Unsupervised Multi-Source Domain Adaptation for Regression

Guillaume Richard, Antoine de Mathelin, Georges Hébrail, Mathilde Mougeot, Nicolas Vayatis

Unsupervised Multi-Source Domain Adaptation for Regression

Guillaume Richard[1,2], Antoine de Mathelin[1], Georges Hébrail[2], Mathilde Mougeot[1,3], and Nicolas Vayatis[1]

[1]Centre Borelli, ENS Paris-Saclay, Saclay, FRANCE
[2]EDF R&D, Palaiseau, FRANCE
[3]ENSIIE, Evry, FRANCE

January 2020

Abstract

We consider the problem of unsupervised domain adaptation from multiple sources in a regression setting. We propose in this work an original method to take benefit of different sources using a weighted combination of the sources. For this purpose, we define a new measure of similarity between probabilities for domain adaptation which we call hypothesis-discrepancy. We then prove a new bound for unsupervised domain adaptation combining multiple sources. We derive from this bound a novel adversarial domain adaptation algorithm adjusting weights given to each source, ensuring that sources related to the target receive higher weights. We finally evaluate our method on different public datasets and compare it to other domain adaptation baselines to demonstrate the improvement for regression tasks.

1 Introduction

In classical machine learning, one assumes that the source data used to train an algorithm comes from the same distribution as the target data it is applied to. This assumption is not true for many applications: for instance, a human activity recognition model trained on young people may not perform well when applied to older ones. Moreover, for many applications, different sources have different relations to the target domain. Including sources that are not related to the target may lead to negative transfer *ie* reduce the performance of adaptation on the target domain. Hence we consider in this paper the problem of unsupervised domain adaptation from multiple sources tackling the issue of adapting from a labelled source domain to a target domain with no labeled data. An abundant literature exists for unsupervised domain adaptation with a single source for classification: [3] introduced a single source adaptation bound for classification. It was later used in several works, notably adversarial methods of [10] and [19]. While those methods can be applied to regression, it is not theoretically founded and often fails in practice. [14] proposed a novel theoretical

bound for regression using the notion of discrepancy between predictors. It is not easy to estimate in the general case but has led to several works using linear regression [1] to train GANs or kernels [8] for domain adaptation.
The main risk of adapting from multiple sources comes from one or several sources being detrimental to adaptation. It is particularly true with adversarial methods trying to match source and target domains. Then one wants to find a way to give high weights to sources the most related to the target. In [15] a weighting scheme is proposed assuming that the target distribution is a convex combination of the sources. A boosting method is used in [21] to derive weights. Recently, [22] extended previous bounds with a maximum over multiple sources leading to an algorithm giving high weights to sources far from the target. In [13] inter-relationships between sources are used to compute the weights.
There are two main contributions in this work: firstly, we prove new a bound for multi-source domain adaptation that is tighter than existing bounds in a regression task. It is based on a new measure of similarity between distributions which we call hypothesis-discrepancy. For a given predictor, it measures how another predictor can give different results on one of the domains while staying close on the other and can be computed with adversarial learning. The second main contribution is a new algorithm optimizing both representations and weights of each source for multi-source domain adaptation. To the best of the authors' knowledge, this is the first adversarial unsupervised domain adaptation tailored for regression. We conduct experiments on both synthetic and real-world datasets and improve on state of the art results for multi-source adversarial domain adaptation for regression.

2 Unsupervised Multiple Source Domain Adaptation with Hypothesis-Discrepancy

Setting. We first define the problem of Multi-Source Domain Adaptation (MSDA). We define K independent source domains $\mathcal{D}_k$ such that $\mathcal{D}_k = \{X_k, f_k\}$ where X_k is the input data with associated marginal distribution $X_k \sim p_k$ and f_k the true labelling function of the domain. Similarly, we define a target domain $\mathcal{D}_t = \{X_t, f_t\}$ with $X_t \sim p_t$. We assume that every input is in the same space $\mathcal{X}$ *ie* $X_k \in \mathcal{X}$ and $X_t \in \mathcal{X}$ which is the case of homogeneous transfer. The prediction task is the same for both domains *ie* $f_k : \mathcal{X} \to \mathcal{Y}$ and $f_t : \mathcal{X} \to \mathcal{Y}$ (f_k and f_t are supposed to be close to each other). For instance, $\mathcal{Y} \subset \mathbb{R}$ for regression or $\mathcal{Y} = \{0, 1\}$ for binary classification. We also consider a loss $L : \mathcal{Y} \times \mathcal{Y} \to \mathbb{R}^+$ and a hypothesis class $\mathcal{H}$ of hypotheses $h : \mathcal{X} \to \mathcal{Y}$. We also assume that the loss L is bounded over $\mathcal{S}_k$ by $M = \sup_k sup_{x \in \mathcal{S}_k, h \in \mathcal{H}} L(h(x), f_k(x))$.
For two hypotheses h and h', we define $\epsilon_k(h, h') = \mathbb{E}_{x \sim p_k}[L(h(x), h'(x))]$ the average loss of two hypotheses over a the source domain $\mathcal{D}_k$ and $\epsilon_t(h, h') = \mathbb{E}_{x \sim p_t}[L(h(x), h'(x))]$ over the target domain. We also consider a labelled source sample $\mathcal{S}_k$ of size m with an associated empirical probability $\hat{p}_k$. Similarly, we consider a unlabelled target sample $\mathcal{S}_t$ of size n with an associated empirical probability of $\hat{p}_t$.

Objective. The goal of Domain Adaptation is to minimize the target risk $\epsilon_t(h, f_t) = \mathbb{E}_{x \sim p_t}[L(h(x), f_t(x))]$. In *unsupervised domain adaptation*, no label

is available in the target task and we cannot directly estimate f_t. Consequently we want to leverage the information about the labels in the source domains f_k to adapt to the target domain. After defining the hypothesis-discrepancy we propose a new bound relating the target risk with a weighted combination of the source risks $\epsilon_k(h, f_k)$.

2.1 Hypothesis-discrepancy

We introduce the concept of hypothesis-discrepancy:

Definition 1. For two distributions P, Q over a set $\mathcal{X}$ and for a hypothesis class $\mathcal{H}$ over $\mathcal{X}$, for any $h \in \mathcal{H}$, the hypothesis-discrepancy (or *HDisc*) associated with h is defined as:

$$HDisc_{\mathcal{H},L}(P, Q; h) = \max_{h' \in \mathcal{H}} |\mathbb{E}_{x \sim P}[L(h(x), h'(x)] - \mathbb{E}_{x \sim Q}[L(h(x), h'(x)]| \quad (1)$$

For any given $h \in \mathcal{H}$ hypothesis-discrepancy measures a similarity between two distributions. It is directly dependent on the hypothesis class $\mathcal{H}$ and the loss L and can be estimated with finite samples. In the definition, h' can be seen as a predictor that would be very close to h on the source domain but far on the target domain (or vice-versa). Using *HDisc*, we are able to show the following proposition for unsupervised single source domain adaptation:

Proposition 1. If L is symmetric and follows the triangle inequality, then the following bound holds for any $k \in \{1, ..., K\}$,

$$\epsilon_t(h, f_t) \leq \epsilon_k(h, f_k) + \eta_{\mathcal{H}}(f_k, f_t) + HDisc_{\mathcal{H},L}(p_t, p_k; h) \quad (2)$$

where

$$\eta_{\mathcal{H}}(f_k, f_t) = \min_{h_0 \in \mathcal{H}} [\epsilon_t(h_0, f_t) + \epsilon_k(h_0, f_k)]$$

Proof. See Appendix A □

This bound gives a good intuition about the conditions under which domain adaptation can work. Indeed, the first term $\epsilon_k(h, f_k)$ corresponds to the error made by h on the source data. The third term is our hypothesis-discrepancy and characterizes the similarity between marginal probabilities over the input data. The second term $\eta_{\mathcal{H}}$ is the sum of the error made by the ideal hypothesis on both domains: it is small when the two labelling functions are close which is the general assumption of unsupervised domain adaptation [4]. As it involves f_t it cannot be controlled in unsupervised domain adaptation without access to labels in the target domain. It follows that, under the assumption that the two labelling functions are close, if the two other terms of the bound can be minimized, the target risk will also be minimized. Another strength of *HDisc* is that it is directly dependent on $\mathcal{H}$ and L: it can be used for any task including regression.

2.2 Multi-Source Domain Adaptation bound

When multiple sources are available, a straightforward idea would be to merge all the source domains into one and transform the problem to single-source domain adaptation where Proposition 1 applies. This solution is clearly not

optimal as different source domains may have different relationships to the target one.

We propose to attribute weights to each source: we introduce the α-weighted source domain $\mathcal{D}_\alpha = \{p_\alpha, f_\alpha\}$ such that for $\alpha \in \Delta = \{\alpha \in \mathbb{R}^K; \alpha_k \geq 0, \sum_{k=1}^K \alpha_k = 1\}$, $f_\alpha : x \to (\sum_{k=1}^K \alpha_k p_k(x) f_k(x))/(\sum_{j=1}^K \alpha_j p_j(x))$ and $p_\alpha = \sum_{k=1}^K \alpha_k p_k$.

The α-weighted sample is $\mathcal{S}_\alpha = \bigcup_{k=1}^K \mathcal{S}_k$ with probabilities $\hat{p_\alpha}(x_i^{(k)}) = \alpha_k/m$. Similarly, we consider an unlabeled target sample $\mathcal{S}_t = \{(x_1^{(t)}, ..., x_n^{(t)})\}$ where $x_i^{(t)} \overset{i.i.d}{\sim} p_t$. We define the sets $\mathcal{H}_k = \{g : x \to L(h(x), f_k(x)); h \in \mathcal{H}\}$. Moreover, the Rademacher complexity of a set $\mathcal{H}_k$ is defined as

$$\mathcal{R}_m(\mathcal{H}_k) = \mathbb{E}_{\mathcal{S}_k}[\mathbb{E}_\sigma[sup_{g \in \mathcal{H}_k} \sum_{i=1}^m \sigma_i g(x_i^{(k)})]]$$

where the expectations is taken over any sample $\mathcal{S}_k = \{x_k^{(1)}, ..., x_k^{(m)}\} \sim \hat{p_k}^{(m)}$ and σ_i are *iid* variables uniformly distributed over $\{-1, 1\}$ independent from $X_1, ..., X_K$.

Theorem 1. Assuming that the loss L is symmetric and follows the triangle inequality, then for any hypothesis $h \in \mathcal{H}$, with probability $1 - \delta$ the following bound holds:

$$\begin{aligned} \epsilon_t(h, f_t) \leq & \sum_{k=1}^K \alpha_k \hat{\epsilon_k}(h, f_k) + HDisc_{\mathcal{H},L}(p_t, p_\alpha) + \eta_{\mathcal{H},\alpha} \\ & + 2 \sum_{k=1}^K \alpha_k \mathcal{R}_m(\mathcal{H}_k) + \|\alpha\|_2 M \sqrt{\frac{ln(1/\delta)}{2m}} \end{aligned} \tag{3}$$

where

- $\eta_{\mathcal{H},\alpha} = \min_{h_0 \in \mathcal{H}} [\epsilon_\alpha(h_0, f_\alpha) + \epsilon_t(h_0, f_t)]$
- $\mathcal{R}_m(\mathcal{H}_k)$ is the Rademacher complexity of $\mathcal{H}_k = \{h : x \to L(h(x), f_k(x)); h \in \mathcal{H}\}$

Proof. We give a sketch of the proof. The full details can be found in Appendix B. Using Proposition 1 with p_α, we get:

$$\epsilon_t(h, f_t) \leq \epsilon_\alpha(h, f_\alpha) + \eta_{\mathcal{H}}(f_t, f_\alpha) + HDisc_{\mathcal{H},L}(p_\alpha, p_k; h) \tag{4}$$

We then define $\phi = \epsilon_\alpha(h, f_\alpha) - \hat{\epsilon_\alpha}(h, f_\alpha)$. Using McDiarmid's inequality [17] for ϕ, we obtain that with probability $1 - \delta$,

$$\epsilon_\alpha(h, f_\alpha) \leq \hat{\epsilon_\alpha}(h, f_\alpha) + \mathbb{E}_{\hat{p_\alpha}}[\phi] + \|\alpha\|_2 M \sqrt{\frac{\log 1/\delta}{2m}}$$

Then one can show using the usual ghost sample argument of Rademacher complexity that:

$$\mathbb{E}_{\hat{p_\alpha}}[\phi] \leq 2 \sum_{k=1}^{K} \alpha_k \mathcal{R}_m(\mathcal{H}_k)$$

Noting that $\hat{\epsilon_\alpha}(h, f_\alpha) = \sum_{k=1}^{K} \alpha_k \hat{\epsilon_k}(h, f_k)$ concludes the proof.

□

Theorem 1 gives a theoretical analysis in the multi-source domain adaptation framework. The first term corresponds to the α-weighted source risks and can be controlled by learning h close to f_k. The second term connects the target distribution with the α-weighted source distribution. The third term is related to how different the labelling functions on the target and the source are and is expected to be small in unsupervised domain adaptation. The last two terms show the convergence rate of this bound and it was proven in [6] that $\mathcal{R}_m(\mathcal{H}_k) = \mathcal{O}(1/\sqrt{m})$ for some functions such as neural networks.
Then in order to adapt from sources $\mathcal{S}_1, ..., \mathcal{S}_k$ to the target, we need to minimize the hypothesis-discrepancy between the α-weighted domain and the target domain. We propose in Section 3 an algorithm to find ideal representations of the sources and weights to select the best sources for adaptation.

3 Adversarial algorithm for Multi-Source Domain Adaptation

3.1 Optimization objectiv: a min-max problem

We now present the practical solution derived from Theorem 1. We introduce a feature extractor parametrized by θ $\phi_\theta : \mathcal{X} \to \mathcal{Z}$ and a class of predictor $\mathcal{H}_\mathcal{Z} : \mathcal{Z} \to \mathcal{Y}$. Given unlabeled target sample $X_t = \{(x_1^{(t)}, ..., x_n^{(t)}\} \in \mathbb{R}^{n \times d}$ and K labeled sources $X_k = \{(x_1^{(k)}, ..., x_m^{(k)}\} \in \mathbb{R}^{m \times d}$ with labels $Y_k = \{y_1^{(k)}, ..., y_m^{(k)}\} \in \mathbb{R}^m$, we want to minimize the combination of the source risk and the hypothesis-discrepancy between the marginal weighted source distribution and target distribution as in Theorem 1.
Using the definition of *HDisc*, we formulate the following objective for our Adversarial Hypothesis-Discrepancy Multi-Source Domain Adaptation (**AHD-MSDA**):

$$\min_{\substack{\phi_\theta, h \in \mathcal{H} \\ \|\alpha\|_1 = 1}} \max_{h' \in \mathcal{H}} \left[\sum_{k=1}^{K} \alpha_k \epsilon_k(h \circ \phi_\theta, y_k) + \lambda \|\alpha\|_2 \right. \\ \left. + |\epsilon_t(h \circ \phi_\theta, h' \circ \phi_\theta) - \sum_{k=1}^{K} \alpha_k \epsilon_k(h \circ \phi_\theta, h' \circ \phi_\theta)| \right] \tag{5}$$

where λ is a hyperparameter. It was shown in [1] that if $\mathcal{H}_\mathcal{Z}$ is a subsect of μ_h-Lipschitz functions and ϕ_θ is continuous in θ then the discrepancy is continuous in θ and it also stands for hypothesis-discrepancy.
The first term of the objective forces h to be a good predictor on the source task. For any given h and h' the discrepancy term constrains both representations ϕ_θ

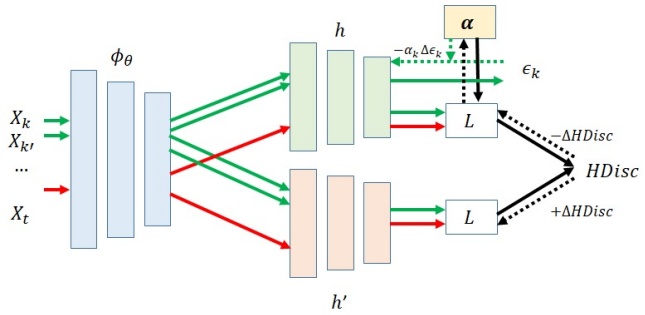


Figure 1: MSDA: The adversarial scheme is similar to single-source with weights α. At each iteration, the weights α are udpated.

and weights α to align domains. The term $\eta_{\mathcal{H}}$ is ignored in our objective and assumed to be small.
This objective involves a min-max formulation that is similar to the ones used in adversarial domain adaptation [10]. While computing the true solution of this min-max problem is still impossible in practice, we derive an alternate optimization algorithm in the next section.

3.2 Adversarial Domain Adaptation

We display the general structure of our algorithm in Figure 1. Similarly to most other adversarial methods, we sequentially optimize differents parameters of our networks according to different objectives. At a given iteration, four losses are minimized sequentially:

1. $\mathcal{L}_h = \alpha_k \epsilon_k$ updates h to minimize the source loss
2. $\mathcal{L}_{h'} = -HDisc$ updates h' to maximize discrepancy
3. $\mathcal{L}_\theta = HDisc + \sum_{k=1}^{K} \alpha_k \epsilon_k$ updates ϕ_θ to minimize discrepancy and source loss
4. $\mathcal{L}_\alpha = HDisc + \lambda ||\alpha||_2$ updates α to minimize the discrepancy between α-weighted domain and target domain

The predictor h' can be seen as a discriminator in traditional adversarial domain adaptation methods. It is trained to give predictions close to h on one domain and far on another. The representations ϕ_θ are gradually updated against the discriminator. We include a source loss in its update as otherwise extracted features would be meaningless for the final task. The loss $\mathcal{L}_h$ ensures that h is performing well on the source domains.
In the loss $\mathcal{L}_\alpha$, we only included the discrepancy term. Indeed, our goal is to select the domains closer to the source in terms of discrepancy. Including the source loss in $\mathcal{L}_\alpha$ may give too high weights to sources that are "easy" to predict. It is possible to keep the term with a μ parameter to control its influence but in our experiments, it did not bring any improvement. It would also be possible

Algorithm 1 Pseudo-algorithm for **AHD-MSDA**

Initialize $\alpha_k = \frac{1}{K}$, h, h' and θ randomly, choose learning rates η_h, η_θ and η_α
for $e = 1...epochs$ **do**
Forward propagation
$\epsilon_k = \frac{1}{m}\sum_{i=1}^{m} L(h^{(e)}(\phi_{\theta^{(e)}}(x_i^{(k)}), y_i^{(k)}))$
$HDisc = \left|\epsilon_t(h^{(e)} \circ \phi_{\theta^{(e)}}, h'^{(e)} \circ \phi_{\theta^{(e)}}) - \sum_{k=1}^{K} \alpha_k \epsilon_k(h^{(e)} \circ \phi_{\theta^{(e)}}, h'^{(e)} \circ \phi_{\theta^{(e)}})\right|$
Backward propagation
$h^{(e+1)} \leftarrow h^{(e)} - \eta_h \left(\sum_{k=1}^{K} \alpha_k^{(e)} \Delta_h \epsilon_k(h^{(e)})\right)$ ▷ $(*)$
$h'^{(e+1)} \leftarrow h'^{(e)} + \eta_h \left(\sum_{k=1}^{K} \alpha_k^{(e)} \Delta_{h'} HDisc(h'^{(e)})\right)$
$\theta^{(e+1)} \leftarrow \theta^{(e)} - \eta_\theta \left(\sum_{k=1}^{K} \alpha_k^{(e)} \Delta_\theta \epsilon_k(\theta^{(e)}) + \Delta_\theta HDisc(\theta^{(e)})\right)$
$\alpha_k^{(e+1)} \leftarrow \alpha_k^{(e)} - \eta_\alpha \left(\Delta_{\alpha_k} HDisc(\alpha_k^{(e)}) + 2\lambda\alpha_k^{(e)}\right)$
Clip weights of $\phi_\theta^{(e+1)}$, $h^{(e+1)}$ and $h'^{(e+1)}$
$\alpha^{(e+1)} = \alpha^{(e+1)} / \|\alpha^{(e+1)}\|_1$
end for
$(*)$ For a parameter p and a loss L, we note $\Delta_p L(p_0)$ the gradient of L with respect to p computed at p_0

to completely update α at each epoch but we found it sub-performing. Our method allows the weights to smoothly adapt to the representations learnt by ϕ_θ.
We present a pseudo-algorithm in Algorithm 1. The order of the steps did not matter in our experiments. It is possible to include a short pre-training phase where hypothesis-discrepancy is not minimized as in the beginning of the training, representations and h may be meaningless and weights may be updated for unrelated sources. Our algorithm can also be applied in the single-source scenario by setting $K = 1$ and $\alpha = 1$.

4 Related works

Relation with other measures. The hypothesis-discrepancy has several advantages. Firstly, it can be estimated with finite samples (see Appendix C). Moreover, it is dependent on the hypothesis class $\mathcal{H}$ and loss L so it is possible to use in both classification and regression settings. The hypothesis-discrepancy is based on the discrepancy introduced in [14]: the original discrepancy is more conservative than our hypothesis-discrepancy as it is defined by $Disc(P, Q) = \sup_{h \in \mathcal{H}} HDisc(P, Q; h)$. Our bound is tighter than the one of [14] or [8] as it involves only one supremum over $\mathcal{H}$. The idea of discrepancy with only one supremum is also used in [12] with the source-discrepancy which is a specific case of our hypothesis-discrepancy with $h = h_s^*$. While the bound is tighter than the original [14] it does not lead to efficient practical solutions.
The popular $d_{\mathcal{H}}$ introduced in [3] for classification also involves only one supremum but often fails in practice for regression problems. Indeed, minimizing it aligns domains in a sense of classification and one can see on Figure 2 how in a simple linear regression problem $d_{\mathcal{H}}$ would fail. Our algorithm presented in the

next section is general to classification and regression.

Discrepancy minimization. While our method minimizes the new hypothesis-discrepancy, several methods worked on discrepancy minimization: in the original work [14], authors derive a quadratic formulation for ℓ^2-loss in regression where the goal is to re-weight each sample in the source domain and was later extended with kernels in [7]. Recently, discrepancy with linear regressors was used as a measure of distance between probabilities to train GANs [1]. Most works focusing on discrepancy have used specific values of the loss L and hypothesis class $\mathcal{H}$ to compute it: for instance, the ℓ^2-norm with linear regression was presented in the original work of [14], later extended to kernels [8] or even used to train GANs [1]. For classification, previous works used $d_{\mathcal{H}}$ as it is a special case of discrepancy for binary classification.

Adversarial Domain Adaptation. The first work using adversarial learning for Domain Adaptation was introduced in [10] with the gradient reversal layer. Using the bound of Ben-David [2], authors find a new representation that is discriminative for the task but where domains are confounded. [19] follows a similar idea. The main benefit of our method is that the adversarial structure is directly dependent on the task at hand. Maximum Classifier Discrepancy [18] is the closest idea to our practical solution even though they use a specific instance of discrepancy. Our *HDisc*-based algorithm is more general as it works for both regression and classification. To the best of authors' knowledge, this is the first work proposing an adversarial domain adaptation tailored for regression.

Multi-Source Domain Adaptation. Other authors also proposed methods to compute weights for different sources. For instance, in [15], authors assume that the target distribution is a convex combination of the source distributions and derive optimal weights. [16] and [11], authors use the Rényi divergence to derive ideal weights for each of the sources.
Other authors tried to attribute weights to different sources with an adversarial objective. Recently, [22] and [13] extended the adversarial framework of [10] to multi-source domain adaptation. Namely, they both prove extended previous bounds from single source DA based on $d_{\mathcal{H}}$ and Wasserstein distance. [22] gives a bound for regression based on $d_{\mathcal{H}}$, but is based on the generalized VC-dimension for regression which is hard to use in practice. Moreover the adopted weighted scheme in their algorithm sequentially adapts the worst source to the target

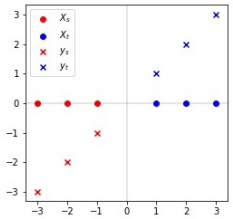


Figure 2: Basic linear regression problem: source domain is in red and target in blue. Here using linear regressors and ℓ^2-loss, for any h, $HDisc(X_s, X_t; h) = 0$ but using linear classifiers $d_{\mathcal{H}}(X_s, X_t) = 1$ as domains are perfectly separable

which we assume to be suboptimal. [13] uses relationships between sources to get weights to adapt to the target using Wasserstein distance, which is not tailored for regression applications. A very recent pre-print [20] proposes a weighting scheme for multi-source DA using the original discrepancy of [14] in the linear regression case, similar to [1].

5 Experiments

In this section, we evaluate our method both in the multi-source (**AHD-MSDA**) and single-source scenario (**AHDA**) on several datasets. It should be noted that unsupervised Domain Adaptation is hard to evaluate as having no labeled samples limits usage of classical comparison tools (cross-validation, hyperparameter tuning, ...). Hence we firstly use a synthetic dataset to test how our algorithm behaves. Secondly, we improve on previous state of the art results on an extended version of the Multi-Domain Amazon Review dataset orginally used in [5]. We also applied our algorithm to a digit classification task for which we get comparable results with methods tailored for classification.

5.1 Synthetic Dataset (Friedman)

To generate synthetic data, we use a modified version of the Friedman regression problem [9]. The Friedman dataset uses inputs x of dimension 5 and the prediction function is

$$y(x) = 10sin(\pi x_1 x_2) + 20(x_3 - 0.5)^2 + 10x_4 + 5x_5 + \epsilon$$

where $\epsilon \sim \mathcal{N}(0, \sigma_y)$.
To highlight the two contributions of this work, our goal is two-fold: we firstly want to demonstrate the effectiveness of *HDisc* in the single-source DA scenario (K=1 and $\alpha = 1$). Then we run different experiments to show when multi-source DA is expected to bring an improvement.
We generate source data as follows: we define three clusters of sources $\{\mathcal{S}_1, \mathcal{S}_2, \mathcal{S}_3\}$, $\{\mathcal{S}_4, \mathcal{S}_5, \mathcal{S}_6\}$ and $\{\mathcal{S}_7, \mathcal{S}_8, \mathcal{S}_9\}$. For each cluster, and for each feature (1...5), a mean of -1 or 1 is selected at random. Then each mean of each source of a given cluster is shifted by a random shift *ie* for a source k associated to a cluster c $\mu_k^{(f)} \sim \mathcal{N}(\mu_c, \sigma_c)$. Finally, each source sample is randomly chosen with a normal distribution $p_k = \mathcal{N}(\mu_k^{(f)}, \sigma_k)$. In Figure 3, we display each feature of the generated data.
In order to separate the effects of our two contributions, we split the experiments in two parts: in the first experiment, demonstrate the effectiveness of our hypothesis-discrepancy minimization in the single source scenario (*ie* when $\alpha_k = \frac{1}{K}$) and why the classical $d_{\mathcal{H}}$ fails in this regression scenario. In a second scenario, we show how our weighting scheme helps adaptation in multi-source, especially to select sources that are related to the target.

Hypothesis-Discrepancy. For single-source, we merge all 9 source domains together to create one. The target sample is generated choosing uniformly one of the source domains for each element and adding a noise of the form $\mathcal{N}(\mu_{shift}$, $\sigma_{shift})$. This experiment helps to understand the purpose of unsupervised domain adaptation. Indeed, as the underlying condition for unsupervised DA to

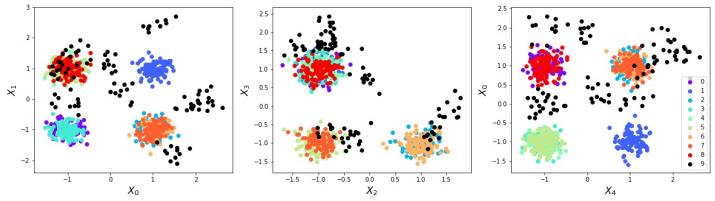
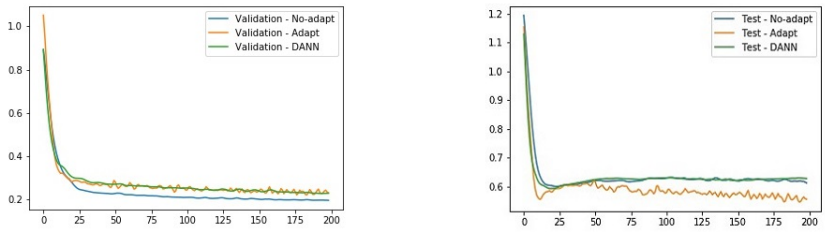


Figure 3: Data for the single-source Friedman experiment over the 5 features with $\sigma_k = 0.2$, $\sigma_c = 0.2$, $\mu_{shift} = 0.5$, $\sigma_{shift} = 0.5$. From right to left: (x_0, x_1) ; (x_2, x_3); $(x_4; x_0)$. Each color corresponds to a source, the target is in black.



(a) Validation loss on source data

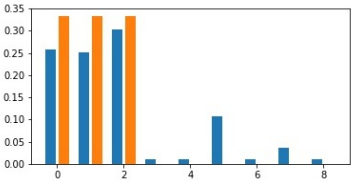
(b) Target loss

Figure 4: Training curves for Single Source DA: without adaptation, the target loss increases as the validation loss keeps decreasing. **DANN** exposes the same behaviour as the target loss of **AHDA** decreases.

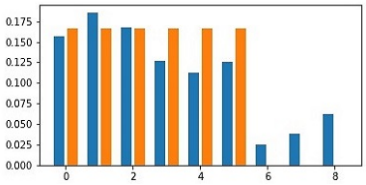
work is that the labelling function is similar in every domain ($\eta_{\mathcal{H}}$ small in our experiments), unsupervised DA is closely related to the issue of generalization. As a consequence, if the algorithm learnt on the source data is able to generalize, domain adaptation will not bring any improvement. As such, one can see unsupervised DA as a data-driven regularization to improve the target risk.

For this experiment, we use a shallow network with 2 layers with 5 neurons and *LeakyRelu* activation for the feature extractor and 1 layer for the final predictor as detailed in Appendix D. We keep this architecture for three methods: Multi-Layer Perceptron (**MLP**) without adaptation, Domain-Adversarial Neural Network (**DANN**) minimizing the $d_{\mathcal{H}}$ between domains and our Adversarial Hypothesis-Discrepancy Adaptation (**AHDA**) our method in the case of single source (no weights α). To get the results for **DANN**, we tuned then hyperparameter μ balancing the regression and domain losses: without this tuning, **DANN** always fails to converge because its adversarial scheme is related to classification.

We conducted the experiments with various amount of shift. In Figure 4, we report the validation loss (computed validation set different from the training set) and target loss for each method for $\sigma_x = 0.2$, $\mu_{shift} = 0.5$ and $\sigma_{shift} = 0.5$, which is the case of target data related but not too close to the source domain plottend on Figure 6. While **MLP** and **DANN** overfit on the source data, our method is able to decrease the target loss. Hence, our adversarial scheme helps the algorithm to better learn for the target data. We report in Table 1 the average MSE scores over ten runs for the three methods and various shifts. One can notice that the further μ_{shift} is, the more useful the adaptation is. We also



(a) One cluster with uniform weights



(b) Two clusters with uniform weights

Figure 5: Friedman Multiple Source experiment: α found by **AHD-MSDA** (blue) vs True α (orange)

report in Appendix D a visualization of the extracted features from **AHDA** and **DANN** which shows that **DANN** tries to align domains only to be able not to separate them while **AHDA** is constrained by the final regression task.

μ_{shift}	0.0	0.1	0.2	0.3	0.4	0.5	0.6	0.7	0.8	0.9	1.0
MLP	**0.221**	**0.214**	0.260	0.360	0.510	0.695	0.909	1.132	1.322	1.484	1.632
DANN	0.223	0.235	0.296	0.412	0.581	0.784	1.017	1.258	1.462	1.629	1.772
AHDA	0.222	**0.214**	**0.255**	**0.350**	**0.490**	**0.670**	**0.881**	**1.044**	**1.197**	**1.392**	**1.446**

Table 1: Single source domain adaptation: MSE for different amounts of shift.

Multi-Source DA. We also tried our multi-source algorithm on the previous target domain. As we expected, multi-source domain adaptation did not bring any improvement in the previous scenario as the target was sharing the same relations with every source. But MSDA is particularly interesting in the case where some of the sources are not useful for adaptation. We demonstrate it in an experiment where we control the weights α given to each source in the creation of the target domain.
In a first experiment, we give equal weights $\alpha_{1,2,3} = 1/3$ to every source in one cluster and $\alpha_{4:9} = 0$ for every other cluster. In a second experiment, we gave equal weights to two clusters $\alpha_{1:6} = 1/6$ and $\alpha_{7:9} = 0$ In both experiments, and on different runs, the algorithm was able to retrieve weights close to the real ones. It was translated by a decrease of the target loss. When putting weights to only one source in a cluster, our algorithm struggled to identify a specific source but still gave high weight to sources in the same cluster. In some experiments where sources were very different from each other (no cluster), we noticed a tendency for α to give 1 for one source and 0 for all the others. It can be meaningful as the target may be close to only one source in that case or be balanced using the λ parameter of the ℓ^2 regularization.

Discussion. Based on this toy experiment, we can conclude that our **AHD-MSDA** and **AHDA** perform well for regression under the condition that the labelling functions are the same. The proposed weighting scheme is performing well when sources have really different relations to the output. The main limitation we found to *HDisc* is that since it is dependent on h, its estimation is hard, especially in regression where values are not constrained. As we have seen the weighting scheme mainly helps when the target data is close to only few of

Dataset	apparel	auto	baby	beauty	books	camera	cellphones	computer	dvd
MLP	0.897	0.902	0.841	0.880	1.024	0.923	0.871	0.842	0.912
DANN	0.929	1.017	0.893	0.878	1.141	0.944	1.051	1.135	1.052
AHDA	**0.837**	0.887	0.840	0.860	**0.945**	0.893	0.859	0.849	0.882
MDAN	0.980	0.797	0.908	0.973	1.234	0.921	0.954	1.749	1.322
AHD-MSDA	0.859	**0.767**	**0.794**	**0.790**	0.969	**0.876**	**0.825**	**0.770**	**0.868**
Dataset	electronics	food	grocery	health	jewelry	kitchen	magazines	music	musical
MLP	0.833	0.882	0.774	0.869	0.759	0.851	0.960	0.976	0.778
DANN	1.064	1.036	0.793	0.955	0.783	0.856	0.985	1.293	0.727
AHDA	0.844	0.866	0.796	0.851	0.815	0.849	**0.909**	**0.955**	0.895
MDAN	1.041	**0.820**	0.789	0.987	0.755	0.850	1.295	1.330	1.117
AHD-MSDA	**0.815**	0.849	**0.716**	**0.840**	**0.715**	**0.848**	0.911	0.991	**0.723**
Dataset	office	outdoor	software	sports	tools	toys	video	Average	Avg rank
MLP	0.969	0.831	0.924	0.844	0.823	0.873	0.866	0.892	3.12
DANN	**0.854**	0.803	1.090	0.850	0.888	0.861	1.00	0.955	3.92
AHDA	0.956	0.851	0.880	0.839	0.813	0.864	**0.878**	0.868	2.52
MDAN	1.041	0.789	0.964	0.910	1.656	**0.843**	1.467	1.060	3.96
AHD-MSDA	0.882	**0.755**	**0.858**	**0.814**	**0.741**	0.873	0.883	**0.838**	**1.48**

Table 2: Average MAE over 5 runs for each method and domain of the Amazon Multi-Domain Dataset

the sources. In the next experiment, we show how our algorithm performs on a real world dataset.

5.2 Multi-Domain Amazon Dataset

We use the extended version of the Multi-Domain Sentiment Dataset[1] with 25 categories (books, dvd, ...). The dataset is made of textual reviews from Amazon and associated ratings. We treat it as a regression problem where the goal is to predict the rating based on the review. As in previous papers [22] [10], we transform the data using tf-idf transform and filtering only the $1,000$ words with highest coefficients.
Similarly to the reduced dataset for classification [5], we kept $2,000$ samples for each category (or less when there was not $2,000$ available). We alternate on each category as a target and use every other category as a source.
In order to study the performance of our method **AHD-MSDA**, we compare it to several other baselines on different datasets:

- **MLP** corresponds to merging all sources and applying to the target dataset without adaptation.
- **DANN** corresponds to merging all sources into one and applying the **DANN** from [10].
- **AHDA** corresponds to merging all sources into one and applying our adversarial scheme using discrepancy specific to the task.
- **MDAN** is a multi-source domain adaptation based on the $d_{\mathcal{H}}$ distance proposed in [22]. It gives high weights to sources far from the target (we used the soft version).
- **AHD-MSDA** is our method described in Algorithm 1.

We also experimented the Multi-Domain Matching Network (**MDMN**) that uses Wasserstein distance between sources and the target to give weights but it did not perform well at all for regression as the adversarial structure is not well conditioned for regression. For **DANN** and **MDAN** the predictors have been specified to regression by changing the last layer by a regression layer and loss with mean squared error. The implementation of **MDAN** is inspired from the original implementation from the authors. [2]. We tried some methods to

[1] `https://www.cs.jhu.edu/~mdredze/datasets/sentiment/`
[2] `https://github.com/KeiraZhao/MDAN`

pre-compute weights for our multi-source algorithm based on different distances between domains but at best it led to similar results to **AHDA** so we do not report it here.
For fair comparison, the basic architecture is kept the same for every method (see Appendix E). We tried different architectures and came to the same conclusion. For every method, we use *Adam* optimizer. Hyperparameter selection in unsupervised domain adaptation is a hard task as no labeled target data is available. For **DANN** and **MDAN**, we tried different sets of values and only report the ones giving the best results on test data which is very advantageous and normally not possible for unsupervised DA ($\lambda = 0.01$ for **DANN**, $\gamma = 10$, $\mu = 0.1$ for **MDAN** which are the hyperparameters they used in the classification setting). For our algorithm, the only hyperparameter is λ for ℓ^2-regularization of α for which we tried several values and we report results for $\lambda = 1$ here as we did not notice significant differences for values betwenn 10^{-3} and 10.
We ran the experiment 5 times and report the average MAE for each method in Table 2. The standard deviations are around 0.01 for **AHD-MSDA** and **AHDA** (see Appendix E) and for most domains, improvements are statistically significants using a Wilcoxon rank test. While the gains over the **MLP** with no adaptation may seem small, we emphasize that obtaining an improvement is already challenging as the **MLP** can already learn general features with the variety of data it is feeded. For most domains, **AHD-MSDA** obtains the best result with multi-source bringing an improvement over **AHDA** for many domains. One can notice the instability of **DANN** and **MDAN** for a few number of domains where the error becomes very large.
Overall, the **AHDA** achieves the best score and shows its efficiency for adversarial regression domain adaptation. The weights obtained by our method were meaningful for some domains (giving high weight for software to transfer to computer for instance) but not for all. We assume that the efficiency of the method is very dependent on the nature of the dataset as even in this case, for some domains, every adaptation method we tried fails.

5.3 Digit classification

Finally, we experiment our method on a digit classification task using 5 different domains: *MNIST*, *MNIST-M*, *Synth*, *SVHN* and *USPS*. *SVHN* and *USPS* are known to be the hardest datasets to classify as they are much more diverse. In each experiment, we use one domain as the target and the 4 others as the sources. We resize all domains to images made of 28×28 pixels. For fair comparison we use the same architecture for every network, using a simple convolutional neural network (**CNN** see Appendix F for details). For this experiment, we use a reduced version of the datasets with $10,000$ samples in each domain.
Since we are in a classification setting, the loss used to train **AHDA** and **AHD-MSDA** cannot be the Mean Squared Error. We found that the cross-entropy loss was performing better than the ℓ^1-loss, so we use it to compute *HDisc* (even though it does not follow triangle inequality). Our adversarial structure becomes very close to the MCD introduced in [18] and our weighting scheme extends this structure to a multi-source setting.
We report the accuracy for each dataset and each method in Table 3. One can see that for a classification task, our *HDisc* still gives state of the art results

Dataset	MNIST	MNIST-M	SVHN	Synth	USPS
CNN	0.916	0.450	0.489	0.562	0.624
DANN	0.918	**0.530**	**0.546**	0.710	0.659
AHDA	0.912	0.512	0.510	0.769	**0.667**
MDAN	**0.923**	0.460	0.488	0.671	0.574
AHD-MSDA	**0.923**	0.518	0.501	**0.793**	0.657

Table 3: Accuracy for the visual adaptations on digit datasets

compared to other methods tailored for classification. For MNIST, even without adaptation, the CNN can learn general features from other datasets. **MDAN** fails as it gives high weights to datasets that are not similar (such as **USPS** for **SVHN**) while our **AHD-MSDA** does not suffer from this drawback.

6 Conclusion

In this work we proposed a new general domain adaptation bound based on a new hypothesis-discrepancy compatible with classification and regression. We proposed an adversarial domain adaptation algorithm tailored for the task at hand, which is novel for regression. We extended these results to multiple sources to find ideal convex combinations of the sources. We demonstrated the efficiency of our method for a regression task for which we improved on previous state of the art results. For a classification task, **AHD-MSDA** obtains comparable results to other state of the art results. We emphasize that in the multi-source setting, our weighting scheme limits the risk of negative transfer. As we saw in our experiments, we mainly expect improvements of the multi-source method when some domains are really closer to other forming clusters. The main limitation of our work comes from the assumption of unsupervised domain adaptation that the labelling functions are the same in every domain. In our future work, we intend to investigate the semi-supervised and few-shot learning settings, where a few labeled target data is available.

References

[1] Adlam, B., Cortes, C., Mohri, M., Zhang, N.: Learning gans and ensembles using discrepancy. In: Advances in Neural Information Processing Systems. pp. 5788–5799 (2019)

[2] Ben-David, S., Blitzer, J., Crammer, K., Kulesza, A., Pereira, F., Vaughan, J.W.: A theory of learning from different domains. Machine learning **79**(1-2), 151–175 (2010)

[3] Ben-David, S., Blitzer, J., Crammer, K., Pereira, F.: Analysis of representations for domain adaptation. In: Advances in neural information processing systems. pp. 137–144 (2007)

[4] Ben-David, S., Lu, T., Luu, T., Pál, D.: Impossibility theorems for domain adaptation. In: International Conference on Artificial Intelligence and Statistics. pp. 129–136 (2010)

[5] Blitzer, J., Dredze, M., Pereira, F.: Biographies, bollywood, boom-boxes and blenders: Domain adaptation for sentiment classification. In: Proceedings of the 45th annual meeting of the association of computational linguistics. pp. 440–447 (2007)

[6] Cortes, C., Gonzalvo, X., Kuznetsov, V., Mohri, M., Yang, S.: Adanet: Adaptive structural learning of artificial neural networks. In: Proceedings of the 34th International Conference on Machine Learning-Volume 70. pp. 874–883. JMLR. org (2017)

[7] Cortes, C., Mohri, M.: Domain adaptation and sample bias correction theory and algorithm for regression. Theoretical Computer Science **519**, 103–126 (2014)

[8] Cortes, C., Mohri, M., Medina, A.M.: Adaptation based on generalized discrepancy. The Journal of Machine Learning Research **20**(1), 1–30 (2019)

[9] Friedman, J.H.: Multivariate adaptive regression splines. The annals of statistics pp. 1–67 (1991)

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

[11] Hoffman, J., Mohri, M., Zhang, N.: Algorithms and theory for multiple-source adaptation. In: Advances in Neural Information Processing Systems. pp. 8246–8256 (2018)

[12] Kuroki, S., Charoenphakdee, N., Bao, H., Honda, J., Sato, I., Sugiyama, M.: Unsupervised domain adaptation based on source-guided discrepancy. In: Proceedings of the AAAI Conference on Artificial Intelligence. vol. 33, pp. 4122–4129 (2019)

[13] Li, Y., Carlson, D.E., et al.: Extracting relationships by multi-domain matching. In: Advances in Neural Information Processing Systems. pp. 6798–6809 (2018)

[14] Mansour, Y., Mohri, M., Rostamizadeh, A.: Domain adaptation: Learning bounds and algorithms. COLT 2009 - The 22nd Conference on Learning Theory (02 2009)

[15] Mansour, Y., Mohri, M., Rostamizadeh, A.: Domain adaptation with multiple sources. In: Advances in neural information processing systems. pp. 1041–1048 (2009)

[16] Mansour, Y., Mohri, M., Rostamizadeh, A.: Multiple source adaptation and the rényi divergence. arXiv preprint arXiv:1205.2628 (2012)

[17] McDiarmid, C.: Concentration. In: Probabilistic methods for algorithmic discrete mathematics, pp. 195–248. Springer (1998)

[18] Saito, K., Watanabe, K., Ushiku, Y., Harada, T.: Maximum classifier discrepancy for unsupervised domain adaptation. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition. pp. 3723–3732 (2018)

[19] Tzeng, E., Hoffman, J., Saenko, K., Darrell, T.: Adversarial discriminative domain adaptation. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition. pp. 7167–7176 (2017)

[20] Wen, J., Greiner, R., Schuurmans, D.: Domain aggregation networks for multi-source domain adaptation. arXiv preprint arXiv:1909.05352 (2019)

[21] Xu, Z., Sun, S.: Multi-source transfer learning with multi-view adaboost. In: International conference on neural information processing. pp. 332–339. Springer (2012)

[22] Zhao, H., Zhang, S., Wu, G., Moura, J.M., Costeira, J.P., Gordon, G.J.: Adversarial multiple source domain adaptation. In: Advances in neural information processing systems. pp. 8559–8570 (2018)

A Proof of Proposition 1

Proof. For any $h \in \mathcal{H}$, and any $k \in \{1, ..., K\}$

$$\begin{aligned}
\epsilon_t(h, f_t) &\leq \epsilon_k(h, f_k) + |\epsilon_t(h, f_t) - \epsilon_k(h, f_k)| \\
&\leq \epsilon_k(h, f_k) + |\epsilon_t(h, h_0) - \epsilon_t(h, f_t)| + |\epsilon_k(h, h_0) - \epsilon_k(h, f_k)| + |\epsilon_t(h, h_0) - \epsilon_k(h, h_0)| \\
&\leq \epsilon_k(h, f_k) + HDisc_{\mathcal{H},L}(p_k, p_t; h) + \mathbb{E}_{x\sim p_t}[|L(h(x), h_0(x)) - L(h(x), f_t(x))|] \\
&+ \mathbb{E}_{x\sim p_k}[|L(h(x), h_0(x)) - L(h(x), f_t(x))|]] \\
&\leq \epsilon_k(h, f_k) + HDisc_{\mathcal{H},L}(p_k, p_t; h) + \mathbb{E}_{x\sim p_t}[|L(h_0(x), f_t(x))|] + \mathbb{E}_{x\sim p_k}[|L(h_0(x), f_k(x))|] \\
&\leq \epsilon_k(h, f_k) + \epsilon_k(h_0, f_k) + \epsilon_t(h_0, f_t) + HDisc_{\mathcal{H},L}(p_k, p_t; h)
\end{aligned}$$

where the first inequality comes from the triangle inequality, the second holds for any $h_0 \in \mathcal{H}$. The third comes from the definition of $HDisc(p_k, p_t)$ and the fourth follows from triangle inequality on L. Finally, the result follows taking the minimum over all h_0. □

B Proof of Theorem 1

Proof. Using Proposition 1 with p_α, we get:

$$\epsilon_t(h, f_t) \leq \epsilon_\alpha(h, f_\alpha) + \eta_{\mathcal{H}}(f_t, f_\alpha) + HDisc_{\mathcal{H},L}(p_\alpha, p_t; h) \tag{6}$$

Let us consider K empirical distributions $\hat{p_k}$ corresponding to a sample $\mathcal{S}_k = \{x_1^{(k)}, ..., x_m^{(k)}$ of size m and a hypothesis $h \in \mathcal{H}$. Then the α-weighted sample $\mathcal{S}_\alpha$ is of size Km and empirical distribution $\hat{p_\alpha}$ defined as $\hat{p_\alpha}(x_i^{(k))} = \frac{\alpha_k}{m}$. We define $\phi = \epsilon_\alpha(h, f_\alpha) - \hat{\epsilon_\alpha}(h, f_\alpha)$. Changing an element $x_i^{(k)}$ modifies ϕ by a maximum of $\frac{\alpha_k}{m}$. We can deduce from the McDiarmid's inequality that:

$$\mathbb{P}(\phi - \mathbb{E}(\phi) \leq t) \leq \exp^{\frac{-2mt^2}{\sum_{k=1}^K \alpha_k^2}}$$

Hence with probability $1 - \delta$,

$$\epsilon_\alpha(h, f_\alpha) \leq \hat{\epsilon_\alpha}(h, f_\alpha) + \mathbb{E}_{\hat{p_\alpha}}[\phi] + \|\alpha\|_2 M\sqrt{\frac{\log 1/\delta}{2m}} \tag{7}$$

We are able to compute $\mathbb{E}_{\hat{p_\alpha}}$ using the classical tool of ghost sample in Rademacher complexity analysis as in the proof of Theorem 2 in [14]. In the following we will use a random variable σ taking its values uniformly over $\{-1, 1\}$. We will write $\{x_i^{(k)}; 1 \leq k \leq K, 1 \leq i \leq m\}$ the sample associated to $\hat{p}_\alpha \sim p_\alpha$ and $\{z_i^{(k)}; 1 \leq k \leq K, 1 \leq i \leq m\}$ $\hat{q}_\alpha \sim p_\alpha$.

$$
\begin{aligned}
\mathbb{E}_{\hat{p_\alpha}\sim p_\alpha}[\phi] &= \mathbb{E}_{\hat{p_\alpha}}[\epsilon_\alpha(h,f_\alpha) - \hat{\epsilon_\alpha}(h,f_\alpha)] \\
&\leq \mathbb{E}_{\hat{p_\alpha}\sim p_\alpha}[\sup_{h\in\mathcal{H}} \epsilon_\alpha(h,f_\alpha) - \hat{\epsilon_\alpha}(h,f_\alpha)] \\
&\leq \mathbb{E}_{\hat{p_\alpha}\sim p_\alpha}[\sup_{h\in\mathcal{H}} \mathbb{E}_{\hat{q_\alpha}\sim p_\alpha}[\mathbb{E}_{x\sim\hat{q_\alpha}}[L(h(x),f_\alpha(x))]] - \mathbb{E}_{x\sim\hat{p_\alpha}}[L(h(x),f_\alpha(x))] \\
&\leq \mathbb{E}_{\hat{p_\alpha},\hat{q_\alpha}\sim p_\alpha}[\sup_{h\in\mathcal{H}} \mathbb{E}_{x\sim\hat{q_\alpha}}[L(h(x),f_\alpha(x)] - \mathbb{E}_{x\sim\hat{p_\alpha}}[L(h(x),f_\alpha(x)] \\
&\leq \mathbb{E}_{\hat{p_\alpha},\hat{q_\alpha}\sim p_\alpha}[\sup_{h\in\mathcal{H}} \sum_{k=1}^{K} \alpha_k(\mathbb{E}_{x\sim\hat{q_k}}[L(h(x),f_k(x)) - \mathbb{E}_{x\sim\hat{p_k}}[L(h(x),f_k(x)))] \\
&\leq \mathbb{E}_{\hat{p_\alpha},\hat{q_\alpha}\sim p_\alpha}[\sup_{h\in\mathcal{H}} \sum_{k=1}^{K} \frac{\alpha_k}{m} \sum_{i=1}^{m}(L(h(x_i^{(k)}),f_k(x_i^{(k)})) - L(h(z_i^{(k)}),f_k(z_i^{(k)})))] \\
&\leq \mathbb{E}_{\substack{\hat{p_\alpha},\hat{q_\alpha}\sim p_\alpha \\ \sigma_i\sim\sigma}}[\sup_{h\in\mathcal{H}} \sum_{k=1}^{K} \frac{\alpha_k}{m} \sum_{i=1}^{m}\sigma_i(L(h(x_i^{(k)}),f_k(x_i^{(k)})) - L(h(z_i^{(k)}),f_k(z_i^{(k)})))] \\
&\leq \mathbb{E}_{\substack{\hat{p_\alpha}\sim p_\alpha \\ \sigma_i\sim\sigma}}[\sup_{h\in\mathcal{H}} \sum_{k=1}^{K} \frac{\alpha_k}{m} \sum_{i=1}^{m}\sigma_i L(h(x_i^{(k)}),f_k(x_i^{(k)}))] \\
&\quad + \mathbb{E}_{\substack{\hat{q_\alpha}\sim p_\alpha \\ \sigma_i\sim\sigma}}[\sup_{h\in\mathcal{H}} -\sum_{k=1}^{K} \frac{\alpha_k}{m} \sum_{i=1}^{m}\sigma_i L(h(z_i^{(k)}),f_k(z_i^{(k)}))] \\
&\leq 2\mathbb{E}_{\substack{\hat{p_\alpha}\sim p_\alpha \\ \sigma_i\sim\sigma}}[\sup_{h\in\mathcal{H}} \sum_{k=1}^{K} \frac{\alpha_k}{m} \sum_{i=1}^{m}\sigma_i L(h(x_i^{(k)}),f_k(x_i^{(k)}))] \\
&\leq 2\sum_{k=1}^{K} \alpha_k \mathcal{R}_m(\mathcal{H}_k)
\end{aligned}
\tag{8}
$$

Finally, noting that with empirical distributions $\hat{\epsilon_\alpha}(h,f_\alpha) = \sum_{k=1}^{K}\alpha_k\hat{\epsilon_k}(h,f_k)$, we have the final result with probability $1-\delta$ over the sample of $\mathcal{S}_1$, ..., $\mathcal{S}_K$:

$$
\begin{aligned}
\epsilon_t(h,f_t) \leq &\sum_{k=1}^{K}\alpha_k\hat{\epsilon_k}(h,f_k) + \mathit{HDisc}_{\mathcal{H},L}(p_t,p_\alpha) + \eta_{\mathcal{H},\alpha} \\
&+ 2\sum_{k=1}^{K}\alpha_k\mathcal{R}_m(\mathcal{H}_k) + \|\alpha\|_2 M\sqrt{\frac{ln(1/\delta)}{2m}}
\end{aligned}
\tag{9}
$$

□

C Empirical bound for the hypothesis-discrepancy

Like the original discrepancy, for any $h \in \mathcal{H}$, $\mathit{HDisc}(.,.;h)$ is symmetric and can be estimated with finite samples. In Theorem 2, we give a bound for empirical distributions. With $\mathcal{H}_s = \{h : x \to L(h(x), f_s(x)\}$ and $\mathcal{H}_t = \{h : x \to L(h(x), f_t(x)\}$, and the empirical source risk $\hat{\epsilon_s}(h,f_s) = \sum_{i=1}^{m}\frac{1}{m}L(h(x_i^{(s)}),f_s(x_i^{(s)}))$.

Theorem 2. We assume that the loss L is symmetric, follows the triangle inequality and verifies $L(h(x), y) \leq M$ for all $h \in \mathcal{H}$ and $(x, y) \in \mathcal{X} \times \mathcal{Y}$. Then for any hypothesis $h \in \mathcal{H}$, with probability $1 - \delta$ over the samples of $\mathcal{S}_s$ of size m according to p_s and $\mathcal{S}_t$ of size n according to p_t the following bound holds:

$$\begin{aligned}\epsilon_t(h, f_t) \leq & \hat{\epsilon}_s(h, f_s) + \eta_{\mathcal{H}}(f_s, f_t) + HDisc_{\mathcal{H},L}(\hat{p_t}, \hat{p_s}; h) \\ & + 4\mathcal{R}_m(\mathcal{H}_s) + 2\mathcal{R}_n(\mathcal{H}_t) + 2M\sqrt{\frac{\log 2/\delta}{2m}} + M\sqrt{\frac{\log 2/\delta}{2n}}\end{aligned} \tag{10}$$

where $\mathcal{R}_m(\mathcal{H}_s)$ and $\mathcal{R}_n(\mathcal{H}_t)$ are the Rademacher complexities of $\mathcal{H}_s$ and $\mathcal{H}_t$ defined in Appendix **??**.

Proof. The proof is similar to the ones of [14]. Let us consider the empirical distribution $\hat{p_s}$ corresponding to a sample $\mathcal{S}_s$ of size m and a hypothesis $h \in \mathcal{H}$. We first define $\phi(h) = \epsilon_s(h, f_s) + HDisc(p_s, p_t; h) - \hat{\epsilon_s}(h, f_s) + HDisc(\hat{p_s}, p_t; h)$. Then as the loss L is bounded by M, changing one element of $\mathcal{S}_s$ will changes $\phi(h)$ by a maximum of $\frac{2M}{m}$. McDiarmid's inequality states that with probability $1 - \frac{\delta}{2}$,

$$\epsilon_s(h, f_s) + HDisc(p_s, p_t; h) \leq \hat{\epsilon_s}(h, f_s) + HDisc(\hat{p_s}, p_t; h) + \mathbb{E}_{\hat{p^s}}[\phi(h)] + 2M\sqrt{\frac{\log(2/\delta)}{2m}}$$

Moreover, from Theorem 2 and Proposition 2 from [14], we know that

- $\mathbb{E}_{\hat{p_s}}[\epsilon_s(h, f_s) - \hat{\epsilon_s}(h, f_s)] \leq 2\mathcal{R}_m(\mathcal{H}_s)$
- $\mathbb{E}_{\hat{p_s}}[HDisc(\hat{p_s}, p_t; h) - HDisc(\hat{p_s}, p_t; h)] \leq 2\mathcal{R}_m(\mathcal{H}_s)$

where $\mathcal{H}_s = \{x \to L(h(x), f_s(x)); \forall h \in \mathcal{H}\}$
As a consequence, with probability $1 - \frac{\delta}{2}$ over the sampling of $\mathcal{S}_s$:

$$\epsilon_s(h, f_s) + HDisc(p_s, p_t; h) \leq \hat{\epsilon_s}(h, f_s) + HDisc(\hat{p_s}, p_t; h) + 4\mathcal{R}_m(\mathcal{H}_s) + 2M\sqrt{\frac{\log(2/\delta)}{2m}}$$

Using Proposition 2 from [14] we obtain that with probability $1 - \frac{\delta}{2}$ over the sampling of $\mathcal{S}_t$:

$$HDisc(\hat{p_s}, p_t; h) \leq HDisc(\hat{p_s}, \hat{p_t}; h) + 2\mathcal{R}_m(\mathcal{H}_s) + M\sqrt{\frac{\log(2/\delta)}{2n}}$$

Hence using an union bound, we have the final result with probability $1 - \delta$:

$$\begin{aligned}\epsilon_t(h, f_t) \leq & \hat{\epsilon}_s(h, f_s) + \eta_{\mathcal{H}}(f_s, f_t) + HDisc_{\mathcal{H},L}(\hat{p_t}, \hat{p_s}; h) \\ & + 4\mathcal{R}_m(\mathcal{H}_s) + 2\mathcal{R}_n(\mathcal{H}_t) + 2M\sqrt{\frac{\log 2/\delta}{2m}} + M\sqrt{\frac{\log 2/\delta}{2n}}\end{aligned} \tag{11}$$

□

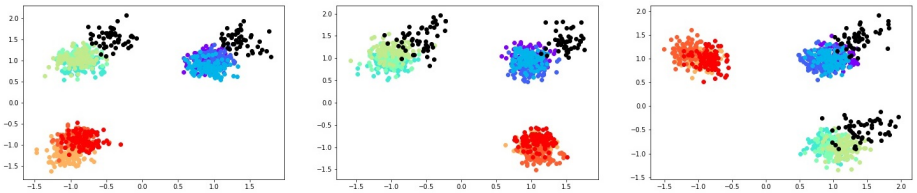


Figure 6: Data for the multiple source domain adaptation with $\alpha = \{1/6, 1/6, 1/6, 1/6, 1/6, 1/6, 0, 0, 0\}$ (2 clusters with uniform weights)

D Details for the Friedman experiment

We display the obtained data in Figure 6 where only two clusters are selected to generate the target data. Every implementation has been made in Python and the method has been implemented using the Pytorch library.
The network architecture is kept the same for every method as follows:

- **Feature extractor:**
  - $Linear(5, 10, ELU)$,
  - $Dropout(0.2)$
  - $Linear(10, 5)$,
  - $Dropout(0.2)$
- **Predictor/Discriminator:**
  - $Linear(5, 1)$

We use SGD optimizer with $lr = 0.001$ and momentum 0.9. The batch size is set to 32 in every experiment.
In Figure 7, we report one of the features extracted by **AHDA** and **DANN** against y. As the final layer is linear, we expect a linear relationship, which is done by **AHDA** while **DANN** tries to align domains in the sense of classification.

E Details about Sentiment analysis Experiments

The architecture is kept the same for each method:

- **Feature extractor:**
  - $Linear(1000, 500, LeakyRELU)$,
  - $Dropout(0.1)$
  - $Linear(500, 20, LeakyRELU)$,
  - $Dropout(0.1)$
- **Predictor/Discriminator:**
  - $Linear(20, 1)$

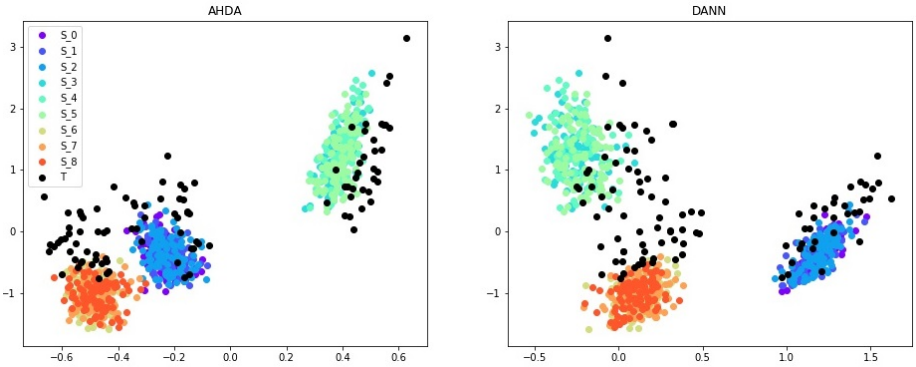


Figure 7: X-axis: Extracted features (before the final predictor) using **AHDA** (left) and **DANN** (right) ; Y-axis: labels to predict (y)

We tried different architectures and the conclusions were the same: adversarial hypothesis discrepancy performs the best out of other compared methods. For every method, we use *Adam* optimizer with learning rate $lr = 0.001$ and a batch size equal to 128. We give equal amount of data from each domain in each batch. Every time, we take one product as the target and every other as the source. Each network is trained for 100 epochs. In

Dataset	**apparel**	**automotive**	**baby**	**beauty**	**books**	**camera**	**cellphones**	**computer**	**dvd**
MLP	0.011	0.007	0.004	0.011	0.016	0.01	0.004	0.008	0.009
DANN	0.037	0.15	0.039	0.04	0.112	0.06	0.08	0.167	0.137
AHDA	0.007	0.016	0.012	0.009	0.008	0.008	0.015	0.012	0.018
MDAN	0.025	0.048	0.051	0.051	0.32	0.051	0.042	0.274	0.162
AHD-MSDA	0.007	0.04	0.002	0.015	0.017	0.009	0.015	0.016	0.014
Dataset	**electronics**	**food**	**grocery**	**health**	**jewelry**	**kitchen**	**magazines**	**music**	
MLP	0.009	0.011	0.004	0.005	0.008	0.005	0.008	0.009	
DANN	0.079	0.166	0.017	0.059	0.019	0.027	0.106	0.236	
AHDA	0.011	0.012	0.011	0.011	0.018	0.006	0.009	0.023	
MDAN	0.049	0.055	0.02	0.057	0.029	0.01	0.438	0.285	
AHD-MSDA	0.01	0.01	0.008	0.006	0.036	0.014	0.014	0.018	
Dataset	**musical**	**office**	**outdoor**	**software**	**sports**	**tools**	**toys**	**video**	
MLP	0.012	0.035	0.007	0.007	0.006	0.028	0.006	0.01	
DANN	0.052	0.118	0.013	0.147	0.034	0.264	0.044	0.101	
AHDA	0.018	0.014	0.009	0.012	0.009	0.031	0.012	0.014	
MDAN	0.082	0.09	0.005	0.049	0.045	0.315	0.024	0.09	
AHD-MSDA	0.035	0.053	0.01	0.013	0.017	0.072	0.009	0.017	

Table 4: Standard deviation of the MAE for each method and each dataset over 5 runs

F Details about Visual adaptation Experiments

Each dataset is formed with $20,000$ samples and resized to 28×28 pixels. The architecture is kept the same for every method:

- **Feature extractor:**
  - $Conv2d(1, 64, 3, padding = 1)$ with $LeakyReLU$
  - $MaxPool2d(2)$
  - $Conv2d(64, 128, 3, padding = 1)$ with $LeakyReLU$
  - $MaxPool2d(2)$

- $Conv2d(128, 256, 3, padding = 1)$ with $LeakyReLU$
- $MaxPool2d(2)$
- $Flatten()$

- **Predictor / Discriminator:**
  - $Linear(2304, 100)$ with $LeakyReLU$
  - $Dropout(0.2)$
  - $Linear(100, 10)$ (predictor) / $Linear(100, 2)$ (discriminator)
  - $Softmax()$

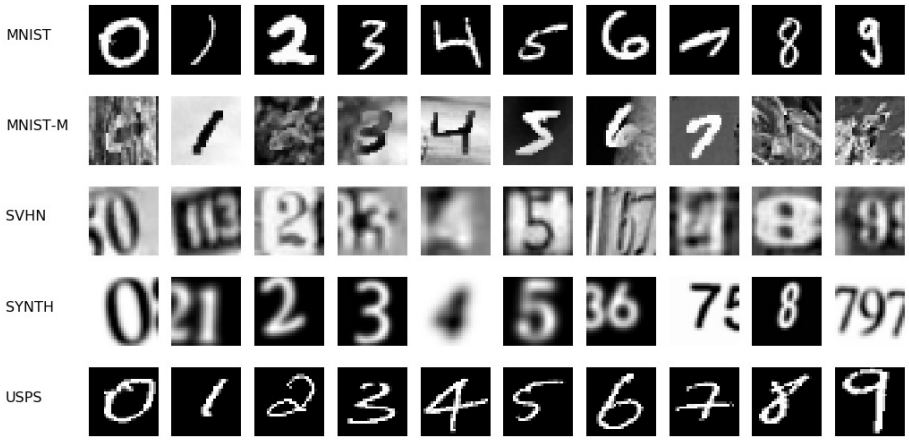


Figure 8: Visualization of digit datasets