Domain Adaptation in the Absence of Source Domain Data

Boris Chidlovskii, Stéphane Clinchant and Gabriela Csurka
Xerox Research Centre Europe
6 chemin Maupertuis, Meylan, France

ABSTRACT

The overwhelming majority of existing domain adaptation methods makes an assumption of freely available source domain data. An equal access to both source and target data makes it possible to measure the discrepancy between their distributions and to build representations common to both target and source domains. In reality, such a simplifying assumption rarely holds, since source data are routinely a subject of legal and contractual constraints between data owners and data customers. When source domain data cannot be accessed, decision making procedures are often available for adaptation nevertheless. These procedures are often presented in the form of classification, identification, ranking etc. rules trained on source data and made ready for a direct deployment and later reuse. In other cases, the owner of a source data is allowed to share a few representative examples such as class means.

In this paper we address the domain adaptation problem in real world applications, where the reuse of source domain data is limited to classification rules or a few representative examples. We extend the recent techniques of feature corruption and their marginalization, both in supervised and unsupervised settings. We test and compare them on private and publicly available source datasets and show that significant performance gains can be achieved despite the absence of source data and shortage of labeled target data.

Keywords

Domain adaptation; classification; machine learning; marginalization; emerging applications

1. INTRODUCTION

While huge volumes of unlabeled data are generated and made available in many domains, the cost of acquiring data labels remains high. Domain Adaptation (DA) problems arise each time when we need to leverage labeled data in one or more related source domains, to learn a classifier for unseen data in a target domain. The domains are assumed to be related, but not identical in which case it becomes a standard machine learning problem. Such a situation occurs in multiple real world applications. Examples include named entity recognition across different text corpora, object recognition in images acquired in different conditions (like background scene, object location and pose, view angle changes), and many others (see [25][34][39] for a survey on transfer learning and domain adaptation methods).

Domain adaptation is particularly critical for service companies operating customer business processes in customer care, human resource and finance. All machine learning components deployed in a given service solution should be customized to a new customer. This entails re-annotating data or calibrating the models in order to achieve a contractual performance. For example, in vehicle classification, when an image classification algorithm is re-deployed in a new location, better performance is obtained by annotating samples collected on the field. In the brand sentiment management, when a sentiment classifier is re-deployed for a new customer, it is critical to tune the classifier to the manner how users talk about their experience with a particular product, etc.

Numerous approaches have been proposed in the last years to address the text and visual domain adaptation to cite just a few [3] [6] [18][19][20][22][23][24][30][32][36]. The majority of existing methods makes an assumption of freely available source domain data. An equal access to both source and target data allows to measure the discrepancy between their distributions and 1) either build representations common to both target and sources, or 2) directly reuse some source instances for a better target classification [43].

In reality, such a simplifying assumption rarely holds. Source data are often a subject of legal, technical and contractual constraints between data owners and data customers. Beyond privacy and disclosure obligations, customers are often reluctant to share their data. When operating customer care, collected data may include information on recent technical problems which is a highly sensitive topic that companies are not willing to share.

It is more common that just decision making procedures but not source domain data are available for adaptation. These procedures are often presented in the form of classification, identification, ranking etc. rules trained on source data and made ready for a direct deployment and later reuse. In other cases, the owner of a source data is not allowed to share the entire dataset, but some representative source examples such as class means.

In this paper we address the domain adaptation problem in real world applications, where the reuse of source data is limited to classification rules or a few representative examples. We adopt the recent techniques of feature corruption and their marginalization, both in the supervised (the Marginalized Corrupted Features [31]) and unsupervised (stacked Marginalized Denoising Autoencoder [7]) frameworks. We propose several extensions to MCF and sMDA methods in order to address the adaptation needs with no access to source domain data. We cope with three most typical cases of what is available for adaptation: 1) classifiers as a black box, 2)
classifiers with known models and parameters, and 3) class representatives.

The remainder of the paper is organized as follows. In Section 2, we present several industrial applications on customer data with a critical need for domain adaptation. In Section 3, we revise the prior art and Section 4 presents how to extend MCF ad sMDA methods to address these adaptation needs, in both supervised and unsupervised settings. Publicly available datasets of Office and Amazon, as well as Xerox’s Smart Document Management data are described in Section 5.1. In Section 5, we describe the experimental setting and report the evaluation results of adapting the source classifiers and representatives on the selected datasets. Section 6 discusses the open questions and concludes the paper.

2. TARGET APPLICATIONS

In this section, we present three real world applications with a strong domain adaptation need and no access to source domain instances.

2.1 Vehicle Passenger Detection System

Upon a strong demand for vehicle occupancy count (VOC) automation in the US tolling and European border crossing, Xerox has developed Vehicle Passenger Detection System (VPDS). The VPDS is a two-camera system that takes one photograph through the front windscreen of the car, and another one on the side of the car, looking into the back seat. Using the two photos, the system calculates whether the car contains just the driver, the driver plus one passenger, or the driver plus two or more passengers. The project started with one week of "training" the system on the field. Compliance with data protection law is ensured by applying non-reversible blurring techniques to the photographs to prevent facial and number plate recognition; and by retaining the photographs for no more than five days. The results proved the system to be 95% accurate in counting occupants.

The technical solution consists of the car detection, car location in the image, and the VOC classifiers. When the front and side cameras take a photo, the object detectors first localize the information zones in these images. In particular, the visor breeze to the front image and the rear window to the side image. The front and side classifiers then estimate probability of passenger presence in the car. The last module aggregates these probabilities into a final estimate of the vehicle occupancy rate.

All classifiers are learned during the training period, when the image annotation tool (see Figure 1) is used to screen the images and to associate each car image with one of the labels, 1, 2 or "3 or more". The training of the algorithm includes annotating the image pairs and estimating optimum parameters to classify the new images.

When a new VPDS installation is requested, the “training” period and annotation cost is expected to be reduced by emerging domain adaptation techniques. As all images are destroyed after five days, the VOC classifiers are the only components to be reused from one VPDS installation to another.

2.2 EMPATH

EMPATH is a product developed by Xerox for customer analytics. It aims to understand customers’ experiences, needs, and opinions in order to improve operations, products and services offered to customers. EMPATH starts by gathering data from a range of text sources (social media, news feeds, emails, chat and surveys). Collected data is then analyzed by an operator with the help of a text analytics engine. The operator tags one to two hundred posts, which are then used for training statistical models. This analysis helps operators better answer customer requests, to report upcoming issues and to provide valuable insights on product development.

In the text analytics engine, topic models, categories and sentiment classifiers are treated as domain and customer dependent. For some domains (such as IT and mobile phone), major concerns are similar and domain adaptation has been shown to reduce the annotation burden.

The main text analytics GUI is presented in Figure 2. Using the interface, the operator extracts topics from the text, creates categories to detect coming issues and monitors customer satisfaction with the sentiment predictions. As customers disallow to share their data, operators face the problem of wasting their time and resource by starting the analytics processes from scratch. As certain components, topics, categories and sentiments are common to many brand customers, the adaptation of classifiers across domain and customers represents a strong opportunity for the EMPATH product.

2.3 Print and Document Usage Analytics

As a part of Print and Document Management Services, Xerox offers to their customers facilities to automate and simplify the management of paper and electronic documents. The dematerialization services and process automation offered by Xerox help

---


http://www.wds.co/product/contact-center/social-media/
streamline client’s business processes to drive growth, and to reduce costs.

As documents vary a lot by their origin, layout and content, there exists a strong need to evolve a workflow for the same client, as well as to redeploy the workflow for new clients, under all legal obligations.

In Section 5 document collections from three customers and evaluation results of domain adaptation are presented in more details.

3. STATE OF THE ART

Domain adaptation for text data has been studied for more than a decade, with applications in statistical machine translation, opinion mining, part of speech tagging and document ranking. Most effective techniques include feature replication [15, 18, 32, 34], pivot features [2, 5, 32] and finding topic models shared by source and target collections [9, 10]. Domain adaptation has equally received a lot of attention in computer vision [3, 6, 17, 18, 19, 20, 22, 23, 24, 30, 32, 35, 38], where domain shift is a consequence of changing conditions, such as background, location and pose, etc. and, more generally, a strong bias between datasets.

We argue that the above principle can be extended to multi-class classification tasks, the refinement may require a significant amount of new labeled data. Recent works by Ganin et al. [20, 32, 34] have proposed better strategies than fine tuning: adding a domain prediction task while learning the deep neural network leads to better domain-invariant feature representation. Their approach obtained a significant performance gain which shows that transfer learning is not completely solved by fine tuning and that transfer tasks should be addressed by appropriate deep learning representations.

Nevertheless, the majority of domain adaptation methods makes an assumption of largely available source collections. Such assumption rarely holds in reality. The source instances may become unavailable for technical reasons, or are disallowed to store for legal and contractual reasons. When objects are to be removed from a database, it also makes privacy preserving machine learning methods inapplicable [11, 41].

The case of available source classifiers have been considered by Duan et al. in [17], where their contribution is limited to regularizing the supervised target classifier. On the other hand, this case has been studied by domain adaptation theory [35]. Particularly, a theoretical analysis was conducted by considering the algorithmic stability of source classifiers. It was shown that the relatedness of source and target domains accelerates the convergence of the Leave-One-Out (LOO) error to the generalization error, thus enabling the use of the LOO error to stop the adaptation learning. In case of unrelated domains, it also suggests how to prevent negative transfer, so that in the limit it does not hurt the performance in the target domain.

4. DOMAIN ADAPTATION WITH CLASSIFIERS

We define a domain $D$ as composed of a feature space $X \subseteq R^D$ and a label space $Y$. Any given task in domain $D$ (classification, regression, ranking, etc.) is defined by a function $h : X \rightarrow Y$.

In the domain adaptation setting, we assume working with $n_S + 1$ domains, including $n_S$ source domains $D^s, s = 1, \ldots, n_S$ and a target domain $D^t$. In conventional setting the source data $X^s = \{X_1^s, \ldots, X_{n_S}^s\}$, with the corresponding labels $Y^s = \{Y_1^s, \ldots, Y_{n_S}^s\}$ are available. In the absence of source data, we assume to have at our disposal either 1) a set $F$ of classifiers $f_k, k = 1, \ldots, K$, each being trained on source data where at least one classifier is available for source domain $D^s$, or 2) a set of representatives for each $D^s, s = 1, \ldots, n_S$.

A close analysis of target applications (see Section 2) reveals three important cases to consider; they reflect different situations and complexity levels, as follows:

1. Models and parameters $\theta_k$ of source classifiers $f_k \in F$ are known; for example we might know they are linear and have access to their parameter vectors $\theta_k = \omega_k$;

2. Models of classifiers $f_k \in F$ are unknown; they are used as a black box and, for a target instance $x^t$, predict a class $c$ probability $P(y^t = c | x^t; F)$;

3. In the target domain, we may have access to both labeled and unlabeled instances, denoted $X^t$ and $X^u$ accordingly.

The domain adaptation problem then consists of leveraging the source labeled and target unlabeled data to derive a hypothesis performing well on the target domain. To achieve this goal, most DA methods compute covariance or correlation between features from source and target domains. With no or limited access to source data, we argue that the above principle can be extended to the correlation between target features and the source class decisions. We thus tune the adaptation trick by considering predicted class probabilities as augmented features for target data. Domain adaptation algorithms can then be used to find relations between features which
were related in the source with features which seem also related in the target collection. In other words, we use the source classifiers as a pivot to transfer knowledge from source to target.

In the following section, we propose solutions to address the three cases mentioned above:

1. When source classifiers are known and they are linear, we extend the Marginalized Corrupted Features (MCF) method to jointly marginalize the corrupted features and predictions;

2. When source classifiers are used as a black box, we can either a) marginalize their predictions with labeled data or b) denoise these predictions with unlabeled target data and Marginalized Denoising Autoencoders;

3. When class means are available as source representatives, we can use them as a Domain Specific Class Means classifier, and denoise their predictions jointly with unlabeled target data.

In the following we will discuss the unsupervised setting where no target instance is labeled, $\mathcal{X}_t = \emptyset$, and the supervised setting where a few target examples (for each class) are labeled. To address these problems, we adapt on one hand the Marginalized Corrupted Features that allows to address the problem when target labels are available and, on the other hand, the stacked Marginalized Denoising Autoencoder (sMDA) framework to address the case when no labeled target instances are available.

### 4.1 Supervised Adaptation with MCF

First, we assume that a set of target instances were manually labeled and we consider the Marginalized Corrupted Features framework to address our specific domain adaptation cases.

To marginalize the corrupted features in the MCF, a corrupting distribution is first defined to transform observations $x$ into corrupted versions $\tilde{x}$. The corrupting distribution is assumed to factorize over all feature dimensions and, each individual distribution is assumed to be a member of the natural exponential family, $p(\tilde{x}|x) = \prod_{d=1}^{D} P(\tilde{x}_d|x_d; \theta_d)$, where $x = (x_1, \ldots, x_D)$ and $\theta_d, d = 1, \ldots, D$ is a parameter of the corrupting distribution on dimension $d$.

The corrupting distribution can be unbiased ($E[\tilde{x}|p(\tilde{x}|x) = x]$) or biased. Known examples of distribution $P$ are the blanket[40], Gaussian, Laplace and Poisson noise [31]. For example, in the unbiased blanket noise, $\tilde{x} = 0$ and $p(\tilde{x} = 0) = q$ and $p(\tilde{x} = \pm 1) = 1 - q$, with expectation $E[\tilde{x}] = x$ and variance $Var[\tilde{x}] = \frac{q}{1-q}x^2$.

For the generalization of a classifier using corrupting distribution $P$, the direct approach is to corrupt each element of the target training set $D^t = \{(x_n, y_n)\}$, $n = 1, \ldots, N$. The extended data set $D^t$ can be used for training the model by minimizing

\[
\mathcal{L}(\tilde{D}^t; \Theta) = \frac{1}{M} \sum_{m=1}^{M} \sum_{n=1}^{N} L(\tilde{x}_{nm}, y_n; \Theta),
\]

where $\tilde{x}_{nm} \sim p(\tilde{x}|x_n)$ is the $m$-th corrupted observation of $x_n$ and $\Theta$ is the set of model parameters. As explicit corruption comes at a high computational cost, the marginalization trick is to consider the limiting case $M \rightarrow \infty$, where the weak law of large numbers is applied to rewrite $\frac{1}{M} \sum_{m=1}^{M} L(\tilde{x}_{nm}, y_n; \Theta)$ as its expectation $E[L(\tilde{x}_{nm}, y_n; \Theta)]$. It has been shown that dropout marginalization plays the role of regularization [31]. Classifiers can be trained efficiently with quadratic, exponential, logistic [31] and hinge loss [3], they generalize better and are more robust to missing feature values at test time. In this paper we focus on the quadratic loss which allows a closed form solution[4].

驱动 by our target applications in Section 2 we consider the multi-class case where label space $\mathcal{Y}$ includes $C$ classes. We denote by $y_s$ the vector of size $C$, where the component $c$ corresponding to the class label of $x_n$ is set to 1, and all others set to 0.

The expected value of the quadratic loss under corrupting distribution $p(\tilde{x}|x)$ is given by:

\[
\mathbf{L}(\mathcal{D}; \Omega) = \mathbb{E}[\sum_n \| \Omega^\top \tilde{x}_n - y_n \|^2] = \text{tr}(\sum_n (\Omega^\top \mathbb{E}[\tilde{x}_n^\top | \Omega] - \Omega^\top \mathbb{E}[\tilde{x}_n] y_n^\top) - y_n \mathbb{E}[\tilde{x}_n^\top | \Omega + \mathbb{E}[y_n y_n^\top])],
\]

where $\Omega$ is a $D \times C$ parameter matrix with column $c$ seen as a linear classifier corresponding to the class $c$, and $\text{tr}(\cdot)$ is the matrix trace. The optimal solution for $\Omega^*$ can be given in a closed form [31] as:

\[
\Omega^* = \left( \sum_{n=1}^{N} \mathbb{E}[\tilde{x}_n \tilde{x}_n^\top] \right)^{-1} \left( \sum_{n=1}^{N} \mathbb{E}[\tilde{x}_n] y_n^\top \right).
\]

As the corruption distribution factorizes over all dimensions, $\Omega^*$ depends only on the expectations $\mathbb{E}[\tilde{x}_{n}]$ and variances $\text{Var}[\tilde{x}_{n}]$ over individual dimensions $d$. Hence, for the unbiased distribution $P$ we can write $\Omega^* = Q^{-1}P$, where $P = \sum_{n=1}^{N} x_n y_n^\top$, $Q = \sum_{n=1}^{N} \{x_n x_n^\top + \text{diag}(\text{Var}[x_n])\}$, and $\text{diag}(\text{Var}[x_n])$ denotes the $D \times D$ diagonal matrix constructed with the variances $\text{Var}[x_n]$.

### 4.2 Integrating source classifiers in MCF

In the case of available source classifiers, the explicit corruption takes into account their predictions on corrupted instances, as $L(\tilde{x}_{nm}, y_n; F; \Omega)$.

In the limiting case $M \rightarrow \infty$, we obtain the loss expectation

\[
\mathbf{L}(\mathcal{D}^t; F; \Omega) = \sum_{n=1}^{N} \mathbb{E}[L(\tilde{x}_n, y_n; F; \Omega)],
\]

where the corrupting distribution $p(\tilde{x}_n|x_n)$ factorizes over $d$ dimensions.

Minimizing the expected loss value under the corruption model leads to learning with marginalized corrupted features and source predictors. The tractability of [3] depends on the choice of the loss function $L$ and the corrupting distribution.

Let us denote the vector of all the predictions for a given corrupted target instance $\tilde{x}_n$ by $f(\tilde{x}_n)$ and let $\Gamma$ be the corresponding parameter matrix to be learned. The quadratic loss can then be written as:

\[
\mathbf{L}(\mathcal{D}; \Omega, \Gamma) = \sum_{n=1}^{N} \mathbb{E} \left[ (\Omega^\top \tilde{x}_n + \Gamma^\top f(\tilde{x}_n) - y_n)^2 \right] = \sum_{n=1}^{N} \mathbb{E} \left[ \left( \Omega^\top \tilde{x}_n + \Gamma^\top f(\tilde{x}_n) - y_n^\top \right)^2 \right],
\]

where source predictions $f(\tilde{x}_n)$ can be seen as augmented corrupted features under the same distribution $p(\tilde{x}_n|x_n)$.

3Other losses have been tested as well; the minimization with the exponential loss requires a gradient descent technique, the logistic and hinge losses are approximated by the upper bounds; they will be included in the extended version.

4For the notation simplicity, we omit subscript $p(\tilde{x}|x)$ in $E[\cdot|p(\tilde{x}|x)]$. 
The corresponding closed form solution can be written as:

$$
\begin{align*}
\begin{bmatrix} \Omega^* \\ \Gamma^* \end{bmatrix}^T &= Q_F^{-1} P_F, \\
Q_F &= \sum_n E \begin{bmatrix} \tilde{x}_n \\ f(\tilde{x}_n) \end{bmatrix} \begin{bmatrix} \tilde{x}_n \\ f(\tilde{x}_n) \end{bmatrix}^T, \\
P_F &= \sum_n E \begin{bmatrix} \tilde{x}_n \\ f(\tilde{x}_n) \end{bmatrix} y_n^T .
\end{align*}
$$

Linear source classifiers. If source classifiers are known to be linear functions (Case 1 in Section 4), we can marginalize them out jointly with the corrupted features. If we know that $f(x) = A^TX$, then we can estimate their expectation on a corrupted instance $x$, $E(f(x)) = E(A^TX) = A^T E(X) = f(E(X))$ and variance $E[f(\tilde{x}_n)f(\tilde{x}_n)^T] = E[AX_n x_n^T A^T] = A E[x_n x_n^T] A^T$. Consequently, shows the diagrams of training MCFA with known linear and corrupted features and source predictions (MCFA). Figure 3 schematically shows the diagrams of training MCFA with known linear and black box classifiers.

4.3 Unsupervised Adaptation

The main drawback of supervised setting presented in the previous section is that it requires target labels and does not exploit the unlabeled target instances $X_n^t$, which is generally much larger set than $X_1^t$. In this section we address the unsupervised domain adaptation case by marginalizing corrupted features in the denoising autoencoder framework.

The stacked Marginalized Denoising Autoencoder (sMDA) is a version of the multi-layer neural network trained to reconstruct input data from partial random corruption [40] proposed by Chen et al. [7] where the random corruption is marginalized out yielding the optimal reconstruction weights in the closed form.

The basic building block of the method is a one-layer linear denoising autoencoder where a set of sample inputs $x_n$ are corrupted $M$ times by random feature dropout with the probability $p$ and reconstructed with a linear mapping $W: D^D \rightarrow D^D$ by minimizing the squared reconstruction loss $\mathcal{L}(W)$.

$$
\mathcal{L}(W) = \sum_{n=1}^N \sum_{m=1}^M \|x_n - W \tilde{x}_{nm}\|^2 .
$$

By applying the marginalization trick described for MCF, the mapping $W$ can be expressed in closed form as $W = E[P] E[Q]^{-1}$, where $E[Q]_{ij} = \begin{cases} S_{ij} q_{ij}, & \text{if } i \neq j, \\ S_{ij} q_{ij}, & \text{if } i = j, \end{cases}$ and $E[P]_{ij} = S_{ij} q_{ij}$, (9)

where $q = [1 - p, \ldots, 1 - p, 1] \in R^{D+1}, p$ is the dropout probability, $D$ is the feature dimension and $S = XX^T$ is the covariance matrix of the uncorrupted data $X$.

Stacking together several MDA layers can create a deep architecture where we feed the representations of the $(l-1)^{th}$ denoising layer as the input to the $l^{th}$ layer and learn the transformation $W^l$ to reconstruct the previous output from its corrupted equivalent. In order to extend the mapping beyond a linear transformation, between layers we apply on each output either a hyperbolic tangent function $h_l = \tanh(W^lh_{l-1})$ or rectified linear units (ReLU) $h_l = \max(W^lh_{l-1}, 0)$. We will denote the final output $h_l$ corresponding to the input $x_n$ by $x_n^t$.

The main advantage of sMDA is that it does not require class labels; hence we can take advantage of the unlabeled target data and apply it for unsupervised domain adaptation. Indeed, [7] applied sMDA to the union of target and source datasets in order to learn domain invariant features.

In our case, $X^S$ is unavailable, we have either access to source classifiers $F$ or a set of representative source examples. We are
interested in one particular case, when the representative examples are the class means for each domain $X_s$. Below we propose two different extensions of sMDA to these two cases.

### 4.4 Adaptation with Class Means

**Class Means.** If class means are available, they can be used both as classifiers and as data points. Used as classifiers, the class means can directly predict labels with a weighted softmax distance \[ \sum \frac{w_d e^{-||x_t - \mu_s^d||}}{\sum_{c'} \sum_d w_d e^{-||x_t - \mu_s^{d,c'}||}} \]

where $w_d$ is the domain weight. In unsupervised domain adaptation, the DSCM make predictions with source class means $\mu_s^d$. If some target instances are labeled, the classifier can additionally use the target class means $\mu_t^d$.

The Adapted Class Means (ACM) classifier is an extension of DSCM to unsupervised and semi-supervised settings. First, a sMDA is used to denoise the source class means. Instead of full source data $X^s$, the source class means $[\mu_1^d, \ldots, \mu_C^d]$ concatenated with the unlabeled target data $X^t$ are fed into the sMDA. The transformation matrix $W$ obtained by (9) (or the set of stacked transformation matrices $W^d$) is then used to jointly denoise the source class means $\mu_s^d$ and the target examples $\hat{x}_n$. The denoised class means $\mu_s^d$ are then used by the classifier to predict labels for denoised target data.

Algorithm 2 reports all steps for the case of a single source; the generalization to multiple sources is straightforward.

**Algorithm 2: Adapting Domain Specific Class Means (ACM).**

**Require:** A large set of unlabeled target dataset $X_t^s$ and optionally a small set of labeled $X_t^s$.

**Require:** A set of source class means $\mu_1^d, \ldots, \mu_C^d$.

1. Compute $W$ using (9) with $X = [\mu_1^d, \ldots, \mu_C^d, x_1^t, \ldots, x_N^t]$ and noise level $p$.
2. (Optionally) repeat $l$ times, while alternating with $h_i = \tanh(W^i h_{i-1})$.
3. Decompose $h_i$ into reconstructed means $\hat{\mu}_s^d$ and reconstructed targets $\hat{x}_n^t$. (Optionally concatenate $\hat{x}_n^t$ with the original features $x_n^t$ and $\hat{\mu}_s^d$ with $\mu_s^d$).
4. If labeled target examples then
   5. $p(c|x_t^s) = \frac{e^{-\parallel x_t^s - \hat{\mu}_s^{c} \parallel} + e^{-\parallel x_t^s - \hat{\mu}_s^{c'} \parallel}}{\sum_{c'} e^{-\parallel x_t^s - \hat{\mu}_s^{c'} \parallel}}$.
   6. Else
   7. $p(c|x_t^s) = \frac{e^{-\parallel x_t^s - \hat{\mu}_s^{c} \parallel}}{\sum_{c'} e^{-\parallel x_t^s - \hat{\mu}_s^{c'} \parallel}}$.
8. End if
9. Label $x_t^s$ with $c = \arg\max_c p(c|x_t^s)$.
10. Return Labels $x_t^s$.

for us, and we propose a Transductive Domain Adaptation (TDA) method which also relies on sMDA but uses it in the way different from the ACM.

Like in Section 4.1 we consider class predictions $f_k(x_n)$ as relevant but corrupted by the domain shift. Hence, we can exploit the correlation between the target data $x_n$ and source predictions $f_k(x_n)$ to reconstruct (or denoise) both the target data and the source classifiers predictions of these target data. We apply the sMDA to the augmented dataset $U = [x_1^t, \ldots, x_N^t]$ and compute $W$ or stack several layers interleaved with nonlinear functions. We denote by $u_n$ the final output corresponding to the input $x_n$.

In this case, we can directly use the reconstructed class predictions $f_k(x_n)$ to make the classification decisions. If the set $F$ includes a single source classifier $f_1$ with one prediction per class, the class with the maximum reconstructed value, $c^* = \arg\max_c u_n^{D+c}$ is assigned as label to $x_n$. In the case of multiple classifiers $f_k$ in $F$, the maximum of the averaged predictions should be considered. Algorithm 3 summarizes all the steps of this transductive version of domain adaptation.

**Algorithm 3: Transductive Domain adaptation with mSDA (TDA).**

**Require:** Target dataset $X_t^s \in R^{N \times D}$ without labels

**Require:** Source classifiers $f_k \in R^D \rightarrow [0,1]^C$.

1. Generate class predictions $f_k(x_n^t)$ for all $x_n^t \in X^t$.
2. Compose an augmented dataset $U$ with $u_n^t = [x_n^t; f(x_n^t)]$.
3. Use sMDA to estimate $W = m\arg\min_W \parallel U - WU \parallel^2$.
4. Get denoised class predictions for $x^t$ as $y^t = \sum_w W_{:,D+1...C}f(x)$.
5. Label $x_t^s$ with $c^* = \arg\max_c \{y^t_n | y^t \}^C$.
6. Return Labels for $x^t$.

### 5. EXPERIMENTAL RESULTS

In this section we present a series of experiments to cope with domain adaptation tasks, along several important dimensions:

1. The amount of supervision in target domain; we consider unsupervised, supervised and semi-supervised settings.
2. Information available from the sources: source linear classi-
fer, source classifier as a black box, and source representatives in the form of class means.

3. When the classifier is a black box, we test different classification models, including common ones like SVM and Logistic Regression, and specific ones like MCF and DSCM.

We recall briefly all algorithms proposed in the previous sections:

**MCFA** (Section 4.2) is primarily supervised domain adaptation algorithm described in Algorithm 1. It includes two versions, **mcfalf** when the source classifiers \( f_s(x) \) are linear, and **mcfab** when \( f_s(x) \) are used as a black box to predict class labels for a target example.

**ACM** (Section 4.4) assumes that class means are available from the source data. In the unsupervised setting, it uses the sMDA reconstructions of the class means to predict the class. In the semi-supervised setting, the target class means (\( \mu_t \)) are learned from labeled target data; then, together with the source class means (\( \mu_s \)), they are used to predict the class labels for target data (see Algorithm 2).

**TDA** (Section 4.5) is the transductive algorithm to denoise source class predictions with target data in unsupervised setting (see Algorithm 3).

The experiments are organized by the supervision type (unsupervised, supervised and semi-supervised); they aim to test performance of all algorithms on different datasets. Below we describe two image and one text datasets used in the experiments.

### 5.1 Datasets and Experimental Framework

**XS3.** This collection is a part of the Xerox’s Smart Document Management data; it contains printed/scanned documents from three Xerox customers denoted by \( C_1, C_2 \) and \( C_3 \). The three datasets contain different types of documents such as invoices, contracts, IDs, filled forms, etc. All the documents have been grouped into 21 different classes, all well represented in each dataset (see examples in the Figure 5). Amongst these classes, some refer to generic concepts such as printed emails, computer programs, drawings, handwritten letters; some others (contracts and filled forms) form a more fine-grained class hierarchy, according to the customers’ requirements. We manually aligned between the fine-grain classes have been manually aligned by their visual similarities as the correspondences is often not obvious.

The number of documents per class and per customer varies from 24 to 859; around 2.5K documents are available for each customer. As image representations we use deep convolutional activation features \[16\]. These features are obtained using publicly available Caffe CNN model \[27\] trained on the 1000 classes of ImageNet \[25\] and fine-tuned for document images on the RVLC-DIP document image dataset \[26\]. In our experiments we use the fully connected layer (caffe_fc6) of dimensionality 4096.

Due to the confidentiality clauses on XS3 collection, we also run experiments on two publicly available benchmark datasets:

**OFF31.** One of the most popular datasets for comparing visual domain adaptation methods is the Office31 collection \[36\]. It consists of four domains: Amazon (A), Caltech (C), dslr (D) and Webcam (W) with images of 31 products (classes). Each domain is considered in its turn as a target, with the other domains considered as sources. For the target set we select 3 instances per class to form the training set, all other instances form the test set. We use the caffe_fc6 deep convolutional activation features \[16\] we obtain with the publicly available VGGNET (16 layer) CNN model \[37\] with no fine-tuning.

**AMT.** The Amazon text dataset consists of products reviews in four domains: Books (b), Dvd (d), Kitchen (k) and Electronics (e). While a book review can be quite different from a kitchen item review, there are nevertheless some common features to assess whether the users were satisfied with their purchase. A part of this collection was preprocessed by Blitzer et al. \[3\] and used subsequently in several domain adaptation studies. The task is to predict whether a customer review is positive or negative, where a review with more than 3 stars is considered as positive and (strictly) less than 3 as negative. Documents are represented by a bag of unigrams and bigrams. In our experiments, we only considered the top 10,000 most-frequent features for each domain. In all experiments, we use the whole source and target collections whereas other studies use a predefined split of 2000 documents per domain.

Finally, we list four classification models tested as source classifiers:

- **DSCM** classifier \[12\] uses a softmax distance to the class specific means to classify a test example.
- **MCF** \[31\] is a multi-class linear classifier with dropout noise probability fixed to 0.5 for image datasets (OFF31 and X3C3) and 0.9 for text dataset (Amazon). Need for a stronger dropout level on text data is explained by its sparsity \[12\].
- **LR** refers to a \( l_2 \) regularized Logistic Regression, cross-validated with the regularization parameter \( C \), where \( C \in [0.0001, 0.001, 0.1, 1, 10, 50, 100] \).
- **SVM** is used with the RBF kernel, two main hyperparameters \( C \) and \( \gamma \) are fit by a random search, \( C \sim \text{Exp}(\text{scale} = 100) \) and \( \gamma \sim \text{Exp}(\text{scale} = .1) \). For both SVM and LR, the Scikit learn implementation is used \[4\].

### 5.2 Unsupervised Setting

In this setting, the baseline consists in classifying target data without any domain adaptation. We compare the performance of the unsupervised ACM with available class means to the TDA with the black box classifier.

![Figure 5: Examples from the XC3 datasets. Each line corresponds to one customer and each column refers to one document class (documents are intentionally blurred for privacy reasons).](https://http://scikit-learn.org/stable/index.html)
Table 1: Unsupervised adaptation in XS3 and OFF31 datasets with DSCM and TDA methods. Underline indicates improvement over the baseline and bold indicates the best performance per task.

<table>
<thead>
<tr>
<th>$f_s$</th>
<th>dscm</th>
<th>mcf</th>
</tr>
</thead>
<tbody>
<tr>
<td>$f_t$</td>
<td>smda$^*$</td>
<td>$f_s(X^t)$</td>
</tr>
<tr>
<td>$S \rightarrow T$</td>
<td></td>
<td></td>
</tr>
<tr>
<td>$C_2 \rightarrow C_1$</td>
<td>37.18</td>
<td>50.75</td>
</tr>
<tr>
<td>$C_3 \rightarrow C_1$</td>
<td>37.58</td>
<td>39.24</td>
</tr>
<tr>
<td>$C_1 \rightarrow C_2$</td>
<td>53.92</td>
<td>47.59</td>
</tr>
<tr>
<td>$C_3 \rightarrow C_2$</td>
<td>62.16</td>
<td>57.92</td>
</tr>
<tr>
<td>$C_1 \rightarrow C_3$</td>
<td>36.11</td>
<td>31.31</td>
</tr>
<tr>
<td>$C_2 \rightarrow C_3$</td>
<td>38.64</td>
<td>30.09</td>
</tr>
<tr>
<td>$D \rightarrow A$</td>
<td>60.18</td>
<td>56.91</td>
</tr>
<tr>
<td>$W \rightarrow A$</td>
<td>60.07</td>
<td>55.74</td>
</tr>
<tr>
<td>$A \rightarrow D$</td>
<td>69.88</td>
<td>71.36</td>
</tr>
<tr>
<td>$W \rightarrow D$</td>
<td>95.56</td>
<td>97.04</td>
</tr>
<tr>
<td>$A \rightarrow W$</td>
<td>65.95</td>
<td>64.1</td>
</tr>
<tr>
<td>$D \rightarrow W$</td>
<td>93.16</td>
<td>93.59</td>
</tr>
</tbody>
</table>

Table 1 reports results for 6 domain adaptation tasks in XS3 and 6 tasks in OFF31 dataset. Each line indicates an adaptation task $S \rightarrow T$, from source $S$ to target $T$. Column 1, smDA$^*$, refers to the dream case of fully available source data; it shows classification accuracy of smDA on full source and target data. Columns 3 to 5 correspond to cases where the source classifier $f_s$ is a DSCM classifier. Column 3 reports the baseline accuracy when the source classifier is applied on the target test without adaptation, $f_s(X^t)$. Columns 4 (ACM) and 5 (TDA) show the results of applying the respective domain adaptation. Both methods in general outperform the baseline and achieve the accuracy values comparable with the dream case with full access to source data, and in general ACM outperforms TDA, showing that it is more interesting to adapt the class means (if available) than the softmax scores we obtain with.

Finally, columns 6 and 7 show the classification accuracy when the target data get combined with the source predictions. Both methods in general outperform the baseline MCF, showing that it is more interesting to adapt the class means classifier than the source classifier's predictions to the original target features brings an improvement over the baseline and is often close to the dream case. To sum up, domain adaptation with TDA is successful when the classifier is used in the black-box ($f_s$), that is trained on the concatenated features. Both methods in general outperform the baseline and achieve the accuracy values comparable with the dream case $sMDA^*$. TDA always improves over the baseline and is often close to the dream case.

Table 2: Unsupervised adaptation in AMT dataset with TDA method and MCF/LR source classifiers. Underline indicates improvement over the baseline and bold indicates the best performance per task.

<table>
<thead>
<tr>
<th>$f_s$</th>
<th>mcf</th>
<th>lr</th>
</tr>
</thead>
<tbody>
<tr>
<td>$f_t$</td>
<td>smda$^*$</td>
<td>$f_s(X^t)$</td>
</tr>
<tr>
<td>$S \rightarrow T$</td>
<td></td>
<td></td>
</tr>
<tr>
<td>$d \rightarrow b$</td>
<td>80.81</td>
<td><strong>80.08</strong></td>
</tr>
<tr>
<td>$e \rightarrow b$</td>
<td>73.39</td>
<td>71.22</td>
</tr>
<tr>
<td>$k \rightarrow b$</td>
<td>74.93</td>
<td>71.17</td>
</tr>
<tr>
<td>$d \rightarrow e$</td>
<td>81.96</td>
<td>80.31</td>
</tr>
<tr>
<td>$e \rightarrow d$</td>
<td>75.91</td>
<td>74.20</td>
</tr>
<tr>
<td>$k \rightarrow d$</td>
<td>77.47</td>
<td>76.03</td>
</tr>
<tr>
<td>$b \rightarrow e$</td>
<td>77.47</td>
<td>74.53</td>
</tr>
<tr>
<td>$d \rightarrow e$</td>
<td>79.02</td>
<td>76.03</td>
</tr>
<tr>
<td>$k \rightarrow e$</td>
<td>87.16</td>
<td>86.28</td>
</tr>
<tr>
<td>$b \rightarrow k$</td>
<td>80.45</td>
<td>76.86</td>
</tr>
<tr>
<td>$d \rightarrow k$</td>
<td>81.44</td>
<td>77.64</td>
</tr>
<tr>
<td>$e \rightarrow k$</td>
<td>88.59</td>
<td>87.72</td>
</tr>
</tbody>
</table>

Experiments with SVM give very similar results.

Table 2 presents evaluation results on XS3 and OFF31 datasets. Each column in the table shows the accuracy of the target classification using linear classifiers (Column 1) and the black box classifiers (Columns 2 to 4). For all tasks, the first supervised baseline is denoted as $T \rightarrow T$ (for ex., $C_1 \rightarrow C_1$), it refers to learning from labeled target data only. The second baseline is reported in the last column, it uses LR as the target classifier ($f_t$) that is trained on the target features concatenated with the source classifier predictions obtained with DSCM ($f_s$).

We can observe that both MCF extensions outperform the baseline LR trained on the concatenated features. Given the small amount of labeled data, the MCF is more appropriate to handle the noise and variability in data, which makes it better suited for domain adaptation and transfer tasks. We can see that MCF from SVM or DSCM predictions provide even better results and outperform both baselines; this witnesses a successful adaptation independently of which classifier is used in the black-box ($f_s$).

We have also conducted experiments on the AMT dataset where 10 target labels per class were used to compare several source classifiers. We do not include results here but report the main findings. MCF shows a better classification performance than LR when using the target features only (similarly to the image datasets). However, when the target data get combined with the source predictions, the LR baseline obtains better results than MCF. Adding the source classifier’s predictions to the original target features brings an improvement over the supervised $T \rightarrow T$ baseline, for both LR and MCF. However, due to a small amount of target labels, the $T \rightarrow T$ baseline is worse than the $S \rightarrow T$ one, and none approach is able to outperform the results obtained with the source classifier’s predictions ($f_s$).

On the other hand, as shown in Table 2 combining the source predictions with the unlabeled target data outperforms significantly the results obtained with source predictions ($f_s$). It points out that using the source predictions and the unsupervised denoising with TDA is a much better option than using any supervised method.

5.3 Supervised Setting

We assume now that some of the target instances are labeled, but no unlabeled target instances are given at the training time. For each adaptation task, there are two baselines. The first one consists in training the classifier on the labeled target instances; the second one relies on the logistic regression model trained on the concatenated of the raw target features and the source predictions. Both baselines are shown in italic in the tables. We run experiments with different source classifiers (MCF, SVM, DSCM) and evaluate the two proposed MCF extensions: MCFAl and MCFAb.
including the baselines. This clearly demonstrates the importance of having access to the unlabeled target data. It shows that the unsupervised domain adaptation can be highly beneficial for the accuracy gain and can considerably reduce the annotation cost.

5.4 Semi-supervised Setting

Lastly, we consider here a transductive semi-supervised setting. We assume that a large set of unlabeled target set $X^t_{\mathcal{U}}$ is available at the training time. The supervised baseline which uses only target data and our algorithm MCFaB are used as baselines. They are compared to the semi-supervised version of ACM, which assumes available source class means. Despite this requirement, it is still interesting to compare these approaches to understand which classifiers can be more amenable to transfer.

Table 3 shows the evaluation results for 12 domain adaptation tasks in XS3 and OFF31 datasets. For each adaptation task, the supervised baseline is denoted $T \rightarrow \mathcal{T}$; it trains a classifier on target labeled data only. Column 2 in the table reports the results of DSCM classifier in the supervised setting. Column 3 indicates the performance of sMDA with a DSCM classifier with full access to source data. The sMDA results help us measure the performance or adaptation methods under the limited or no access to the source data. Column 4 indicates the performance of the ACM model and Column 5 reports the supervised baseline MCFaB. The table reveals two important findings. First, the semi-supervised ACM brings additional gains comparing to the baselines. But the most important finding is that methods with access to source data (ACM, MFCaB) achieve performances close to the methods with access to source data (sMDA+DSCM, Column 2). This indicates that the classifier contains enough information for adaptation and upholds the result in unsupervised adaptation.

6. CONCLUSION

In this paper we address the problem of domain adaptation in real world applications, where the reuse of source data is limited, due to legal and contractual obligations, to classification rules or a few representative examples. We adapt the recent techniques of feature corruption and their marginalization developed for the supervised (MCF) and unsupervised (sMDA) settings. We propose several extensions to MCF and sMDA methods in order to address cases when the source classifiers are available either as a black box or with known parameters, or a few class representatives are from a source domain. We conduct a series of experiments on two public and one in-house datasets in supervised and unsupervised settings. We present the evaluation results and show that significant performance gains can be achieved despite the absence of source data and shortage of labeled target data. This result is an highly encouraging for a range of applications with machine learning components, whose adaptation to new domains and customers is expected to be low and without sacrificing the stated performance.

7. REFERENCES


