Riemannian Similarity Learning

Li Cheng CHENGLI@BII.A-STAR.EDU.SG
Bioinformatics Institute, A*STAR, Singapore
School of Computing, National University of Singapore, Singapore

Abstract

We consider a similarity-score based paradigm to address scenarios where either the class labels are only partially revealed during learning, or the training and testing data are drawn from heterogeneous sources. The learning problem is subsequently formulated as optimization over a bilinear form of fixed rank. Our paradigm bears similarity to metric learning, where the major difference lies in its aim of learning a rectangular similarity matrix, instead of a proper metric. We tackle this problem in a Riemannian optimization framework. In particular, we consider its applications in pairwise-based action recognition, and cross-domain image-based object recognition. In both applications, the proposed algorithm produces competitive performance on respective benchmark datasets.

1. Introduction

We consider a similarity-score based learning problem: Denote by $\hat{x} \in \mathbb{R}^{\hat{n}}$ and $x \in \mathbb{R}^{n}$ instances from two different sources, predict whether a pair of unseen instances $(\hat{x}, x)$ belongs to the same class. The problem is key to many seemingly-unrelated real-life applications, where we are asked to make predictions given access to either heterogeneous sources that differ from training to testing phases, or provided only incomplete knowledge of the set of possible class labels, or even both. In particular, they include the applications of domain adaptation as well as pair-matching.

In domain adaptation (Pan & Yang, 2010), data from the source and target domains are often very different, and might reside in separate spaces. The closest research here might be the work of (Saenko et al., 2010; Kulis et al., 2011): The *Symm* method of (Saenko et al., 2010) is developed based on metric learning with cross-domain constraints. *Symm* aims to learn a symmetric transformation to project the source and target domain data into a domain-invariant space. The *ARC-t* method of (Kulis et al., 2011) extends this idea further by instead learning a asymmetric transformation to map the target domain data to the source domain. Designated to work with square similarity matrices, it remains cumbersome to address the $\hat{n} \neq n$ scenarios often seen when dealing with heterogeneous data sources such as sound tracks and videos. The applications of pair-matching have been raised in e.g. face or action recognitions (Chen et al., 2005; Kliper-gross et al., 2012), where there is a practical demand to identify a new face image or a clip of unseen action, a class label that does not exist in the training set. Note this pair-matching application is also referred to as face verification (Kumar et al., 2009; Yin et al., 2011).

One distinct feature of pairwise similarity learning lies in its ability to make predictions on novel class labels not provided in training set. Recently its potential has drawn increasing attention, ranging from theoretical studies (Balcan & Blum, 2006; Wang et al., 2009b) to real-life applications such as face and action recognition (Chen et al., 2005; Huang et al., 2007; Kliper-gross et al., 2012). This pairwise similarity learning problem is in spirit similar to the metric learning research work (Davis et al., 2007; Jain et al., 2012; Bellet et al., 2012; Mensink et al., 2012): One main difference is here we need to deal with non-symmetric and non-square similarity matrices, versus the more well-posed metric functions [1] usually considered in metric learning. This line of research is also related to one-shot learning or learning from one example (Miller et al., 2000; Fei-fei et al., 2006). The algorithm development is usually emphasized on the specific scenario where

Proceedings of the 30th International Conference on Machine Learning, Atlanta, Georgia, USA, 2013. JMLR: W&CP volume 28. Copyright 2013 by the author(s).

[1] A metric is a function that satisfies three conditions: (a) non-negativity and identity at zero, (b) symmetry, and (c) triangle inequality.

the similarity function is symmetric – in other words, $\hat{x}$ and x have to come from the same source.

We consider a general similarity learning scenario where $\hat{x}$ and x are from the different sources and even reside in different dimensions, $\hat{n} \neq n$. Our aim is to learn this $\hat{n}$ by n similarity matrix W, which is well-known to be a smooth (C^∞) manifold (e.g. Example 8.14 of (Lee, 2003)). However, this often leads to a *huge* number of unknowns, as both dimensions could be large. Two strategies natural follow as a consequence: (1) Work with sparse methods to exploit the sparsity (e.g. ℓ_1-norm) or structural sparsity (e.g. sparsity induced norm) of the matrix (Bach et al., 2012), or (2) Utilize matrix factorization techniques (Hubert et al., 2000; Eldén, 2007), in order to uncover the inherit matrix structure. Meanwhile, an important observation here is that the rank of this matrix is usually very *low*, as being reported by related research work (Ying et al., 2009; Bi et al., 2011), which is also empirically supported by our synthetic experiments in Sec. 4. This inspires us to consider a rank-related matrix factorization, and a natural choice is SVD. Recent development in manifold-based optimization (Absil et al., 2008) offers new geometric insight into the underlining geometric structures for this type of low rank matrices. By resorting to computational differential geometry principles and techniques, competitive or even superior performance has been achieved on matrix-related applications (Shalit et al., 2012; Vandereycken, 2013).

Our Contributions The main contributions in this paper are three-fold. First, we propose to work with similarity learning in the *general* setting, as such we aim at learning a non-symmetric, non-square similarity matrix with fixed rank. This is less-restrictive comparing to the existing work discussed earlier. It thus enables to directly work with the $\hat{n} \neq n$ settings. Second, our algorithm offers a new geometric interpretation for similarity learning, and by this we wish to bring some insights in understanding the characteristics of the similarity matrix. Third, our work brings interesting connections between the applications of pair-matching and domain adaptation. The proposed geometric algorithm also connects to and makes possible the exploitation of Riemannian manifold-based optimization (Absil et al., 2008). Empirical experiments verify the competitive performance of our algorithm in real-life vision tasks of object and action recognition.

Related Work The closest work is that of Kulis *et al.* (Kulis et al., 2011), which also aims to learn a non-symmetric similarity matrix. Meanwhile, there is no explicit rank or sparsity constraints on the matrix to be learned, leaving the potential for overfitting. The works in Mahalanobis metric learning *e.g.* (Bellet, 2012) are closely related, where the main focus is on symmetric positive semi-definite matrices. Another thread of intimately related works is bilinear learning (Tenenbaum & Freeman, 2000; Pirsiavash et al., 2009; Pirsiavash & Ramanan, 2012; Akhter et al., 2012). Denote the trace operator of a matrix as $\mathrm{tr}(\cdot)$, and factorize the matrix W as $W := W_l W_r^T$, with W_l and W_r being a $\hat{n} \times d$ and a $n \times d$ matrix, respectively. The bilinear function of interest usually takes the following form

$$\hat{f}_W(X) := \mathrm{tr}\big(W_l^T X W_r\big) = \mathrm{tr}\Big(\big(W_l W_r^T\big)^T X\Big).$$

On the other hand, significant progress has been made over the past few years in optimization on Riemannian manifolds (Absil et al., 2008). Recently research has also been carried on toward the topic of geometric low-rank matrix factorization (Vandereycken, 2013; Mishra et al., 2012). Our Riemannian approach can be viewed as an adaptation of the works of (Vandereycken, 2013; Absil et al., 2008) to similarity learning.

2. Preparation

We briefly recall the related notions of matrix manifold and retraction. Motivated readers can refer to (do Carmo, 1992; Lee, 2003; Absil et al., 2008) for further details.

Stiefel Manifold (Lee, 2003) Define the set of $\hat{n} \times d$ orthonormal matrices as $st(\hat{n}, d) := \{U \in \mathbb{R}^{\hat{n} \times d} : U^T U = I_d\}$, with I_d an identity matrix of size $d \times d$, and $\hat{n} \geq d$. It is a differentiable manifold that can be locally approximated by a set of Euclidean spaces. Consider an arbitrary point in the manifold, $U \in st(\hat{n}, d)$. To perform differential calculus, define the *tangent space* at U as $T_U st(\hat{n}, d)$, which is also a subset of $\hat{n} \times d$ matrices. It is easy to check $U^T \rho = -\rho U^T$ for any tangent vector $\rho \in T_U st(\hat{n}, d)$.

Retraction (Absil et al., 2008) Intuitively, *retraction* generalizes the notion of moving in the direction of a vector in Euclidean space to manifolds. An ideal retraction is the *exponential map* (Lee, 2003): In our context, the exponential map at point W, $R_W(\eta)$, maps a tangent vector $\eta \in T_W \mathcal{M}$ to a point in the manifold $\mathcal{M}$, as projecting along a geodesic curve started at W in the direction of η. In practice, usually a computationally less demanding retraction is used instead of the exact exponential map. Formally, a retraction on a manifold $\mathcal{M}$ at point W is a function

$R_W : T_W\mathcal{M} \to \mathcal{M}$ that satisfies two properties (Absil et al., 2008). (1) $R_W(0) = W$. (2) The geodesic curve defined by $\gamma_W(\bar{t}) := R_W(\bar{t}\eta)$ satisfies $\dot{\gamma}_W(0) = \eta$, with $\bar{t}$ being a real. In general, a retraction can approximate the exponential map at least to its first order Taylor expansion.

Vector Transport (do Carmo, 1992; Absil et al., 2008) In a way similar to retractions vs. exponential mapping, the vector transport (Absil et al., 2008) is an approximation of parallel transport (do Carmo, 1992), by transporting a tangent vector ξ from a point $W \in \mathcal{M}$ to a point $W' := R_W(\eta) \in \mathcal{M}$ along a geodesic curve defined by η. More specifically, it is defined w.r.t. an existing retraction R as $(\eta, \xi) \mapsto \mathcal{T}_\eta(\xi) \in T_{W'}\mathcal{M}$ for any $\eta, \xi \in T_W\mathcal{M}$ and $W' := R_W(\eta)$, satisfying $\mathcal{T}_0(\xi) = \xi$, and $\mathcal{T}_\eta(a\xi + b\varsigma) = a\mathcal{T}_\eta(\xi) + b\mathcal{T}_\eta(\varsigma)$, with $a, b \in \mathbb{R}$.

Riemannian Connection (do Carmo, 1992) and Riemannian Hessian (Absil et al., 2008) In an Euclidean space, A connection $\nabla_\eta \xi$ amounts to the directional derivative of ξ along η at a point W, as $\nabla_\eta \xi := \lim_{t\to 0} \frac{\xi_{W+t\eta} - \xi_W}{t}$, where ξ and η are both vector fields. This is the special case of affine connection on a manifold $\mathcal{M}$, which satisfies three properties: i) $\nabla_{f\eta + g\iota}\xi = f\nabla_\eta\xi + g\nabla_\iota\xi$, ii) $\nabla_\eta\big(a\xi + b\varsigma\big) = a\nabla_\eta\xi + b\nabla_\eta\varsigma$, and iii) Leibniz' rule: $\nabla_\eta\big(f\xi\big) = \big(\eta f\big)\xi + f\nabla_\eta\xi$. where $\eta, \iota, \xi, \varsigma \in T\mathcal{M}$, f, g are real functions of $\mathcal{M}$, and $a, b \in \mathbb{R}$. A Riemannian metric on a differentiable manifold is a correspondence that associates to each point $W \in \mathcal{M}$ an inner product, $\langle \cdot, \cdot \rangle_W$ in the tangent space of $\mathcal{M}_W$. A differentiable manifold with a given Riemannian metric is termed a Riemannian manifold. A *Riemannian Connection* is thus the unique affine connection ∇ of the Riemannian manifold $\mathcal{M}$ that is i) symmetric and ii) compatible with its Riemannian metric. The *Riemannian Hessian* is related to the Riemannian Connection as $\mathrm{Hess}J(W)[\eta] := \nabla_\eta \mathbf{Grad}_W J(W)$, for any η in the tangent space of $\mathcal{M}_W$.

Riemannian Trust Region Methods (Nocedal & Wright, 2006; Absil et al., 2008) Trust region methods (Nocedal & Wright, 2006) work by simultaneously choose the descending direction and the step-size, by explicitly approximating the objective function $J(W)$ with a quadratic model m_W. In analogy to its Euclidean space counterparts, at each iteration of the Riemannian trust region methods (Absil et al., 2008), the following trust-region subproblem is to be solved:

$$\min_{\xi \in T_W\mathcal{M}} m_W(\xi) \tag{1}$$
$$:= J(W) + \langle \mathbf{Grad}_W J(W), \xi \rangle + \frac{1}{2}\langle \mathrm{Hess}J(W)[\xi], \xi \rangle$$

subject to constraint $\langle \xi, \xi \rangle \le \Delta^2$, where Δ is current trust-region radius. The solution this subproblem provides a descending direction in the tangent space of current point $W \in \mathcal{M}$.

3. Our approach

In our context, an example is represented as a triplet $(\hat{x}, x, y)$, with pair of instances $\hat{x} \in \mathbb{R}^{\hat{n}}$ and $x \in \mathbb{R}^n$, as well as $y \in \{\pm 1\}$ depending on whether the labels of the two instances are equivalent. We consider a bilinear form $f_W : (\hat{x}, x) \mapsto \hat{x}^T W x$, with W a matrix of size $\hat{n}$ by n taking a fixed rank $d \ll \min\{\hat{n}, n\}$. In general, the Left Hand Side (LHS) feature vector's dimension $\hat{n}$ is not necessarily the same as the RHS dimension n, therefore W is rectangular. An important fact is the existence of a factorization $W = U\Sigma V^T$ by SVD (Golub & Loan, 1996): $\Sigma := \mathrm{diag}(\sigma_1, \ldots, \sigma_d)$ contains positive singular values *uniquely* determined by W, while U and V form the eigen-bases of WW^T and W^TW, respectively. As a result, through f_W the vector $\hat{x}$ is mapped to a d-dimensional vector space, $\Sigma^{\frac{1}{2}}U^T\hat{x}$, and x mapped to the *same* space via $\Sigma^{\frac{1}{2}}V^Tx$. This fact can be subsequently used to construct a hypothesis function h for prediction problems, for example, $h(\hat{x}, x) := \mathrm{sgn}(f_W(\hat{x}, x)) = \mathrm{sgn}\Big(\big(\Sigma^{\frac{1}{2}}U^T\hat{x}\big)^T\big(\Sigma^{\frac{1}{2}}V^Tx\big)\Big)$. Intuitively, $h(\hat{x}, x) = 1$ if the inner product is positive (i.e. along the same direction).

The problem is then ready to be formulated as solving an optimization problem over a set of t training triplets

$$\min_{W: rank(W) = d} J(W) := \frac{\lambda}{2}\Omega(W) + \sum_{i=1}^{t} l(\hat{x}_i, x_i, y_i; f_W) \tag{2}$$

A variety of differentiable loss function can be considered in our context w.r.t. $f_W(\hat{x}, x)$, including for example the squared hinge loss $l_{\mathrm{hinge}^2}(\hat{x}, x, y; f_w) = \max\{0, 1 - yf_W(\hat{x}, x)\}^2$, the log loss $l_{\log}(\hat{x}, x, y; f_w) = \log\big(1 + e^{-yf_W(\hat{x}, x)}\big)$, and the ridge loss $l_2(\hat{x}, x, y; f_w) = \frac{1}{2}\big(f_W(\hat{x}, x) - y\big)^2$. We have empirically experimented with a number of loss functions, and it turns out that the log loss consistently delivers a top performance. So from now on we focus our attention to the log loss. Note here we choose not to consider the non-differentiable loss functions such as the hinge loss, which are more intricate for convergence analysis. Nevertheless we would like to mention that the set of manifold subgradients can be obtained for non-differentiable functions, and certain types of subgradient descent algorithms are also known to converge (Ferreira & Oliveira, 1998). Furthermore, the regularizer term is necessary to ensure W a bounded matrix. In this paper we restrict our attention to the Frobenius norm, $\Omega(W) := \|W\|_F$.

Geometric Meaning of SVD Denote a set $\mathcal{M}_W := \{W = U\Sigma V^T\}$, s.t. $U \in \mathrm{st}(\hat{n}, d)$, $V \in \mathrm{st}(n, d)$, and $\Sigma = \mathrm{diag}(\sigma_1, \ldots, \sigma_d)$, with $\sigma_i > 0$, $\forall i$. In other words, $\mathrm{diag}(\sigma_1, \ldots, \sigma_d)$ denotes a $d \times d$ positive diagonal matrix. It has been shown by (Absil et al., 2008; Vandereycken, 2013) that $\mathcal{M}_W$ is indeed a Riemannian manifold, with its tangent space at point W being established by

$$T_W\mathcal{M} := \left\{ UMV^T + U_pV^T + UV_p^T \right\} \qquad (3)$$

for M, U_p, V_p being a $d \times d$, a $\hat{n} \times d$, and a $n \times d$ matrix respectively, with $U_p^T U = 0$, and $V_p^T V = 0$. The Riemannian metric becomes $\langle \zeta, \eta \rangle = tr(\zeta^T \eta)$, with $tr(\cdot)$ being the matrix trace.

From Euclidean gradient to Riemannian Gradient and Riemannian Hessian Denote the *Euclidean* gradient of our objective function w.r.t. W as $\mathrm{grad}_W J \in \mathbb{R}^{\hat{n} \times n}$, by (3) its *Riemannian* gradient is computed as a projection onto the tangent space of $\mathcal{M}_W$

$$\mathbf{Grad}_W J = P_U^{\mathcal{H}}\, \mathrm{grad}_W J\, P_V^{\mathcal{H}} + P_U^{\mathcal{V}}\, \mathrm{grad}_W J\, P_V^{\mathcal{H}} + P_U^{\mathcal{H}}\, \mathrm{grad}_W J\, P_V^{\mathcal{V}}$$

with shorthand notations $P_U^{\mathcal{H}} := UU^T$ and $P_U^{\mathcal{V}} := I - UU^T$ for U, as well as similar notations for V.

It turns out impossible in our context to compute an analytic form of the Riemannian Hessian $\mathrm{Hess}J(W)$. We instead perform finite difference approximation of the Hessian by forward difference of the Riemannian gradients (Nocedal & Wright, 2006; Absil et al., 2008), where the vector transport is utilized to move the Riemannian gradients from the forward point back to the tangent space of the current point.

Retraction For any tangent vector $\eta \in T_W\mathcal{M}$, its retraction $R_W(\eta)$ into the manifold is naturally determined by

$$R_W(\eta) := \operatorname*{argmin}_{X \in \mathcal{M}} \left\| W + \eta - X \right\|_F.$$

It is easy to verify that it results in a closed form solution

$$R_W(\eta) = \sum_{i=1}^{d} \sigma_i u_i v_i^T,$$

with σ_i, u_i and v_i being the i-th singular values and vectors of SVD of $W + \eta$, respectively.

RSL As an adaptation of the Riemannian trust-region method of (Absil et al., 2008), the proposed algorithm for Riemannian Similarity Learning, or *RSL* in short, is presented in Algorithm 1. At each iteration, the trust-region subproblem is solved following the truncated conjugate gradient method presented in (Absil et al., 2008) chapter 7, which produces a candidate tangent vector ξ_k. To decide whether to accept ξ_k as well as the new trust-region radius Δ_k, we evaluate the following ρ_k value

$$\rho_k := \frac{J(W_k) - J\big(R_W(\xi_k)\big)}{m_{W_k}(0) - m_{W_k}(\xi_k)}. \qquad (4)$$

Ideally ρ_k is expected be close to 1, then we accept ξ_k and the trust-region radius Δ_{k+1} may be enlarged; If ρ_k is however less than a small positive ρ', then the model is not accurate and in this case we must reject the candidate and reduce the trust-region radius; If ρ_k is not too small ($\geq \rho'$) but still away from 1 (*i.e.,* $\rho_k < \frac{1}{4}$), we accept the candidate while still reducing the radius; Lastly if $\rho_k \gg 1$, the model is inaccurate while there is a significant decrease in objective function J, we can choose to accept the candidate and increase the radius, in the hope of a greater decrease in J value. This explains the logic underlying Algorithm 1.

The initial model parameter W_0 is started randomly as $U_0\, I_d\, V_0^T$, with the orthonormal matrices U_0 and V_0 obtained from SVD of a randomly generated real matrix A of size $\hat{n} \times n$. Throughout experiments, the following parameters are fixed: $\overline{\Delta}$ is computed as $(\hat{n} + n) \times k$, $\Delta_0 = \frac{\overline{\Delta}}{4}$, $k_{\max} = 300$, $\rho' = 0.1$, and $\epsilon = 1e^{-4}$.

Kernelization It is straightforward to work with the kernelized input instances, by considering implicit feature maps $\phi : x \mapsto \phi(x)$ for x, and likewise $\hat{\phi}$ for $\hat{x}$. The kernelized version of $\hat{x}^T W x$ becomes $\hat{K}^{\frac{1}{2}} L K^{\frac{1}{2}}$, with $\hat{K}(i,j) = \hat{\phi}(\hat{x}_i)^T \hat{\phi}(\hat{x}_j)$, $K(i,j) = \phi(x_i)^T \phi(x_j)$, and L being a $t \times t$ similarity matrix. The corresponding kernelized optimization problem is thus written as

$$\min_{L:rank(L)=d} \frac{\lambda}{2}\Omega(L) + \sum_{i=1}^{t} l(\hat{\delta}_i^T \hat{K}^{\frac{1}{2}} L K^{\frac{1}{2}} \delta_i), \qquad (5)$$

Where $\hat{\delta}, \delta$ denote the selector functions, and for the log loss $l_{\log} := \log\left(1 + e^{-y_i \hat{\delta}_i^T \hat{K}^{\frac{1}{2}} L K^{\frac{1}{2}} \delta_i}\right)$. With little modification, the proposed Algorithm 1 can also be applied to learn the similarity matrix L. Denote $\hat{X}$ and X the column-stacked LHS and RHS examples during training, respectively. At test run, a new pair of examples $\hat{x}, x$ is observed, and its corresponding score can be computed as

$$\hat{\phi}(\hat{x})^T W \phi(x) = \hat{k}(\hat{x}, \hat{X})\, \hat{K}^{-\frac{1}{2}} L K^{-\frac{1}{2}}\, k(X, x),$$

Algorithm 1 Riemannian Similarity Learning: *RSL*

```
**Input:** The set of triplets $(\hat{x}, x, y)$
**Output:** $W \leftarrow W_k$
**Init:** $\epsilon$, $W_0$, $k_{\max}$, $\rho'$, $\overline{\Delta}$, $\Delta_0 \in (0, \overline{\Delta})$, and $k \leftarrow 0$
**repeat**
    Obtain $\xi_k$ by solving (1)
    Evaluate $\rho_k$ using (4)
    **if** $\rho_k > \rho'$ **then**
        $W_{k+1} \leftarrow R_W(\xi_k)$
    **else**
        $W_{k+1} \leftarrow W_k$
    **end if**
    **if** $\rho_k < \frac{1}{4}$ **then**
        $\Delta_{k+1} \leftarrow \frac{1}{4}\Delta_k$
    **else if** $\rho_k > \frac{3}{4}$ and $\|\xi_k\| = \Delta_k$ **then**
        $\Delta_{k+1} \leftarrow \min(2\,\Delta_k, \overline{\Delta})$
    **else**
        $\Delta_{k+1} = \Delta_k$
    **end if**
    $k \leftarrow k + 1$
**until** $\left\|\mathbf{Grad}_W J\right\|_F^2 < \epsilon$ or $k \geq k_{\max}$
```

with $\hat{k}(\hat{x}, \hat{X}) := \hat{\phi}(\hat{x})^T \hat{\phi}(\hat{X})$ a $1 \times t$ row vector, and $k(X, x) := \phi(X)^T \phi(x)$ a $t \times 1$ column vector, respectively.

Generalization Analysis One may concern the generalization ability of the proposed algorithm on unseen examples. Bounds on the generalization error is related to the capacity of the function class, which can be captured by its VC-dimension for 0-1 loss as well as its pseudo-dimension (Mohri et al., 2012) generalization for other loss functions. Denote $e := exp(1)$. It has been shown in (Srebro et al., 2004) that $\hat{d}$, the VC-dimension / pseudo-dimension of our matrix W, is upper bounded by:

$$d\,(\hat{n} + n)\,\log\left(\frac{16e\min\{\hat{n}, n\}}{d}\right). \tag{6}$$

Note that the VC-dimension of our rank-d matrix is much smaller than the VC-dimension of a full rank matrix which is roughly $\hat{n}$ by n. This is key in our scenario of learning a similarity function with satisfactory prediction performance. Following this guidance, in practice we usually choose a relative small d, *e.g.* $d = 10$.

Convergence Analysis Trust region methods in Euclidean space $\mathbb{R}^n$ are well-known to converge to local fixed points under mild conditions (see *e.g.* (Nocedal & Wright, 2006)). It is also known (Yang, 2007; Absil et al., 2008) that on matrix manifolds they enjoy similar convergence properties as their analogs in Euclidian space. Similarly it can be shown that our *RSL* algorithm is guaranteed to decrease the objective function value monotonically as iteration increases, thus converges to local fixed points. Moreover, it has also been proved by Theorem 7.4.11 of (Absil et al., 2008) that Riemannian trust region methods such as our *RSL* algorithm have a super-linear convergence rate.

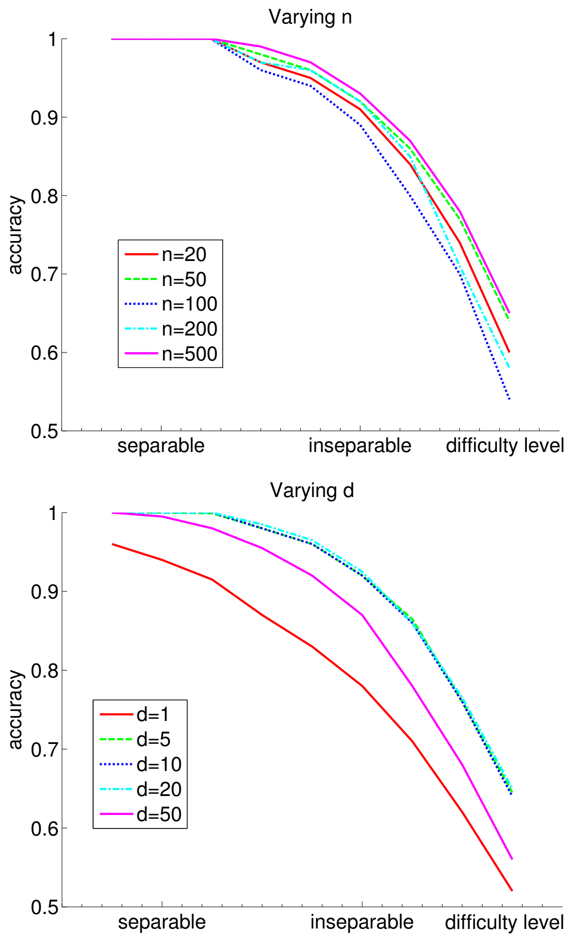

Figure 1. Synthetic experiments: Each curve displays the accuracy over increasingly more difficult datasets (from linearly separable to highly inseparable). Top: Comparison of performance by varying n, the dimension of RHS instances. Bottom: Comparison of performance by varying d, the rank of W. See text for details.

Computational Analysis At each iteration, the main computational load is from computing the Euclidean gradient $\text{grad}_W J$, its projection onto the manifold tangent space $\mathbf{Grad}_W J$, and the retraction step R_W. It takes up to $t\hat{n}n$ flops to compute the Euclidean gradient for our loss functions. Since $d \ll \min\{\hat{n}, n\}$, the computational cost of computing the manifold gradient is up to $4(\hat{n} + n)d^2$, and up to $14(\hat{n} + n)d^2$ for retraction. The overall complexity is

$O\Big(18(\hat{n}+n)d^2+t\hat{n}n\Big)$, with a mild constant when e.g. the maximum number of iterations $k_{\max}$ is considered.

4. Experiments

We first examine the proposed *RSL* algorithm on synthetic datasets. This is followed by evaluations on two real-life applications: the ASLAN challenge (Kliper-gross et al., 2012) of pairwise similarity prediction for action recognition, and domain adaptation for object recognition (Saenko et al., 2010). The following parameter values are used: The trade-off parameter λ is fixed to 1; Except for the synthetic experiments, the subspace dimension d is set to 10.

Synthetic Experiments Experiments are conducted on synthetic datasets, in order to analyze the characteristics of the *RSL* algorithm under controlled settings. As illustrated in both plots of in Fig 1, We focus on the accuracy measure as a function over increased difficulty levels, ranging from being linearly separable to highly inseparable. The synthetic datasets are constructed for binary classes, with each class being sampled from a Gaussian distribution with fixed covariance matrix. For the linearly separable cases, all instances that would be misclassified by the Bayes optimal classifier are dropped off. The difficulty level is increased as we decrease the ratio of between-class dispersions and within-class dispersions (similar to what has been used in Linear Discriminant Analysis (Bishop, 2006)). In practice, this is achieved by shortening the distance of two Gaussian means while keeping the rest unchanged. Unless stated explicitly, by default the following set-up is adopted: The number of training examples is $t = 5,000$ with a half for each class, the LHS feature vector's dimension is $\hat{n} = 50$ and for RHS $n = 100$, the subspace dimension (i.e. rank of W) is set to $d = 5$. Each accuracy value is obtained by averaging over 20 repeats.

The first experiment is carried out by varying the RHS dimension n while fixing other factors. As displayed in Fig 1 top panel, there is no clear trend of performance changes, although there are slight variations of accuracies w.r.t. different n. This supports our hypothesis that the influence of the two dimensions ($\hat{n}$ and n) being the same or not is insignificant.

To examine the influence of different subspace dimensions d (i.e. rank of W) toward the overall performance, we run a second experiment by varying d values, which is presented in Fig 1 bottom panel. The empirical results suggest that the model matrix W here resides in a low-dimensional subspace. Moreover, except for the extreme cases where d equals to 1 for taking a too small (or possibly too large) values, the overall performance seems otherwise rather close for a rather broad set of d values between 5 and 20. In other words, the performance is relatively stable over a rather broad range of matrix ranks for W. As a result, throughout the remaining experiments of this paper, we consider a fixed low rank of $d = 10$ for W.

Pairwise Action Recognition The ASLAN Challenge (Kliper-gross et al., 2012) is a recent action recognition benchmark. It contains 3,697 human action clips collected from YouTube, spanning over 432 unique action categories that is organized around a tree hierarchy from CMU Motion Capture Dataset (cmu). The sample distribution over categories is highly uneven, with 116 classes possessing only one video clip. Instead of explicitly performing a multiclass classification, the aim of the benchmark is to make "same" or "not-same" binary prediction each time on a pair of actions.

Following (Kliper-gross et al., 2012), from each video local salient points are obtained (Wang et al., 2009a), where each point possesses three types of descriptors: Histogram of Oriented Gradients (HoG), Histogram of Optical Flow (HoF), and a concatenation of both (referred to as HnF). k-means clustering method is then used to produce a codebook of c=5,000 codewords from the training data, which naturally gives rise to a bag-of-words (BoW) representation (Fei-Fei & Perona, 2005). An action video is thus collectively represented as a histogram vector of length c. We work with the view-2 benchmark (the major one for performance evaluation), where the categories are split into 10 disjoint subsets, to be used for 10-fold cross validation. For each subset, 300 same and 300 not-same pair of action videos are randomly selected. Together it forms a benchmark of 10-splits that each contains mutually exclusive action labels: If videos of a certain action appear in one split, no videos of that same action will appear in any other split.

Under the same protocol of (Kliper-gross et al., 2012), results of the proposed algorithm are produced and are compared with the state-of-the-art methods, which include the 13 methods reported in (Kliper-gross et al., 2012). To save space, for each experiment, only the best of the 13 results are reported here and is collectively referred to as "*best* of (Kliper-gross et al., 2012)". The detailed performance of these methods can be found in (Kliper-gross et al., 2012). The best result of (Kliper-Gross et al., 2011) on one-short similarity metric, using CSML initialization and cosine similarity score (Kliper-Gross et al., 2011), is also included and referred to as "*OSS-CS*".

The results of competing algorithms are shown in Table 1 on the basic ASLAN benchmark as well as in Table 2 on the ASLAN-KTH benchmark (Kliper-gross et al., 2012). Overall *RSL* outperforms the comparison methods over these different spatial-temporal features. Moreover, *RSL* is able to work reasonably well in scenarios where the LHS and RHS features are from different sources. This is in contrast with existing methods of this benchmark, which are unable to cope with such a situation. In particular, on the basic ASLAN benchmark, *RSL* produces 58.43% for HoG $\rightarrow$ HoF, and 60.19% for HoG $\rightarrow$ HnF, respectively.

Table 2. Comparison of pairwise action recognition accuracy (%) on ASLAN-KTH Benchmark. See text for details.

local features	best of (Kliper-gross et al., 2012)	*RSL*
HoG $\rightarrow$ HoG	82.78	**84.21**
HoF $\rightarrow$ HoF	85.44	**88.01**
HnF $\rightarrow$ HnF	90.00	**90.44**

Cross-domain Object Recognition We further evaluate on the cross-domain object recognition application, using the benchmark dataset of Berkeley31 (Saenko et al., 2010; Kulis et al., 2011). This Berkeley31 dataset contains images taken from three different image domains, with each having *31* object categories. The first domain consists of product images downloaded from Amazon, which are in a canonical pose and with a white background. The second domain comprises images taken with a digital SLR camera in office. They are high-resolution images with varying poses and backgrounds. The third domain includes low-resolution webcam images with varying poses and backgrounds. These three domains are also referred to as *amazon*, *dslr*, and *webcam*, respectively. A summary of the dataset is presented in Table 3. Our protocol follows that of (Kulis et al., 2011). Specifically, all images are resized to the same size (300×300) and converted to gray-scale, and are extracted to form SURF feature points. Each image is then represented in BoW: 800 codewords are used for *webcam* and *amazon* domains, and 600 for *dslr* domain.

The methods are evaluated on two experimental settings: (1) All 31 categories are available in both training and testing phases; (2) During training only the first 15 categories are available. The trained model is then evaluated in testing phase on the rest 16 categories. For training in both settings, *20* images per category from source domain and *3* images per category from target domain are randomly selected, while the rest is retained for evaluation. Since the number of training instances is much lower than the feature dimension, the competing methods considered here are applied with kernels. Here RBF kernel is used for all methods. The final results presented are averaged over 10 rounds of randomly sampled training examples.

Table 3. A summary of the object recognition dataset of Berkeley31.

Domains	# dims	# images
amazon	800	2,813
dslr	600	498
webcam	800	795

Table 4. Comparison of Cross-domain Object Recognition accuracy (%) on Berkeley31 with *dslr* as the default target domain. Rows 2-3 is for setting (1): All 31 categories are seen during training. Bottom row is for setting (2): Training on partial categories, and evaluating on the remaining (unseen) categories. See text for details.

Source	*SVM-t*	*Symm*	*ARC-t*	*HFA*	*RSL*
webcam	49.1	53.2	53.0	54.3	**54.6**
amazon	49.1	50.1	53.2	**55.4**	55.3
webcam		37.7	48.3		**51.8**

A list of competing methods is provided below. In particular, three state-of-the-art methods, namely *Symm* (Saenko et al., 2010), *ARC-t* (Kulis et al., 2011), and *HFA* (Duan et al., 2012), are considered:

SVM-t A standard SVM model is trained only with examples from the *target* domain.

Symm (Saenko et al., 2010) Labels from both domains are used to learn a *symmetric* transformation.

ARC-t (Kulis et al., 2011) Labels from both domains are used to learn a *asymmetric* transformation.

HFA (Duan et al., 2012) Labels from both domains are used by first projecting to a common subspace, followed by a feature augmentation step.

The average classification accuracies are reported in rows 2-3 of Table 4 for setting (1), and the bottom row for setting (2). Here 2 cross-domain tasks are considered: from *webcam* or *amazon* as "source domain" to *dslr* as the default "target domain". In *both* settings, our *RSL* method performs competitively comparing with the state-of-the-art methods. In particular, *RSL* outperforms the *ARC-t* method, which is a dedicated method (Kulis et al., 2011) for addressing the object recognition problem using this particular dataset, and is closely related to our approach. The

Table 1. Comparison of pairwise action recognition accuracy (%) on ASLAN Challenge. See text for details.

local features	*best* of (Kliper-gross et al., 2012)	*OSS-CS* (Kliper-Gross et al., 2011)	*RSL*
HoG $\to$ HoG	58.55	60.63	**61.84**
HoF $\to$ HoF	56.82	59.53	**61.75**
HnF $\to$ HnF	58.87	60.83	**62.61**

competitive performance can be largely attributed to the manifold-based matrix factorization approach we have adopted.

5. Summary and Outlook

In this paper, we consider a similarity-score based paradigm that can be used to address scenarios where either the class labels are only partially available in training, or the training and testing data are drawn from heterogeneous sources. Our aim in this context becomes that of learning a rectangular similarity matrix of a fixed rank. We formulate the problem as manifold-based optimization, propose a trust-region type algorithm, and apply to applications in recognizing visual objects and actions. Empirical performance on both applications suggest the competitiveness of the proposed approach.

For future work, we plan to develop and analyze improved algorithms, as well as investigate their applications into related problems such as bilinear learning.

Acknowledgements

This research was partially supported by A*STAR JCO grants (1231BEG034, 1231BFG040, 12302FG010, and 1231BFG047). We would like to thank Nengli Lim, Sinno Jialin Pan, and Hwee Kuan Lee for helpful discussions.

References

CMU motion capture database. http://mocap.cs.cmu.edu/.

Absil, P., Mahony, R., and Sepulchre, R. *Optimization Algorithms on Matrix Manifolds.* Princeton University Press, Princeton, NJ, 2008.

Akhter, I., Simon, T., Khan, S., Matthews, I., and Sheikh, Y. Bilinear spatiotemporal basis models. *ACM Trans. Graph.*, 31(2), 2012.

Bach, F., Jenatton, R., Mairal, J., and Obozinski, G. Optimization with sparsity-inducing penalties. *Foundations and Trends in Machine Learning*, 4(1):1–106, 2012.

Balcan, M. and Blum, A. On a theory of learning with similarity functions. In *IMCL*, 2006.

Bellet, A. *Supervised Metric Learning with Generalization Guarantees.* PhD thesis, University of Saint-Etienne, 2012.

Bellet, A., Habrard, A., and Sebban, M. Similarity learning for provably accurate sparse linear classication. In *ICML*, 2012.

Bi, J., Wu, D., Lu, L., Liu, M., Tao, Y., and Wolf, M. AdaBoost on low-rank PSD matrices for metric learning. In *CVPR*, 2011.

Bishop, C. *Pattern Recognition and Machine Learning.* Springer-Verlag, 2006.

Chen, H., Liu, T., and Fuh, C. Learning effective image metrics from few pairwise examples. In *ICCV*, 2005.

Davis, J., Kulis, B., Jain, P., Sra, S., and Dhillon, I. Information-theoretic metric learning. In *ICML*, 2007.

do Carmo, M. *Riemannian Geometry.* Springer, 1992.

Duan, L., Xu, D., and Tsang, I. Learning with augmented features for heterogeneous domain adaptation. In *ICML*, 2012.

Eldén, Lars. *Matrix Methods in Data Mining and Pattern Recognition.* SIAM, 2007.

Fei-Fei, L. and Perona, P. A Bayesian hierarchical model for learning natural scene categories. In *CVPR*, 2005.

Fei-fei, L., Fergus, R., and Perona, P. One-shot learning of object categories. *IEEE Trans. PMAI*, 28:594–611, 2006.

Ferreira, O. and Oliveira, P. Subgradient algorithm on Riemannian manifolds. *J. Optim. Theory Appl.*, 97:93–104, 1998.

Golub, G. and Loan, C. *Matrix Computations.* The Johns Hopkins University Press, 1996.

Huang, G., Ramesh, M., Berg, T., and Learned-Miller, E. Labeled faces in the wild: A database for studying face recognition in unconstrained environments. Technical Report 07-49, University of Massachusetts, Amherst, 2007.

Hubert, L., Meulman, J., and Heiser, W. Two purposes for matrix factorization: A historical appraisal. *SIAM Rev.*, 42(1):68–82, 2000.

Jain, P., Kulis, B., Davis, J., and Dhillon, I. Metric and kernel learning using a linear transformation. *JMLR*, 13: 519–547, 2012.

Kliper-Gross, O., Hassner, T., and Wolf, L. One shot similarity metric learning for action recognition. In *International conference on Similarity-based pattern recognition*, pp. 31–45, 2011.

Kliper-gross, O., Hassner, T., and Wolf, L. The action similarity labeling challenge. *IEEE Trans. PAMI*, 2012.

Kulis, B., Saenko, K., and Darrell, T. What you saw is not what you get: Domain adaptation using asymmetric kernel transforms. In *CVPR*, pp. 1785–1792. 2011.

Kumar, N., Berg, A., Belhumeur, P., , and Nayar, S. Attribute and simile classifiers for face verification. In *ICCV*, 2009.

Lee, J. *Introduction to Smooth Manifolds*. Springer, 2003.

Mensink, T., Verbeek, J., Perronnin, F., and Csurka, G. Metric learning for large scale image classication: Generalizing to new classes at near-zero cost. In *ECCV*, 2012.

Miller, E., Miller, E., Matsakis, N., and Viola, P. Learning from one example through shared densities on transforms. In *CVPR*, 2000.

Mishra, B., Meyer, G., Bonnabel, S., and Sepulchre, R. Fixed-rank matrix factorizations and riemannian low-rank optimization. Technical report, arXiv, 2012.

Mohri, M., Rostamizadeh, A., and Talwalkar, A. *Foundations of Machine Learning*. The MIT Press, 2012.

Nocedal, J. and Wright, S. *Numerical optimization*. Springer, 2006.

Pan, S. and Yang, Q. A survey on transfer learning. *IEEE Transactions on Knowledge and Data Engineering*, 22 (10):1345–1359, 2010.

Pirsiavash, H. and Ramanan, D. Steerable part models. In *CVPR*, 2012.

Pirsiavash, H., Ramanan, D., and Fowlkes, C. Bilinear classifiers for visual recognition. In *NIPS*, 2009.

Saenko, K., Kulis, B., F., M., and Darrell, T. Adapting visual category models to new domains. In *ECCV*, pp. 213–226, 2010.

Shalit, U., Weinshall, D., and Chechik, G. Online learning in the embedded manifold of low-rank matrices. *J. Mach. Learn. Res.*, 13:429–458, 2012.

Srebro, Nathan, Alon, Noga, and Jaakkola, Tommi. Generalization error bounds for collaborative prediction with low-rank matrices. In *NIPS*, 2004.

Tenenbaum, J. and Freeman, W. Separating style and content with bilinear models. *Neural Comput.*, 12(6):1247–1283, 2000.

Vandereycken, B. Low-rank matrix completion by riemannian optimization. *SIAM Journal on Optimization*, 2013. Accepted.

Wang, H., Ullah, M., Kläser, A., Laptev, I., and Schmid, C. Evaluation of local spatio-temporal features for action recognition. In *BMVC*, 2009a.

Wang, L., Sugiyama, M., Yang, C., Hatano, K., and Feng, J. Theory and algorithm for learning with dissimilarity functions. *Neural Comput.*, 21(5):1459–1484, 2009b.

Yang, Y. Globally convergent optimization algorithms on Riemannian manifolds: Uniform framework for unconstrained and constrained optimization. *Journal of Optimization Theory and Applications*, 132(2):245–265, 2007.

Yin, Q., Tang, X., and Sun, J. An associate-predict model for face recognition. In *CVPR*, 2011.

Ying, Y., Huang, K., and Campbell, C. Sparse metric learning via smooth optimization. In *NIPS*, 2009.