Robust Domain Adaptation on the L_1-Grassmannian Manifold

Sriram Kumar Andreas Savakis

Rochester Institute of Technology, Rochester, New York
{sk3100, andreas.savakis}@rit.edu

Abstract

Domain adaptation aims to remedy the loss in classification performance that often occurs due to domain shifts between training and testing datasets. This problem is known as the dataset bias attributed to variations across datasets. Domain adaptation methods on Grassmann manifolds are among the most popular, including Geodesic Subspace Sampling and Geodesic Flow Kernel. Grassmann learning facilitates compact characterization by generating linear subspaces and representing them as points on the manifold. However, Grassmannian construction is based on PCA which is sensitive to outliers. This motivates us to find linear projections that are robust to noise, outliers, and dataset idiosyncrasies. Hence, we combine L_1-PCA and Grassmann manifolds to perform robust domain adaptation. We present empirical results to validate improvements and robustness for domain adaptation in object class recognition across datasets.

1. Introduction

In practical machine learning problems, the data presented at test time can be quite different from the data used to train the classifier due to variations in pose or illumination, sensor variability and changes in the environment. For example, in Figure 1 sample images from three different domains are shown. The three domains have the same object categories, but contain visually dissimilar images due to domain shift. The process of adaptation comes naturally to humans, but it is hard to achieve in computer vision. This may be attributed to the fact that the model learned during training is biased on the particular training dataset [1]. Torralba and Efros [1] pointed out that each dataset has a distinct inherent bias or idiosyncrasy that causes the classifier to learn a biased model. This often results in poor cross-dataset generalization. The problem we address here is visual domain adaptation (DA), which falls under the category of transductive transfer learning, where the train and test data have the same object categories but the domain-shift is unknown [2].

Figure 1. Sample images from different domains such as DSLR, Amazon and Webcam dataset showing possible variations in the same object category bust across domains.

One of the main problems in visual domain adaptation is how to select proper features, given that the nature of the domain-shift is unknown. Ben-David et al. [3] presented a theoretical analysis indicating the choice of features representing the domains is such that the divergence between the distributions in the feature space is minimized. There is need for a robust system that is able to perform reasonably well on any dataset without being idiosyncratic towards any particular one. Although this is desirable, it is hard to achieve.

There have been several approaches proposed to tackle this problem [4]. Subspace based methods try to find a latent space that is domain invariant, and then project the data from different domains onto this space where classification is performed. This paper introduces robustness to domain adaptation by incorporating recent work on L_1-PCA. The particular methods explored are Geodesic Subspace Sampling (GSS) [5] and Geodesic Flow Kernel (GFK) [6].

The paper is organized as follows. Section 2 reviews the related work in visual domain adaptation. Section 3 reviews Grassmannian geometry and Section 4 overviews the PCA based Grassmann approaches used in this paper for evaluation. In Section 4 we present a robust approach for subspace generation during Grassmann manifold construction based on L_1-PCA. We outline the experiment setup and related results in Section 5 and conclude in Section 6.

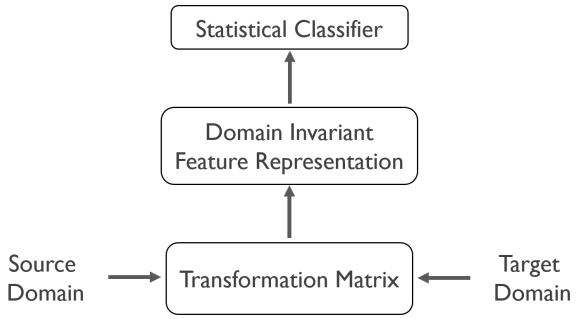

Figure 2. The pipeline for subspace based domain adaptation methodologies.

2. Related Work

Visual domain adaptation can be broadly categorized into semi-supervised and unsupervised. The former exploits the presence of target labels at hand. Successful approaches include transformative learning [19] and metric learning [2]. Although having access to a few labelled samples from the target domain helps in improving performance, these can be difficult to acquire and in many applications may not available. Unsupervised domain adaptation strategies often make use of linear representations such as principal component analysis (PCA) for domain representations [5, 6, 20-22] in a lower dimensional subspace. Although dimensionality reduction finds a low dimensional space common to both the domains, it does not guarantee the reduction in the divergence mismatch between the two domains.

Dimensionality reduction based DA methods are presented in [21, 22]. These approaches try to find a latent space by minimizing the mismatch in the distribution between the two domains using maximum mean discrepancy (MMD), a non-parametric method that compares two statistical distributions by mapping the data points to reproducible kernel Hilbert space (RKHS). A limitation of [22] is the computation of the kernel matrix via semi-definite programming which can be computationally challenging.

Manifold alignment based methods find a projection that exploits the local geometry by preserving the local neighborhood information [23, 24]. In [27], adaptation is performed by aligning the basis vectors of the source domain to the target domain by learning a transformation that minimizes the Bregman divergence. Metric learning and canonical correlation analysis (CCA) methods have been explored for DA in [2, 25, 26]. In [25], they assume the existence of a linear predictor for both the domains. A robust approach based on low rank reconstruction that is similar to manifold learning was proposed in [26].

While the above approaches are restricted to the use of source and target domain representation alone, Grassmannian based approaches exploit the intermediate representations [5, 6, 20]. In [5], intermediate subspaces are sampled from the geodesic curve on the manifold and are combined to create a domain invariant space, while [6] and [20] integrate the subspaces on the geodesic between the source and target domain to learn a transformation matrix. In [21], a lower dimensional representation is learnt on the Grassmann manifold that minimizes the MMD in RKHS space.

Subspace based methods in general try to find a latent space that is domain invariant, and then project data from different domains onto this space, where classification is performed. This process is summarized in Figure 2. Under the paradigm of domain adaptation, the training data is called the source domain D_S and the test data is called the target domain D_T. Domain adaptation deals with learning a classifier that performs well on the target domain, which is sampled from a different distribution than the source domain and often has few or no labeled samples.

Let X_S and X_T denote the samples present in the source and target domain respectively. Y_S and Y_T are the class labels corresponding to the conditional probabilities $\mathbb{P}(Y_S|X_S)$ and $\mathbb{P}(Y_T|X_T)$. Under transductive transfer learning, $\mathbb{P}(Y_S|X_S) \approx \mathbb{P}(Y_T|X_T)$ while $\mathbb{P}(X_S) \neq \mathbb{P}(X_T)$ [4]. This kind of scenario occurs frequently in computer vision problems such as face and object recognition in the wild. In the subsequent sections we discuss the techniques used for analysis.

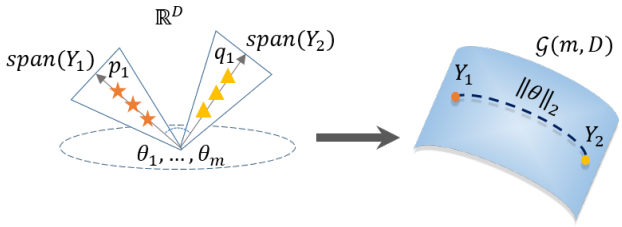

Figure 3. Mapping from Euclidean space to Grassmann space. The subspaces Y_1 and Y_2 are mapped as points on the Grassmann manifold. The angle θ is a metric used to define the similarity between the two subspaces spanned by Y_1 and Y_2.

3. Grassmannian Domain Adaptation

3.1. Grassmannian Geometry

Grassmann manifolds are a special class of Riemannian manifolds. The Grassmann manifold $\mathcal{G}(m, D)$ is defined as the set of all m-dimensional linear subspaces in $\mathbb{R}^D$ [7]. A simple visualization of the Grassmann manifold is shown in Figure 3. Let Y_1 and Y_2 be the representations of linear subspaces corresponding to two different image sets. These subspaces are mapped as two different points on the Grassmann manifold. Various metrics on the Grasmannian based on the angle θ between subspaces have been considered [8].

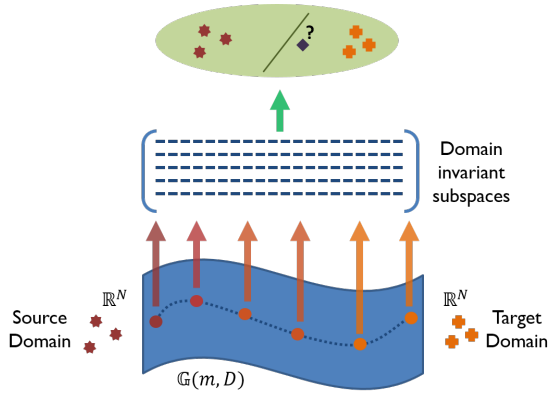

Figure 4. Graphical representation of Geodesic Subspace Sampling domain adaptation.

3.2. Geodesic Subspace Sampling (GSS)

Gopalan et.al [5] proposed a method to perform visual domain adaptation on the Grassmann manifold by sampling points along the geodesic connecting the two domains, namely the source and target domain, on the manifold. A graphical illustration of GSS is shown in Figure 4. The sampled points along the geodesic are essentially intermediate subspaces between the source and target domains, and provide intermediate feature representations that contain information present in both source and target domains. By concatenating the sampled subspaces, a domain invariant feature representation is obtained, which can be used as a projection matrix. The source and domain data can be projected onto this domain invariant space, which is followed by learning a classifier such as a nearest neighbor. We now briefly describe the sampling procedure on the manifold. The geodesic on the Grassmann manifold $\mathcal{G}(m, D)$ starting from source S to target T is given by

$$\phi(t) = Qe^{(tB)}J \tag{1}$$

and $J = \begin{bmatrix} I_D \\ 0_{m-D,D} \end{bmatrix}$, I_D is the $D \times D$ identity matrix. B is a skew-symmetric matrix given by $\begin{bmatrix} 0 & A^T \\ -A^T & 0 \end{bmatrix}$ and $A \in \mathbb{R}^{(m-D)\times D}$ is a real matrix. A denotes the direction matrix and is typically computed using the inverse exponential map [10]. Once the direction matrix A is obtained, we can then sample points along the geodesic on the manifold.

An efficient implementation for the orthogonal completion is presented by Gallivan et al [10]. To make the paper self-sufficient, we present the algorithms for computing the exponential map and sampling along the geodesic in Algorithm 1 and 2 respectively.

The process is summarized as follows: a) Generate subspaces for the source and target domains by performing PCA. b) Compute the geodesic flow between the source and target subspace, by computing the direction matrix. c) Sample points along the flow by substituting different value for t between 0 and 1, to obtain intermediate representations. d) Project the source and target data onto these intermediate subspaces and perform one nearest neighbor.

Algorithm 1: Computing the direction matrix via inverse exponential map [10]

1. Given the source P_S and target P_T subspaces,
2. Find the orthogonal completion Q of P_S.
3. Perform economic CS decomposition of $Q^T P_T$ given by, $Q^T P_T = \begin{pmatrix} X_A \\ Y_A \end{pmatrix} = \begin{pmatrix} U_1 & 0 \\ 0 & \widetilde{U}_2 \end{pmatrix} \begin{pmatrix} \Gamma(1) \\ -\Sigma(1) \end{pmatrix} V_1^T$
4. Compute θ_i , given by $arccos$ and $arcsine$ of the diagonal elements of Γ and Σ (i.e.) $\gamma_i = \cos(\theta_i)$ and $\sigma_i = \sin(\theta_i)$. Construct a diagonal matrix Θ, using θ as its diagonal elements.
5. Compute the direction matrix $A = \widetilde{U}_2 \Theta {U_1}^T$.

Algorithm 2: Computing the Exponential Map and sampling along the geodesic [10]

1. Given a point on the Grassmann manifold P_S, and the tangent vector $B = \begin{bmatrix} 0 & A^T \\ -A^T & 0 \end{bmatrix}$.
2. Find the orthogonal completion Q of P_S.
3. Compute the compact SVD of the direction matrix $A = \widetilde{U}_2 \Theta {U_1}^T$.
4. Compute the diagonal matrices $\Gamma(t)$ and $\Sigma(t)$ such that $\gamma_i = \cos(t\theta_i)$ and $\sigma_i = \sin(t\theta_i)$, where θ's are the diagonal elements of Θ.
5. Compute $\phi(t) = Q \begin{pmatrix} U_1 \Gamma(t) \\ -\widetilde{U}_2 \Sigma(t) \end{pmatrix}$, for different values of $t \in [0,1]$.

Despite its promising results and intuitive idea, GSS had some limitations. The first one is that there is no proper way to sample the intermediate points. Next is how to select the number of subspaces to retain for each domain. These parameters need to be tuned. Hence, the number of samples is determined empirically through cross-validation, which is cumbersome and time consuming.

3.3. Geodesic Flow Kernel (GFK)

Gong et.al [6], proposed a new approach which integrates all the intermediate subspaces between the source and target subspaces. The geodesic distance is the shortest distance connecting between two points on the Grassmann manifold. One can think of the geodesic as the line connecting the two subspaces in the manifold. The intermediate points on the geodesic line represent the intermediate subspaces.

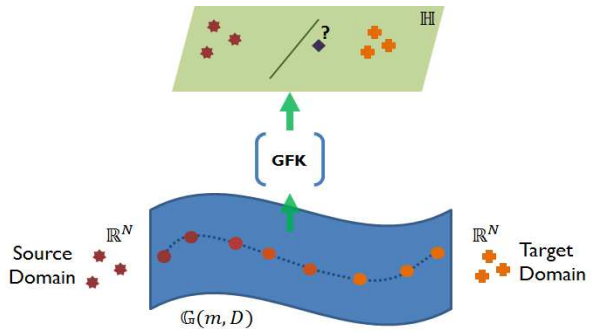

Figure 5. Graphical representation of the Geodesic Flow Kernel domain adaptation.

A pictorial representation of the GFK process is shown in Figure 5. We denote the source and target subspace as P_S and P_T. Intuitively this works as all the subspaces between the source and target domains are utilized, unlike the previous approach where one samples intermediate points along the geodesic between P_S and P_T. The GFK approach consists of two major steps: (a) construct a geodesic curve between the two subspaces, (b) find a kernel that integrates all the intermediate subspaces; in this case it is the Geodesic Flow Kernel.

The geodesic curve is a generalization of the straight line on curved surfaces. We parameterize the curve by $t \in [0, 1]$ and define a subspace as $\Phi(t)$, where at $t = 0$ the subspace represents the source domain, P_S, and at $t = 1$ the subspace is represents the target domain, P_T. There are infinite number of subspaces between the source and target domains. The projection of a feature vector on a subspace is given by $\Phi(t)^T x$. Such a projection enables representations that are invariant and unbiased to both source and target domains. If x_i and x_j represent the features from source and target domain respectively, the flow kernel can be obtained as follows:

$$\begin{aligned} \langle z_i^{\infty} | z_j^{\infty} \rangle &= \int_0^1 (\Phi(t)^T x_i)^T \left(\Phi(t)^T x_j\right) dt \\ &= \int_0^1 x_i^T (\Phi(t)\Phi(t)^{\mathrm{T}}) x_j \; dt \\ &= x_i G_K x_j. \end{aligned} \tag{2}$$

where, $G_K \in \mathbb{R}^{D \times D}$ is a positive definite matrix. By doing this, the feature vectors are implicitly projected on infinite number of subspaces that are lying in between. This G_K is essentially the kernel matrix and accomplishes the "kernel trick". The elements of G_K are the pairwise dot products between the subspaces. The closed form solution for G_K is given by

$$G_K = [P_S U_1 \quad R_S U_2] \begin{bmatrix} \Lambda_1 & \Lambda_2 \\ \Lambda_2 & \Lambda_3 \end{bmatrix} \begin{bmatrix} {U_1}^T {P_S}^T \\ {U_2}^T {R_S}^T \end{bmatrix}. \tag{3}$$

where Λ_1, Λ_3 are the diagonal elements and Λ_2 is the off diagonal elements and are defined as follows:

$$\begin{aligned} \lambda_{1i} &= 1 + \frac{\sin 2\theta_i}{2\theta_i}, \\ \lambda_{2i} = \frac{\cos 2\theta_i - 1}{2\theta_i}, &\qquad \lambda_{3i} = 1 - \frac{\sin 2\theta_i}{2\theta_i} \end{aligned} \tag{4}$$

Although these methods work well, they are often susceptible to noise and are not robust [6, 11]. This is due the fact that the subspaces are generated using the L_2-norm constraint. Hence, we use the L_1-norm constraint to generate our subspaces. The robustness of L_1-based PCA has been shown to work well for face recognition [16]. We show that by combining the L_1-norm for generating subspaces in the Grassmann framework, we obtain better recognition rates.

4. Robust Grassmann Learning

Subspace generation is an important part of constructing the Grassmann manifold. Traditional PCA is based on L_2-norm and as such it is susceptible to noisy projections. This affects the recognition process and a robust alternative is needed. By using L_1-norm based PCA, we can extract robust projections and thus create a robust Grassmann manifold. Below we review the L_2-norm and L_1-norm approaches for PCA subspace generation.

4.1. L_2 - Principal Component Analysis

Principal Component Analysis provides a signal decomposition using a linear combination of orthogonal basis vectors. The principal components are computed by finding the eigenvectors of the covariance matrix. The contribution of each component for a particular signal can reconstruct the original signal from the principal components. This is achieved by reducing the squared loss between the original signal and its reconstruction. For example if we define the matrix $X \in \mathbb{R}^{D \times N}$ and $D \gg N$ such that each column represents a signal after vectorising. The principal components can be found by solving:

$$(XX^T)V = V\Lambda \tag{5}$$

where V is the matrix of eigenvectors of (XX^T) and Λ is a diagonal matrix, containing the eigenvalues associated with each eigenvector.

4.2. L_1 - Principal Component Analysis

In this section we begin with L_2-PCA and move into the formulation of L_1-PCA. Again, the data matrix is $X = [x_1, x_2, \dots, x_N], X \in \mathbb{R}^{D \times N}$; where D is the dimension of the data and N is the number of samples. Assuming the data is zero mean, in L_2-PCA formulation we try to

minimize the projection error which is equivalent to maximizing the variance in the data.

$$E_2 = \|X - RV\|_2^2 \tag{6}$$

where, E is the reconstruction error, $R \in \mathbb{R}^{D\times m}$ and $V \in \mathbb{R}^{m\times N}$ given by

$$V = R^T X \tag{7}$$

$$R_{L_2} = \arg\max_R \|X^T R\|_2, \tag{8}$$

where R is the matrix containing columns denoting the principal components. The problem with this approach is that the error value is sensitive to outliers [12]. Even a single outlier can affect the direction of the principal components. In order to make it robust, we may formulate the problem under the L_1 norm. Under the L_1 formulation, Equations (6) and (8) become

$$E_1 = \|X - RV\|_1 \tag{9}$$

$$R_{L_1} = \arg\max_R \|X^T R\|_1. \tag{10}$$

As a consequence of this reformulation, Equations (9) and 10 are no longer equivalent under L_1 constraints. The optimal solution to the above L_1 problem is NP-hard [13] and suboptimal approaches have been developed [14]. To find one optimal principal component, Equation (10) can be rewritten as,

$$r_{L_1} = \arg\max_{r\in\mathbb{R}^{D\times 1},\, r^T r=1} \|X^T r\|_1. \tag{11}$$

It has been shown [13] that the optimal solution for a single L_1 principal component is given by,

$$r_{L_1} = \frac{Xb_{opt}}{\|Xb_{opt}\|_2}, \tag{12}$$

where

$$b_{opt} = \arg\max_{b\in\{\pm 1\}^N} \|Xb\|_2 = \arg\max_{b\in\{\pm 1\}^N} \{b^T X^T X b\}. \tag{13}$$

The vector b_{opt} is a binary vector having length N and entries either 1 or -1. It has also been shown in [13] that $\|X^T r_{L_1}\|_1 = \|Xb_{opt}\|_2$. Thus, after finding b_{opt} it is straight forward to obtain r_{L_1}, the L_1 principal component. A fast computation of the principal component was proposed in [15] and is discussed next.

4.2.1 Fast Computation of Eigenvectors

Kundu et.al [15] introduced a new approach for fast computation of one principal component. We briefly explain the method in this section. A binary vector is initialized from the column of covariance matrix of the input data. The bits that contribute negatively to the projection energy are flipped and the process is repeated until the optimal binary vector is found. The quadratic form of Equation (12) is given by,

$$b^T X^T X b = Trace(X^T X) + \sum_i 2b_i \left\{ \sum_{j>i} b_j (X^T X)_{i,j} \right\} \tag{14}$$

where $i, j \in 1,2,\dots,N.$

In [15], the variable α is defined, that finds the bits that contribute negatively to each location.

$$\alpha_i \triangleq \pm 4b_i \textstyle\sum_{j\neq i} b_j (X^T X)_{i,j}. \tag{15}$$

The bits that negatively contribute are found through this process and are flipped to obtain b_{opt}. After finding b_{opt} we can determine the corresponding eigenvector using Equation (12).

This approach was extended to obtaining multiple eigenvectors with a greedy strategy that preserves orthogonality [14]. The pseudocode for the greedy approach is presented below in Algorithm 3. After we find the first principal component, we remove its contribution from the data and find the next principal component using the bit flipping method. This guarantees orthogonality of the principal components [14] and the process is repeated until all principal components are found.

Algorithm 3: Greedy search for L_1-PCA [14]

For $j = 2{:}m$

// m is the number of principal components

$x_i^j = x_i^{j-1} - r_{j-1}\left(r_{j-1}^T x_i^{j-1}\right) \forall i \in \{1,\dots,N\}$

$X^j = \left[x_1^j, x_2^j, \dots, x_N^j\right]$

4.3. L_1-Grassmann manifold

During Grassmann manifold construction, PCA is the common method for subspace generation. The apparent problem is that the method lacks robustness to outliers and noise, while blindly trying to find the projection along the direction maximum variance. This leads to finding projections that may point towards noise, which is not desirable [11, 14].

In order to obtain a robust subspace that is not susceptible to noise or outliers, we utilize the L_1-PCA to obtain the subspaces P_S and P_T from source and target domain data X_S and X_T respectively. The L_1-PCA has shown promising results in face recognition with noise [16]. We maintain that by integrating L_1-PCA approach for Grassmann framework, we might get a more robust subspace. The L_1-Grassmannian [11] approach for subspace mapping is more robust to noise that may occur while mapping the subspace on the Grassmann manifold.

5. Experimental Results

In this section we start by discussing the datasets used and report empirical results on two subspace based visual domain adaptation methods, namely Geodesic Subspace Sampling [5] and Geodesic Flow Kernel [6]. We perform object recognition experiments and show the validity of the proposed robust Grassmann manifold approach.

5.1. Object Class Recognition

For this experiment, we use the Office and Caltech-256 datasets. The office dataset contains images taken from Webcam, DSLR camera and Amazon. In each domain there are a total of 31 categories, such as headphones, monitor, laptop, cycle, etc. In addition, images from Caltech–256 database were used. The categories which overlapped with the other three domains were selected from the Caltech database.

We used the precomputed features provided in [2]. In short, SURF (Speeded-Up Robust Features) features, were extracted from images in the Amazon dataset and a codebook was learnt with 800 codewords by selecting a subset of features and by performing K-means clustering. Finally, each image was represented by a histogram of 800 codewords. This representation was used to represent images in each domain. The features were normalized to have zero mean and unit standard deviation.

We performed object recognition experiments on the Office and Caltech-256 datasets. Sample images are shown in Figure 1. We denote the four domains as A, C, W and D for Amazon, Caltech-256, Webcam and DSLR respectively. There are in total 12 possible combinations of datasets for the Domain Adaptation tasks.

We report the recognition rates for unsupervised object recognition in Tables 1 and 2. We compared GSS and GFK, the two Grassman manifold based techniques, with standard subspace generation using L_2-PCA and robust subspace generation using L_1-PCA. We report the average accuracies over the trials based on one nearest neighbor and SVM. For SVM classification, we used the kernelized version where we precomputed the kernel and cross-validated for the parameter C. For SVM computation, we used the libsvm package [17]. For GSS**,** we used Partial Least Squares [18] to learn discriminative classifier.

6. Analysis of Results

In the unsupervised learning setting, the labels of images in the source domain are known but those of the images in the target domain are unknown. For each domain we generate L_1-norm constrained basis vectors such that the number of eigenvectors is same as the dimension of the data. After this, we performed domain adaptation using Geodesic Subspace Sampling (GSS) and Geodesic Flow Kernel (GFK). We performed 20 random trials, for which we randomly choose a subset of samples from the source domain and performed the domain adaptation on the target domain. We empirically chose the optimal value for subspace dimensions. For the subspace generation, we used L_1-PCA and L_2-PCA for both the source and target domains and evaluated the two approaches.

The results in Tables 1 and 2 shown in bold indicate improvement when using the L_1 approaches. In order to validate the significance of the proposed approaches we conducted two-way ANOVA (analysis of variance) across different domain pairs. The results which are underlined in Tables 1 and 2 indicate statistically significant with 95% confidence. This shows that the results obtained with the L_1 approach are significant and not due to chance. In specific, for the webcam-DSLR pair the accuracy of the GFK approach improved from 78.79% to 87.19%.

6.1. Finding the optimal L_1 subspace dimensionality

In order to estimate the optimal subspace dimension for each of the 12 domain pairs, we varied the subspace dimensions from 5 to 200. The variation of accuracy against the subspace dimension for four domain pairs based on 1-NN approach is shown in Figure 6 and 7. Similar results were obtained for both GSS and GFK. The optimal dimension was found to be around 20-30.

In Figure 7, it can be observed that as the dimensionality increases above a certain point, the performance decreases and finally tapers off. This is attributed to the fact that higher dimensions begin to overfit to the source domain and do not work as well for targets domains that appear visually similar such as webcam and DSLR benefit the most from the proposed L_1 approach.

7. Conclusion

In this paper we explored robust domain adaptation methods by learning the L_1-Grassmannian. Our method of generating robust subspaces is attributed to the L_1-norm and improves the recognition rate. Two popular Grassmann manifold based domain adaptation techniques, Geodesic Subspace Sampling and Geodesic Flow Kernel, were analyzed and evaluated.

The L_1 approaches achieved improvements over the standard methods across multiple domain pairs. In particular, the L_1 Grassmannian approaches boosted the accuracy significantly when the domain pairs were visually similar.

Table 1. Recognition accuracy (in %) with unsupervised Domain Adaptation using NN classification. Datasets: A: AMAZON, C: CALTECH, D: DSLR, W: WEBCAM

Method	**C→A**	**C→W**	**C→D**	**A→C**	**A→W**	**A→D**
L_2-GSS [5]	34.15±1.8	28.47±4.3	32.61±4.5	33.01±2.2	30.10±3.7	29.52±3.8
L_1-GSS (ours)	**34.93±2.2**	**28.71±3.7**	**34.04±4.3**	**33.54±1.6**	**32.41±4.3**	**32.48±3.9**
L_2-GFK [6]	36.75±2.1	34.19±4.7	35.25±3.2	**35.66±1.3**	36.08±2.9	**36.37±4.2**
L_1-GFK (ours)	**37.88±2.0**	**34.37±4.3**	**38.12±3.8**	34.62±1.6	**37.76±3.5**	33.66±2.9

Method	**W→C**	**W→A**	**W→D**	**D→C**	**D→A**	**D→W**
L_2-GSS [5]	25.45±1.6	31.96±2.1	73.47±3.0	28.05±1.7	28.19±3.4	65.10±3.5
L_1-GSS (ours)	**26.53±1.7**	**32.53±1.9**	**75.54±2.5**	**30.00±1.8**	**30.36±2.3**	**68.20±2.7**
L_2-GFK [6]	**29.93±1.2**	30.27±1.5	78.79±2.2	29.64±1.3	33.06±1.8	74.49±2.3
L_1-GFK (ours)	27.80±0.6	**31.64±1.5**	**87.19±2.1**	29.39±1.0	**35.25±1.6**	**78.17±2.4**

Table 2. Recognition accuracy (in %) with unsupervised Domain Adaptation using SVM classification. Datasets: A: AMAZON, C: CALTECH, D: DSLR, W: WEBCAM)

Method	**C→A**	**C→W**	**C→D**	**A→C**	**A→W**	**A→D**
L_2-GFK [6]	46.57±3.8	39.83±4.1	42.39±4.9	40.49±2.3	40.49±3.0	38.41±3.5
L_1-GFK (ours)	46.69±3.2	**41.00±3.8**	42.32±4.2	**41.03±1.6**	40.61±2.8	38.98±3.7

Method	**W→C**	**W→A**	**W→D**	**D→C**	**D→A**	**D→W**
L_2-GFK [6]	30.51±1.7	34.87±2.4	74.81±3.3	32.89±2.0	**38.55±2.0**	72.93±3.3
L_1-GFK (ours)	**31.24±1.4**	**37.71±2.1**	**79.36±2.8**	**34.82±2.4**	36.47±2.5	**75.68±2.5**

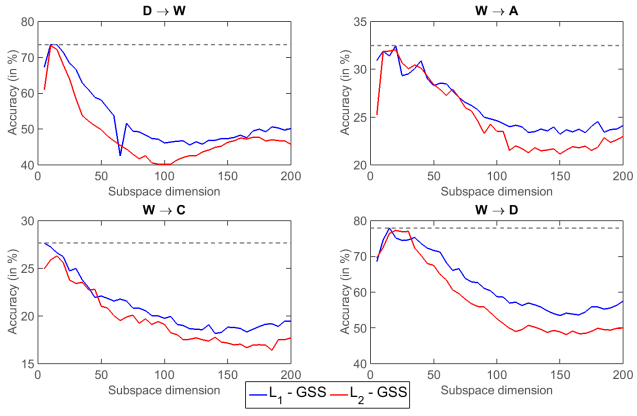

Figure 6. Plot of domain adaptation accuracies versus number of subspace dimensions for GSS method.

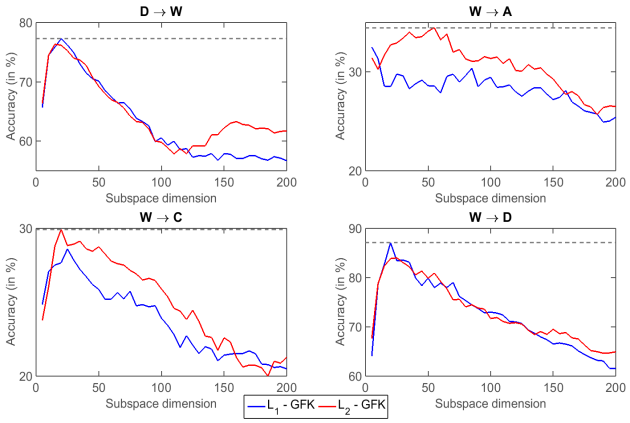

Figure 7. Plot of domain adaptation accuracies versus number of subspace dimensions for GFK method.

Acknowledgement

This research was supported in part by Qualcomm.

References

[1] A. Torralba and A. A. Efros. Unbiased look at dataset bias. In Computer Vision and Pattern Recognition, CVPR, 2011.

[2] K. Brian, K. Saenko and T. Darrell. What you saw is not what you get: Domain adaptation using asymmetric kernel transforms. In Computer Vision and Pattern Recognition, CVPR, 2011.

[3] S. Ben-David, J. Blitzer, K. Crammer and F. Pereira. Analysis of representations for domain adaptation. In Advances in Neural Information Processing Systems, NIPS, 2007.

[4] S. J. Pan and Q. Yang. A survey on transfer learning. In IEEE Trans. Knowledge and Data Engineering, pages 1345-1359, 2010.

[5] R. Gopalan, Ruonan Li and R. Chellappa. Domain adaptation for object recognition: An unsupervised approach. In IEEE Int. Conf. Computer Vision, ICCV, 2011.

[6] B. Gong, F. S. Yuan Shi and K. Grauman. Geodesic flow kernel for unsupervised domain adaptation. In Computer Vision and Pattern Recognition, CVPR, 2012.

[7] P. Turaga, A. Veeraraghavan, A. Srivastava and R. Chellappa. Statistical computations on Grassmann and Stiefel manifolds for image and video-based recognition. In IEEE Trans. Pattern Analysis and Machine Intelligence, pages 2273-2286, 2011.

[8] J. Hamm and D. Lee. Grassmann discriminant analysis: a unifying view on subspace-based learning. In Proceedings of the 25th ACM International Conference on Machine learning, 2008.

[9] G. H, Golub and C. F. Van Loan. Matrix computations. Vol. 3. JHU Press, 2012.

[10] K. A. Gallivan, A. Srivastava, X. Liu and P. Van Dooren. Efficient algorithms for inferences on grassmann manifolds. In IEEE Workshop on Statistical Signal Processing, 2003.

[11] M. Johnson and A. Savakis. L_1-Grassmann manifolds for robust face recognition. In IEEE Int. Conf. Image Processing, ICIP, 2015.

[12] P. Markopoulos, G. Karystinos and D. Pados. Optimal Algorithms for-subspace Signal Processing. In IEEE Trans. Signal Processing, pages 5046-5058, 2014.

[13] P. Markopoulos, G. Karystinos and D. Pados. Some options for L_1-subspace signal processing. In Proc. Tenth International Symposium on Wireless Communication Systems, ISWCS, 2013.

[14] N. Kwak. Principal component analysis based on L_1-norm maximization. In IEEE Trans. Pattern Analysis and Machine Intelligence, pages 1672-1680, 2008.

[15] S. Kundu, P. Markopoulos and D. Pados. Fast computation of the L_1-principal component of real-valued data. In IEEE Int. Conference on Acoustics, Speech and Signal Processing, ICASSP, 2014.

[16] M. Johnson and A. Savakis. Fast L_1-eigenfaces for robust face recognition. IEEE Western New York Image and Signal Proc. Workshop, WNYISPW, 2014.

[17] C. C. Chang and C. J. Lin. LIBSVM: a library for support vector machines. ACM Transactions on Intelligent Systems and Technology, 2011.

[18] H. Wold. Partial least squares. Encyclopedia of statistical sciences, 1985.

[19] K. Saenko, B. Kulis, M. Fritz, and T Darrell. Adapting visual category models to new domains. In European Conf. Computer Vision, ECCV, 2010.

[20] J. Zheng, M. Yu-Liu, R. Chellappa and P. J. Phillips. A Grassmann manifold-based domain adaptation approach. In Int. Conf. Pattern Recognition, ICPR, 2012.

[21] M. Baktashmotlagh, M. Harandi, B. Lovell and M. Salzmann. Unsupervised domain adaptation by domain invariant projection. In International Conference on Computer Vision, ICCV, 2013.

[22] S. K. Pan, J. Kwok and Q. Yang. Transfer Learning via Dimensionality Reduction. In AAAI, 2008.

[23] C. Wang and S. Mahadevan. Manifold Alignment without Correspondence. In International Joint Conference on Artificial Intelligence, 2009.

[24] C. Wang and S. Mahadevan. Heterogeneous domain adaptation using manifold alignment. In International Joint Conference on Artificial Intelligence, 2011.

[25] M. Chen, K. Weinberger, and J. Blitzer. Co-training for domain adaptation. In Advances in Neural Information Processing Systems, NIPS, 2011.

[26] I. Jhuo, D. Liu, D. Lee and S. Chang. Robust visual domain adaptation with low-rank reconstruction. In Computer Vision and Pattern Recognition, CVPR, 2012.

[27] B. Fernando, A. Habrard, M. Sebban and T. Tuytelaars. Unsupervised Visual Domain Adaptation Using Subspace Alignment. In International Conference on Computer Vision, ICCV, 2013.