Adapted Domain Specific Class Means

Gabriela Csurka, Boris Chidlovskii and Stéphane Clinchant
Xerox Research Centre Europe
6 Rue de Maupertuis, 38240 Meylan, France
Firstname.Lastname@xrce.xerox.com

Abstract

We address the problem of domain adaptation (DA) from one or multiple source domains to a target domain. Most of the existing DA methods assume that source data is largely available. Such an assumption rarely holds in real applications, for both technical and legal reasons. More realistic are situations where source domain observations become quickly unavailable, but only some domain representatives can be retained, either as source instances or as their aggregation. In this paper therefore we focus on the Domain Specific Class Means (DSCM) classifier [5] that can handle such scenario and we combine it with the sMDA framework [4]. We show, on a variety of datasets and tasks, that the method can be applied successfully even when no labeled target is available and also that it can provide performance comparable to the case where dense knowledge (all source data) is available.

1. Introduction

Domain adaptation addresses the problem of leveraging labeled data in one or more related domains, often referred as "source" domains, when learning a classifier for unseen data in a "target" domain. The domains are assumed to be related but not identical and when we apply directly models learned on the source domain the performance can be often poor due to a frequent domain shift. However in general these sources can still bring useful information to build classifiers in the target domain especially when no labels are available in the target domain.

The majority of domain adaptation methods [7, 18, 11, 14, 2, 12, 1, 9, 10, 17] search to compensate this mismatch by making use of information coming from both source and target domains during the learning process; they learn or adapt automatically the classifiers to the target either by exploiting also labeled target examples (known as semi-supervised DA) or assuming that the target domain data is fully unlabeled (unsupervised DA).

In contrast to these approaches, in this paper we assume that we do not have access to the actual source data, but we own only an aggregation of the data for each class (class means). We propose a new method that combines these class means with the unsupervised Stacked Marginalized Denoising Autoencoders (sMDA) [4] to solve both unsupervised (US) and semi-supervised (SS) domain adaptation tasks. The main idea is that the proposed method exploits the correlation between the target data and source prototypes without the need of target labels and automatically adapts these class means to the target dataset.

The main advantages of the proposed approach are the followings:

1) The method is has low computational cost with results comparable to many complex state-of-the art methods. Furthermore it is easy to retrain the models if updated or new class means or new sources became available.

2) In contrast to many DA methods in the literature that assumes single source versus target scenarios, the method can exploit multiple independent sources to predict labels for new target examples. Furthermore, the method requires only the class means from the different sources that can be precomputed beforehand and independently for each source, in contrast to methods which need to access simultaneously the whole source data to train models for the target.

3) The method can be applied in applications where customers care about their privacy issues (*e.g.* document images, medical field, etc) and are not ready to share the whole source data only class averages from which it is not possible to reconstruct the original documents.

4) The method has only a few parameters. While fine-tuning them yields better performances, it is possible to select reasonable prefixed values based on the data type and the document representation used.

2. The proposed method

The aim of the proposed method is to use the sMDA[4] framework to adapt directly the Domain Specific Class Means (DSCM) classifiers. Therefore, we will call the proposed method *Adapted Class Means* and denote it by ACM.

The DSCM [5] classifier considers the means

$$\boldsymbol{\mu}_d^c = \frac{1}{N_d^c} \sum_{x_i \in \mathcal{D}_d \cap \mathcal{C}_c} \boldsymbol{x}_i, \tag{1}$$

where N_d^c is the number of instances from class $\mathcal{C}_c$ in the domain $\mathcal{D}_d$, to predict the class label of an unlabeled target instance, based on a weighted softmax distance to these domain-specific class means:

$$p(c|\boldsymbol{x}_i) = \frac{1}{Z_i} \sum_{d \in D} w_d e^{(-\frac{1}{2}\|\boldsymbol{x}_i - \boldsymbol{\mu}_d^c\|)},$$

where $Z_i = \sum_{c'} \sum_d w_d e^{(-\frac{1}{2}\|\boldsymbol{x}_i - \boldsymbol{\mu}_d^{c'}\|)}$ is the normalizer, and w_d are the domain weights[1].

The *stacked marginalized Denoising Autoencoder* (sMDA) is a version of the multi-layer neural network trained to reconstruct input data from partial random corruption [19] proposed by Chen *et al*. in [4] 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 $\boldsymbol{x}_1, \ldots, \boldsymbol{x}_m$ are corrupted n times by random feature removal with the probability p (denoted by $\tilde{\boldsymbol{x}}_{ij}$ the j^{th} corrupted version of the input $\boldsymbol{x}_i$) and reconstructed with a linear mapping $\mathbf{W} : R^d \rightarrow R^d$ by minimizing the squared reconstruction loss[2] :

$$\mathcal{L}(\mathbf{W}) = \frac{1}{2mn} \sum_{j=1}^{n} \sum_{i=1}^{m} ||\mathbf{x}_i - \mathbf{W}\tilde{\mathbf{x}}_{i,j}||^2. \tag{2}$$

We design the matrix $\mathbf{X} = [\mathbf{x}_1, \ldots, \mathbf{x}_m]$ of inputs, its m-times repeated version as $\bar{\mathbf{X}} = [\mathbf{X}, \ldots, \mathbf{X}]$ and the corresponding corrupted version by $\tilde{\mathbf{X}} = [\tilde{\mathbf{x}}_{11}, \ldots, \tilde{\mathbf{x}}_{mn}]$, the solution of (2) reduces to

$$\mathcal{L}(\mathbf{W}) = \frac{1}{2mn} tr\Big[\big(\bar{\mathbf{X}} - \mathbf{W}\tilde{\mathbf{X}}\big)^\top \big(\bar{\mathbf{X}} - \mathbf{W}\tilde{\mathbf{X}}\big)\Big]. \tag{3}$$

The solution of (3) can be expressed as the closed-form solution $\mathbf{W} = \mathbf{P}\mathbf{Q}^{-1}$ from ordinary least squares, where $\mathbf{Q} = \tilde{\mathbf{X}}\tilde{\mathbf{X}}^T$ and $\mathbf{P} = \bar{\mathbf{X}}\tilde{\mathbf{X}}^T$.

While the solution $\mathbf{W}$ depends on the corrupted sample inputs $\tilde{\mathbf{x}}_{11}, \ldots, \tilde{\mathbf{x}}_{mn}$, it was shown in [4] that by the weak law of large numbers, the matrices $\mathbf{P}$ and $\mathbf{Q}$ converge to their expected values $\mathbb{E}[\mathbf{P}]$ and $\mathbb{E}[\mathbf{Q}]$ as more copies of the corrupted data are created (letting $n \rightarrow \infty$). In the limit, the corresponding mapping for $\mathbf{W}$ can be expressed in closed form as $\mathbf{W} = \mathbb{E}[\mathbf{P}]\,\mathbb{E}[\mathbf{Q}]^{-1}$, where:

$$\mathbb{E}[\mathbf{Q}]_{ij} = \begin{cases} \mathbf{S}_{ij} q_i q_j, & \text{if} \quad i \neq j, \\ \mathbf{S}_{ij} q_i, & \text{if} \quad i = j, \end{cases} \qquad \mathbb{E}[\mathbf{P}]_{ij} = \mathbf{S}_{ij} q_j \tag{4}$$

[1] We used $w_s = 1$ for sources and $w_s = 2$ for target in the experiments.

[2] A constant is added to the input, $\mathbf{x_i} = [\boldsymbol{x}_i; 1]$, and an appropriate bias incorporated within $\mathbf{W}$ which is is never corrupted.

with $q = [1-p, \ldots, 1-p, 1] \in R^{f+1}$, p being the noise level, f the feature dimension and $\mathbf{S} = \mathbf{X}\mathbf{X}^T$ being the covariance matrix of the uncorrupted data $\mathbf{X}$.

We can create a *deep* architecture by stacking together several MDA layers where we feed the representations of the $(t-1)^{th}$ denoising layer as the input to the t^{th} layer and learn the transformation $\mathbf{W}^t$ 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 tangent-hyperbolic nonlinearities $\mathbf{h}_t = \texttt{tanh}(\mathbf{W}^t\mathbf{h}_{t-1})$ or alternatively rectified linear units (RELU) $\mathbf{h}_t = max(\mathbf{W}^t\mathbf{h}_{t-1}, 0)$. We will denote the final output $\mathbf{h}_t$ corresponding to the input $\mathbf{x}_t$ by $\hat{\mathbf{x}}_t$.

The main advantage of sMDA is that the loss in (2) does not require class labels and hence we can take advantage of the unlabeled target data and successfully apply it for unsupervised domain adaptation. Indeed, by concatenating the target and source dataset to form $\mathbf{X}$, sMDA learns correlations between the source and target features and hence the reconstructed features will better describe what is common between the source and the target set [4].

In our case we assume that we do not have access to the source data, but we claim that we can instead still learn from the correlation between the source class representatives $\mathbf{X}_S = [\mu_{s_1}^{c_1}, \ldots \mu_{s_S}^{c_C}]$, with $c_i \in C$ and $s_d \in D \setminus \{t_d\}$ and the target instances $\mathbf{X}_T = [\mathbf{x}_{t_1}, \ldots, \mathbf{x}_{t_n}]$ to improve DSCM accuracies. The main idea is therefore to consider these class means $\mu_{s_d}^{c_i}$ as the inputs features from the source and together with the target instances $\mathbf{x}_{t_j}$ to form our uncorrupted $\mathbf{X} = [\mathbf{X}_T, \mathbf{X}_S]$ input of the sMDA framework. Then we can compute W using (4) to reconstruct both the source class means $\hat{\mu}_{s_d}^{c_i}$ and the target examples $\hat{\mathbf{x}}_{t_j}$.

To predict class labels for the reconstructed unlabeled target examples $\hat{\mathbf{x}}_t$ we can use the DSCM classifier with the reconstructed class means $\hat{\mu}_{s_d}^{c_i}$. Or alternatively, we can concatenate the original features with the reconstructed outputs (both for the class means and target examples) and apply DSCM in this concatenated feature space. We opted for the latter as it generally outperform the former. In the semi-supervised (SS) case we compute also class means from the labeled target examples ($s_d \in D$) in the concatenated space and use in them in the DSCM.

We can also generalize the method, by considering different noise level for the target instances and class means. If we denote the fractions of source class labels and of target instances by $\alpha_t = \frac{n_t}{n_t+n_s}$ and $\alpha_s = \frac{n_s}{n_t+n_s}$, where n_t is the number of target instances, $n_s = S \cdot C$, and the two noise vectors $q_t = [1-p_t, \ldots, 1-p_t, 1] \in R^{n_t}$ and $q_s = [1-p_s, \ldots, 1-p_s, 1] \in R^{n_s}$, the expectations in (4) become:

$$\begin{aligned} \mathbb{E}[\mathbf{Q}]_{ij} &= \begin{cases} \mathbf{S}_{ij}(\alpha_t q_{ti} q_{tj} + \alpha_s q_{si} q_{sj}), & \text{if } i \neq j \\ \mathbf{S}_{ij}(\alpha_t q_{ti} + \alpha_s q_{si}), & \text{if } i = j. \end{cases} \\ \mathbb{E}[\mathbf{P}]_{ij} &= \mathbf{S}_{ij}(\alpha_t q_{tj} + \alpha_s q_{sj}) \end{aligned}$$

Table 1. Notations used in the experimental section.

US/SS	unsupervised/semi-supervised setting
ase/sse	mean with all data/sub sampling
BC/AS	best source configuration/all sources
sMDAt/sMDAr	tangent-hyperbolic/RELU
sMDA*/sMDA^{pf}	best noise per configuration/fixed noise
nn.nn/nn.nn	best method/advised fixed parameters

Table 2. Results using BOV features.

Dataset	DSCM	ACMt*	ACMt^f	ACMr*	ACMr^f
OFF31 *ase* US	26.18	**26.95**	**26.81**	25.62	25.09
OFF31 *ase* SS	45.6	**46.31**	**46.24**	45.25	44.77
OC10 *ase* US	49.84	49.01	**48.25**	**51.36**	49.96
OC10 *ase* SS	55.31	**58.63**	**56.86**	57.82	55.84

3. Datasets

OFF31 and OC10: The Office31 and Office+Caltech10 are two datasets frequently used for testing DA techniques [18, 12, 14, 8, 1]. The Off31 dataset consists of three domains: *Amazon* (A), *Dslr* (D) and *Webcam* (W) and 31 classes, while in OC10 only 10 classes were kept but added a new domain *Caltech* (C). Each domain is considered in its turn as a *target*, with the others as *sources*. In our experiments we use either the provided Bag-of-visual words (BOV) built on SURF features or deep convolutional activation feature [6]. The latter features are obtained with the publicly available Caffe (8 layer) CNN model[3] [16] trained on the 1000 classes of ImageNet used in the ILSVRC 2012 challenge. We use the activation features from the 6^{th} (fully connected) layer; we will refer to it as fc6.

3VC. The 3 Virtual Customer dataset is an in house collection of document images, where we distinguish different types of documents such as forms, invoices, handwritten and typed letters, checks, bank statements, drawings, reports, etc. We have in total 21 classes with variable number of documents per class (from 24 to 859). On these document images we extracted the fc6 features of a similar (8 layer) CNN model as above, but the model was trained on the TobaccoBig dataset of 400K document images [15].

AMT. The Amazon text dataset consists of products reviews with 2 classes (positive and negative) represented by TFIDF normalized bag-of-words, used in several studies for domain adaptation [3]. We only considered the most frequent 10,000 features and the domains used in most studies: *kitchen* (K), *dvd* (D), *books* (B) and *electronics* (E). We have roughly 5,000 documents for each class per domain.

4. Validation framework

In this section we want to make a few general remarks and explain our notations (resumed in Table 1):

Feature normalization. As most DA papers, we do z-score normalization of the features except for text where the classical TFIDF normalization was used. However, in contrast to those methods, as we handle each domain independently, the z-score normalization is also done for each domain individually and hence all results including the baseline are obtained with these z-score normalized features.

[3] https://github.com/BVLC/caffe

Table 3. OC10 *ase* (SS) results per target using BOV features.

T	DSCM T→T	ACMt^f T→T	DSCM BC→T	ACMt^f BC→T	DSCM AS→T	ACMt^f AS→T
A	43.86	44.94	53.56	**54.09**	50.	**50.22**
C	33.49	33.58	**42.82**	42.09	40.44	**42.45**
D	59.8	59.84	**72.44**	70.87	67.72	**69.29**
W	58.49	60	76.98	**78.49**	72.45	**73.21**

Table 4. Comparison with the SOA on OC10 using BOV features.

T	ACMt^f	GFK [13]	SA-NN [8]	DIP-CC [1]
OC10 *sse* (US)	42.1	42.5	44.2	50.79
OC10 *sse* (SS)	54.5	48.5	53.66	63.7

sMDA settings. We consider 5 layers for the sMDA and the output of the fifth layer is concatenated with the original features to build the ACM classifiers. Between layers we apply either a tangent-hyperbolic nonlinearities (denoted by ACMt) or the rectified units (denoted by ACMr). We consider two main cases. On one hand, for each configuration we perform a grid search varying the noise level from 0.05 to 0.95 with step 0.05 and keep the best value on the grid. The corresponding results are denoted by ACMr* and ACMt* respectively. In addition we perform experiments with fixed noise for all configurations and datasets given a feature type and denote the corresponding results by ACMr^f and ACMt^f. In the tables we show best results in bold face and in red the values of the results obtained with the suggested fix parameters.

Multiple source configurations: In general, given a dataset we shows results averaged over all domain adaptation tasks. Thus, for OFF31, the classification accuracies are averaged over the following 9 tasks: D→A, W→A, (D,W)→A, A→D, W→D, (A,W)→D, A→W, D→W and (A,D)→W, while for OC10 we have an average over C→A, D→A, . . ., (A,C,D)→W, in total 4*7=28 tasks. We will denote by AS the case when all sources are considered *e.g.* (A, C, D) for the target T and by BC the best found configuration amongst all except AS.

Source data size. We distinguish three cases depending on the amount of source examples used to pre-compute the class means. We denote by *ase* the case when all source examples are used to build the class means and by *sse* when we consider only a subsample 20 examples (except for D or W as source, where as in the literature we selected only 8) for each class given a source domain. The latter is repeated 10 times and the averaged results are reported.

Multiple noises. We made a preliminary experiments using different noise level for the source class means and the target data, but observed only non-significant improvement of the results when we select the best multi-noise combinations per configuration. Furthermore we could not find fixed

Table 5. Results using fc6 features on OFF31 and OC10.

Dataset	DSCM	ACMt*	ACMt^f	ACMr*	ACMr^f
OFF31 *ase* (US)	77.72	80.59	78.4	**82.46**	**80.61**
OFF31 *ase* (SS)	85.07	86.83	85.99	**87.19**	**86.54**
OFF31 *sse* (US)	75.77	79.03	76.32	**81.21**	**79.73**
OFF31 *sse* (SS)	83.55	85.11	84.34	**85.72**	**85.23**
OC10 *ase* (US)	93.38	96.02	93.17	**96.25**	**94.95**
OC10 *ase* (SS)	90.08	**96.4**	94.3	96.17	**94.94**
OC10 *sse* (US)	92.49	**96.07**	92.52	95.94	**93.89**
OC10 *sse* (SS)	93.75	**95.73**	94.03	95.67	**94.3**

Table 6. OC10 *sse* (SS) results per target using fc6 features.

T	DSCM T→T	ACMt^f T→T	DSCM BC→T	ACMt^f BC→T	DSCM AS→T	ACMt^f AS→T
A	93.34	93.91	94.99	95.23	95.15	**95.36**
C	84.51	86.9	90.33	91.15	90.7	**91.73**
D	94.02	92.52	97.32	**97.64**	97.09	**97.56**
W	90.79	90.42	97.47	98.15	97.66	**98.23**

noise pairs for (ACM^f) that performed better over all configuration given a dataset that a single noise level. Therefore, we only show classification results where we used a global noise level, *i.e*. $\mathbf{W}$ is computed with (4).

5. Experimental results

BOV features. We experimented with BOV as OFF3 and OC10 are provided with these features and many papers are still report results on them. In the Table 2 we show results[4] averaged over all configurations with the *ase* setting. First we can see that using ACMr performs in general less well (except OC10 US) than using ACMt with the BOV features and ACMr results are sometimes below the baseline DSCM. On the contrary, ACMt outperforms DSCM even with a fixed noise level (wee used $p_f = 0.1$) and with setting appropriate noise level for each configuration can further improve the average accuracy.

In Table 3, we show results for individual target domains in the case of OC10 *ase* (SS) using ACMt^f with $p_f = 0.1$. We compare DSCM and ACMt results when using only the target training set (T→T), the best configuration (BC→T) and all source domains (AS→T). We can see that while best results are often obtained with some specific configuration choice, considering all available sources domains (AS) is a good compromise and still allows for significant improvements over the T→T with or without sMDA.

In Table 4 we compare our method to a few recent state-of the art methods on OC10. We show the average results over the 12 source-target pairs. We can see that except for DIP-CC[1] our methods performs not too compared to GFK and SA in spite of the fact that we use only the class means from the source and prefixed parameters.

Results with fc6 features. From the results with fc6 features shown in in Table 5 we can see that with these deep features using ACMr performs better than ACMt, especially with fixed noise level (we used $p_f = 0.5$) and compared to the baseline, ACMr^f, in average and most often, we have significant improvements with both of them. Finally, we can see that using class means obtained even with a small set of source examples (*sse*) allows to get results close (*ase*) suggesting that we do not need necessarily a large amount of source examples to get useful class prototypes.

If we analyze the ACMr^f per target results shown in Table 6 and compare them with DSCM with or without sMDA, we can see again a significant gain with ACM. Furthermore with the fc6 features best results are obtained when all the sources domains (AS) are considered.

Note that while we are not in the same application scenario than the state of the art methods which assumes the access to the whole source data when they learn the target classifiers, ACM can be obviously applied in that case too. We obtain on OFF31 (*ase*) SS with ACMr^f using fc6 for (A→W) an accuracy of 93.3% and for (D→W) 94.59%. Best results reported in [6] using SVM with fc6 are 82.14% for (A→W) and 94.79% for D→W respectively.

On the US setting, best accuracy reported in [10] are 73% for (A→W) and 96.4% for D→W while we with our ACMr^f obtain 75.21% for (A→W) and 92.59% (D→W).

Finally, Tsable 7 we compare our results on OC10 (*ase*) US with ATTM (Adaptive Transductive Transfer Machines) [17] which is one of the current SOA obtained on this dataset. Note that again, in contrast to us, they need the whole source dataset to train their model.

Table 7. ACM compared to ATTM using fc6.

$S \rightarrow T$	DSCM	ACMr^f	ATTM[17]
C →A	**95.26**	95.15	92.17
D →A	90.3	**94.07**	91.65
W →A	92.24	**94.83**	92.27
A →C	89.48	**91.03**	86.55
D →C	87.56	**90.85**	82.28
W →C	89.02	**91.77**	83.44
A →D	90.55	**91.34**	90.45
C →D	93.7	**94.49**	92.99
W →D	**100**	**100**	**100**
A →W	87.17	**92.45**	89.15
C →W	91.7	**93.21**	90.84
D →W	97.74	98.87	**98.98**
Average	92.06	**94**	90.9

Table 8. Results using fc6 features on the 3VC dataset.

Dataset	DSCM	ACMt*	ACMt^f	ACMr*	ACMr^f
3VC *ase* (US)	44.07	**48.2**	39.28	47.08	**45.85**
3VC *ase* (SS)	90.37	94.33	92.69	**94.77**	**93.68**
3VC *sse* (US)	42.99	47.2	40.02	**48.94**	**45.48**
3VC *sse* (SS)	89.61	**93.92**	92.28	93.58	**92.6**

Results on the 3VC dataset. In Table 8 we compare DSCM with ACM using different source settings with $p_f = 0.5$ on the fc6 features extracted from the 3VC document image dataset. We can observe again that ACMr outperforms ACMt consistently with the fixed noise level and in general is better with the optimal configuration.

Results on the AMT text dataset. Table 9 compares DSCM with ACM using the different source settings where the fixed noise level was set to $p_f = 0.5$. The results were

[4]Numbers are percentages corresponding to average class accuracies. Best results in bold, results of the suggested configuration in red.

Table 9. Results using TFIDF features on the AMT dataset.

Dataset	DSCM	ACMt*	ACMt^f	ACMr*	ACMr^f
AMT *ase* (US)	75.35	**77.84**	**76.73**	75.61	61.78
AMT *ase* (SS)	69.59	**77.85**	**75.48**	72.78	75.48
AMT *sse* (US)	64.17	**72.94**	**68.73**	66.1	58.86
AMT *sse* (SS)	66.71	**76.32**	**73.58**	70.9	68.19

Table 10. Comparing results with the dense case (DSCM+SMDA).

Dataset	DSCM	+sMDA*	+sMDA^f	ACM*	ACM^f
OFF31 (US) bov	26.18	26.95	**26.81**	**27.98**	27.31
OFF31 (SS) bov	45.6	**46.31**	**46.24**	45.64	45.29
OC10 (US) bov	49.84	49.01	**48.25**	**51.73**	50.04
OC10 (SS) bov	55.31	**58.63**	**56.86**	57.1	55.21
OFF31 (US) fc6	77.72	**82.46**	**80.61**	80.11	78.48
OFF31 (SS) fc6	83.55	**87.19**	**86.54**	86.92	85.52
OC10 (US) fc6	93.38	**96.25**	**94.95**	95.25	94.8
OC10 (SS) fc6	90.08	**96.17**	**94.94**	95.78	93.44
3VC (US) fc6	44.07	47.08	**45.85**	**50.59**	43.76
3VC (SS) fc6	90.37	**94.77**	**93.68**	94.42	93.45
AMT (US)	75.35	**77.84**	**76.73**	77.52	76.91
AMT (SS)	69.59	77.85	**75.48**	**77.91**	75.89

again obtained as average over as configurations, D→B, E→B, K→B, (D,E)→B, ...(B,D,E)→K. With these features ACMt outperformed ACMr.

Comparison with ACM using the whole source data. In Table 10 we compare ACM to DSCM+sMDA, *i.e.* the dense case where we use the whole source dataset in the sMDA framework to adapt the source data (instead of adapting the means) and compute the class means from adapted source examples before applying the DSCM. We consider both the fixed and the tuned noise level and we show results when all source examples were used (*ase*). For fair comparison we used 5 layers and concatenated the fifth on to the original features also in the case of DSCM+sMDA. The fixed noise level was set to 0.5 (except for BOV where we used 0.1) and we used ACMt (except for fc6 for which we used RELU)s. From these results we can see that both for the fixed noise level and even when we consider the optimal choice of the noise for each configuration, ACM often outperforms DSCM+sMDA and when this is not the case, it remains still highly competitive.

6. Conclusion

We proposed a simple and still powerful method with low computational cost for domain adaptation that requires only class means/prototypes from the source domains and hence can be applied in a larger set of real scenarios *e.g.* also those concerned with privacy issues. The method has only a few parameters for which we suggested reasonable default values depending on the representation used. Finally, we have shown that the method performs similarly to the dense case (DSCM+sMDA) and can be applied successfully even if the source data is available.

References

[1] M. Baktashmotlagh, M. T. Harandi, B. C. Lovell, and M. Salzmann. Unsupervised domain adaptation by domain invariant projection. In *ICCV*, 2013. 1, 3, 4

[2] O. Beijbom. Domain adaptations for computer vision applications. *CoRR*, arXiv:1211.4860, 2012. 1

[3] J. Blitzer, D. Foster, and S. Kakade. Domain adaptation with coupled subspaces. In *ICAIS*, 2011. 3

[4] M. Chen, Z. Xu, K. Q. Weinberger, and F. Sha. Marginalized denoising autoencoders for domain adaptation. *ICML*, arXiv:1206.4683, 2012. 1, 2

[5] G. Csurka, B. Chidlovskii, and F. Perronnin. Domain adaptation with a domain specific class means classifier. In *TASK-CV, ECCV workshop*, 2014. 1, 2

[6] J. Donahue, Y. Jia, O. Vinyals, J. Hoffman, N. Zhang, E. Tzeng, and T. Darrell. Decaf: A deep convolutional activation feature for generic visual recognition. In *ICML*, 1999. 3, 4

[7] L. Duan, I. W. Tsang, D. Xu, and T. Chua. Domain adaptation from multiple sources via auxiliary classifiers. In *ICML*, 2009. 1

[8] B. Fernando, A. Habrard, M. Sebban, and T. Tuytelaars. Unsupervised visual domain adaptation using subspace alignment. In *ICCV*, 2013. 3

[9] B. Fernando, A. Habrard, M. Sebban, and T. Tuytelaars. Subspace alignment for domain adaptation. *CoRR*, arXiv:1409.5241, 2014. 1

[10] Y. Ganin and V. Lempitsky. Unsupervised domain adaptation by backpropagation. *CoRR*, arXiv:1409.7495, 2014. 1, 4

[11] X. Glorot, A. Bordes, and Y. Bengio. Domain adaptation for large-scale sentiment classification: A deep learning approach. In *ICML*, 2011. 1

[12] B. Gong, K. Grauman, and F. Sha. Reshaping visual datasets for domain adaptation. In *NIPS*, 2013. 1, 3

[13] B. Gong, Y. Shi, F. Sha, and K. Grauman. Geodesic flow kernel for unsupervised domain adaptation. In *CVPR*, 2012. 3

[14] R. Gopalan, R. Li, and R. Chellappa. Domain adaptation for object recognition: An unsupervised approach. In *ICCV*, 2011. 1, 3

[15] A. Harley, A. Ufkes, and K. Derpanis. Evaluation of deep convolutional nets for document image classification and retrieval. In *ICDAR*, 2015. 3

[16] A. Krizhevsky, I. Sutskever, and G. Hinton. ImageNet classification with deep convolutional neural networks. In *NIPS*, 2012. 3

[17] N.Farajidavar, T. deCampos, and J. Kittler. Adaptive transductive transfer machines. In *BMVC14*, 2014. 1, 4

[18] K. Saenko, B. Kulis, M. Fritz, and T. Darrell. Adapting visual category models to new domains. In *ECCV*, 2010. 1, 3

[19] P. Vincent, H. Larochelle, Y. Bengio, and P.-A. Manzagol. Extracting and composing robust features with denoising autoencoders. In *ICML*, 2008. 2