Deep Multi-Source Supervised Domain Adaptation with Class Imbalance

Thomas Ranvier, Haytham Elghazel, Emmanuel Coquery, Khalid Benabdeslem

Deep Multi-Source Supervised Domain Adaptation with Class Imbalance

Thomas Ranvier[1*], Haytham Elghazel[1], Emmanuel Coquery[1] and Khalid Benabdeslem[1]

[1]Université Lyon 1, LIRIS, UMR 5205, F-69622, France.

*Corresponding author(s). E-mail(s): thomas.ranvier@univ-lyon1.fr;
Contributing authors: firstname.lastname@univ-lyon1.fr;

Abstract

Deep multi-source domain adaptation is based on deep neural networks and exploits knowledge from multiple source domains to improve predictions on a target domain. In this paper, we are especially interested in investigating domain adaptation in a supervised context, with limited data and class imbalance. We propose a new multi-source supervised domain adaptation approach that is able to transfer both shared knowledge across all source domains and source domain specific knowledge toward a target domain. Transfer contribution weights are computed during training based on domain divergence. They are used to balance each source domain impact on learning during the model training phase, limiting negative transfer as much as possible. We conduct extensive experiments to show that our approach competes and even outperforms other state-of-the-art domain adaptation approaches on both image benchmark datasets and real-world tabular medical data. We perform statistical analysis to better evaluate our experimental results, and conduct an ablation study to evaluate the usefulness of each component of the method.

Keywords: Deep Learning, Domain Adaptation, Multi-Source, Supervised, Imbalanced

1 Introduction

Learning from imbalanced data requires a specific learning approach in order to pay specific attention to the class distribution during the training phase. Deep Learning is notoriously hard when dealing with limited data. Indeed, if the data is too limited, it is impossible to properly train a deep model, and simpler Machine Learning approaches should be preferred. When independent limited but similar datasets are available, it becomes adequate and beneficial to use deep multi-source domain adaptation to improve predictions on the target domain [1–4]. Exploiting knowledge from several source domains can help to minimize the negative impact of both limited data and class imbalance. In this paper, we propose a new interesting multi-source supervised domain adaptation approach, which we ultimately aim to apply on real-world limited and imbalanced medical tabular data.

In transfer learning, we aim to exploit knowledge from one or several source dataset(s) to improve learning performance on another target dataset. For transfer learning to be beneficial, the dataset(s) used as source(s) should be similar enough to the target dataset. A source that is not similar enough to the target will negatively impact learning performance, and should not be used in this context. We talk about domain adaptation when we aim to learn a single and common task by transferring knowledge from one or several source domain(s) to a target domain. A very well-researched area of domain adaptation is single-source domain adaptation [5**?** –7], that is, when we use only one source domain to transfer towards the target domain. A more complex and less researched area is multi-source domain adaptation [8–11], where we use several source domains to transfer as much knowledge as possible to the target. Domain adaptation can help largely improve prediction performance on the target domain by exploiting more knowledge from source domain(s) than available on the sole target domain. Domain adaptation is often used to make prediction possible on an entirely unlabeled target domain, that is, unsupervised domain adaptation. In our work, we are interested in a case where the target domain is labeled as any standard dataset, in this case we talk about supervised domain adaptation, which is a less researched domain adaptation area, despite being a common real-world occurrence.

In this paper, we propose a new original approach for multi-source domain adaptation in a supervised context, and demonstrate its performance on limited and imbalanced data. Namely, Weighted Multi-Source Supervised Domain Adaptation (WMSSDA). WMSSDA transfers knowledge from s source domains to a similar target domain. It learns a domain invariant latent space, regularized using both statistical and adversarial approaches, where shared knowledge across source domains is transferred to the target domain, and s source domain specific latent spaces, in which source specific knowledge is transferred. With such an architecture, our proposed approach WMSSDA is able to exploit both common knowledge across all domains and source specific knowledge that is useful for inference on the target domain. We compute source domain specific transfer contribution weights during training, those are applied

during training to weight the importance of each source domain on learning, reducing as much as possible Negative Learning. We conduct extensive experiments to compare our approach with other baseline and state-of-the-art domain adaptation approaches in a data-limited and class imbalance context. We show that WMSSDA outperforms other state-of-the-art domain adaptation approaches on both image benchmark datasets and real-world medical tabular data. We perform an ablation study to validate the pertinence and positive impact of each component in our method.

The source code used to conduct our complete experiments is available at the following GitHub repository[1].

In the rest of the paper, we first formally describe our domain adaptation learning scenario in section 2, we then present related works from transfer learning and learning on imbalanced data literature in section 3. Section 4 describes our proposed approach, we show our experimental results in section 5 and conclude with a summary of our contributions, results and future perspectives.

2 Learning Scenario

In this section, we formally describe the considered learning scenario.

2.1 Notations and Preliminaries

Notation	Description	Notation	Description
$\mathcal{X}$	Feature space	$\mathbb{T}$	Target domain
$\mathcal{Y}$	Label space	$P(\cdot)$	Distribution
X	Data sample	$f(\cdot)$	Labeling function
Y	Labels sample	s	Number of source domains
$\mathbb{D}$	Domain	c	Number of classes
$\mathbb{S}$	Source domain	n	Number of instances

Table 1: Main notations used in this paper.

Table 1 summarizes the used notations in this section and the rest of the paper. We introduce definitions and concepts needed for the following sections, our notations are inspired by the work of [12], we took liberties to adapt them to our multi-source supervised domain adaptation context.

Let $\mathcal{X} \in \mathbb{R}^d$ denote an input feature space, with d the number of features, and $\mathcal{Y} = \{1, \ldots, c\}$ a multi-class output label space, with c the total number of classes. We define a domain as a pair formed by a distribution over $\mathcal{X}$ and a labeling function mapping from $\mathcal{X}$ to $\mathcal{Y}$. We note $\mathbb{D} = (P(X_{\mathbb{D}}), f_{\mathbb{D}})$ the domain $\mathbb{D}$, with $P(X_{\mathbb{D}})$ the marginal distribution of $\mathbb{D}$ over $\mathcal{X}$, $f_{\mathbb{D}} : \mathcal{X} \rightarrow \mathcal{Y}$ is the labeling function mapping from feature to label space, $X_{\mathbb{D}}$ is the data sample

[1]Temporary private link, to be updated after acceptance: https://drive.google.com/file/d/1XvgWqkHA4LuK6bPE9ktIlO9SFY1MemnC/view?usp=sharing

defined as $X_{\mathbb{D}} = \{x_i \in \mathcal{X}\}_{i=1}^{n_{\mathbb{D}}}$, with $n_{\mathbb{D}}$ the number of instances in the data sample of the domain $\mathbb{D}$.

In the scenario of multi-source supervised domain adaptation we consider, we are given s source domains, noted $\mathbb{S}_i$ for $i \in [1, s]$, that we want to exploit to improve classification over one target domain, noted $\mathbb{T}$. A unique label space $\mathcal{Y}$ is shared across all domains, feature space of each domain can be different from other domains, we note $\mathcal{X}_{\mathbb{D}}$ the feature space of domain $\mathbb{D}$. In our scenario, we have access to s labeled source domains where $\mathbb{S}_i = \{(x_j^{\mathbb{S}_i}, y_j^{\mathbb{S}_i})\}_{j=1}^{n_i}$, with $\{x_1^{\mathbb{S}_i}, \ldots, x_{n_i}^{\mathbb{S}_i}\} \sim P(X_{\mathbb{S}_i})$, and $y_j^{\mathbb{S}_i} = f_{\mathbb{S}_i}(x_j^{\mathbb{S}_i})$. We have access to a labeled target domain, similarly, $\mathbb{T} = \{(x_j^{\mathbb{T}}, y_j^{\mathbb{T}})\}_{j=1}^{n_{\mathbb{T}}}$, where $\{x_1^{\mathbb{T}}, \ldots, x_{n_{\mathbb{T}}}^{\mathbb{T}}\} \sim P(X_{\mathbb{T}})$ and $y_j^{\mathbb{T}} = f_{\mathbb{T}}(x_j^{\mathbb{T}})$.

We want to exploit knowledge from labeled source domains and the labeled target domain, to improve classification on an unknown and unusable part of $\mathbb{T}$. As we consider a scenario in which the three types of shifts are present between domains, we consider that the covariate shift assumption does not hold, $f_{\mathbb{S}_1} \neq \ldots \neq f_{\mathbb{S}_s} \neq f_{\mathbb{T}}$. Solving such a problem is only possible if the target domain is labeled, as it is necessary to rely on supervision to properly align domains with concept shifts. Therefore, we want our supervised domain adaptation model to learn to estimate the labeling function $f_{\mathbb{T}}$, while exploiting knowledge from the source domains through the learning of the different source labeling functions $\{f_{\mathbb{S}_1}, \ldots, f_{\mathbb{S}_s}\}$.

2.2 Domain Shift in Our Adaptation Scenario

In their 2012 paper, to unify the terms and definitions used for the various domain shifts that appear in domain adaptation literature and provide consistent terminology, Moreno-Torres et al. [13] proposed formalization of three types of shifts: covariate shift, prior shift, and concept shift. More recently, Kouw and Loog [4] reviewed those defined shifts and provided more precise and up-to-date definitions. To present our adaptation scenario in terms of domain shift we use those same terms and define them mathematically using our notations in the following. Figure 1 illustrates each of the presented domain shifts.

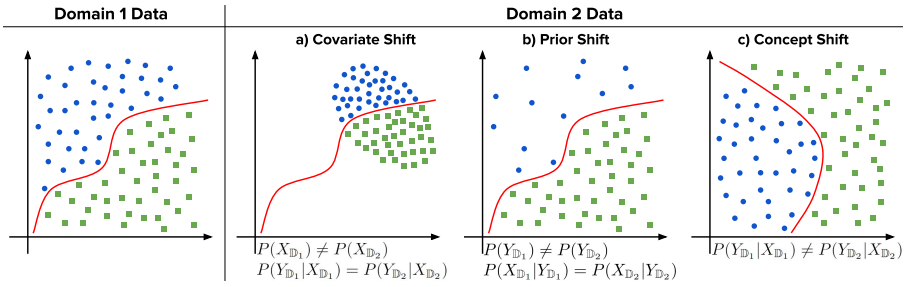

Fig. 1: Illustration of the three kinds of domain shifts.

Formally, covariate shift is when the data marginal distributions of two domains are different, while their conditional distributions are equal, $P(X_{\mathbb{D}_1}) \neq P(X_{\mathbb{D}_2})$ and $P(Y_{\mathbb{D}_1}|X_{\mathbb{D}_1}) = P(Y_{\mathbb{D}_2}|X_{\mathbb{D}_2})$. Intuitively, covariate shift exists between domains when the feature space of a domain is different from the one of another domain, that is $\mathcal{X}_{\mathbb{D}_1} \neq \mathcal{X}_{\mathbb{D}_2}$, or when there exists a domain specific form of sample selection bias [4]. Under covariate shift, a classifier trained on one domain might struggle when applied on another domain. This is a domain shift that is present in almost all domain adaptation applications and is a point of interest of this paper.

There is a prior shift between two domains when the label marginal distributions of both domains are different, while their conditional distributions are equal, $P(Y_{\mathbb{D}_1}) \neq P(Y_{\mathbb{D}_2})$ and $P(X_{\mathbb{D}_1}|Y_{\mathbb{D}_1}) = P(X_{\mathbb{D}_2}|Y_{\mathbb{D}_2})$. This happens when the class balance is not the same in each domain. This is a common occurrence that can happen when data from similar domains are gathered differently [14], leading to different label marginal distributions between the domains. This type of domain shift appears less often in domain adaptation literature, we are specifically interested in prior shift, as it occurs in cases where domains are all differently imbalanced.

Formally, concept shift occurs when the conditional distributions of two domains are different, $P(Y_{\mathbb{D}_1}|X_{\mathbb{D}_1}) \neq P(Y_{\mathbb{D}_2}|X_{\mathbb{D}_2})$. This means that the decision boundary between classes is not the same from one domain to another, meaning that the causal relation between features and labels is semantically different from one domain to another. Concept shift might occur between two domains if classes are semantically inaccurate, which might lead to slightly differently labeled data between them. In an unsupervised context, concept shift would render transfer from source domains to the target domain impossible. As we are in a supervised domain adaptation setting, transferring knowledge is possible in our scenario.

In this paper, we are specifically interested in the case of multi-source domain adaptation with imbalanced data. We want to use multiple source domains with different feature spaces to improve classification performance over a similar target domain, with the particularity that each used domain is imbalanced in a different way. That is, each domain's data and label marginal distributions are different from other domains. This is a standard case of covariate shift between domains, with the addition of a prior shift, due to domain specific class imbalance. In our learning scenario we assume that the covariate shift assumption does not hold, as we cannot entirely rule out the hypothesis of a concept shift across domains.

3 Related Works

3.1 Domain Adaptation

Domain adaptation (DA) can be considered as a special case of transfer learning [1]. Transfer learning includes all approaches that are able to use knowledge from a source to improve inference on a target. Domain adaptation includes

all approaches that aim to learn a single and common task by transferring knowledge from one or several source domain(s) to a target domain [3**?**], most domain adaptation scenarios do not include target domain labels, that is, unsupervised DA. Less researched DA scenarios are semi-supervised and supervised DA where the target domain includes partially or entirely labeled target data, the goal of transferring knowledge from source(s) to target in those cases remains identical. As can be understood by the names, single-source DA is transferring knowledge from one source domain to a target domain, while multi-source DA transfers knowledge from multiple source domains to the target. Since about 2015, deep neural networks have been explored for domain adaptation, obtained results are significantly better compared to prior shallow transfer learning approaches, consequently, most recent DA approaches are based on deep models [1, 15**?**].

Most domain adaptation approaches in the literature focus on learning a shared domain invariant latent space between domains to capture shared information between source(s) and target [1]. This leads to a latent space where instances of a domain are indistinguishable from instances of other domains, while classification relevant information is conserved, leading to better inference results on the target domain. There are two main ways of reaching this goal, relying on statistic distribution matching, or relying on an adversarial loss that encourages samples from different domains to lose all domain specific information.

Maximum Mean Discrepancy (MMD) is the most commonly used statistic to measure domain discrepancy to match source(s) and target learned distributions in DA literature. MMD has first been used and democratized by DAN [5], that uses MMD to minimize the distance between the learned representation of a source domain and the representation of the target domain. As an alternative, [10] proposed M^3SDA, a multi-source approach that uses Moment matching Distance (MD) to match the distributions moments between domains. They demonstrate that MD is more pertinent than MMD in a multi-source adaptation context.

Other approaches from the DA literature use an adversarial approach to learn a domain invariant latent space. DANN [6] is the first DA approach that made use of a domain classifier trained adversarially on the learned latent representation of both target and source data. The domain classifier tries to discriminate the domain in the latent features, while the feature extractor learns to fool the domain classifier, successfully leading to a common invariant latent space between domains. MDAN [8] can be considered as a multi-source version of DANN, with s sources domains, s domain classifiers are trained to discriminate between the i-th source domain and the target domain, leading to an invariant latent space between each source domain and the target domain. Adversarial approaches are notorious for reaching better results than statistic distribution matching in Domain Adaptation [7].

We believe that learning a shared domain invariant latent space for multi-source domain adaptation is limited, and that learning pairwise invariant latent

spaces between the target domain and each source domain allows for the capture and transfer of more useful information between sources and target. When learning a domain invariant latent space between two domains, intuitively, the only information that is captured within the shared representation is the common information. Therefore, when building a shared representation across multiple domains, only the common information across all domains is captured, which becomes a limiting factor as the number of domains and the dissimilarities between them increases. This is why we believe that learning several latent spaces in a pairwise manner between target and sources is pertinent in order to capture and transfer as much relevant information as possible. For this reason, in our proposed multi-source domain adaptation approach, we used an architecture where a shared domain invariant latent space is learned across all domains in one branch, while s source specific latent spaces are learned between the target domain and each of the s source domains in another branch.

There exist two multi-source domain adaptation methods in the literature that also rely on learning both a shared domain invariant latent space while also learning pairwise latent spaces between sources and target domain: ML-MSDA [16], and MLAN [17]. Unlike us, their methods have been proposed and applied in a unsupervised domain adaptation context, which differs from supervised domain adaptation. Mutual Learning Network for Multiple-Source Domain Adaptation (ML-MSDA) [16] is composed of two branches. The first branch learns a shared invariant latent space across all domains, while the second learns pairwise latent spaces between the target and each source domain. By jointly learning those multiple latent representations, they obtain better experimental results than all previously presented multi-source domain adaptation approaches. They rely on adversarial learning to ensure the domain invariance of the learned latent spaces. Similarly, [17] extended the work of [16] by proposing a Mutual Learning based Alignment Network (MLAN). The model architecture is identical to ML-MSDA, but is trained slightly differently, through the proposed mutual learning module. The module relies on pseudo-labeling of target instances to maximize target prediction performance. With MLAN, [17] currently obtains state-of-the-art results compared to other multi-source and single-source models of the domain adaptation literature.

In our work, we propose a multi-source supervised domain adaptation method with a two-branch architecture, similarly to ML-MSDA and MLAN. We exploit the fact that we are working in a supervised domain adaptation context to train the model on labeled instances from all sources and target domains.

3.2 Negative Transfer

The goal of domain adaptation is to exploit knowledge from one or several source domain(s) to improve prediction quality on a target domain. But a common issue with domain adaptation is negative transfer [2, 3, 18]. Negative transfer occurs in domain adaptation when transferring knowledge from a source domain to a target domain harms the learning performance on the

target. Consequently, instead of improving the inference model performance, negative transfer leads to a decrease in prediction performance on the target domain. One of the most common reasons for negative transfer is a too large dissimilarity between source and target domains [18]. This risk is multiplied in the multi-source domain adaptation field, as multiple sources can contribute to negative transfer. Negative transfer is an important issue in the transfer learning field, limiting negative transfer is an important matter, which should be addressed when designing new domain adaptation approaches.

In their paper, [11] proposed the ABMSDA method, which avoids negative transfer by weighting each source domain depending on its contribution to the adaptation process. Their model architecture is composed of a domain classifier, a common feature extractor regularized using WMD (a modified version of Moment Distance), and a shared task-specific classifier. They train the domain classifier, separately and prior from the rest of the model, to predict the probability that target images belong to each source domain. They use the probability output of the domain classifier as a metric that indicates the statistical similarity between the target domain and each source domains, with the intuition that source domains that are most similar to the target domain should be attributed a higher weight. They apply those weights to source instances when computing WMD. They also apply those weights when combining the probability outputs of the classifier during training, leading to a classifier less prone to negative transfer. This is a way of avoiding negative transfer during multi-source domain adaptation.

In our proposed approach, we compute transfer contribution weights with a discrepancy measure, directly during training. Those weights are associated to each source domain, based on supervised target domain results, and applied to scale the importance of source instances during training. The goal being to increase the importance of instances from relevant source domains, while decreasing the importance of instances from less related source domains.

3.3 Dealing With Class Imbalance

Class imbalance occurs when the labels we aim to predict are not uniformly distributed over the dataset, resulting in certain classes having a much higher, or lower, number of instances compared to others. When trained on an imbalanced dataset, standard Machine Learning approaches will lead to poorer results than when trained with a similar balanced dataset [19]. When dealing with class imbalance it is primordial to use approaches that help improve learning performance. Nowadays, two main ways are used to deal with imbalanced data in the literature, sampling approaches, and cost-sensitive approaches [20].

Sampling approaches aim to artificially adjust the class distribution, by either removing instances the majority class(es), and/or adding more instances from the minority class(es). Under-sampling aims to reduce the number of majority class instances to achieve a more balanced class distribution. The simplest under-sampling approach that can be used to artificially re-balance a dataset is random under-sampling [20–22]. With random under-sampling

instances from the majority class(es) are randomly removed until the desired class distribution is achieved. This simple approach can lead to improved inference results but can also result in the loss of important information from the majority class(es), and so, reduce the generalization ability of the model as it misses crucial information. To assess this drawback more researched and advanced under-sampling approaches have been proposed, such as: Condensed Nearest Neighbor [23], or Tomek Links [24]. Overall, under-sampling leads to improved inference results but might also lead to losing important information for inference if the amount of instances is too low to afford removing instance from the majority class(es). On the other hand, over-sampling involves increasing the representation of minority classes by duplicating or generating synthetic examples. The simplest over-sampling approach that can be used to artificially re-balance a dataset is random over-sampling [20–22]. Random over-sampling duplicates randomly selected instances from the minority class(es) until the desired class distribution is achieved. As with random under-sampling, over-sampling leads to improved inference results but highly increases the risk of overfitting, leading to biased inference models that lack in generalization capacity. The Synthetic Minority Oversampling Technique (SMOTE) method is the most popular and most widely used advanced over-sampling method, it has been proposed in [25] and it is known to largely improve inference results on imbalanced data [20, 21]. SMOTE works by creating synthetic examples of the minority class(es) by interpolating between existing instances of the minority class(es). SMOTE is known to largely improve prediction results on imbalanced data. A known drawback of SMOTE, and over-sampling in general, is the risk of leading to overfitting, and the risk of generating synthetic examples that are unrealistic or less informative, leading to a limited improvement in prediction quality.

Removing and generating synthetic data leads to improved learning results, but with important drawbacks. Removing data is often non viable in real-world scenarios with limited data, and generating synthetic data comes with the disadvantage of potentially generating implausible instances. Another approach that can be used in Machine Learning to deal with class imbalance is cost-sensitive learning, where modifications are made to the algorithm, and/or to the training process, to take account of imbalance and improve prediction results. It has been shown in several empirical studies that cost-sensitive learning leads to superior inference results than sampling approaches on imbalanced data [19, 26]. Therefore, cost-sensitive techniques are usually a better solution than sampling methods. In neural network training, the most popular way of implementing cost-sensitive learning is to adapt the error function to take into account the class cost of each training instance during the learning phase, as defined in [26]. The error function is corrected by introducing the cost factor of the class as a weight that is applied during training. Class weights applied to the loss function are commonly computed as the inverse of the class distribution of training data, though other weighting techniques can be used. This approach obtained by far the best results in [26], it is still a very commonly

used approach as it is very easy to implement and use, and it reaches better results than most other existing approaches to handle imbalanced data. This is why, in this paper, we use a cost-sensitive approach to account for the class imbalance between domains to maximize prediction quality.

4 Proposed Approach

In this section, we describe in details our proposed approach: Weighted Multi-Source Supervised Domain Adaptation (WMSSDA).

The idea behind WMSSDA is to create a common domain invariant latent space and s source domain specific latent spaces, perform classification on each latent space, and draw final weighted prediction results in an ensemble manner. The common domain invariant latent space is trained on a label classification task on both source and target batches. Two regularization techniques are employed to minimize domain-specific information into the common latent space. First, by using a Moment Distance (MD) regularization to match the distributions of source and target batches, and secondly, through the adversarial training of a domain discriminator. For the s source domain specific latent spaces the ideology is the opposite, we want each latent space to retain as much source specific information as possible. This is possible as in a supervised domain adaptation context, the target domain is labeled, it is therefore possible to train each classifier on a supervised classification task on both source and target instances, leading to a pairwise fine-tuning between each source domain and the target domain. Those specific latent representations are regularized using a collaboratively trained specific domain classifier, while performing label classification, leading to latent spaces retaining as much domain specific information as possible. Using multi-source domain adaptation naturally helps dealing with the class imbalance problem, as it increases the total amount of available data for training. We further deal with the class imbalance in each domain using a cost-sensitive learning approach, by scaling the loss with a class weight, computed as the inverse of the class distribution of training data. We use the output of the Moment Distance measure between the target domain and each source domain to determine transfer contribution weights, higher weights are attributed to source domains with closer latent distributions compared to the target domain latent representation, and inversely. Those transfer contribution weights are then applied to weight the classification loss on source instances, giving less importance to less relevant source domains in training. Ultimately, the final predictions for the target domain are obtained by passing a target batch through all trained modules. The average of the outputs from all specific classifiers is computed and combined with the outputs of the common classifier to obtain the final probabilities for the target classes. Figure 2 shows a simple representation of the architecture of the approach.

When training the approach, we iterate through one batch from each source domain for each target domain batch. Batches are processed as a pair between

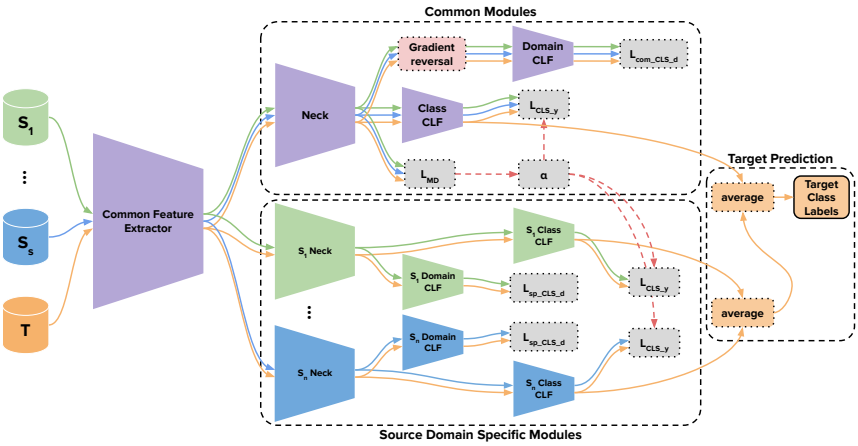

Fig. 2: Architecture of our approach WMSSDA, common modules appear in purple, i-th source domain specific modules appear in the same color as the i-th associated source domain. Lines of color symbolize the data flow, each color corresponding to a batch of the corresponding domain. Dashed red lines symbolize the computation and application of transfer contribution weights, computed from the MD measure and applied as scaling in the classification loss terms.

the b-th target batch and the b-th batch of the i-th source domain, which we will refer to as a pair of batches in the following. Each pair of batches is fed through a common feature extractor used to extract low-level features on all domains alike. The approach architecture is then divided in two main parts: common modules and source domain specific modules.

- Common modules. They are fed pairs of batches between the target domain and any source domain. The first component of this part is a common neck, comparable to the previous feature extractor, which extracts higher-level features on all domains. We call the output of the previous component the common domain invariant latent space, which we note $Z_{\mathbb{T}com}$ and $Z_{\mathbb{S}com}$, for the target and source batch latent representations respectively. To ensure that this latent space is domain invariant, we use both statistical distribution matching and adversarial domain discrimination.
  We regularize the latent representations by minimizing the standard Moment Distance, such as defined in [10, 11], between target and source representation, $\mathcal{L}_{MD} = \sum_{i=1}^{k} \left\| \mathbb{E}(Z^i_{\mathbb{S}com}) - \mathbb{E}(Z^i_{\mathbb{T}com}) \right\|_{\mathcal{F}}$.
  We use the output of the MD measure between the target domain batch and batches of all source domains to compute transfer contribution weights. We note those weights $\alpha \in \mathbb{R}^s$, with α_i the weight associated to the i-th source domain. Those transfer contribution weights are used to scale the classification loss of each source domains, giving more weight to close and related source domains and less weight to less useful domains. They are

computed as:

$$\alpha_i = \frac{s+1}{s} - \frac{e^{D_i - \max(D)}}{\sum e^{D_i - \max(D)}}$$

with $D = \{\mathrm{MD}(Z_{\mathbb{S}_i com}, Z_{\mathbb{T} com})\}_{i=1}^{s}$ the set of MD measures between the target domain batch and each source domain batch. The weights are computed such as $\sum_{i=1}^{s} \alpha_i = s$, which ensures that source instances are given as much importance as target instances overall, while a different weight is given to each source domain.

We associate the MD regularization with the training of an adversarial common domain classifier applied to the latent space. This common domain classifier learns to discriminate the domain from which originates each sample in the pair of batches, while the common feature extractor and neck try to fool the discriminator, leading to a domain invariant representation in this latent space. Parameters of the common feature extractor and neck are tuned to maximize the loss of the domain discriminator, while the discriminator parameters are tuned to minimize their own loss. As it is hard to optimize minimax problems using gradient descent algorithms, a common practice in domain adaptation research is to use a gradient reversal operation, and apply it between the latent representation and the domain classifier, such as defined in [6]. Which solves the adversarial problem by minimizing a single loss. We note $\mathcal{L}_{adv\_d}$ the loss of the common domain discriminator applied on the gradient reversed common latent representation of the pair of batches.

We found that using both statistical and adversarial strategies simultaneously led to better empirical results, which suggests that using both approaches cooperatively leads to a better domain invariant representation. Finally, the latent representation is fed through a task-specific classifier that discriminates samples on their class label.

- Source domain specific modules. They are only fed pairs of batches between the target domain and their associated source domain, that is, a pair of batches between the i-th source domain and the target domain is fed to the i-th specific module. Each specific module is composed of a specific neck, which extracts higher-level features, a specific task-specific classifier that discriminates samples on their class label, and a specific domain classifier that discriminates samples on the domain they originate from. In the opposite way to the above, this specific domain classifier is trained collaboratively, which pushes each specific latent space to retain as much domain specific information as possible. The source and target batches are treated differently:
  - The source batch from domain $\mathbb{S}_i$ is fed through the i-th specific neck, the resulting latent representation is noted $Z_{\mathbb{S}i}$. The i-th domain classifier is fed $Z_{\mathbb{S}i}$ and is trained to recognize that those samples originate from domain $\mathbb{S}_i$. The i-th task-specific classifier is fed $Z_{\mathbb{S}i}$ and is trained to discriminate the class of each sample.

– The target batch is fed through all specific necks, and the resulting latent representations are noted $\{Z_{\mathbb{T}1}, \ldots, Z_{\mathbb{T}s}\}$. The i-th domain classifier is fed $Z_{\mathbb{T}i}$ and is trained to recognize that those samples originate from domain $\mathbb{T}$. Each task-specific classifier is fed $Z_{\mathbb{T}i}$, their outputs are noted $\{\hat{Y}_{\mathbb{T}1}, \ldots, \hat{Y}_{\mathbb{T}s}\}$. Only the i-th classifier is trained to discriminate the class of each target sample to avoid overfitting.

We note $\mathcal{L}_{sp\_d}$ the loss of the i-th specific domain discriminator applied on the i-th latent representations of the pair of batches.

We note $\mathcal{L}_y$ the global task-specific classification loss, which is the average between common and specific classification probabilities on the pair of batches. To take account of class imbalance during training we compute class weights, noted W, that are applied in a cost-sensitive learning manner during task-specific classification loss computation, they are computed as the inverse of the class distribution observed in the training data of each domain: $W_{\mathbb{D}} = 1/P(Y_{\mathbb{D}})$. Finally, the loss we minimize using gradient descent is computed for each pair of batches in the following way:

$$\mathcal{L} = \mathcal{L}_y + \lambda_1 \mathcal{L}_{adv\_d} + \lambda_2 \mathcal{L}_{sp\_d} + \lambda_3 \mathcal{L}_{MD}$$

Where $\lambda_1, \lambda_2, \lambda_3$ are hyper-parameters that are defined to balance each component of the final loss.

Final prediction results are obtained by feeding target instances through all task-specific classifiers, the average of source specific classifiers outputs is computed and averaged with the output of the common classifier, leading to final class probabilities.

Pseudo-code 1 describes, in a more formal way, the training steps of the entire approach. In this pseudo-code, *model* includes a common feature extractor model, common and specific necks, and common and specific classifiers, com_clf_d is the common domain classifier and sp_clf_d is the set of i specific domain classifiers. We note $adv(\cdot)$ the gradient reversal operation, classification outputs are noted $\{\hat{Y}_{\mathbb{D}com}, \hat{Y}_{\mathbb{D}1}, \ldots, \hat{Y}_{\mathbb{D}s}\}$ and correspond to the label probabilities obtained on the common latent representation and all specific representations of the domain $\mathbb{D}$ respectively. Latent spaces are referred to with the same notations as above, $Z_{\mathbb{D}com}$ for the common latent representation and $Z_{\mathbb{D}i}$ for the i-th specific latent representation of domain $\mathbb{D}$. Parameter E is the number of epochs to perform, s is the number of source domains and λ is the set of hyper-parameters used to balance all loss components together. Finally, W are class weights specific to each domain computed as $1/P(Y_{\mathbb{D}})$, W_i for the i-th source domain and $W_{\mathbb{T}}$ for the target domain.

Our proposal combines the advantages of a domain invariant latent space with a set of domain specific representations, with transfer contribution weights applied to minimize negative transfer during training, leading to a supervised domain adaptation approach able to transfer knowledge from multiple source domains to a target domain. Similarly to DANN [6], our proposed WMSSDA learns a domain invariant latent space on which is performed classification and

Algorithm 1: WMSSDA Training Pseudo-Code.

input: E, s, $model$, com_clf_d, sp_clf_d, W, λ
for $epoch \leftarrow 1$ *to* E **do**
  $X_{\mathbb{T}}, Y_{\mathbb{T}} \leftarrow$ extract_batch($\mathbb{T}$);
  for $i \leftarrow 1$ *to* s **do**
    // Feed target batch to model and compute target loss components
    $\{\hat{Y}_{\mathbb{T}com}, \hat{Y}_{\mathbb{T}1}, \ldots, \hat{Y}_{\mathbb{T}s}\} \leftarrow$ forward($model$, $X_{\mathbb{T}}$);
    $\hat{Y}_{\mathbb{T}} \leftarrow (\hat{Y}_{\mathbb{T}com} + \hat{Y}_{\mathbb{T}i})/2$;
    $\mathcal{L}_{\mathbb{T}y} =$ cross_entropy($Y_{\mathbb{T}}$, $\hat{Y}_{\mathbb{T}}$, $W_{\mathbb{T}}$);
    // Feed source batch to model and compute source loss components
    $X_{\mathbb{S}}, Y_{\mathbb{S}} \leftarrow$extract_batch($\mathbb{S}_i$);
    $\{\hat{Y}_{\mathbb{S}com}, \hat{Y}_{\mathbb{S}1}, \ldots, \hat{Y}_{\mathbb{S}s}\} \leftarrow$ forward($model$, $X_{\mathbb{S}}$);
    $\hat{Y}_{\mathbb{S}} \leftarrow (\hat{Y}_{\mathbb{S}com} + \hat{Y}_{\mathbb{S}i})/2$;
    $\mathcal{L}_{\mathbb{S}y} =$ cross_entropy($Y_{\mathbb{S}}$, $\hat{Y}_{\mathbb{S}}$, W_i);
    // Feed adversarially trained common domain classifier
    $\{\hat{Y}_{\mathbb{S}adv_d}, \hat{Y}_{\mathbb{T}adv_d}\} \leftarrow$forward($com\_clf\_d$, adv($\{Z_{\mathbb{S}com}, Z_{\mathbb{T}com}\}$));
    $\mathcal{L}_{\mathbb{S}adv\_d} \leftarrow$ cross_entropy($\hat{Y}_{\mathbb{S}adv\_d}$, $\{i, \ldots, i\}$);
    $\mathcal{L}_{\mathbb{T}adv\_d} \leftarrow$ cross_entropy($\hat{Y}_{\mathbb{T}adv\_d}$, $\{0, \ldots, 0\}$);
    $\mathcal{L}_{adv\_d} \leftarrow (\mathcal{L}_{\mathbb{S}adv\_d} + \mathcal{L}_{\mathbb{T}adv\_d})/2$;
    // Feed i-th collaboratively trained specific domain classifier
    $\{\hat{Y}_{\mathbb{S}sp\_d}, \hat{Y}_{\mathbb{T}sp\_d}\} \leftarrow$ forward($sp\_clf\_d_i$, $\{Z_{\mathbb{S}sp_i}, Z_{\mathbb{T}sp_i}\}$);
    $\mathcal{L}_{\mathbb{S}sp\_d} \leftarrow$ cross_entropy($\hat{Y}_{\mathbb{S}sp\_d}$, $\{1, \ldots, 1\}$);
    $\mathcal{L}_{\mathbb{T}sp\_d} \leftarrow$ cross_entropy($\hat{Y}_{\mathbb{T}sp\_d}$, $\{0, \ldots, 0\}$);
    $\mathcal{L}_{sp\_d} \leftarrow (\mathcal{L}_{\mathbb{S}sp\_d} + \mathcal{L}_{\mathbb{T}sp\_d})/2$;
    // Compute MD regularization
    $\gamma \leftarrow 2/(1 + exp(-10e/E))) - 1$;
    $D_i \leftarrow MD(Z_{\mathbb{S}com}, Z_{\mathbb{T}com})$;
    $\mathcal{L}_{MD} \leftarrow \gamma \times D_i$;
    $\alpha_i \leftarrow ((s+1)/s) - (e^{D_i - \max(D)} / \sum e^{D_i - \max(D)})$;
    // Compute global loss and back-propagate
    $\mathcal{L}_y \leftarrow (\mathcal{L}_{\mathbb{T}y} + \alpha_i \times \mathcal{L}_{\mathbb{S}y})/(1 + \alpha_i)$;
    $\mathcal{L} = \mathcal{L}_y + \lambda_1 \mathcal{L}_{adv\_d} + \lambda_2 \mathcal{L}_{sp\_d} + \lambda_3 \mathcal{L}_{MD}$;
    Train $model$, com_clf_d and sp_clf_d by back-propagating $\mathcal{L}$;
  end
end

similarly to MFSAN [9], WMSSDA builds s domain specific latent spaces. The main difference between the second part of WMSSDA and MFSAN is that MFSAN aims to match source and target distributions between all specific latent spaces, which we find counterproductive as it means that all specific latent spaces are pushed toward an identical latent space. We choose to combine both the adversarial and statistical approaches to regularize the common latent space, since we found better results in this way, and could use the MD

results to compute contribution weights. As in the two recent Unsupervised multi-source domain adaptation approaches, ML-MSDA [16] and MLAN [17], we chose to organize our architecture in two branches, one to build a common latent space, and the other to build s domain specific latent spaces. We do so as we believe that only learning a shared latent space across all domains in a multi-source domain adaptation context prevents part of useful knowledge to be transferred from source to target domain. We also noted superior experimental results by using two branches instead of one. We choose to use the Moment Distance to match both target and source common distributions, since [10] demonstrated that Moment Distance is better suited as a statistical distribution matching approach for multi-source domain adaptation than the most common Maximum Mean Discrepancy. We compute a scale variable γ as described in [7], to scale the impact of the MD regularization throughout the training. Its value starts at a 0 and increases logarithmically towards a value of 1 over the total number of epochs, giving more importance to this regularization at the middle and end of the training phase. We apply the computed transfer contribution weights from the MD output on the classification loss of source instances, minimizing as much as possible Negative Learning during training. We apply class weights to scale the computed classification loss to take account of class imbalance during training in a cost-sensitive learning manner. Using a domain adaptation approach also naturally helps in dealing with limited data since source domain knowledge helps improving the overall learning performance. Our proposed WMSSDA approach allows the exploitation of both common knowledge and source domain specific information that is useful for target domain classification.

As we consider, in our learning scenario, that the covariate shift assumption might not hold, meaning that there might be concept shift between domains, we implement a second version of our method. In their paper, [**?**] proposed an interesting and easy-to-implement way to allow domain adaptation approaches to handle concept shift by successfully aligning conditional distribution between two domains $P(Y_{\mathbb{D}_1}|X_{\mathbb{D}_1}) = P(Y_{\mathbb{D}_2}|X_{\mathbb{D}_2})$. Similarly to [9], we implement a second version of our method in which we include this modification, we name this variation WMSSDA-β. We replace each task-specific classifier in WMSSDA-β with a pair of classifiers and follow the following three steps in our training:

1. We train our entire model as previously defined, where label probabilities are defined as the mean of each pair of classifier outputs.
2. We then fix the feature extractor and necks and train the classifier pairs to maximize their discrepancy. The discrepancy between two classifiers C and C' for an instance x is defined as $|C(x) - C'(x)|$.
3. Finally, we train the feature extractor and necks to minimize this same discrepancy with fixed classifiers.

We repeat those steps until global convergence to simultaneously align both marginal and conditional distributions, leading to successful domain adaptation given our scenario. This β version of WMSSDA should be able to better handle concept shift than the standard version, leading to better inference results when in presence of concept shift.

5 Experimental Results

This section presents the experiments we led to evaluate and compare our proposed approach WMSSDA to other state-of-the-art domain adaptation approaches in a data-limited and class imbalance context.

5.1 Datasets

With our experiments we aim to show that our method performs well compared to other state-of-the-art approaches, on both popular benchmark domain adaptation datasets and a real-world medical domain adaptation dataset composed of mixed-type tabular data. We are interested in comparing state-of-the-art domain adaptation approaches in the context of limited data and class imbalance.

The 5-Digits multi-domain dataset is widely used in domain adaptation studies [6–8, 10, 11], it is composed of five digits recognition datasets, with grayscale or color images of various sizes:

1. MNIST[2], the widely known handwritten digit recognition dataset.
2. MNIST-M[3], a more complex version of MNIST, created by combining MNIST images with randomly extracted patches of photos of the BSDS500 dataset as their background.
3. Street View House Numbers (SVHN)[4], a Real-World image dataset of house numbers extracted from Google Street View images.
4. Synthetic Digits (SYN)[5], synthetically generated images of digits with random backgrounds.
5. USPS[6], a handritten digit recognition dataset similar to MNIST.

The DomainNet dataset is a multi-domain dataset recently released with paper [10], and accessible at the following webpage[7]. It aims to provide a new difficult and more advanced benchmark dataset for domain adaptation approaches. It is composed of six domains of color images of various sizes, with a total of 345 common categories across all domains:

1. Clipart, a collection of clipart images.
2. Infograph, infographic images of specific objects.
3. Painting, artistic depictions of objects in the form of paintings.

[2] http://yann.lecun.com/exdb/mnist
[3] https://www.kaggle.com/datasets/aquibiqbal/mnistm
[4] http://ufldl.stanford.edu/housenumbers
[5] https://www.kaggle.com/datasets/prasunroy/synthetic-digits
[6] https://www.kaggle.com/datasets/bistaumanga/usps-dataset
[7] http://ai.bu.edu/M3SDA/#dataset

4. Quickdraw, drawings of the worldwide players of the game "Quick Draw!".
5. Real, photos and real-world images.
6. Sketch, sketches of specific objects.

The 5-Digits and DomainNet datasets are domain adaptation benchmark image datasets that are known for their covariate shift between domains. We perform experiments on those two datasets to demonstrate the capacity of our approach to perform well on known experimental domain adaptation datasets. As we are interested in an experimental scenario with limited data in each domain and class imbalance, we preprocessed the datasets to follow our setting of interest. We randomly selected subsets of each dataset domain to create both a limited amount of data and a class imbalance. After this selection step, the class representation in the 5-Digits dataset spans from 4.22% (10 samples) for the least represented class to 21.09% (500 samples) for the most represented class in each domain. For the DomainNet dataset, the class representation spans from 1.1% (10 samples) for the least represented class to 13.33% (120 samples) for the most represented class in each domain, out of 17 classes, leading to a total of 900 instances in each domain.

The Covid dataset was provided by the Mexican government and is composed of health data about patients that suffer from, or have symptoms that could be related to, Covid19. It is downloadable from the following Kaggle repository[8], it originally contains $1,048,576$ samples from patients suffering from Covid19, with 20 tabular mixed-type (continuous and categorical) features. The goal of this dataset is to predict the survival outcome of patients. The dataset is naturally imbalanced, odds of survival to Covid19 being, fortunately, higher than the odds of death. We used the categorical feature "Medical Unit" to split the original data into 5 domains depending on the type of medical institution that provided the care to the patient. We could not find more information about those kinds of medical institutions, apart from their ID in the dataset, we consider in the following that they correspond to different hospitals as we observe a covariate shift between the institutions. During our preprocessing, we selected 800 unique patients per domain to simulate a limited amount of training data, and we dropped two features containing too many missing values. With the subtraction of the feature used to split the data into separate domains, there remain 17 features after preprocessing.

Figure 3 shows the univariate marginal distributions of four features of the Covid dataset across the five domains. We observe different marginal distributions between all domains, with domain 4 the most different one, with drastically different Sex and Obesity distributions. This simple visualization of the distributions across the Covid dataset pairs of domains is enough to conclude that there is indeed a covariate shift between the five domains.

Table 2 shows the class representation in each domain of the Covid dataset, revealing a prior shift between the domains of the dataset. Indeed, we observe that the label distribution is imbalanced in each domain and the representation

[8] https://www.kaggle.com/datasets/meirnizri/covid19-dataset

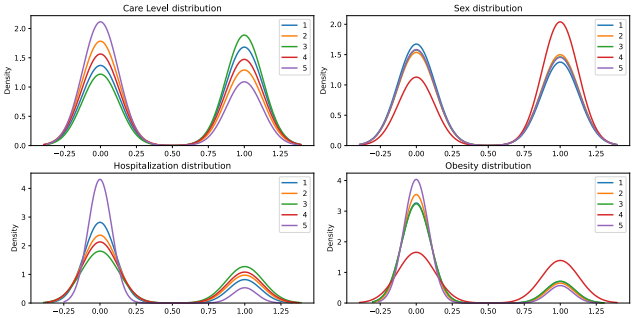

Fig. 3: We observe a covariate shift between the five domains, as univariate marginal distributions are different across pairs of features, $P(X_{\mathbb{D}_1}) \neq P(X_{\mathbb{D}_2}) \neq \ldots \neq P(X_{\mathbb{D}_5})$.

is different from one domain to the other, thus, $P(Y_{\mathbb{D}_1}) \neq P(Y_{\mathbb{D}_2}) \neq \ldots \neq P(Y_{\mathbb{D}_5})$.

Class	Representation	Domain 1	Domain 2	Domain 3	Domain 4	Domain 5	Overall
Negative	Percentage	7.37%	12.37%	13.75%	10.5%	4.87%	9.78%
	Samples Count	59	99	110	84	39	391
Positive	Percentage	92.63%	87.63%	86.25%	89.5%	95.13%	90.22%
	Samples Count	741	701	690	716	761	3,609

Table 2: Covid label distribution per domain, showing that there is a prior shift across domains, $P(Y_{\mathbb{D}_1}) \neq P(Y_{\mathbb{D}_2}) \neq \ldots \neq P(Y_{\mathbb{D}_5})$.

With the preprocessing described above for each dataset, we observe both covariate and prior shifts in all datasets, with limited and imbalanced data in each domain. All datasets might suffer from concept shift, in precaution, the β version of our approach is designed to better handle concept shift.

5.2 Compared Approaches

We compared our proposed WMSSDA to four single-source and four multi-source domain adaptation state-of-the-art approaches. Most of those approaches are initially unsupervised domain adaptation approaches, that is, approaches that do not use labels from the target domain. We modified those approaches to allow them to use target domain labels in their training phase for a fair comparison. The modification for each method is usually as simple as adding the supervised classification task on the target domain into the loss term of each approach. This phase of adapting unsupervised domain adaptation approaches for the supervised domain adaptation context is crucial as it would not be fair to compare our approach, that is able to use target domain labels in its supervised learning, with unsupervised domain adaptation methods that are able to use them. This would lead to far better inference results for

our approach compared to unsupervised ones and would not be representative of the real capacity of unsupervised approaches.

In our experiments, single-source approaches are evaluated using two settings, such as in [10]. In the single best setting, we evaluate the approach on all possible pairs of domains as source and target and select the best-obtained results for each target domain. In the source combine setting, we combine all source domains as one unique domain to obtain only one source domain. We used the following single-source domain adaptation approaches from the literature:

- DAN, Deep Adaptation Network [5] is among the first deep learning unsupervised domain adaptation approaches to have been proposed. DAN uses multi-kernel MMD to minimize the distribution divergence between features extracted on the source and target data, fed through common layers followed by domain specific classifiers. We based our implementation on the following PyTorch implementation[9].
- DANN, Domain-Adversarial Neural Network [6], a model that builds a domain invariant latent space, using an adversarial domain classifier, on which classification is performed. We based our implementation on this PyTorch implementation[10].
- MCD, Maximum Classifier Discrepancy [**?**], an approach that trains a generator and a pair of classifiers by alternating between training the classifiers to maximize their discrepancy, and training the generator to minimize their discrepancy. We based our implementation on this PyTorch implementation[11].
- DSAN, Deep Subdomain Adaptation Network [7], a similar approach to DAN that replaces the MK-MMD with a Local MMD that aligns the distributions of the relevant subdomains. We based our implementation on this PyTorch implementation[12].

We used the following multi-source domain adaptation approaches from the literature:

- MDAN, Multi-source Domain Adversarial Network [8], a similar approach to DANN that uses as many adversarial domain classifiers as source domains. We based our implementation on this PyTorch implementation[13].
- MFSAN, Multiple Feature Spaces Adaptation Network [9], the architecture of this model is composed of a common feature extractor followed by source domain specific parallel layers blocks. Those layers are regularized using MMD and domain specific classifier outputs are aligned with an L1 operation. Classifiers outputs are combined in an ensemble way to obtain

[9] https://github.com/CuthbertCai/pytorch_DAN
[10] https://github.com/fungtion/DANN
[11] https://github.com/mil-tokyo/MCD_DA
[12] https://github.com/easezyc/deep-transfer-learning/tree/master/UDA/pytorch1.0/DSAN
[13] https://github.com/hanzhaoml/MDAN

the final target prediction. We based our implementation on this PyTorch implementation[14].

- M^{3}SDA, Moment Matching for multi-source domain adaptation [10], the architecture is composed of a common feature extractor followed by source domain specific parallel classifiers. The output of the common feature extractor is regularized using a moment-matching distribution distance between source and target data. We based our implementation on this PyTorch implementation[15].
- ABMSDA, Attention-Based Multi-Source Domain Adaptation [11], the architecture of the model is comparable to that of M^{3}SDA, the main difference is that ABMSDA uses a common classifier instead of domain specific ones. ABMSDA computes attention weights with a domain classifier that is fed raw data, the domain classifier outputs are used to derive weights that are applied to the moment matching regularization and to the training loss in an attempt to weight each source domain and avoid negative transfer.

We also used two simple baseline approaches to get reference results:

- NN, a simple neural network, in the first evaluation setting, the model is trained only on the target domain, in the second setting, the model is trained on all combined domains, sources, and target domains alike.
- FT, a simple fine-tuning approach in which a neural network is pre-trained on source data and is then fine-tuned on target data.

5.3 Experimental Protocol

We performed three main experiments, one to evaluate our approach compared to other domain adaptation methods on popular benchmark datasets, a second one to evaluate WMSSDA on a real-world medical multi-source domain adaptation dataset, and finally, an ablation study to evaluate the impact and usefulness of each component of our method. We evaluate all approaches on a supervised domain adaptation classification task with limited data and class imbalance, all unsupervised domain adaptation methods have been slightly modified to handle a labeled target domain as described in the previous section. All our experimental results are compared using the three following classification metrics: the balanced Accuracy (bACC), the Area Under the Receiver Operating Characteristic Curve (AUC), and the F_1-score.

All datasets are split between a training set and a test set, all approaches are trained on the same training data and evaluated on the same test data. To obtain significant results we conduct each experiment 5 times and report the mean and standard deviation of each evaluation metric. To be as fair as possible in our comparisons, in each experiment, we define the model architecture for each approach as similarly as possible across them, while taking account of their architectural differences. For image datasets, our feature extractor is a convolutional network, and all other modules are fully connected networks, for

[14] https://github.com/easezyc/deep-transfer-learning/blob/master/MUDA/MFSAN
[15] https://github.com/VisionLearningGroup/VisionLearningGroup.github.io

the Covid tabular dataset all modules are fully-connected. In the same line of thought, identical hyper-parameters, tuned on the NN approach, are used across all compared approaches for each experiment. All approaches are given the same class weights, those are applied in a cost-sensitive learning manner during loss computation to better handle class imbalance. Those class weights are computed, for each domain, as the inverse of the class distribution observed in the training data: $W_{\mathbb{D}} = 1/P(Y_{\mathbb{D}})$.

To better asses the obtained experimental results, we statistically compare WMSSDA results to each of the compared state-of-the-art approaches using t-tests. The results of those statistical tests are used to determine if our approach performs significantly better, even, or worse than each other, based on a p-values set to 0.05. The results of the t-tests are symbolized in result tables as either a bullet $\bullet$, a circle $\circ$, or an equivalent symbol $\equiv$. The bullet is used to signify that our method is significantly better than the method we compared it to, the circle signifies the opposite, and the equivalent means that there is no significant difference between WMSSDA and the compared method. In the following sections, we will refer as "significantly better" all results that have been evaluated using a t-test and that were classified as significantly better in regard to a p-value of 0.05, and "significantly worse" the opposite.

5.4 Comparative Study on Benchmark Datasets

This first experiment is a comparative study of several baseline and state-of-the-art domain adaptation approaches to evaluate WMSSDA performance on benchmark domain adaptation datasets. We aim to show that our approach is able to compete against, and even outperform, other state-of-the-art domain adaptation approaches on well-known image benchmark multi-domain datasets in a context of limited data and class imbalance.

Table 3 reports our entire experimental results on the 5-Digits dataset, with visual indications of the results of the t-tests between our best-performing models and each compared approach in all settings. As can be seen in the table, our two models, WMSSDA and WMSSDA-β, perform largely better than other approaches in the vast majority of cases, obtaining the best or second-best result for almost all metrics and target domains. We note that apart from our approach, other state-of-the-art multi-source approaches do not reach particularly better results than single-source approaches, we hypothesize that this is a manifestation of negative transfer that multi-source approaches are not yet able to fully avoid. The statistical comparison of the results on 5-Digits shows that our best-performing approach, WMSSDA-β, obtains significantly better results in the vast majority of cases, with only a few settings in which our approach leads to equivalent results to those of other methods. If we consider average performance across all target domains, we can conclude that WMSSDA-β obtains the best prediction results overall, with significantly better results than any other compared approach.

Table 4 reports our entire experimental results on the DomainNet dataset. Prediction results are overall quite low, as this dataset is notoriously hard.

Setting	Method	Metric	MNIST		MNIST-M		SVHN		SYN		USPS		Avg	
Single Best	NN	bACC	84.74 ± 0.29	•	63.08 ± 1.07	•	46.36 ± 1.07	•	61.72 ± 0.70	•	89.49 ± 0.53	•	69.08	•
		AUC	.9915 ± .0004	•	.9353 ± .0046	•	.8578 ± .0081	•	.9333 ± .0028	•	.9934 ± .0007	•	.9423	•
		F1	83.85 ± 0.35	•	58.98 ± 1.25	•	31.47 ± 0.81	•	56.77 ± 1.36	•	88.91 ± 0.40	•	64.00	•
	DAN	bACC	77.56 ± 1.54	•	53.95 ± 2.10	•	41.12 ± 0.14	•	56.46 ± 2.01	•	78.70 ± 1.13	•	61.56	•
		AUC	.9832 ± .0005	•	.8963 ± .0077	•	.8196 ± .0036	•	.9073 ± .0082	•	.9900 ± .0010	•	.9193	•
		F1	75.10 ± 2.15	•	48.61 ± 2.61	•	24.88 ± 0.39	•	49.47 ± 2.79	•	76.74 ± 1.58	•	54.96	•
	DANN	bACC	90.12 ± 0.51	•	69.62 ± 0.85	•	69.37 ± 0.53	•	78.40 ± 0.52	•	96.02 ± 0.24	•	80.71	•
		AUC	.9967 ± .0002	•	.9622 ± .0030	•	.9479 ± .0013	•	.9734 ± .0008	•	.9989 ± .0002	•	.9758	•
		F1	89.43 ± 0.71	•	68.38 ± 1.14	•	**68.26 ± 0.63**	≡	78.06 ± 0.53	•	96.21 ± 0.28	•	80.07	•
	DSAN	bACC	91.00 ± 0.25	•	70.42 ± 1.78	•	67.57 ± 1.03	•	77.30 ± 0.79	•	96.30 ± 0.31	•	80.52	•
		AUC	.9971 ± .0001	•	.9549 ± .0052	•	.9394 ± .0022	•	.9669 ± .0022	•	.9988 ± .0002	•	.9714	•
		F1	90.73 ± 0.29	•	69.44 ± 2.05	•	67.52 ± 1.04	≡	77.27 ± 0.80	•	96.35 ± 0.39	•	80.26	•
	FT	bACC	88.95 ± 0.56	•	68.11 ± 0.98	•	59.32 ± 1.97	•	67.50 ± 1.13	•	93.91 ± 0.34	•	75.56	•
		AUC	.9958 ± .0002	•	.9568 ± .0029	•	.9274 ± .0049	•	.9562 ± .0029	•	.9980 ± .0002	•	.9669	•
		F1	88.54 ± 0.67	•	65.58 ± 0.97	•	51.94 ± 2.35	•	65.03 ± 1.57	•	94.10 ± 0.26	•	73.04	•
	MCD	bACC	92.47 ± 1.00	•	74.09 ± 2.43	•	59.94 ± 4.38	•	76.84 ± 3.16	•	94.95 ± 0.28	•	79.66	•
		AUC	.9968 ± .0005	•	.9666 ± .0084	•	.8972 ± .0178	•	.9665 ± .0081	•	.9982 ± .0005	•	.9651	•
		F1	92.25 ± 1.10	•	73.32 ± 2.27	≡	58.00 ± 4.22	•	76.38 ± 3.41	•	94.81 ± 0.29	•	78.95	•
Source Combine	NN	bACC	91.92 ± 1.00	•	68.66 ± 1.78	•	62.21 ± 1.05	•	78.36 ± 0.76	•	95.77 ± 0.29	•	79.39	•
		AUC	.9971 ± .0005	•	.9572 ± .0029	•	.9354 ± .0030	•	.9737 ± .0013	•	.9983 ± .0002	•	.9724	•
		F1	91.57 ± 1.19	•	67.77 ± 1.52	•	61.04 ± 1.99	•	78.23 ± 0.74	•	95.93 ± 0.34	•	78.91	•
	DANN	bACC	90.12 ± 1.62	•	66.76 ± 1.68	•	54.77 ± 8.47	•	73.98 ± 3.86	•	93.65 ± 1.38	•	75.86	•
		AUC	.9945 ± .0008	•	.9471 ± .0038	•	.9078 ± .0291	•	.9617 ± .0090	•	.9971 ± .0009	•	.9616	•
		F1	89.94 ± 1.62	•	66.22 ± 1.39	•	54.63 ± 7.70	•	73.97 ± 3.77	•	94.06 ± 1.13	•	75.76	•
	DAN	bACC	75.91 ± 1.34	•	53.67 ± 1.02	•	42.48 ± 1.48	•	56.00 ± 0.69	•	81.27 ± 1.65	•	61.87	•
		AUC	.9847 ± .0017	•	.9018 ± .0056	•	.8449 ± .0100	•	.9138 ± .0059	•	.9910 ± .0013	•	.9272	•
		F1	72.56 ± 1.96	•	47.78 ± 1.02	•	24.84 ± 1.13	•	48.14 ± 0.98	•	79.88 ± 2.36	•	54.64	•
	DSAN	bACC	94.11 ± 0.54	•	72.40 ± 0.48	•	65.76 ± 1.01	•	80.41 ± 0.29	•	96.35 ± 0.16	•	81.80	•
		AUC	.9978 ± .0003	•	.9608 ± .0017	•	.9348 ± .0018	•	.9730 ± .0005	•	.9986 ± .0004	•	.9730	•
		F1	94.02 ± 0.57	•	71.88 ± 0.24	•	63.40 ± 1.08	•	80.39 ± 0.31	•	96.38 ± 0.25	•	81.21	•
	FT	bACC	92.76 ± 0.47	•	70.71 ± 0.55	•	59.97 ± 1.08	•	73.49 ± 0.22	•	95.85 ± 0.26	•	78.56	•
		AUC	.9977 ± .0001	•	.9627 ± .0013	•	.9296 ± .0032	•	.9671 ± .0009	•	.9985 ± .0002	•	.9711	•
		F1	92.62 ± 0.50	•	69.38 ± 0.70	•	52.48 ± 1.26	•	72.69 ± 0.30	•	95.93 ± 0.24	•	76.62	•
	MCD	bACC	93.33 ± 0.91	•	64.97 ± 3.06	•	51.95 ± 3.47	•	70.68 ± 0.99	•	93.71 ± 0.91	•	74.93	•
		AUC	.9950 ± .0016	•	.9067 ± .0193	•	.8475 ± .0236	•	.9315 ± .0040	•	.9880 ± .0035	•	.9337	•
		F1	93.33 ± 0.91	•	66.18 ± 2.56	•	50.63 ± 4.02	•	70.52 ± 0.99	•	94.22 ± 0.78	•	74.98	•
Multi Source	MDAN	bACC	87.63 ± 0.98	•	53.44 ± 1.66	•	43.47 ± 2.08	•	60.64 ± 1.27	•	87.58 ± 2.12	•	66.55	•
		AUC	.9946 ± .0005	•	.9194 ± .0036	•	.8925 ± .0113	•	.9469 ± .0042	•	.9946 ± .0016	•	.9496	•
		F1	87.29 ± 1.11	•	49.77 ± 1.59	•	26.62 ± 1.26	•	57.22 ± 1.39	•	87.85 ± 2.02	•	61.75	•
	MFSAN	bACC	90.97 ± 0.46	•	70.75 ± 1.19	•	62.76 ± 2.64	•	75.06 ± 0.97	•	95.87 ± 0.34	•	79.08	•
		AUC	.9970 ± .0003	•	.9607 ± .0055	•	.9378 ± .0050	•	.9734 ± .0011	•	.9986 ± .0002	•	.9735	•
		F1	90.67 ± 0.55	•	69.45 ± 1.30	•	57.28 ± 3.75	•	74.07 ± 1.14	•	96.10 ± 0.30	•	77.51	•
	M3SDA	bACC	92.38 ± 1.41	•	65.45 ± 1.19	•	57.56 ± 2.61	•	75.85 ± 0.43	•	94.69 ± 0.46	•	77.19	•
		AUC	.9963 ± .0009	•	.9421 ± .0039	•	.9178 ± .0078	•	.9660 ± .0012	•	.9976 ± .0004	•	.9639	•
		F1	92.20 ± 1.57	•	64.67 ± 1.46	•	56.41 ± 2.87	•	75.56 ± 0.39	•	94.84 ± 0.52	•	76.74	•
	ABMSDA	bACC	93.53 ± 0.50	•	67.01 ± 1.84	•	53.47 ± 4.63	•	77.45 ± 0.56	•	95.23 ± 0.77	•	77.34	•
		AUC	.9977 ± .0003	•	.9471 ± .0041	•	.9079 ± .0112	•	.9677 ± .0014	•	.9983 ± .0004	•	.9637	•
		F1	93.37 ± 0.58	•	66.78 ± 2.09	•	52.12 ± 5.24	•	77.22 ± 0.62	•	95.57 ± 0.75	•	77.01	•
	WMSSDA	bACC	**95.30 ± 0.21**		75.05 ± 1.06		70.50 ± 1.21		82.85 ± 0.49		96.77 ± 0.30		84.10	
		AUC	.9988 ± .0001		.9737 ± .0017		.9519 ± .0014		.9838 ± .0005		.9992 ± .0001		.9815	
		F1	**95.25 ± 0.21**		74.62 ± 1.17		68.17 ± 2.01		82.70 ± 0.49		96.90 ± 0.29		83.53	
	WMSSDA-β	bACC	95.21 ± 0.53		**77.58 ± 1.75**		**71.83 ± 1.57**		**84.74 ± 0.59**		**96.91 ± 0.24**		**85.26**	
		AUC	**.9989 ± .0001**		**.9790 ± .0025**		**.9596 ± .0027**		**.9876 ± .0006**		**.9993 ± .0002**		**.9849**	
		F1	95.14 ± 0.56		**76.68 ± 2.05**		67.80 ± 1.95		**84.60 ± 0.62**		**97.00 ± 0.22**		**84.24**	

WMSSDA is: • significantly better, ≡ equivalent, ∘ significantly worse, p-value: 0.05

Table 3: Comparative Study on the 5-Digits domain adaptation Benchmark Dataset, with Limited and Imbalanced Data. The best and second-best results for each metric and each target domain appear in bold and underlined respectively. Results are evaluated on a multi-class classification task using metrics: balanced Accuracy, AUC, and the F_1-Score.

The fact that we drastically reduced the amount of available training data worsens classification results, but this does not affect the comparison potential of the results. We can see that our proposed WMSSDA-β obtains the best or second-best results in the majority of cases, while WMSSDA obtains overall good results but not better than other approaches. It is important to note that the overall best prediction results for the Quickdraw domain are reached by a simple neural network trained on the target domain only. This means that no domain adaptation method is currently able to avoid negative transfer enough to match those results, nor are they able to surpass them. This shows that negative transfer in domain adaptation is still an open-problem, and that it is crucial to find better ways of solving this issue. The second best results on the Quickdraw domain are reached by the Fine-Tuning approach,

Setting	Method	Metric	CLIP		INFO		PAINT		REAL		SKETCH		QUICK		Avg	
Single Best	NN	bACC	27.43 ± 0.90	≡	16.16 ± 0.81	≡	23.14 ± 0.19	•	35.51 ± 1.36	•	17.04 ± 1.75	•	**44.76 ± 0.77**	○	27.34	•
		AUC	.7466 ± .0044	•	.6283 ± .0043	•	.7171 ± .0062	•	.8372 ± .0063	•	.6489 ± .0086	•	**.8707 ± .0052**	○	.7415	•
		F1	22.85 ± 0.94	•	13.21 ± 0.82	•	21.22 ± 0.48	•	32.26 ± 1.47	•	14.76 ± 1.60	•	**39.89 ± 0.97**	○	24.03	•
	DAN	bACC	23.08 ± 1.90	•	13.78 ± 1.02	•	17.78 ± 2.99	•	28.84 ± 1.42	•	15.29 ± 0.63	•	36.08 ± 1.23	•	22.48	•
		AUC	.7072 ± .0101	•	.6070 ± .0152	•	.6504 ± .0100	•	.7722 ± .0163	•	.6367 ± .0060	•	.8129 ± .0029	•	.6977	•
		F1	19.22 ± 2.11	•	11.20 ± 1.25	•	14.01 ± 4.13	•	24.50 ± 0.91	•	13.30 ± 0.84	•	29.57 ± 1.20	•	18.63	•
	DANN	bACC	28.37 ± 0.33	≡	17.24 ± 0.77	≡	23.75 ± 1.17	≡	37.22 ± 1.09	•	19.02 ± 1.42	•	34.12 ± 1.65	•	26.62	•
		AUC	.7593 ± .0040	•	.6514 ± .0058	•	.7354 ± .0068	•	.8499 ± .0054	•	.6849 ± .0082	•	.8188 ± .0077	•	.7499	•
		F1	25.85 ± 0.70	•	**16.39 ± 0.64**	≡	22.42 ± 0.97	•	36.51 ± 1.07	•	17.41 ± 1.26	•	29.53 ± 1.67	•	24.69	•
	DSAN	bACC	5.90 ± 0.40	•	5.88 ± 0.00	•	14.82 ± 1.65	•	25.51 ± 5.35	•	5.88 ± 0.00	•	18.57 ± 2.51	•	12.76	•
		AUC	.5741 ± .0105	•	.5465 ± .0061	•	.6466 ± .0113	•	.7632 ± .0443	•	.5444 ± .0027	•	.7574 ± .0109	•	.6387	•
		F1	0.95 ± 0.53	•	0.65 ± 0.00	•	11.14 ± 2.14	•	21.23 ± 7.06	•	0.65 ± 0.00	•	13.32 ± 2.35	•	7.99	•
	FT	bACC	29.41 ± 1.16	≡	16.39 ± 1.16	≡	23.51 ± 1.07	•	37.37 ± 0.92	•	19.61 ± 0.74	≡	44.51 ± 1.10	○	28.45	○
		AUC	.7664 ± .0077	•	.6446 ± .0038	•	.7386 ± .0069	•	.8513 ± .0027	•	.6858 ± .0055	•	.8663 ± .0023	≡	.7588	•
		F1	25.61 ± 1.21	•	13.97 ± 1.20	≡	21.95 ± 0.83	•	34.63 ± 1.21	•	17.87 ± 0.44	•	39.81 ± 1.18	○	25.64	•
	MCD	bACC	26.47 ± 1.43	≡	16.90 ± 0.74	≡	21.90 ± 0.66	•	34.94 ± 0.83	•	19.96 ± 1.03	≡	37.98 ± 0.86	•	26.36	•
		AUC	.7328 ± .0142	•	.6389 ± .0056	•	.6952 ± .0055	•	.8216 ± .0039	•	.6667 ± .0046	•	.8253 ± .0055	•	.7301	•
		F1	25.48 ± 1.07	•	15.86 ± 0.20	≡	20.85 ± 0.67	•	34.27 ± 0.57	•	18.84 ± 1.15	≡	33.68 ± 1.00	≡	24.83	•
Source Combine	NN	bACC	26.51 ± 1.60	≡	14.65 ± 1.22	≡	22.88 ± 1.48	•	34.67 ± 1.45	•	19.73 ± 1.25	≡	26.92 ± 0.98	•	24.23	•
		AUC	.7451 ± .0122	•	.6357 ± .0075	•	.7103 ± .0072	•	.8223 ± .0098	•	.6903 ± .0085	•	.7518 ± .0083	•	.7259	•
		F1	25.78 ± 1.35	≡	14.52 ± 1.09	≡	22.08 ± 1.32	•	34.78 ± 1.74	•	18.25 ± 0.72	•	21.65 ± 1.54	•	22.84	•
	DANN	bACC	25.41 ± 0.99	•	14.82 ± 1.18	≡	22.10 ± 1.44	•	32.53 ± 2.46	•	20.51 ± 1.63	≡	27.98 ± 1.07	•	23.89	•
		AUC	.7344 ± .0031	•	.6324 ± .0069	•	.7104 ± .0126	•	.8073 ± .0179	•	.6920 ± .0061	•	.7465 ± .0161	•	.7205	•
		F1	24.79 ± 1.16	•	14.85 ± 1.06	≡	21.15 ± 1.35	•	32.86 ± 2.76	•	19.42 ± 1.77	≡	22.21 ± 1.84	•	22.55	•
	DAN	bACC	21.67 ± 1.02	•	11.41 ± 1.41	•	16.78 ± 1.20	•	28.75 ± 0.77	•	15.71 ± 0.75	•	33.31 ± 0.94	•	21.27	•
		AUC	.6988 ± .0065	•	.5923 ± .0160	•	.6602 ± .0167	•	.7782 ± .0103	•	.6264 ± .0049	•	.7946 ± .0079	•	.6918	•
		F1	17.33 ± 1.23	•	7.76 ± 2.36	•	14.49 ± 0.79	•	25.02 ± 1.06	•	12.56 ± 0.97	•	26.22 ± 0.65	•	17.23	•
	DSAN	bACC	5.88 ± 0.06	•	5.76 ± 0.11	•	5.84 ± 0.05	•	5.92 ± 0.08	•	5.84 ± 0.08	•	5.88 ± 0.00	•	5.86	•
		AUC	.5056 ± .0137	•	.5107 ± .0115	•	.5153 ± .0232	•	.4794 ± .0377	•	.5054 ± .0096	•	.5037 ± .0234	•	.5034	•
		F1	0.82 ± 0.11	•	0.68 ± 0.06	•	0.85 ± 0.31	•	0.73 ± 0.15	•	0.68 ± 0.06	•	0.65 ± 0.00	•	0.74	•
	FT	bACC	29.76 ± 1.00	○	17.00 ± 0.58	≡	24.22 ± 0.55	•	38.69 ± 1.04	≡	20.71 ± 0.72	≡	41.37 ± 0.67	≡	**28.62**	○
		AUC	.7762 ± .0061	•	.6463 ± .0050	•	.7362 ± .0053	•	.8567 ± .0044	•	.6978 ± .0092	•	.8431 ± .0028	•	.7594	•
		F1	26.83 ± 0.68	≡	15.49 ± 0.28	≡	22.80 ± 0.31	•	36.67 ± 0.93	•	19.08 ± 0.72	≡	36.52 ± 0.75	≡	26.23	•
	MCD	bACC	25.63 ± 0.64	•	13.04 ± 0.78	•	21.88 ± 1.36	•	33.47 ± 1.74	•	18.18 ± 1.50	•	27.80 ± 2.22	•	23.33	•
		AUC	.7315 ± .0076	•	.6166 ± .0066	•	.6899 ± .0116	•	.8050 ± .0161	•	.6542 ± .0111	•	.7579 ± .0111	•	.7092	•
		F1	25.14 ± 0.77	•	12.52 ± 0.67	•	20.32 ± 1.24	•	32.12 ± 2.07	•	16.82 ± 1.18	•	23.50 ± 2.48	•	21.74	•
Multi Source	MDAN	bACC	27.88 ± 0.89	≡	15.82 ± 0.90	≡	22.43 ± 0.32	•	35.69 ± 1.41	•	18.73 ± 0.85	•	30.00 ± 2.42	•	25.09	•
		AUC	.7789 ± .0074	•	.6515 ± .0035	•	.7310 ± .0111	•	.8560 ± .0018	•	.6887 ± .0083	•	.8269 ± .0094	•	.7555	•
		F1	24.60 ± 1.31	•	13.45 ± 1.06	•	20.40 ± 0.72	•	32.99 ± 1.46	•	16.36 ± 1.08	•	23.14 ± 2.85	•	21.82	•
	MFSAN	bACC	**30.29 ± 0.72**	○	**17.39 ± 0.50**	○	**26.18 ± 0.41**	≡	38.51 ± 0.83	≡	21.96 ± 0.75	≡	36.41 ± 1.01	•	28.45	○
		AUC	.7708 ± .0035	•	.6548 ± .0058	•	.7437 ± .0057	•	.8379 ± .0042	•	.7066 ± .0072	•	.8104 ± .0125	•	.7540	•
		F1	26.43 ± 0.45	≡	15.69 ± 0.78	≡	24.23 ± 0.51	≡	35.77 ± 1.15	•	19.98 ± 0.93	≡	29.52 ± 1.13	•	25.27	•
	M3SDA	bACC	26.14 ± 1.19	•	15.55 ± 0.90	≡	22.12 ± 0.42	•	34.24 ± 1.82	•	19.88 ± 0.89	≡	23.10 ± 1.70	•	23.50	•
		AUC	.7386 ± .0070	•	.6342 ± .0056	•	.7102 ± .0048	•	.8120 ± .0057	•	.6804 ± .0087	•	.7107 ± .0058	•	.7143	•
		F1	25.47 ± 1.49	≡	15.42 ± 1.20	≡	21.22 ± 0.45	•	34.11 ± 1.63	•	18.84 ± 1.08	≡	16.84 ± 2.03	•	21.98	•
	ABMSDA	bACC	25.43 ± 1.50	•	14.84 ± 0.82	≡	21.73 ± 0.71	•	32.20 ± 1.32	•	19.80 ± 0.67	≡	24.94 ± 2.02	•	23.16	•
		AUC	.7269 ± .0063	•	.6201 ± .0024	•	.6991 ± .0046	•	.7966 ± .0122	•	.6755 ± .0065	•	.7245 ± .0129	•	.7071	•
		F1	24.75 ± 1.55	•	14.87 ± 1.11	≡	20.80 ± 0.48	•	32.03 ± 1.34	•	18.59 ± 0.76	≡	19.24 ± 2.14	•	21.71	•
	WMSSDA	bACC	29.84 ± 1.38		16.25 ± 1.19		24.96 ± 0.89		39.04 ± 0.70		**22.02 ± 0.69**		32.88 ± 2.08		27.50	
		AUC	.7737 ± .0103		.6504 ± .0078		.7447 ± .0048		.8500 ± .0059		.7044 ± .0094		.8007 ± .0088		.7540	
		F1	**28.31 ± 1.28**		16.06 ± 1.09		24.08 ± 0.87		**38.87 ± 0.64**		**20.74 ± 0.74**		28.08 ± 3.06		26.02	
	WMSSDA-β	bACC	28.18 ± 0.58		15.96 ± 1.04		25.27 ± 0.70		**39.29 ± 1.26**		21.14 ± 1.15		40.04 ± 0.99		28.31	
		AUC	**.7971 ± .0033**		**.6791 ± .0055**		**.7663 ± .0059**		**.8821 ± .0036**		**.7162 ± .0063**		.8620 ± .0045		**.7838**	
		F1	27.28 ± 0.64		15.50 ± 1.11		**24.45 ± 0.83**		38.72 ± 1.45		20.17 ± 1.25		34.85 ± 1.48		**26.83**	

WMSSDA is: • significantly better, ≡ equivalent, ○ significantly worse, p-value: 0.05

Table 4: Comparative Study on the DomainNet domain adaptation Benchmark Dataset, with Limited and Imbalanced Data. The best and second-best results for each metric and each target domain appear in bold and underlined respectively. Results are evaluated on a multi-class classification task using metrics: balanced Accuracy, AUC, and the F_1-Score.

followed by our WMSSDA-β approach, showing that our method is currently the best-performing multi-source domain adaptation approach to avoid negative transfer. The simple Fine-Tuning approach leads to significantly better average balanced Accuracy results than ours and almost all other methods on both single best and source combine settings. This probably shows that adding a short phase of pre-training to other state-of-the-art approaches would drastically improve the overall results of all methods. Another well-performing approach in this setting is MFSAN, which leads to significantly better balanced Accuracy than ours on average. Overall, our two versions of WMSSDA lead to competitive experimental results on this dataset, with WMSSDA-β leading to significantly better results than other approaches in the vast majority of cases. When considering the average over all target domains, WMSSDA obtains significantly better results than all other approaches on the three evaluation metrics except for Fine-Tuning and MFSAN on the balanced accuracy metric.

We can conclude from this experiment on benchmark datasets that our proposed approach is able to compete, and even surpass, other baseline and state-of-the-art domain adaptation approaches in our supervised multi-domain with limited and imbalanced data context.

5.5 Comparative Study on Real-World Tabular Medical Dataset

The second experiment is a comparative study between all tested domain adaptation approaches and our proposed WMSSDA, on the real-world mixed-type tabular medical Covid dataset. We aim to show that our approach can reach good results on mixed-type tabular data in addition to image data. We also aim to show that WMSSDA competes and outperforms other domain adaptation approaches in a real-world medical context with limited data and class imbalance.

Table 5 reports our entire experimental results on the Covid dataset. Results show that both WMSSDA variations lead to the best or second-best results in most cases, and in all cases on average. Those experimental results show that our approach can perform very well on tabular data. In this particular setting, we note that it is our standard version of WMSSDA that performs the best, which is probably an indication that there is almost no concept shift in this dataset, unlike with the two benchmark image datasets, rendering the β version of the approach no better than the standard one. The results between the two versions of WMSSDA are close, with slightly better results for our standard version, thus, we performed the statistical tests evaluation based on the standard version of WMSSDA. The statistical comparison of the results shows that our approach WMSSDA leads to significantly better results than most other approaches, in the majority of cases. Our approach leads to significantly better results than any other state-of-the-art approach when considering the average performance over all target domains.

5.6 Ablation Study

Our method WMSSDA is composed of several important elements, in this section we perform an ablation study in order to evaluate the pertinence and usefulness of each component of WMSSDA. An ablation study is a type of experiment that is conducted to investigate the impact of removing, or disabling, specific components or features of an approach. To do so, we eliminate parts of our model and evaluate the results to understand their individual contributions to the overall performance and validate the pertinence of each component.

In this ablation study we compare five ablated versions of our WMSSDA approach with our complete method. Table 6 shows all WMSSDA versions compared in this study. In the column “Branches” it is indicated if the method contains both the common modules branch and the source domain specific modules branch, or only one of the two. Column “Regus” indicates if both

Setting	Method	Metric	1		2		3		4		5		Avg	
Single Best	NN	bACC	86.15 ± 1.15	•	85.57 ± 0.46	•	85.41 ± 0.37	•	83.47 ± 0.30	•	85.91 ± 2.92	•	85.30	•
		AUC	.9257 ± .0083	•	.9145 ± .0042	•	.8983 ± .0042	•	.8878 ± .0066	•	.9621 ± .0048	•	.9177	•
		F1	81.52 ± 1.14	•	82.06 ± 0.83	•	80.79 ± 1.00	•	78.80 ± 0.68	•	81.79 ± 3.55	•	80.99	•
	DAN	bACC	87.09 ± 1.01	•	80.96 ± 7.17	≡	79.50 ± 4.82	•	82.36 ± 1.14	•	92.90 ± 1.20	•	84.56	•
		AUC	.9261 ± .0074	•	.9052 ± .0079	•	.8910 ± .0069	•	.8928 ± .0055	•	.9646 ± .0090	•	.9159	•
		F1	85.64 ± 2.25	≡	75.02 ± 13.25	≡	71.73 ± 9.18	•	78.83 ± 2.88	≡	92.00 ± 1.36	•	80.64	•
	DANN	bACC	85.10 ± 0.82	•	85.82 ± 0.58	•	84.44 ± 0.97	•	83.31 ± 1.54	•	92.63 ± 1.05	•	86.26	•
		AUC	.9331 ± .0021	•	.9210 ± .0011	•	.9047 ± .0043	•	.9071 ± .0043	≡	.9701 ± .0025	•	.9272	•
		F1	80.01 ± 1.14	•	81.68 ± 1.31	•	78.74 ± 2.01	•	77.18 ± 2.39	•	90.60 ± 1.35	•	81.64	•
	DSAN	bACC	84.62 ± 3.17	•	84.82 ± 1.52	•	82.99 ± 1.32	•	83.30 ± 0.62	•	91.20 ± 2.03	•	85.39	•
		AUC	.9146 ± .0089	•	.9112 ± .0088	•	.8941 ± .0073	•	.8944 ± .0067	•	.9575 ± .0088	•	.9143	•
		F1	81.45 ± 4.51	•	84.56 ± 1.14	≡	77.88 ± 2.07	•	79.67 ± 1.03	•	90.94 ± 2.63	•	82.90	•
	FT	bACC	84.47 ± 0.97	•	85.12 ± 0.63	•	84.43 ± 0.61	•	83.31 ± 1.42	•	90.17 ± 2.39	•	85.50	•
		AUC	.9311 ± .0040	•	.9179 ± .0022	•	.9061 ± .0018	•	.9049 ± .0018	≡	.9689 ± .0014	•	.9258	•
		F1	79.23 ± 1.03	•	81.33 ± 1.98	•	78.70 ± 1.02	•	77.48 ± 2.00	•	86.71 ± 3.09	•	80.69	•
	MCD	bACC	86.67 ± 1.35	•	85.49 ± 0.34	•	85.23 ± 0.40	•	84.35 ± 0.85	≡	91.82 ± 2.42	•	86.71	•
		AUC	.9350 ± .0016	•	.9207 ± .0029	≡	.9089 ± .0014	•	**.9114 ± .0026**	○	.9718 ± .0010	•	.9296	•
		F1	82.38 ± 1.70	•	82.51 ± 1.18	•	79.79 ± 0.64	•	78.94 ± 1.69	•	89.26 ± 2.97	•	82.58	•
Source Combine	NN	bACC	87.93 ± 0.44	•	86.53 ± 0.22	•	85.26 ± 0.18	•	83.90 ± 0.34	•	94.45 ± 0.23	•	87.61	•
		AUC	.9389 ± .0019	•	.9168 ± .0033	•	.9111 ± .0037	≡	.9036 ± .0031	≡	.9751 ± .0009	•	.9291	•
		F1	83.86 ± 0.77	•	83.95 ± 0.52	•	79.07 ± 0.48	•	78.66 ± 0.44	•	93.94 ± 0.28	•	83.90	•
	DANN	bACC	88.12 ± 0.27	•	86.79 ± 0.40	≡	85.41 ± 0.15	•	83.98 ± 0.32	•	94.78 ± 0.13	≡	87.82	•
		AUC	.9391 ± .0023	≡	.9188 ± .0038	•	.9118 ± .0007	•	.9074 ± .0016	≡	.9747 ± .0002	•	.9304	•
		F1	84.34 ± 0.42	•	84.27 ± 0.51	•	79.15 ± 0.28	•	78.66 ± 0.28	•	94.05 ± 0.16	•	84.09	•
	DAN	bACC	87.40 ± 1.23	•	86.22 ± 0.22	•	84.78 ± 0.75	•	83.99 ± 0.38	•	94.25 ± 0.24	•	87.33	•
		AUC	.9306 ± .0077	•	.9238 ± .0023	≡	.9078 ± .0034	•	.9026 ± .0029	•	.9726 ± .0012	•	.9275	•
		F1	84.57 ± 1.18	•	83.13 ± 1.24	•	80.11 ± 1.31	•	79.18 ± 0.73	•	93.38 ± 0.83	≡	84.07	•
	DSAN	bACC	87.53 ± 1.02	•	85.85 ± 0.23	•	85.67 ± 1.02	•	84.21 ± 0.77	•	92.27 ± 1.70	•	87.11	•
		AUC	.9307 ± .0027	•	.9197 ± .0052	≡	.9069 ± .0046	•	.9053 ± .0051	≡	.9585 ± .0082	•	.9242	•
		F1	84.15 ± 1.52	•	84.15 ± 0.89	•	80.44 ± 1.71	•	78.76 ± 0.83	•	92.11 ± 1.57	•	83.92	•
	FT	bACC	85.88 ± 0.52	•	86.08 ± 0.50	•	85.39 ± 0.42	•	84.43 ± 1.12	≡	92.43 ± 1.53	•	86.84	•
		AUC	.9358 ± .0031	•	.9210 ± .0022	•	.9083 ± .0030	•	.9055 ± .0039	≡	.9706 ± .0019	•	.9282	•
		F1	81.00 ± 0.68	•	82.40 ± 0.72	•	79.76 ± 0.78	•	79.51 ± 1.02	•	89.73 ± 2.06	•	82.48	•
	MCD	bACC	84.74 ± 1.75	•	84.25 ± 1.73	•	83.81 ± 0.87	•	83.59 ± 0.60	•	90.98 ± 1.33	•	85.47	•
		AUC	.9288 ± .0028	•	.9103 ± .0094	•	.9060 ± .0028	•	.9091 ± .0027	≡	.9674 ± .0033	•	.9243	•
		F1	79.64 ± 2.83	•	80.80 ± 1.98	•	77.31 ± 1.40	•	77.65 ± 1.33	•	88.38 ± 1.78	•	80.76	•
Multi Source	MDAN	bACC	66.96 ± 7.40	•	73.42 ± 8.83	•	77.47 ± 2.40	•	72.11 ± 4.53	•	71.34 ± 8.91	•	72.26	•
		AUC	.8415 ± .0451	•	.8474 ± .0244	•	.8460 ± .0285	•	.8215 ± .0259	•	.8669 ± .0363	•	.8447	•
		F1	52.83 ± 17.89	•	65.41 ± 19.93	≡	72.76 ± 3.70	•	63.68 ± 11.61	•	63.02 ± 20.23	•	63.54	•
	MFSAN	bACC	86.42 ± 0.35	•	85.22 ± 0.39	•	83.07 ± 0.43	•	83.75 ± 0.77	•	92.20 ± 0.85	•	86.13	•
		AUC	.9300 ± .0035	•	.9196 ± .0018	•	.9059 ± .0027	•	.9108 ± .0024	≡	.9643 ± .0050	•	.9261	•
		F1	82.82 ± 0.40	•	81.87 ± 1.11	•	76.25 ± 0.84	•	77.36 ± 1.10	•	90.07 ± 1.06	•	81.67	•
	M3SDA	bACC	86.45 ± 0.73	•	85.00 ± 0.47	•	84.51 ± 0.27	•	82.92 ± 0.77	•	93.70 ± 0.86	•	86.52	•
		AUC	.9351 ± .0022	•	.9174 ± .0057	≡	.9067 ± .0015	•	.9010 ± .0024	•	.9695 ± .0022	•	.9259	•
		F1	82.24 ± 1.04	•	81.22 ± 0.76	•	78.10 ± 0.57	•	76.76 ± 1.20	•	92.85 ± 1.60	≡	82.23	•
	ABMSDA	bACC	86.78 ± 1.09	•	85.88 ± 0.97	≡	83.66 ± 1.16	•	81.00 ± 0.93	•	93.45 ± 1.91	≡	86.15	•
		AUC	.9281 ± .0043	•	.9190 ± .0033	•	.9068 ± .0034	•	.9049 ± .0021	≡	.9695 ± .0022	•	.9256	•
		F1	83.52 ± 1.48	•	82.60 ± 2.07	•	77.00 ± 2.01	•	73.44 ± 1.48	•	92.08 ± 2.63	≡	81.73	•
	WMSSDA	bACC	**89.24 ± 0.30**		**86.97 ± 0.28**		**87.03 ± 0.14**		**85.33 ± 0.24**		**94.88 ± 0.20**		**88.69**	
		AUC	**.9413 ± .0006**		.9240 ± .0014		**.9147 ± .0003**		.9074 ± .0016		**.9764 ± .0006**		**.9328**	
		F1	**86.87 ± 0.37**		**85.65 ± 0.21**		**83.05 ± 0.33**		**81.32 ± 0.26**		**94.34 ± 0.13**		**86.25**	
	WMSSDA-β	bACC	88.85 ± 0.61		86.61 ± 0.17		86.29 ± 0.24		84.60 ± 0.62		94.21 ± 0.27		88.11	
		AUC	.9378 ± .0027		**.9242 ± .0017**		.9112 ± .0017		.9065 ± .0064		.9735 ± .0010		.9306	
		F1	85.31 ± 1.39		84.50 ± 0.74		82.05 ± 0.89		80.36 ± 0.53		92.86 ± 0.50		85.02	

WMSSDA is: • significantly better, ≡ equivalent, ○ significantly worse, p-value: 0.05

Table 5: Comparative Study on the Real-World Medical Covid Dataset Dataset, with Limited and Imbalanced Data. The best and second-best results for each metric and each target domain appear in bold and underlined respectively. Results are evaluated on a binary classification task using metrics: balanced Accuracy, AUC, and the F_1-Score.

Approach	Branches	Regus	Weights
WMSSDA-A	Common	MD+ADV	No
WMSSDA-B	Specific	-	No
WMSSDA-C	Common+Specific	ADV	No
WMSSDA-D	Common+Specific	MD	No
WMSSDA-E	Common+Specific	MD+ADV	No
WMSSDA	Common+Specific	MD+ADV	Yes

Table 6: Ablation study compared approaches.

the statistical and adversarial regularizations of the common branch are used, or only one of the two, or none in the case where only the specific branch is

used. Finally, column "Weights" indicates if transfer contribution weights are computed and used during training to minimize negative transfer or not.

Method	Metric	MNIST	MNIST-M	SVHN	SYN	USPS	Avg
WMSSDA-A	bACC	94.95 ± 0.58	72.02 ± 1.24	67.69 ± 2.28	80.94 ± 0.49	96.57 ± 0.22	82.43
	AUC	.9985 ± .0001	.9620 ± .0024	.9429 ± .0054	.9783 ± .0012	.9989 ± .0002	.9761
	F1	94.89 ± 0.61	71.20 ± 1.35	65.64 ± 2.70	80.63 ± 0.53	96.83 ± 0.19	81.84
WMSSDA-B	bACC	93.77 ± 0.90	74.62 ± 1.50	66.34 ± 2.24	79.72 ± 1.20	95.89 ± 0.35	82.07
	AUC	.9980 ± .0003	.9721 ± .0021	.9414 ± .0048	.9805 ± .0013	.9982 ± .0005	.9781
	F1	93.71 ± 0.92	73.79 ± 1.72	62.83 ± 2.40	79.48 ± 1.18	96.06 ± 0.47	81.17
WMSSDA-C	bACC	93.95 ± 0.61	73.52 ± 1.32	69.61 ± 1.60	81.58 ± 0.95	96.60 ± 0.16	83.05
	AUC	.9982 ± .0002	.9709 ± .0017	.9482 ± .0049	.9813 ± .0014	.9984 ± .0004	.9794
	F1	93.79 ± 0.71	72.98 ± 1.58	67.29 ± 2.87	81.41 ± 0.99	96.73 ± 0.11	82.44
WMSSDA-D	bACC	95.01 ± 0.46	74.60 ± 1.33	69.27 ± 2.05	81.87 ± 1.42	96.28 ± 0.21	83.41
	AUC	.9988 ± .0002	.9736 ± .0025	.9505 ± .0035	.9828 ± .0019	.9987 ± .0003	.9809
	F1	94.93 ± 0.50	74.19 ± 1.40	67.16 ± 2.34	81.70 ± 1.43	96.38 ± 0.14	82.87
WMSSDA-E	bACC	95.16 ± 0.25	74.94 ± 0.76	69.20 ± 1.17	82.60 ± 0.46	96.60 ± 0.28	83.70
	AUC	**.9989 ± .0002**	.9737 ± .0024	.9501 ± .0016	.9833 ± .0008	.9987 ± .0003	.9809
	F1	95.11 ± 0.26	74.50 ± 0.88	67.11 ± 1.56	82.45 ± 0.43	96.65 ± 0.30	83.17
WMSSDA	bACC	**95.30 ± 0.21**	**75.05 ± 1.06**	**70.50 ± 1.21**	**82.85 ± 0.49**	**96.77 ± 0.30**	**84.10**
	AUC	.9988 ± .0001	**.9737 ± .0017**	**.9519 ± .0014**	**.9838 ± .0005**	**.9992 ± .0001**	**.9815**
	F1	**95.25 ± 0.21**	**74.62 ± 1.17**	**68.17 ± 2.01**	**82.70 ± 0.49**	**96.90 ± 0.29**	**83.53**

Table 7: Ablation study results.

Table 7 shows our experimental results on the Covid dataset for this ablation study. We compare our method with only the common modules branch with method WMSSDA-A and only the source domain specific modules branch with method WMSSDA-B. In both cases, average results are similar, with slightly better results for WMSSDA-A, showing the importance of a shared latent space. Method WMSSDA-E contains both branches and obtains largely better results than WMSSDA-A and WMSSDA-B, showing the pertinence of an architecture combining both the common and specific branches to obtain the best possible results. We believe that the two branches architecture naturally decreases negative transfer as classifiers with higher confidence are given more importance in the ensemble pooling of results, which highly contributes to improving prediction quality. We also evaluate the pertinence of using both statistical and adversarial regularizations to learn a shared domain invariant latent space, with method WMSSDA-C using only the adversarial regularization, and method WMSSDA-D using only the statistical MD regularization. The results of WMSSDA-C and WMSSDA-D show that using a statistical only regularization leads to slightly better results than the adversarial one alone. This is contradictory with the actual consensus in the literature, that states that adversarial regularization is superior to statistical regularization for learning a common domain-invariant latent space. This can probably be explained by the fact that MD has been shown to be more pertinent and lead to better learning performance than MMD in a multi-source domain adaptation context in [10], making it slightly superior to adversarial alignment in this case. Method WMSSDA-E, which uses both MD and adversarial regularization for its shared latent space obtains better results than both WMSSDA-C and WMSSDA-D, showing that using both kind of regularizations allows to further align the latent space and lead to even better adaptation results. Finally, we observe that our complete WMSSDA approach leads to the best results overall, which seems to indicate that our transfer contribution weights are useful

to limit negative transfer during training and have a positive effect on learning performance.

6 Discussion and Conclusion

In this paper, we proposed an innovative multi-source supervised domain adaptation approach, Weighted Multi-Source Supervised Domain Adaptation (WMSSDA). We evaluated and compared WMSSDA and compared our results to those obtained by other state-of-the-art approaches, on limited and imbalanced data, on both benchmark and real-world medical datasets. Our proposed approach is composed of a two branch architecture, learning both a shared domain invariant latent space and source domain specific latent spaces. The shared latent representation is learned and regularized both statistically and adversarially, the statistical regularization relies on a MD measure between source and target domains. The output of the MD regularization is used to compute transfer contribution weights that are applied to weight the impact of each source domain during training, limiting negative transfer. We show that our proposed WMSSDA outperforms most state-of-the-art approaches on both image benchmarks datasets and a real-world tabular medical dataset. We further analyze the relevance and importance of each component of our method by performing an ablation study, validating the overall architecture of our approach.

Overall, our experimental results seem to show that most multi-source domain adaptation approaches do not obtain significantly better results than single-source approaches in our experimental scenario. This seems to show that despite researchers efforts, negative transfer in multi-source domain adaptation is still an open critical problem that seems to limit the overall potential performance of state-of-the-art multi-source domain adaptation approaches. Best performing domain adaptation approaches are still not able to fully avoid negative transfer. In our proposal of a new multi-source domain adaptation approach, we tried to limit negative transfer through the computation of transfer contribution weights that are applied as a scaling of the impact of each source domain in the training of the entire model. Our experimental results and ablation study show that this element of our approach is relevant and improves overall results. But even with this component, our proposed approach is yet not able to fully avoid negative transfer. Future works in the domain adaptation field should focus on finding better ways to handle this important matter.

Declarations

Funding

This research is supported by the European Union's Horizon 2020 research and innovation program under grant agreement No 875171, project QUALITOP (Monitoring multidimensional aspects of QUAlity of Life after cancer

ImmunoTherapy - an Open smart digital Platform for personalized prevention and patient management).

Conflicts of interest/Competing interests

The authors declare that there is no conflict of interests regarding the publication of this paper.

Ethics approval

The authors adhere to the accepted ethical standards of a genuine research study.

Consent to participate

All authors give consent for their participation to the research work.

Consent for publication

The authors give to the publisher the permission to publish this work if it is accepted.

Availability of data and material

- Covid dataset: https://www.kaggle.com/datasets/meirnizri/covid19-dataset.
- Digits dataset:
  - MNIST: http://yann.lecun.com/exdb/mnist.
  - MNIST-M: https://www.kaggle.com/datasets/aquibiqbal/mnistm.
  - Street View House Numbers (SVHN): http://ufldl.stanford.edu/housenumbers.
  - Synthetic Digits (SYN): https://www.kaggle.com/datasets/prasunroy/synthetic-digits.
  - USPS: https://www.kaggle.com/datasets/bistaumanga/usps-dataset.
- DomainNet dataset: http://ai.bu.edu/M3SDA/#dataset.

Code availability

Temporary private link, to be updated after acceptance: https://drive.google.com/file/d/1XvgWqkHA4LuK6bPE9ktIlO9SFY1MemnC/view?usp=sharing.

References

[1] Wilson, G., Cook, D.J.: A Survey of Unsupervised Deep Domain Adaptation. ACM Transactions on Intelligent Systems and Technology **11**(5), 51–15146 (2020). https://doi.org/10.1145/3400066

[2] Day, O., Khoshgoftaar, T.M.: A survey on heterogeneous transfer learning. Journal of Big Data **4**(1), 29 (2017). https://doi.org/10.1186/s40537-017-0089-0

[3] Pan, S.J., Yang, Q.: A Survey on Transfer Learning. IEEE Transactions on Knowledge and Data Engineering **22**(10), 1345–1359 (2010). https://doi.org/10.1109/TKDE.2009.191

[4] Kouw, W.M., Loog, M.: An introduction to domain adaptation and transfer learning. arXiv (2019). http://arxiv.org/abs/1812.11806

[5] Long, M., Cao, Y., Wang, J., Jordan, M.I.: Learning transferable features with deep adaptation networks. In: Proceedings of the 32nd International Conference on International Conference on Machine Learning - Volume 37. ICML'15, pp. 97–105. JMLR.org, Lille, France (2015). https://doi.org/10.5555/3045118.3045130

[6] Ganin, Y., Ustinova, E., Ajakan, H., Germain, P., Larochelle, H., Laviolette, F., Marchand, M., Lempitsky, V.: Domain-Adversarial Training of Neural Networks. In: Domain Adaptation in Computer Vision Applications, pp. 189–209. Springer, Cham (2017). Series Title: Advances in Computer Vision and Pattern Recognition. https://jmlr.org/papers/volume17/15-239/15-239.pdf

[7] Zhu, Y., Zhuang, F., Wang, J., Ke, G., Chen, J., Bian, J., Xiong, H., He, Q.: Deep Subdomain Adaptation Network for Image Classification. IEEE Transactions on Neural Networks and Learning Systems **32**(4), 1713–1722 (2021). https://doi.org/10.1109/TNNLS.2020.2988928

[8] Zhao, H., Zhang, S., Wu, G., Moura, J.M.F., Costeira, J.P., Gordon, G.J.: Adversarial Multiple Source Domain Adaptation. In: Bengio, S., Wallach, H., Larochelle, H., Grauman, K., Cesa-Bianchi, N., Garnett, R. (eds.) Advances in Neural Information Processing Systems, vol. 31. Curran Associates, Inc., ??? (2018). https://proceedings.neurips.cc/paper/2018/file/717d8b3d60d9eea997b35b02b6a4e867-Paper.pdf

[9] Zhu, Y., Zhuang, F., Wang, D.: Aligning Domain-specific Distribution and Classifier for Cross-domain Classification from Multiple Sources. Proceedings of the AAAI Conference on Artificial Intelligence **33**(01), 5989–5996 (2019). https://doi.org/10.1609/aaai.v33i01.33015989

[10] Peng, X., Bai, Q., Xia, X., Huang, Z., Saenko, K., Wang, B.: Moment Matching for Multi-Source Domain Adaptation. In: 2019 IEEE/CVF International Conference on Computer Vision (ICCV), pp. 1406–1415. IEEE, Seoul, Korea (South) (2019). https://doi.org/10.1109/ICCV.2019.00149

[11] Zuo, Y., Yao, H., Xu, C.: Attention-Based Multi-Source Domain Adaptation. IEEE Transactions on Image Processing **30**, 3793–3803 (2021). https://doi.org/10.1109/TIP.2021.3065254

[12] Cortes, C., Mohri, M., Medina, A.M.: Adaptation Based on Generalized Discrepancy. Journal of Machine Learning Research **20**(1), 1–30 (2019). https://doi.org/10.5555/3322706.3322707

[13] Moreno-Torres, J.G., Raeder, T., Alaiz-Rodríguez, R., Chawla, N.V., Herrera, F.: A unifying view on dataset shift in classification. Pattern Recognition **45**(1), 521–530 (2012). https://doi.org/10.1016/j.patcog.2011.06.019

[14] Redko, I., Morvant, E., Habrard, A., Sebban, M., Bennani, Y.: Advances in Domain Adaptation Theory. Computer engineering. ISTE Press Ltd ; Elsevier Ltd, London, UK : Kidlington, Oxford, UK (2019). OCLC: ocn988168970. https://www.elsevier.com/books/advances-in-domain-adaptation-theory/redko/978-1-78548-236-6

[15] Zhu, Y., Zhuang, F., Wang, J., Chen, J., Shi, Z., Wu, W., He, Q.: Multi-representation adaptation network for cross-domain image classification. Neural Networks **119**, 214–221 (2019). https://doi.org/10.1016/j.neunet.2019.07.010

[16] Li, Z., Zhao, Z., Guo, Y., Shen, H., Ye, J.: Mutual Learning Network for Multi-Source Domain Adaptation. arXiv. arXiv:2003.12944 [cs] (2020). http://arxiv.org/abs/2003.12944 Accessed 2023-05-22

[17] Xu, Y., Kan, M., Shan, S., Chen, X.: Mutual Learning of Joint and Separate Domain Alignments for Multi-Source Domain Adaptation. In: 2022 IEEE/CVF Winter Conference on Applications of Computer Vision (WACV), pp. 1658–1667. IEEE, Waikoloa, HI, USA (2022). https://doi.org/10.1109/WACV51458.2022.00172. https://ieeexplore.ieee.org/document/9707089/ Accessed 2023-05-22

[18] Zhang, W., Deng, L., Zhang, L., Wu, D.: A Survey on Negative Transfer. IEEE/CAA Journal of Automatica Sinica, 1–25 (2022). https://doi.org/10.1109/JAS.2022.106004

[19] Haibo He, Garcia, E.A.: Learning from Imbalanced Data. IEEE Transactions on Knowledge and Data Engineering **21**(9), 1263–1284 (2009). https://doi.org/10.1109/TKDE.2008.239

[20] Wang, S., Liu, W., Wu, J., Cao, L., Meng, Q., Kennedy, P.J.: Training deep neural networks on imbalanced data sets. In: 2016 International Joint Conference on Neural Networks (IJCNN), pp. 4368–4374. IEEE, Vancouver, BC, Canada (2016). https://doi.org/10.1109/IJCNN.2016.

7727770

[21] Fernández, A., García, S., Galar, M., Prati, R.C., Krawczyk, B., Herrera, F.: Learning from Imbalanced Data Sets. Springer, Cham (2018). https://doi.org/10.1007/978-3-319-98074-4. http://link.springer.com/10.1007/978-3-319-98074-4 Accessed 2023-05-03

[22] Krawczyk, B.: Learning from imbalanced data: open challenges and future directions. Progress in Artificial Intelligence **5**(4), 221–232 (2016). https://doi.org/10.1007/s13748-016-0094-0. Accessed 2023-05-02

[23] Hart, P.: The condensed nearest neighbor rule (Corresp.). IEEE Transactions on Information Theory **14**(3), 515–516 (1968). https://doi.org/10.1109/TIT.1968.1054155. Accessed 2023-05-03

[24] Tomek, I.: Two Modifications of CNN. IEEE Transactions on Systems, Man, and Cybernetics **SMC-6**(11), 769–772 (1976). https://doi.org/10.1109/TSMC.1976.4309452. Accessed 2023-05-03

[25] Chawla, N.V., Bowyer, K.W., Hall, L.O., Kegelmeyer, W.P.: SMOTE: Synthetic Minority Over-sampling Technique. Journal of Artificial Intelligence Research **16**, 321–357 (2002). https://doi.org/10.1613/jair.953

[26] Kukar, M., Kononenko, I.: Cost-Sensitive Learning with Neural Networks. (1998). https://www.semanticscholar.org/paper/Cost-Sensitive-Learning-with-Neural-Networks-Kukar-Kononenko/bdef7eb9b62e2a12b870957879f7a097b41f6012