the computing infrastructure used. (e.g., type of GPUs, internal cluster, or cloud provider). [Yes]

4. If you are using existing assets (e.g., code, data, models) or curating/releasing new assets, check if you include:
   (a) Citations of the creator If your work uses existing assets. [Yes]
   (b) The license information of the assets, if applicable. [Not Applicable]
   (c) New assets either in the supplemental material or as a URL, if applicable. [Not Applicable]
   (d) Information about consent from data providers/curators. [Not Applicable]
   (e) Discussion of sensible content if applicable, e.g., personally identifiable information or offensive content. [Not Applicable]

5. If you used crowdsourcing or conducted research with human subjects, check if you include:
   (a) The full text of instructions given to participants and screenshots. [Not Applicable]
   (b) Descriptions of potential participant risks, with links to Institutional Review Board (IRB) approvals if applicable. [Not Applicable]
   (c) The estimated hourly wage paid to participants and the total amount spent on participant compensation. [Not Applicable]

A APPENDIX

A.1 Theoretical properties

For the purpose of our theoretical analysis, we assume that $\mathcal{X} = \mathbb{R}^d$ that $\mathcal{Y}$ is a vector set , i.e., $\mathbf{y} \in \mathbb{R}^C$, and label $\mathbf{y}_i$ associated to a sample $\mathbf{x}_i$ is a one-hot encoded vector, i.e., $\mathbf{y}_i \in \{0,1\}^C$, where C is the number of classes. Remember that our goal is to learn a function $h : \mathbb{R}^d \to \mathbb{R}^C$ that maps an input sample $\mathbf{x}$ to a vector $\mathbf{y} = h(\mathbf{x})$, with h being the composition of a feature extractor $g : \mathbb{R}^d \to \mathbb{R}^K$ and a classifier $f : \mathbb{R}^K \to \mathbb{R}^C$, i.e., $h(\mathbf{x}) = f(g(\mathbf{x}))$.

Instance and Bag-level losses. We consider the problem as a vector-valued regression and thus we define our losses using *mean squared-error (mse)*. For any function $h : \mathbb{R}^d \to \mathbb{R}^C$, the loss w.r.t. to a distribution $\boldsymbol{D}$ over $\mathbb{R}^d \times \mathbb{R}^C$ is

$$\xi(\boldsymbol{D}, h) := \mathbb{E}_{(\mathbf{x},y)\leftarrow\boldsymbol{D}}\left[\|h(\mathbf{x}) - \mathbf{y}\|^2\right],$$

where we shall let $\boldsymbol{D}$ be $\boldsymbol{D}_S$ or $\boldsymbol{D}_T$ for our purpose. The loss over a finite sample $\mathcal{U}$ of labeled points is:

$$\hat{\xi}(\mathcal{U}, h) := \frac{1}{|\mathcal{U}|}\sum_{(\mathbf{x},\mathbf{y})\in\mathcal{U}}\left[\|h(\mathbf{x}) - \mathbf{y}\|^2\right]$$

Finally, we have the loss on sampled bags:

$$\bar{\xi}(\mathcal{B}, h) := \frac{1}{|\mathcal{B}|}\sum_{(\boldsymbol{B},y_B)\in\mathcal{B}}\left\|\left(\frac{1}{|B|}\sum_{\mathbf{x}\in\boldsymbol{B}} h(\mathbf{x})\right) - \mathbf{y}_B\right\|^2$$

where $\mathcal{B}$ is a set of bags $\boldsymbol{B}$ with their corresponding label proportions $\mathbf{y}_B$.

For our specific problem, we consider the source and target domains as two distributions $\boldsymbol{D}_S$ and $\boldsymbol{D}_T$ over $\mathbb{R}^d \times \mathbb{R}^C$. We assume that we have access to finite samples in those domains. For $\mathcal{S} = \{(\mathbf{z}_i, \mathbf{l}_i)\}_{i=1}^{mk}$, and $\mathcal{B} = \{(\boldsymbol{B}_j, \mathbf{y}_{B_j})\}_{j=1}^{m}$ be the bags constructed from $\mathcal{T}$ with k samples per bag. Note that we have set the number of samples in the source domain to be $n_S = mk$ A as a simplifying assumption. However, the results can be easily extended to the case where the number of samples in the source domain is different from mk. We also assume that f is a linear function, i.e., $f(\mathbf{z}) = \mathbf{R}_h^\top \mathbf{z}$, $\mathbf{R}_h \in \mathbb{R}^{K\times C}$ is the weight vector of the classifier.

In this case, the instance-level loss for the source domain is:

$$\hat{\xi}(\mathcal{S}, h) = \frac{1}{mk}\sum_{i=1}^{mk}\left[\|h(\mathbf{z}_i) - \mathbf{l}_i\|^2\right] \tag{9}$$

$$= \frac{1}{mk}\sum_{i=1}^{mk}\left[\|h(\mathbf{z}_i)\|^2 + \|\mathbf{l}_i\|^2 - 2(\mathbf{R}_h^\top g(\mathbf{z}_i))^\top \mathbf{l}_i\right] \tag{10}$$

$$\tag{11}$$

The bag level loss for the target domain is:

$$\begin{aligned}
\bar{\xi}(\mathcal{B}, h) &= \frac{1}{m}\sum_{j=1}^{m}\left\|\frac{1}{k}\sum_{\mathbf{x}\in\boldsymbol{B}_j} h(\mathbf{x}) - \mathbf{y}_{B_j}\right\|^2 \\
&= \frac{1}{m}\sum_{j=1}^{m}\left[\left(\frac{1}{k}\sum_{\mathbf{x}\in B_j} h(\mathbf{x})\right)^T\left(\frac{1}{k}\sum_{\mathbf{x}\in B_j} h(\mathbf{x})\right) + \mathbf{y}_{B_j}^T\mathbf{y}_{B_j} - 2\left(\frac{1}{k}\sum_{\mathbf{x}\in B_j} h(\mathbf{x})\right)^T \mathbf{y}_{B_j}\right] \\
&\leq \frac{1}{m}\sum_{j=1}^{m}\left[\frac{1}{k}\sum_{\mathbf{x}_i\in B_j}\left(\|h(\mathbf{x_i})\|^2 + \|\mathbf{y}_i\|^2\right) - 2\left(\frac{1}{k}\sum_{\mathbf{x}\in B_j} h(\mathbf{x})\right)^T \mathbf{y}_{B_j}\right]
\end{aligned}$$

where the last inequality holds because the square of the mean is less than the mean of the squares (Jensen's inequality) and the last term is the inner product between the mean of the outputs and the label vector. We also remind that the label vector $\mathbf{y}_{B_j}$ is the mean of the vector labels in the bag $\boldsymbol{B}_j$.

Based on the above definitions, we can write the difference between the bag-level loss and the instance-level loss as follows:

$$\bar{\xi}(\mathcal{B},h)-\hat{\xi}(\mathcal{S},h)\leq \frac{1}{m}\sum_j\left[\frac{1}{k}\sum_{i:\mathbf{x}_i\in B_j}(\|h(\mathbf{x}_i)\|^2+\|\mathbf{y}_i\|^2)-2\left(\frac{1}{k}\sum_{i:\mathbf{x}_i\in B_j}h(\mathbf{x}_i)\right)^T\mathbf{y}_{B_j}\right] \tag{12}$$

$$-\frac{1}{mk}\sum_i\left[\|h(\mathbf{z}_i)\|^2+\|\mathbf{l}_i\|^2-2(h(\mathbf{z}_i))^T\mathbf{l}_i\right]$$

$$\leq \frac{1}{mk}\sum_i(\|h(\mathbf{x}_i)\|^2-\|h(\mathbf{z}_i)\|^2)+\frac{1}{mk}\sum_i(\|\mathbf{y}_i\|^2-\|\mathbf{l}_i\|^2)$$

$$+\frac{2}{m}\left[\frac{1}{k}\sum_{i=1}^{mk}(h(\mathbf{z}_i))^T\mathbf{l}_i-\sum_j\left(\frac{1}{k}\sum_{i:\mathbf{x}_i\in B_j}h(\mathbf{x}_i)\right)^T\mathbf{y}_{B_j}\right]$$

$$\leq \frac{1}{mk}\sum_i(\|h(\mathbf{x}_i)\|^2-\|h(\mathbf{z}_i)\|^2)+\frac{1}{mk}\sum_i(\|\mathbf{y}_i\|^2-\|\mathbf{l}_i\|^2)$$

$$+\frac{2}{m}\left[\frac{1}{k}\sum_{i=1}^{mk}(\mathbf{R}_h^\top g(\mathbf{z}_i))^T\mathbf{l}_i-\sum_j\left(\frac{1}{k}\sum_{i:\mathbf{x}_i\in B_j}\mathbf{R}_h^\top g(\mathbf{x}_i)\right)^T\mathbf{y}_{B_j}\right] \tag{13}$$

if we denote as $\mathbf{u}_j=\frac{1}{k}\sum_i g(\mathbf{x}_i)$ then the last term in the equation above can be rewritten as:

$$\frac{2}{m}\left[\frac{1}{k}\sum_{i=1}^{mk}(\mathbf{R}_h^\top g(\mathbf{z}_i))^T\mathbf{l}_i-\sum_j\left(\frac{1}{k}\sum_{i:\mathbf{x}_i\in B_j}\mathbf{R}_h^\top g(\mathbf{x}_i)\right)^T\mathbf{y}_{B_j}\right]=2\left(\frac{1}{mk}\sum_i(\mathbf{R}_h^\top g(\mathbf{z}_i))^T\mathbf{l}_i-\frac{1}{m}\sum_j(\mathbf{R}_h^\top\mathbf{u}_j)^T\mathbf{y}_{B_j}\right)$$

Now we would like to write this last term

$$\left(\frac{1}{mk}\sum_i(\mathbf{R}_h^T g(\mathbf{z}_i))^T\mathbf{l}_i-\frac{1}{m}\sum_j(\mathbf{R}_h^T\mathbf{u}_j)^T\mathbf{y}_{B_j}\right)$$

in a form that allows us to factor out $\mathbf{R}_h$. We will transform this expression by introducing the trace operator and use the following properties of the trace: (i) For column vectors $\mathbf{a}$ and $\mathbf{b}$, their dot product $\mathbf{a}^\top\mathbf{b}$ can be expressed as $\text{Tr}(\mathbf{b}\mathbf{a}^\top)$, (ii) For matrices $\mathbf{X},\mathbf{Y},\mathbf{Z}$ of compatible dimensions, the cyclic property of the trace states that $\text{Tr}(\mathbf{XYZ})=\text{Tr}(\mathbf{YZX})=\text{Tr}(\mathbf{ZXY})$.

Hence, in the first term $\frac{1}{mk}\sum_i(\mathbf{R}_h^T g(\mathbf{z}_i))^T\mathbf{l}_i$, we can rewrite the dot product $(\mathbf{R}_h^T g(\mathbf{z}_i))^T\mathbf{l}_i$ as

$$(\mathbf{R}_h^T g(\mathbf{z}_i))^T\mathbf{l}_i=\text{Tr}(\mathbf{l}_i(\mathbf{R}_h^T g(\mathbf{z}_i))^T)=\text{Tr}(\mathbf{l}_i(g(\mathbf{z}_i))^T\mathbf{R}_h)=\text{Tr}(\mathbf{R}_h\mathbf{l}_i(g(\mathbf{z}_i))^T) \tag{14}$$

For the second term $\frac{1}{m}\sum_j(\mathbf{R}_h^T\mathbf{u}_j)^T\mathbf{y}_{B_j}$, we can apply the same transformation:

$$(\mathbf{R}_h^T\mathbf{u}_j)^T\mathbf{y}_{B_j}=\text{Tr}(\mathbf{y}_{B_j}(\mathbf{R}_h^T\mathbf{u}_j)^T)=\text{Tr}(\mathbf{y}_{B_j}\mathbf{u}_j^T\mathbf{R}_h)=\text{Tr}(\mathbf{R}_h\mathbf{y}_{B_j}\mathbf{u}_j^T) \tag{15}$$

where we have used the same properties of the trace as before. Substituting these transformed terms back into the original expression gives us:

$$\left(\frac{1}{mk}\sum_i\text{Tr}(\mathbf{R}_h\mathbf{l}_i(g(\mathbf{z}_i))^T)-\frac{1}{m}\sum_j\text{Tr}(\mathbf{R}_h\mathbf{y}_{B_j}\mathbf{u}_j^T)\right)$$

Since the trace operator is linear (i.e., $\text{Tr}(A+B)=\text{Tr}(A)+\text{Tr}(B)$ and $\text{Tr}(cA)=c\text{Tr}(A)$), we can factor out $\text{Tr}(\mathbf{R}_h\cdot)$ from the sums and get the final equality :

$$\frac{2}{m}\left[\frac{1}{k}\sum_{i=1}^{mk}(\mathbf{R}_h^\top g(\mathbf{z}_i))^T\mathbf{l}_i-\sum_j\left(\frac{1}{k}\sum_i\mathbf{R}_h^T g(\mathbf{x}_i)\right)^T\mathbf{y}_{B_j}\right]=2\left(\frac{1}{mk}\sum_i(\mathbf{R}_h^T g(\mathbf{z}_i))^T\mathbf{l}_i-\frac{1}{m}\sum_j(\mathbf{R}_h^T\mathbf{u}_j)^T\mathbf{y}_{B_j}\right)$$

$$=2\text{Tr}\left(\mathbf{R}_h\left(\frac{1}{mk}\sum_i\mathbf{l}_i g(\mathbf{z}_i)^T-\frac{1}{m}\sum_j\mathbf{y}_{B_j}\mathbf{u}_j^T\right)\right) \tag{16}$$

Now, based on the above derivations, we can claim the following lemma that provides an upper bound on the difference between the bag-level loss and the instance-level loss and by defining

$$\xi(\mathcal{S},\mathcal{T}) = \left\| \left(\frac{1}{mk} \sum_i \mathbf{l}_i g(\mathbf{z}_i)^T - \frac{1}{m} \sum_j \mathbf{y}_{B_j} \mathbf{u}_j^T \right) \right\|_F$$

Lemma 2. *For any $h \in \mathcal{F}$,*

$$\bar{\xi}(\mathcal{B},h) - \hat{\xi}(\mathcal{S},h) \leq 2\xi(\mathcal{S},\mathcal{T})\|\mathbf{R}_h\|_F + \lambda'(\mathcal{S},\mathcal{T}) + R(h,\mathcal{S},\mathcal{T})$$

where $\lambda'(\mathcal{S},\mathcal{T}) = \frac{1}{mk}\sum_i(\|\mathbf{y}_i\|^2 - \|\mathbf{l}_i\|^2)$ is independent of h and $R(h,\mathcal{S},\mathcal{T}) = \frac{1}{mk}\sum_i(\|h(\mathbf{x}_i)\|^2 - \|h(\mathbf{z}_i)\|^2)$ is a label-independent regularization on $\mathcal{S}$ and $\mathcal{T}$

Proof. The proof is based on Equation (13) and by injecting Equation (16) in it and applying Cauchy-Schwarz inequality to the last term. □

Note that the regularization term $R(h,\mathcal{S},\mathcal{T})$ is a label-independent regularization in the sense that it does not depend on the labels in the source and target domains. However, it enforces the model to have average output norm across the source and target domains. In practice, this regularization term is small when the model is well-trained and the outputs are close to one-hot encoded vectors for any inputs or when the models outputs uniformly distributed vectors.

Now, by expanding the definition of $\xi(\mathcal{S},\mathcal{T})$ class-wise, we can define our class-conditional discrepancy measure as follows:

Definition 1 (Class-conditional discrepancy measure)**.** *The class-conditional discrepancy measure $\xi_{class}(\mathcal{S},\mathcal{T})$ is defined as:*

$$\xi_{class}(\mathcal{S},\mathcal{T}) = \sum_{c=1}^{C} \left\| \frac{1}{mk} \sum_{i:\mathbf{l}_{i,c}=1} g(\mathbf{z}_i) - \frac{1}{m} \sum_j \pi_j(c)\mathbf{u}_j \right\|_2 \tag{17}$$

where $\pi_j(c)$ is the proportion of samples in the bag $\boldsymbol{B}_j$ that belong to class c (ie the c-th component of vector $\mathbf{y}_{B_j}$), ℓ_i is the one-hot encoded label vector for the sample $\mathbf{x}_i$ in the source domain, and $\mathbf{u}_j = \frac{1}{k}\sum_i g(\mathbf{x}_i)$ is the mean of the features in the bag $\boldsymbol{B}_j$.

A.2 Complexity analysis of `bagMME`

The computational complexity of `bagMME` is dominated by the forward and backward passes through the neural networks during training.

Indeed, at each training iteration, we need to compute the forward pass through the feature extractor and the label classifier for all samples in a batch of size B. This has a complexity of $O(B \cdot F)$ where F is the complexity of a single forward pass through the network. Then, we need to compute the bag-level predictions for the target domain samples in a bag. This involves averaging the outputs of the feature extractor for samples in the bag and then passing the averaged features through the label classifier. The complexity of this step is $O(B \cdot F)$ as well, since we need to process all samples in the bag and assuming that bag size is also B. For computing the domain discrepancy loss, we compute the mean of the target bag and the source class-conditional means, then the discrepancy between them which is a sum of C ℓ_2 norms which are simple vector operations on features that have been already computed. Overall, the complexity of a single training iteration is $O(B \cdot F)$.

A.3 Datasets and protocols

The VisDA 2017 problem (Peng et al., 2017). is a 12-class classification problem with source and target domain being simulated and real images. We have considered two sets of problem, a 3-class one (based on the classes *aeroplane, horse* and *truck*) and the full 12-class problem.

The Office-31 (Saenko et al., 2010) is an object categorization problem involving 31 classes with a total of 4652 samples. There exists 3 domains in the problem based on the source of the images : Amazon (A), DSLR (D) and WebCam (W). We have considered all possible pairwise source-target domains.

The Office-Home is another object categorization problem involving 65 classes with a total of 15500 samples. There exists 4 domains in the problem based on the source of the images : Art, Product, Clipart (Clip), Realworld (Real).

For the Visda and Office datasets, we have considered Imagenet pre-trained ResNet-50 features and our feature extractor (which is a fully-connected feedforword networks) aims at adapting those features. We have used pre-trained features freely available at https://github.com/jindongwang/transferlearning/blob/master/data/dataset.md.

For the digits problem, We considered the MNIST, USPS and SVHN datasets (LeCun et al., 1998; Hull, 1994). We have used the default splits of the datasets (train for source and test for target) and have resized images to 32×32 pixels for MNIST and SVHN problems and to 28×28 for MNIST-USPS.

For the object recognition problem, we have considered the following datasets: CIFAR-10 (Krizhevsky et al., 2009) and STL-10 (Coates et al., 2011). In order to have the same classes in both datasets, we have removed the *frog* class from CIFAR-10 and have the *monkey* class from STL-10. Hence, we have 9 classes in total. Default train/test splits are used for both datasets and images are resized to 32×32 pixels.

The hyperparameters and protocols used in the experiments are the following. Model architectures are described in the next section and for all problems, we have used the Adam optimizer with a learning rate of 0.001 and a batch size of 128. Bag size is set by default to 50 for all problems. λ_B and λ_D are chosen by grid search using the validation bags. We have the following range for $\lambda_B : \{0.5, 1, 2\}$. The range of hyperparameters for the domain classifier is set to $\lambda_D \in \{0.1, 0.2, 0.5, 1\}$ for `bagMME`, and $\lambda_D \in \{10^{-4}, 10^{-3}, 10^{-2}, 10^{-1}\}$ for `daWD`, `bagCasual`, `bagWD`. We have checked that these are the best ranges for λ_D. The number of epochs is set to 50 for the Office problems and 30 for the VisDA, digits and object recognition problems. All results are averaged over 5 runs with different random seeds which impact the bag construction and the model initialization.

A.4 Bag construction

For simulating a situation where bags from the target domain comes with a large diversity of label proportion, we have considered the following stratified sampling strategy. For each bag, given a chosen pre-defined number of class in the bag, we sample the class present in the bag. Given the number of times a class is present in all the bag, we randomly split the samples of each class to be assigned to bags and then assign samples to bags. Note that for this sampling strategy, since bag size can vary, the training data is split based on a number of bag instead of bag size. However, we have set the number of bags for each dataset to be the ratio of training samples and bag size so that the mean bag size is approximately a pre-defined bag size.

The number of class in each bag has been set to 5 for all the digits problems and 4 for the object recognition problem. For

the Office and VisDA problems, we have set the number of class in each bag to 10 and the number of class respectively.

A.5 Neural Network Architectures

For Office31, OfficeHome and Visda, we have used the following neural network architectures for the feature extractor and the classifier. The feature extractor is a fully-connected feedforward network with two hidden layers and LeakyReLU activation function. The classifier is a simple linear layer that outputs the class logits. The dropout layer is used to prevent overfitting. The classifier takes the output of the feature extractor and outputs the class logits. Input dimension of the feature extractor is 2048 and the hidden layer size is set to 128.

```
    class FeatureExtractor(nn.Module):
        def __init__(self, input_dim=100,n_hidden=256,output_dim=256):
            super(FeatureExtractor, self).__init__()
            self.fc1 = nn.Linear(input_dim, n_hidden)
            self.fc2 = nn.Linear(n_hidden, output_dim)
            self.activation = nn.LeakyReLU(0.2)
            self.dropout = nn.Dropout(p=0.1)

        def forward(self, input):
            x = self.activation(self.fc1(input))
            x = self.activation(self.fc2(x))
            x = self.dropout(x)
            return x
    class DataClassifier(nn.Module):
        def __init__(self, input_dim=256,n_class=10):
            super(DataClassifier, self).__init__()
            self.fc1 = nn.Linear(input_dim,n_class)

        def forward(self, input):
            x = (self.fc1(input.view(input.size(0), -1)))
            return x

    class DomainClassifier(nn.Module):
        def __init__(self, input_dim=256,n_hidden=100):
            super(DomainClassifier, self).__init__()
            self.fc1 = nn.Linear(input_dim, n_hidden, bias=True)
            self.fc2 = nn.Linear(n_hidden, n_hidden, bias=True)
            self.fc3 = nn.Linear(n_hidden, 1, bias=True)
            self.activation = nn.LeakyReLU(0.2)

        def forward(self, input):
            x = self.activation(self.fc1(input))
            x = self.activation(self.fc2(x))
            x = torch.sigmoid(self.fc3(x))
            return x
```

For the digits problem, we have used a convolutional neural network as the feature extractor and a fully-connected feedforward network as the classifier. The feature extractor is composed of three convolutional layers with batch normalization and max pooling, followed by a fully-connected layer that outputs the features. The classifier is a simple batch-norm feedforward network with two hidden layers and LeakyReLU activation function. These models are those used in the paper of Kirchmeyer et al. (2022) and we used the default parameters.

```

class FeatureExtractorDigits(nn.Module):
    def __init__(self, channel, kernel_size=5, output_dim=128):
        super(FeatureExtractorDigits, self).__init__()
        self.conv1 = nn.Conv2d(channel, 64, kernel_size=kernel_size)
        self.pool1 = nn.MaxPool2d(2)
        self.conv2 = nn.Conv2d(64, 64, kernel_size=kernel_size)
        self.pool2 = nn.MaxPool2d(2)
        self.conv3 = nn.Conv2d(64, output_dim, kernel_size=kernel_size)
        self.act = nn.LeakyReLU(0.2)

    def forward(self, input):
        x = self.conv1(input)
        x = self.act(self.pool1(x))
```

```
        x = self.conv2(x)
        x = self.act(self.pool2(x))
        x = self.conv3(x)
        x = x.view(x.size(0), -1)
        return x

class DataClassifierDigits(nn.Module):
    def __init__(self, n_class, input_size=128):
        super(DataClassifierDigits, self).__init__()
        self.fc1 = nn.Linear(input_size, 100)
        self.fc2 = nn.Linear(100, 100)
        self.fc3 = nn.Linear(100, n_class)
        self.act = nn.LeakyReLU(0.2)

    def forward(self, input):
        x = self.act((self.fc1(input)))
        x = self.act((self.fc2(x)))
        x = self.fc3(x)
        return x
```

A.6 Details on experiments

- For all experiments, we have used the balanced classification accuracy as the main metric.
- For the comparison experiments, we have reported the mean and standard deviation of the accuracy over 5 runs with different random seeds.
- Computations have been performed on a single NVIDIA V100 GPU with 32GB of RAM.

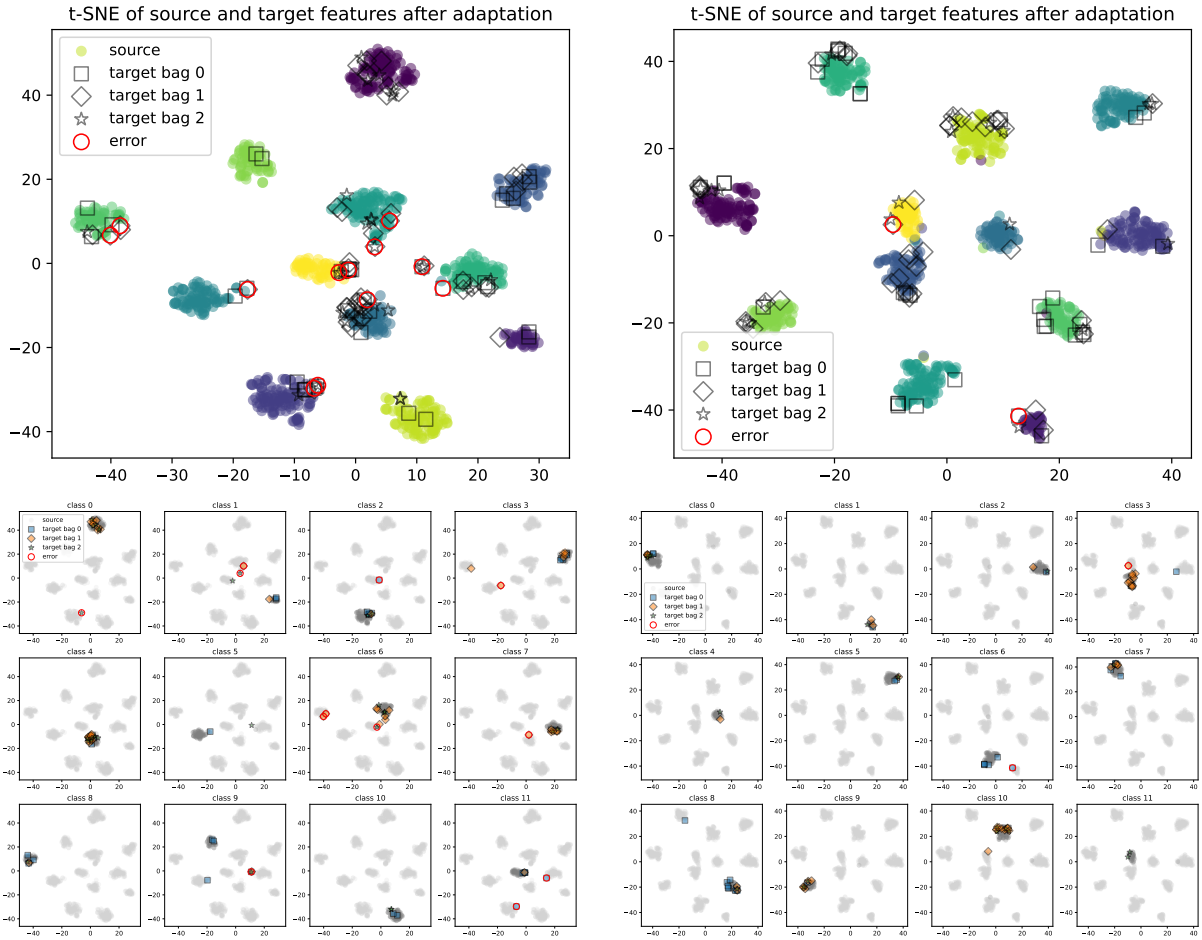

Figure 4: t-SNE visualization of the learned features on VisDA 2017 problem. Top row: Source domain samples colored by class and the three-top accuracy target bags shown using different symbols. Missclassified target samples are shown in a red circle. Bottom row: samples depicted by their true class. Grey samples are the source samples with dark grey samples being the depicted class. Colored samples are the target samples and red circled ones are the missclassified ones. Left column: $\lambda_B = 0.005$ and $\lambda_D = 0.005$. Right column: $\lambda_B = 1$ and $\lambda_D = 1$. We clearly see that with small λ_B and λ_D, the target samples are not well aligned with the source ones and the classification performance is poor. With larger λ_B and λ_D, the target samples are well aligned with the source ones and the classification performance is significantly improved.

Table 3: Comparison of different versions of `bagMME`. `CE` and ℓ_2 refer to cross-entropy and squared loss respectively used for the source domain loss. `Unmix` refers to the use of the exact unmixing strategy described in Equation (4) for estimating the target class-conditional mean embeddings. Results are averaged over 5 runs with different random seeds. Best results are in bold. We report the mean and standard deviation of the accuracy (%) over the target test set. We note that for the digits and object recognition problems, `CE` and ℓ_2 perform similarly on average. However, for the Office problems, ℓ_2 outperforms the cross-entropy loss. These latter problems are more challenging as the number of classes is larger and the domain shift is more significant and we believe that the ℓ_2 loss is more robust in such situations and as corroborated our theoretical analysis helps in controlling the target error. Exact unmixing does not lead to significant and robust improvement in performance compared to the approximated unmixing strategy. We assume that this is due to lack of stability induced by the matrix inversion in Equation (4) and the need for a proper regularization (for this experiment, a ridge penalty was used with a $\lambda = 1e-3$).

Data	Problem	`CE`	`CE + Unmix`	ℓ_2	ℓ_2 `+ Unmix`
Office31	A $\rightarrow$ D	87.79 ± 1.4	$\mathbf{88.20 \pm 2.4}$	87.49 ± 2.0	84.24 ± 4.8
	A $\rightarrow$ W	89.57 ± 2.6	90.48 ± 1.9	$\mathbf{90.85 \pm 2.1}$	90.49 ± 2.4
	D $\rightarrow$ A	71.37 ± 0.8	70.28 ± 1.4	$\mathbf{74.32 \pm 1.8}$	73.73 ± 1.1
	D $\rightarrow$ W	97.44 ± 0.5	96.48 ± 0.6	$\mathbf{97.46 \pm 0.8}$	95.48 ± 0.7
	W $\rightarrow$ A	76.23 ± 1.6	75.90 ± 1.8	$\mathbf{80.72 \pm 1.6}$	77.79 ± 1.4
	W $\rightarrow$ D	$\mathbf{99.39 \pm 0.4}$	98.71 ± 0.6	98.32 ± 0.6	93.57 ± 3.6
OfficeHome	A $\rightarrow$ C	54.16 ± 2.2	52.08 ± 4.9	$\mathbf{66.42 \pm 1.3}$	40.48 ± 3.4
	A $\rightarrow$ P	72.40 ± 0.5	71.17 ± 1.5	$\mathbf{84.45 \pm 1.7}$	65.42 ± 4.7
	A $\rightarrow$ RW	73.85 ± 1.0	72.73 ± 1.3	$\mathbf{79.12 \pm 0.8}$	62.94 ± 3.6
	C $\rightarrow$ A	47.98 ± 3.0	45.69 ± 2.7	$\mathbf{52.31 \pm 1.4}$	49.20 ± 2.6
	C $\rightarrow$ P	68.95 ± 1.2	68.74 ± 1.8	$\mathbf{82.73 \pm 0.4}$	74.54 ± 3.0
	C $\rightarrow$ RW	66.96 ± 1.4	65.91 ± 2.1	$\mathbf{74.63 \pm 1.6}$	65.69 ± 4.1
	P $\rightarrow$ A	53.22 ± 1.4	51.45 ± 2.5	$\mathbf{57.27 \pm 1.3}$	53.18 ± 1.6
	P $\rightarrow$ C	52.03 ± 1.6	54.38 ± 1.5	$\mathbf{64.92 \pm 1.8}$	53.98 ± 1.8
	P $\rightarrow$ RW	73.41 ± 0.8	73.42 ± 1.8	$\mathbf{77.30 \pm 1.5}$	71.46 ± 5.9
	RW $\rightarrow$ A	61.31 ± 1.0	61.90 ± 1.9	$\mathbf{63.39 \pm 1.1}$	59.82 ± 0.7
	RW $\rightarrow$ C	55.34 ± 1.0	56.86 ± 1.8	$\mathbf{66.26 \pm 1.2}$	58.24 ± 1.0
	RW $\rightarrow$ P	79.81 ± 1.1	80.79 ± 1.2	$\mathbf{87.43 \pm 1.3}$	65.69 ± 32.2
VisDA	0	79.13 ± 0.6	78.85 ± 0.6	$\mathbf{79.76 \pm 0.7}$	79.58 ± 0.5
	1	$\mathbf{96.11 \pm 0.2}$	95.95 ± 0.2	96.01 ± 0.5	95.96 ± 0.2
MNIST USPS	-	$\mathbf{98.20 \pm 0.3}$	97.13 ± 0.3	97.98 ± 0.2	97.74 ± 0.2
USPS MNIST	-	$\mathbf{97.78 \pm 0.4}$	84.21 ± 3.5	97.52 ± 0.3	24.63 ± 29.3
MNIST SVHN	-	$\mathbf{83.32 \pm 0.8}$	81.73 ± 1.4	81.14 ± 1.2	79.93 ± 0.9
SVHN MNIST	-	95.74 ± 0.7	19.47 ± 12.0	$\mathbf{97.82 \pm 0.4}$	10.00 ± 0.0
STL CIFAR	-	45.79 ± 1.4	46.39 ± 0.6	$\mathbf{48.18 \pm 1.0}$	47.65 ± 0.9
CIFAR STL	-	42.05 ± 1.0	40.87 ± 1.1	$\mathbf{44.60 \pm 1.5}$	41.27 ± 0.9