Supplementary Material : Learning an Invariant Hilbert Space for Domain Adaptation

Samitha Herath[1,2], Mehrtash Harandi[1,2] and Fatih Porikli[1]
[1]Australian National University, [2]DATA61-CSIRO
Canberra, Australia
{samitha.herath, mehrtash.harandi}@data61.csiro.au, fatih.porikli@anu.edu.au

1. Introduction

In this supplementary material,

1. We provide detailed derivations used for our implementation.
2. We elaborate the Riemannian tools used for optimization on the product manifold.
3. We present an experiment elaborating the sensitivity of our solution to the used combination weight λ in the loss function.
4. We show the improvement obtained by using the orthogonality constraint over the transformation matrices.
5. We provide experimental results for Office+Caltech10 dataset when VGG-FC7 features are used.

2. Derivations

We recall our cost function from the main text,

$$\mathcal{L} = \mathcal{L}_d + \lambda \mathcal{L}_u \; . \tag{1}$$

In Eq. 1, $\mathcal{L}_d$ is a measure of dissimilarity between labeled samples. The term $\mathcal{L}_u$ quantifies a notion of statistical difference between the source and target samples in the latent space. In brief, the cost $\mathcal{L}_d$ was based on the proposed generalized soft-margin loss, ℓ_β on labeled pairs. The statistical loss, $\mathcal{L}_u$ was based on Stein divergence, δ_s between source and target domain covariances in the latent space. Here, we intend to derive the derivative of the proposed ℓ_β and $\mathcal{L}_u$. Note that all the variable dimensions and the notations are similar to what we have used in the main text.

2.1. Derivative of soft-margin ℓ_β

First we shall recall the structure of the proposed discriminative loss function, $\mathcal{L}_d$,

$$\mathcal{L}_d = \frac{1}{N_p} \sum_{k=1}^{N_p} \ell_\beta\big(\boldsymbol{M}, y_k, \boldsymbol{z}_{1,k} - \boldsymbol{z}_{2,k}, 1 + y_k \epsilon_k\big) + r(\boldsymbol{M}) + \frac{1}{N_p} \sqrt{\sum \epsilon_k^2}, \tag{2}$$

with

$$\ell_\beta\big(\boldsymbol{M}, y, \boldsymbol{x}, u\big) = \frac{1}{\beta} \log\Big(1 + \exp\big(\beta y (\boldsymbol{x}^T \boldsymbol{M} \boldsymbol{x} - u)\big)\Big). \tag{3}$$

In Eq. 2, y_k denotes whether the k-th pair is similar or dissimilar (*i.e.*, $y_k = +1$ if $\boldsymbol{z}_{1,k}$ and $\boldsymbol{z}_{2,k}$ are from the same class and $y_k = -1$ otherwise). For the sake of discussion, assume $\boldsymbol{z}_{1,k}$ and $\boldsymbol{z}_{2,k}$ are embedded from the source and target domains, respectively. That is $\boldsymbol{z}_{1,k} = \boldsymbol{W}_s^T \boldsymbol{x}_i^s$ and $\boldsymbol{z}_{2,k} = \boldsymbol{W}_t^T \boldsymbol{x}_j^t$. By expanding ℓ_β for such a pair, we get

$$\ell_\beta = \frac{1}{\beta} \log(1 + \exp(\beta y_k((\boldsymbol{z}_{1,k} - \boldsymbol{z}_{2,k})^T \boldsymbol{M}(\boldsymbol{z}_{1,k} - \boldsymbol{z}_{2,k}) - 1 - y_k \epsilon_k))) \tag{4}$$

We will use $d(\boldsymbol{M}, \boldsymbol{W}_s, \boldsymbol{W_t}) = (\boldsymbol{z}_{1,k} - \boldsymbol{z}_{2,k})^T \boldsymbol{M}(\boldsymbol{z}_{1,k} - \boldsymbol{z}_{2,k})$ to simplify our calculations. Hence, Eq. 4 could be re-written as,

$$\ell_\beta = \frac{1}{\beta} \log(1 + \exp(\beta y_k(d(\boldsymbol{M}, \boldsymbol{W}_s, \boldsymbol{W_t}) - 1 - y_k \epsilon_k))). \tag{5}$$

We provide the gradients of Eq. 5 with respect to $\boldsymbol{M}$, $\boldsymbol{W_t}$, $\boldsymbol{W_s}$ and the slack ϵ_k below. To simplify the presentation we use $r = \exp(\beta y_k(d(\boldsymbol{M}) - y_k \epsilon_k - 1))$.

2.1.1 Derivative w.r.t. $\boldsymbol{M}$

$$\begin{aligned}
\nabla_{\boldsymbol{M}} \ell_\beta &= \frac{y_k r}{(1+r)} \nabla_{\boldsymbol{M}} d(\boldsymbol{M}) \\
&= \frac{y_k r}{(1+r)} (\boldsymbol{W}_s^T \boldsymbol{x}_i^s - \boldsymbol{W}_t^T \boldsymbol{x}_j^t)(\boldsymbol{x}_i^{sT} \boldsymbol{W}_s - \boldsymbol{x}_j^{tT} \boldsymbol{W}_t) \\
&= y_k (1 + r^{-1})^{-1} (\boldsymbol{W}_s^T \boldsymbol{x}_i^s - \boldsymbol{W}_t^T \boldsymbol{x}_j^t)(\boldsymbol{x}_i^{sT} \boldsymbol{W}_s - \boldsymbol{x}_j^{tT} \boldsymbol{W}_t).
\end{aligned} \tag{6}$$

2.1.2 Derivative w.r.t. $\boldsymbol{W}_s$ (or w.r.t. $\boldsymbol{W}_t$)

$$\begin{aligned}
\nabla_{\boldsymbol{W}_s} \ell_\beta &= \frac{y_k r}{(1+r)} \nabla_{\boldsymbol{W}_s} d(\boldsymbol{W}_s) && (7) \\
&= 2 \frac{y_k r}{(1+r)} \boldsymbol{x}_i^s (\boldsymbol{x}_i^{sT} \boldsymbol{W}_s - \boldsymbol{x}_j^{tT} \boldsymbol{W}_t) \boldsymbol{M}. \\
&= 2 y_k (1 + r^{-1})^{-1} \boldsymbol{x}_i^s (\boldsymbol{x}_i^{sT} \boldsymbol{W}_s - \boldsymbol{x}_j^{tT} \boldsymbol{W}_t) \boldsymbol{M}. && (8)
\end{aligned}$$

For the case where both the pair instances are from the same domain (*i.e.* $\boldsymbol{z}_{1,k} = \boldsymbol{W}_s^T \boldsymbol{x}_i^s$ and $\boldsymbol{z}_{2,k} = \boldsymbol{W}_s^T \boldsymbol{x}_j^s$), it could be shown that,

$$\nabla_{\boldsymbol{W}_s} d(\boldsymbol{W}_s) = 2 y_k (\boldsymbol{x}_i^s - \boldsymbol{x}_j^s)(\boldsymbol{x}_i^{sT} - \boldsymbol{x}_j^{sT}) \boldsymbol{W}_s \boldsymbol{M}. \tag{9}$$

Considering Eq. 7 and Eq. 9,

$$\begin{aligned}
\nabla_{\boldsymbol{W}_s} \ell_\beta &= 2 \frac{y_k r}{(1+r)} (\boldsymbol{x}_i^s - \boldsymbol{x}_j^s)(\boldsymbol{x}_i^{sT} - \boldsymbol{x}_j^{sT}) \boldsymbol{W}_s \boldsymbol{M}. \\
&= 2 y_k (1 + r^{-1})^{-1} (\boldsymbol{x}_i^s - \boldsymbol{x}_j^s)(\boldsymbol{x}_i^{sT} - \boldsymbol{x}_j^{sT}) \boldsymbol{W}_s \boldsymbol{M}.
\end{aligned} \tag{10}$$

2.1.3 Derivative w.r.t. a Slack variable ϵ_k.

The slacks by origin are non-negative. To avoid using a non-negative constraint we make the substitution $\epsilon_k = e^{v_k}$ to Eq. 5.

$$\therefore \ell_\beta = \frac{1}{\beta} \log(1 + \exp(\beta y_k(d(\boldsymbol{M}, \boldsymbol{W}_s, \boldsymbol{W_t}, v_k) - 1 - y_k e^{v_k}))) \tag{11}$$

Table 1. Summarized Gradients of the Cost Function. Note we use $r = \exp(\beta y_k(d(\boldsymbol{M}) - y_k\epsilon_k - 1))$

$\nabla_{\boldsymbol{W}_s}\ell_\beta$; $x_i^s \in \mathbb{R}^s$, $x_j^t \in \mathbb{R}^t$	$2y_k(1+r^{-1})^{-1}\boldsymbol{x}_i^s(\boldsymbol{x}_i^{sT}\boldsymbol{W}_s - \boldsymbol{x}_j^{t\,T}\boldsymbol{W}_t)\boldsymbol{M}$
$\nabla_{\boldsymbol{W}_s}\ell_\beta$; $x_i^s \in \mathbb{R}^s$, $x_j^s \in \mathbb{R}^s$	$2y_k(1+r^{-1})^{-1}(\boldsymbol{x}_i^s - \boldsymbol{x}_j^s)(\boldsymbol{x}_i^{sT} - \boldsymbol{x}_j^{sT})\boldsymbol{W}_s\boldsymbol{M}$
$\nabla_{\boldsymbol{M}}\ell_\beta$	$2y_k(1+r^{-1})^{-1}(\boldsymbol{W}_s^T\boldsymbol{x}_i^s - \boldsymbol{W}_t^T\boldsymbol{x}_j^t)(\boldsymbol{x}_i^{sT}\boldsymbol{W}_s - \boldsymbol{x}_j^{t\,T}\boldsymbol{W}_t)$
$\nabla_{v_k}\ell_\beta$	$-e^{v_k}(1+r^{-1})^{-1}$
$\nabla_{\boldsymbol{W}_s}\mathcal{L}_u$	$\frac{1}{p}\boldsymbol{\Sigma}_s\boldsymbol{W}_s\Big(2\big\{\boldsymbol{W}_s^T\boldsymbol{\Sigma}_s\boldsymbol{W}_s + \boldsymbol{W}_t^T\boldsymbol{\Sigma}_t\boldsymbol{W}_t\big\}^{-1} - \big\{\boldsymbol{W}_s^T\boldsymbol{\Sigma}_s\boldsymbol{W}_s\big\}^{-1}\Big)$

Obtaining the derivative of Eq. 11 w.r.t. v_k,

$$\begin{aligned}\nabla_{v_k}\ell_\beta &= \frac{-e^{v_k}\exp(\beta y_k(d(\boldsymbol{M},\boldsymbol{W}_s,\boldsymbol{W_t},v_k) - 1 - y_k e^{v_k}))}{(1+\exp(\beta y_k(d(\boldsymbol{M},\boldsymbol{W}_s,\boldsymbol{W_t},v_k) - 1 - y_k e^{v_k}))} \\ &= \frac{-e^{v_k}r}{(1+r)} = -e^{v_k}(1+r^{-1})^{-1} \end{aligned} \tag{12}$$

2.2. Derivative of Statistical loss $\mathcal{L}_u$

The statistical loss (*i.e.* unsupervised loss) in Eq. 1 is defined with the stein divergence,δ_s of the domain covarainces in the latent space. This could be written as,

$$\begin{aligned}\mathcal{L}_u &= \frac{1}{p}\delta_s(\boldsymbol{W}_s^T\boldsymbol{\Sigma}_s\boldsymbol{W}_s, \boldsymbol{W}_t^T\boldsymbol{\Sigma}_t\boldsymbol{W}_t) \\ &= \frac{1}{p}\left\{\log\det\left(\frac{\boldsymbol{W}_s^T\boldsymbol{\Sigma}_s\boldsymbol{W}_s + \boldsymbol{W}_t^T\boldsymbol{\Sigma}_t\boldsymbol{W}_t}{2}\right) - \frac{1}{2}\log\det(\boldsymbol{W}_s^T\boldsymbol{\Sigma}_s\boldsymbol{W}_s\boldsymbol{W}_t^T\boldsymbol{\Sigma}_t\boldsymbol{W}_t)\right\} \\ &= \frac{1}{p}\left\{\log\det\left(\frac{\boldsymbol{W}_s^T\boldsymbol{\Sigma}_s\boldsymbol{W}_s + \boldsymbol{W}_t^T\boldsymbol{\Sigma}_t\boldsymbol{W}_t}{2}\right) - \frac{1}{2}\log\det(\boldsymbol{W}_s^T\boldsymbol{\Sigma}_s\boldsymbol{W}_s) - \frac{1}{2}\log\det(\boldsymbol{W}_t^T\boldsymbol{\Sigma}_t\boldsymbol{W}_t)\right\}, \end{aligned} \tag{13}$$

where $\boldsymbol{\Sigma}_s$ and $\boldsymbol{\Sigma}_t$ are respectively the source and target domain covariances. By following that (*see* Proof 2.11 of [2]),

$$\nabla_{\boldsymbol{W}_s}\log\det(\boldsymbol{W}_s^T\boldsymbol{\Sigma}_s\boldsymbol{W}_s) = 2\boldsymbol{\Sigma}_s\boldsymbol{W}_s(\boldsymbol{W}_s^T\boldsymbol{\Sigma}_s\boldsymbol{W}_s)^{-1}. \tag{14}$$

The derivative of Eq. 13 could be obtained w.r.t $\boldsymbol{W}_s$ (or similarly for $\boldsymbol{W}_t$[1]),

$$\nabla_{\boldsymbol{W}_s}\mathcal{L}_u = \frac{1}{p}\boldsymbol{\Sigma}_s\boldsymbol{W}_s\left\{\left(\frac{\boldsymbol{W}_s^T\boldsymbol{\Sigma}_s\boldsymbol{W}_s + \boldsymbol{W}_t^T\boldsymbol{\Sigma}_t\boldsymbol{W}_t}{2}\right)^{-1} - (\boldsymbol{W}_s^T\boldsymbol{\Sigma}_s\boldsymbol{W}_s)^{-1}\right\}. \tag{15}$$

The above derivatives are summarized in Table 1.

[1]The Stein divergence is symmetric over its two arguments.

3. Product Topology

As the constraints of the optimization problem depicted in Eq. 1 are indeed Riemannian manifolds, the whole set of constraints can be given a Riemannian structure through the concept of product topology. In particular, the constraints can be modeled as

$$\mathcal{M}_{prod.} = \mathrm{St}(p,s) \times \mathrm{St}(p,t) \times \mathcal{S}_{++}^{p} \times \mathbb{R}^{N_p}, \tag{16}$$

The tangent space of such a product topology [9, 6] could be written as,

$$\mathcal{T}_{(\boldsymbol{W}_s,\boldsymbol{W}_t,\boldsymbol{M},\boldsymbol{\epsilon})}\mathcal{M}_{prod.} = T_{\boldsymbol{W}_s}\mathrm{St}(p,s) \times T_{\boldsymbol{W}_t}\mathrm{St}(p,t) \times T_{\boldsymbol{M}}\mathcal{S}_{++}^{p} \times T_{\boldsymbol{\epsilon}}\mathbb{R}^{N_p}. \tag{17}$$

In Table 3, the metric and, the form of Riemannian gradient and the retraction for $\mathcal{M}_{prod.}$ are provided. Here, we follow the notations of Table 2.

4. Parameter Sensitivity and Orthogonality

In all the above experiments, we keep $\lambda = 1$ (see Eq. 1). To analyze the sensitivity of our method to the changes in parameter λ, we performed an experiment using the unsupervised protocol. This is because the statistical loss plays a significant role in establishing the correspondence between the source and the target in the unsupervised DA. We consider two random splits from each of the Office+Caltech10 dataset along VGG-FC6 features here.

Our results are shown in Fig. 1. When $\lambda = 0$, no statistical loss term is considered. It is clear that for this case the performance drops considerably. For other values of λ, the performance is superior and there is little variation in performance. In other words, our method remains robust.

We further investigate the benefit of orthogonality constraint on $\boldsymbol{W}_s$ and $\boldsymbol{W}_t$ against free-form and unconstrained transformations. Using the orthogonality constraint provides a considerable performance gain as shown in Fig. 1. While orthogonality makes the optimization more complicated, it seems it guides the learning to better uncovering the form of adaptation.

Table 2. Riemannian metric, gradient and retraction on $\mathrm{St}(p,n)$ and $\mathcal{S}_{++}^{p}$. Here, $\mathrm{uf}(\boldsymbol{A}) = \boldsymbol{A}(\boldsymbol{A}^T\boldsymbol{A})^{-1/2}$, which yields an orthogonal matrix, $\mathrm{sym}(\boldsymbol{A}) = \frac{1}{2}(\boldsymbol{A} + \boldsymbol{A}^T)$ and $\mathrm{expm}(\cdot)$ denotes the matrix exponential.

	$\mathrm{St}(p,n)$	$\mathcal{S}_{++}^{p}$
Matrix representation	$\boldsymbol{W} \in \mathbb{R}^{n\times p}$	$\boldsymbol{M} \in \mathbb{R}^{p\times p}$
Riemannian metric	$g_\nu(\xi,\varsigma) = \mathrm{Tr}(\xi^T\varsigma)$	$g_{\mathcal{S}}(\xi,\varsigma) = \mathrm{Tr}\left(\boldsymbol{M}^{-1}\xi\boldsymbol{M}^{-1}\varsigma\right)$
Riemannian gradient	$\nabla_{\boldsymbol{W}}(f) - \boldsymbol{W}\mathrm{sym}\left(\boldsymbol{W}^T\nabla_{\boldsymbol{W}}(f)\right)$	$\boldsymbol{M}\mathrm{sym}\left(\nabla_{\boldsymbol{M}}(f)\right)\boldsymbol{M}$
Retraction	$\mathrm{uf}(\boldsymbol{W}+\xi)$	$\boldsymbol{M}^{\frac{1}{2}}\,\mathrm{expm}(\boldsymbol{M}^{-\frac{1}{2}}\xi\boldsymbol{M}^{-\frac{1}{2}})\boldsymbol{M}^{\frac{1}{2}}$

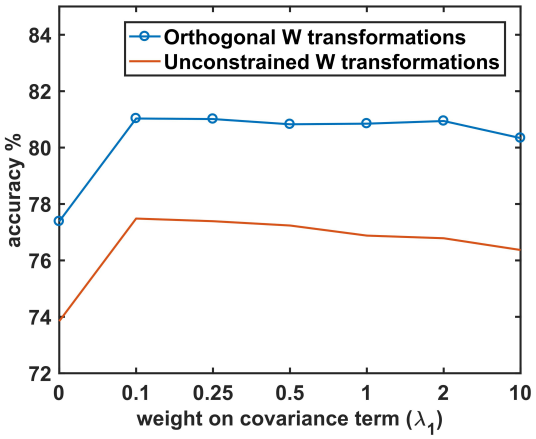

Figure 1. Sensitivity to λ in the unsupervised DA.

Table 3. Riemannian metric, gradient and retraction on the proposed Product Manifold in Eq. 16. The Riemannian metrics g_{ν_s} and g_{ν_t} are respectively defined on the Stiefel manifolds of $\boldsymbol{W}_s$ and $\boldsymbol{W}_t$. Furthermore, the Riemannian metrics g_S and g_E are respectively on the SPD manifold and the Euclidean manifold. As we have used in the main text, ξ and ς are elements from the tangent spaces of the corresponding manifolds. Here, $\text{uf}(\boldsymbol{A}) = \boldsymbol{A}(\boldsymbol{A}^T\boldsymbol{A})^{-1/2}$, which yields an orthogonal matrix, $\text{sym}(\boldsymbol{A}) = \frac{1}{2}(\boldsymbol{A} + \boldsymbol{A}^T)$ and $\text{expm}(\cdot)$ denotes the matrix exponential.

	$\mathcal{M}_{prod.}$
Matrix representation	$(\boldsymbol{W}_s, \boldsymbol{W}_t, \boldsymbol{M}, \boldsymbol{\epsilon})$
Riemannian metric	$g_{\nu_s}(\varsigma_s, \xi_s) + g_{\nu_t}(\varsigma_t, \xi_t) + g_{\mathcal{S}}(\varsigma_M, \xi_M) + g_E(\varsigma_E, \xi_E)$
Riemannian gradient	$\Big(\nabla_{\boldsymbol{W}_s}(f) - \boldsymbol{W}_s\text{sym}\Big(\boldsymbol{W}_s^T\nabla_{\boldsymbol{W}_s}(f)\Big),\ \nabla_{\boldsymbol{W}_t}(f) - \boldsymbol{W}_t\text{sym}\Big(\boldsymbol{W}_t^T\nabla_{\boldsymbol{W}_t}(f)\Big),\ \boldsymbol{M}\text{sym}\Big(\nabla_{\boldsymbol{M}}(f)\Big)\boldsymbol{M},\ \nabla_{\boldsymbol{\epsilon}}(f)\Big)$
Retraction	$\big(\text{uf}(\boldsymbol{W}_s + \xi_s),\ \text{uf}(\boldsymbol{W}_t + \xi_t),\ \boldsymbol{M}^{\frac{1}{2}}\,\text{expm}(\boldsymbol{M}^{-\frac{1}{2}}\xi_M\boldsymbol{M}^{-\frac{1}{2}})\boldsymbol{M}^{\frac{1}{2}},\ \boldsymbol{I}_p\big)$

5. Office+Caltech10 : VGG-FC7 Feature Experiments

In the main text we compared our results with SURF [1] and VGG-FC6 [10] features of the Office+Caltech10 dataset. Here, we provide the comparison on VGG-FC7 [10] features for both the semi-supervised (see Table 4) and unsupervised (see Table 5) setups Similarly to the VGG-FC6 experiments we use a dimensionality of 20 for all the DA-SL algorithms. Furthermore, we use $\lambda = 1$ (see Eq. 1). In general for all the DA algorithms, we see a performance reduction in VGG-FC7 features than the VGG-FC6 features. In the semi-supervised setup, our solution tops in 7 cases. In the unsupervised setup our solution leads in 8 cases out of 12.

Table 4. Semi-supervised domain adaptation results using VGG-FC7 features on Office+Caltech10 [5] dataset with the evaluation setup of [7]. The best (in bold blue), the second best (in blue).

	A→W	A→D	A→C	W→A	W→D	W→C	D→A	D→W	D→C	C→A	C→W	C→D
1-NN-t	81.8	78.2	68.3	77.8	77.6	67.4	78.1	81.5	66.9	79.0	80.6	77.4
SVM-t	87.5	**85.4**	76.8	86.2	85.6	75.8	87.0	87.1	76.0	87.1	86.4	84.4
HFA [3]	86.6	85.3	75.2	84.9	85.5	74.8	85.8	86.5	75.1	86.0	85.3	**84.8**
MMDT [7]	76.9	73.3	78.1	83.6	79.5	72.2	82.3	83.8	71.7	85.3	77.8	72.6
CDLS [8]	**90.0**	85.0	78.5	87.2	86.5	79.0	**87.7**	89.5	**78.8**	87.8	**89.7**	84.6
ILS (1-NN)	89.3	84.0	**81.9**	**88.4**	**91.0**	**80.8**	86.9	**94.4**	**78.8**	**88.9**	88.7	83.3

Table 5. Unsupervised domain adaptation results using VGG-FC7 features on Office+Caltech10 [5] dataset with the evaluation setup of [5].The best (in bold blue), the second best (in blue).

	A→W	A→D	A→C	W→A	W→D	W→C	D→A	D→W	D→C	C→A	C→W	C→D
1-NN-s	64.0	50.8	72.6	64.5	83.1	60.2	61.2	88.2	52.8	82.6	65.3	54.9
SVM-s	68.0	51.8	76.2	70.1	87.4	65.5	58.7	91.2	56.0	86.7	74.8	61.3
GFK-PLS [5]	74.0	57.6	76.6	75.0	89.6	62.1	67.5	**91.9**	62.9	84.1	73.6	63.4
SA [4]	75.0	60.7	76.2	74.6	88.8	67.5	66.0	89.5	59.4	82.6	73.6	63.2
CORAL [11]	71.8	61.3	**78.6**	81.4	**90.1**	73.6	71.2	93.5	63.0	**88.6**	76.0	63.8
ILS (1-NN)	**80.9**	**71.3**	78.4	**85.7**	84.8	**75.1**	**76.5**	91.8	**66.2**	87.1	**80.1**	**67.1**

References

[1] H. Bay, T. Tuytelaars, and L. Van Gool. Surf: Speeded up robust features. In *European conference on computer vision*, pages 404–417. Springer, 2006. 5

[2] M. Brookes. The matrix reference manual. *Imperial College London*, 2005. 3

[3] L. Duan, D. Xu, and I. W. Tsang. Learning with augmented features for heterogeneous domain adaptation. In *Proc. Int. Conference on Machine Learning (ICML)*, pages 711–718, Edinburgh, Scotland, June 2012. Omnipress. 5

[4] B. Fernando, A. Habrard, M. Sebban, and T. Tuytelaars. Unsupervised visual domain adaptation using subspace alignment. In *Proc. Int. Conference on Computer Vision (ICCV)*, pages 2960–2967, 2013. 5

[5] B. Gong, Y. Shi, F. Sha, and K. Grauman. Geodesic flow kernel for unsupervised domain adaptation. In *Proc. IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 2066–2073, 2012. 5

[6] V. Guillemin and A. Pollack. *Differential topology*, volume 370. American Mathematical Soc., 2010. 4

[7] J. Hoffman, E. Rodner, J. Donahue, B. Kulis, and K. Saenko. Asymmetric and category invariant feature transformations for domain adaptation. *Int. Journal of Computer Vision*, 109(1):28–41, 2014. 5

[8] Y.-H. Hubert Tsai, Y.-R. Yeh, and Y.-C. Frank Wang. Learning cross-domain landmarks for heterogeneous domain adaptation. In *The IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2016. 5

[9] J. M. Lee. Smooth manifolds. In *Introduction to Smooth Manifolds*, pages 1–29. Springer, 2003. 4

[10] K. Simonyan and A. Zisserman. Very deep convolutional networks for large-scale image recognition. *CoRR*, abs/1409.1556, 2014. 5

[11] B. Sun, J. Feng, and K. Saenko. Return of frustratingly easy domain adaptation. In *Thirtieth AAAI Conference on Artificial Intelligence*, 2016. 5