Parallel Field Ranking

Ming Ji[†], Binbin Lin[‡], Xiaofei He[‡], Deng Cai[‡], Jiawei Han[†]

[†]Department of Computer Science, University of Illinois at Urbana-Champaign, Urbana, IL, USA
[‡]State Key Lab of CAD&CG, College of Computer Science, Zhejiang University, China

[†]{mingji1, hanj}@illinois.edu, [‡]binbinlin@zju.edu.cn, {xiaofeihe, dengcai}@cad.zju.edu.cn

ABSTRACT

Recently, ranking data with respect to the intrinsic geometric structure (manifold ranking) has received considerable attentions, with encouraging performance in many applications in pattern recognition, information retrieval and recommendation systems. Most of the existing manifold ranking methods focus on learning a ranking function that varies smoothly along the data manifold. However, beyond smoothness, a desirable ranking function should vary monotonically along the geodesics of the data manifold, such that the ranking order along the geodesics is preserved. In this paper, we aim to learn a ranking function that varies linearly and therefore monotonically along the geodesics of the data manifold. Recent theoretical work shows that the gradient field of a linear function on the manifold has to be a parallel vector field. Therefore, we propose a novel ranking algorithm on the data manifolds, called Parallel Field Ranking. Specifically, we try to learn a ranking function and a vector field simultaneously. We require the vector field to be close to the gradient field of the ranking function, and the vector field to be as parallel as possible. Moreover, we require the value of the ranking function at the query point to be the highest, and then decrease linearly along the manifold. Experimental results on both synthetic data and real data demonstrate the effectiveness of our proposed algorithm.

Categories and Subject Descriptors

H.3.3 [**Information Storage and Retrieval**]: Information Search and Retrieval—*Retrieval models*

General Terms

Algorithms, Theory

Keywords

Manifold, Ranking, Vector field

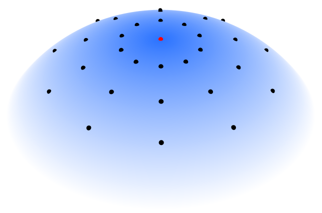
1. INTRODUCTION

Ranking is a fundamental problem in many areas including data mining, machine learning and information retrieval [17, 10, 6]. Given a query, we aim to learn a real-valued function f that ranks the data points according to their relevance to the query. In other words, for any two data points x_i and x_j, $f(x_i) > f(x_j)$ if x_i is more relevant to the query than x_j, and vice-versa.

In many real-world applications, one is often confronted with high dimensional data, such as images, documents and videos [21, 22]. However, there is a strong intuition that the high dimensional data may have a lower dimensional intrinsic representation. Various work in literature have considered the case where the data is sampled from a submanifold embedded in the ambient Euclidean space [2, 19, 23]. In this paper, we are interested in the ranking problem on the data represented by vectors in Euclidean space, under the manifold assumption [28, 1, 29].

Most of the existing ranking algorithms on data manifolds are based on the Laplacian regularization framework [28, 29], with promising performance observed on various data types such as image [9], video [27], and text [25]. These methods usually construct nearest neighbor graphs over the data to model the intrinsic geometric structure, and use the Graph Laplacian [5] to ensure that the ranking function varies smoothly along the manifold. They essentially spread the ranking scores via the graph iteratively until a stationary state is achieved. The Laplacian-based ranking framework has some nice interpretations including close relationships to personalized PageRank [28, 17] and HITS [10]. However, recent theoretical analysis [12, 13] shows that the Laplacian regularizer is way too general for measuring the smoothness of the function. Although it is ensured that data points connected by edges in the graph have similar ranking scores, the actual variation of the ranking function along the geodesics of the data manifold is unknown. Ideally, beyond smoothness, a desirable ranking function should vary monotonically along the geodesics of the manifold. Therefore, the ranking order of the data points along the geodesics of the manifold could be well preserved. Moreover, according to the nature of the ranking problem, no sample in the data set should be more relevant to the query than the query itself. So the ranking function should have the highest value at the query point, and then decrease to other points nearby.

In this paper, we propose a novel ranking algorithm on the data manifolds termed Parallel Field Ranking (PFRank), which learns a ranking function that has the highest value at the query point, and varies linearly and therefore mono-


(a) Data samples

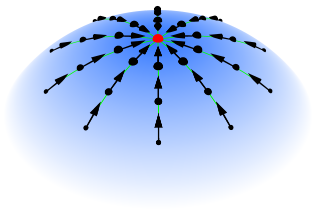

(b) Ranking results

Figure 1: We aim to design a ranking function that has the highest value at the query point marked by red, and then decreases linearly along the geodesics of the manifold, which is equivalent to its gradient field being parallel along the geodesics. The arrows above denote the gradient field of the ranking function, and the green lines denote the geodesics of the data manifold.

tonically along the geodesics of the data manifold. A function that varies linearly along the geodesics of the manifold is called a linear function on the manifold [18], which could still be nonlinear in the Euclidean space. It was shown that the gradient field of a linear function on the manifold has to be a parallel vector field (or parallel field in short) [18]. Besides, in order to ensure that the query has the highest ranking score, the gradient field of the ranking function should point to the query at the neighborhood around the query, since the value of a function increases towards the direction pointed by its gradient field. Figure 1 shows an example of the ranking function that we aim to learn on a data set sampled from a 2D sphere manifold in the 3D Euclidean space, where the red point denotes the query, the green lines denote the geodesics, and the arrows denote the gradient field of the ranking function. The size of each point denotes the ranking order generated by the ranking function, where large points are ranked higher than small ones. As can be observed, the ranking results generated by a function that has the highest value at the query point and varies linearly along the geodesics are quite reasonable. The gradient field of the function points to the query, and is parallel along the geodesics which pass through the query.

However, it is very difficult to design constraints on the gradient field of a function directly [13]. Instead, motivated by the recent progress in semi-supervised regression [13], we propose to learn a vector field to approximate the gradient field of the ranking function, and then design constraints on the vector field. In this paper, we propose to learn a ranking function and a vector field simultaneously based on three intuitions:

1. The vector field should be close to the gradient field of the ranking function.
2. The vector field should be as parallel as possible.
3. The vector field at the neighborhood around the query should all point to the query.

By encoding the above three intuitions, the gradient field of the learned ranking function should be as parallel as possible, and is forced to point to the query at the neighborhood around the query. Hence the ranking function learned by our method decreases linearly from the query to other points along the geodesics of the data manifold.

The rest of this paper is organized as follows. We briefly review some background knowledge related to ranking under the manifold assumption and vector fields in Section 2. Our Parallel Field Ranking algorithm encoding all the three intuitions proposed above is introduced in Section 3. Section 4 presents an optimization scheme that solves our objective function. Section 5 provides the experimental results on both synthetic data and real data. Finally, we conclude this work in Section 6.

2. BACKGROUNDS

Our algorithm is fundamentally based on vector fields in geometry and vector calculus. Also, for ranking methods under the manifold assumption, the most related work is the Laplacian-based manifold ranking [28]. In this section, we give a brief introduction of [28], and review some background knowledge related to vector fields.

2.1 Laplacian-based Manifold Ranking

Let $\mathcal{M}$ be a d-dimensional submanifold in the Euclidean space $\mathbb{R}^m$. Given n data points $\{x_1, \ldots, x_n\} \in \mathbb{R}^m$ on $\mathcal{M}$, where x_q is the query ($1 \leq q \leq n$), we aim to learn a ranking function $f : \mathcal{M} \to \mathbb{R}$, such that $\forall i, j \in \{1, \ldots, n\}$, $f(x_i) > f(x_j)$ if x_i is more relevant to the query x_q than x_j, and vice-versa.

The intuition behind [28] is that if two points x_i and x_j are sufficiently close to each other, then their ranking scores $f(x_i)$ and $f(x_j)$ should be close as well. Let $W \in \mathbb{R}^{n\times n}$ be a symmetric similarity matrix, and $y = [y_1, \ldots, y_n]^T$ be an initial ranking score vector which encodes some prior knowledge about the relevance of each data point to the query. Then [28] minimizes the following objective function:

$$J_{MR}(f) = \sum_{i,j=1}^{n} w_{ij}\left(\frac{f(x_i)}{\sqrt{D_{ii}}} - \frac{f(x_j)}{\sqrt{D_{jj}}}\right)^2 + \mu \sum_{i=1}^{n} (f(x_i) - y_i)^2 \tag{1}$$

where w_{ij} denotes the element at the i-th row and j-th column of W, $\mu > 0$ is a regularization parameter and D is a diagonal matrix used for normalization with $D_{ii} = \sum_{j=1}^{n} w_{ij}$. The first term in the above function aims to learn a ranking function that varies smoothly along the data manifold. The second term ensures the final ranking results to be close to the initial ranking score assignment y.

Let $f = [f(x_1), \ldots, f(x_n)]^T$. The closed form solution of Eq. (1) is the following:

$$f^* = (\mu I + L)^{-1} y \tag{2}$$

where I is an identity matrix of size $n \times n$. $L = I - D^{-1/2} W D^{-1/2}$ is the normalized Graph Laplacian [5]. We can also obtain the same solution via an iterative scheme:

$$f^{t+1} = \frac{1}{\mu + 1} S f^t + \frac{\mu}{\mu + 1} y \tag{3}$$

where each data point spreads the ranking score to its neighbors iteratively.

The above ranking framework based on Laplacian regularization has enjoyed long-lasting popularity in the community, with successful extensions in content-based image retrieval [9], ranking on both undirected and directed graphs [1], ranking on multi-typed interrelated web objects [8], ranking tags over a tag similarity graph [14], etc. However, for the Laplacian-based ranking framework, we do not exactly know how the ranking function varies, and whether the ranking order of the data points is preserved along the geodesics of the data manifold. [29] points out some drawbacks of [28] and proposes a more robust method by using an iterated graph Laplacian. But [29] still works under the Laplacian-based ranking framework, addressing the smoothness of the function. Besides, the Laplacian-based ranking framework requires the final ranking results to be close to the initial ranking score assignment y. In the situation when there is no prior knowledge about the relevance of each data point, people usually assign $y_i = 1$ if x_i is the query and $y_i = 0$ otherwise. This might not be very reasonable, and the final ranking results are likely to be biased towards the initial ranking score assignment.

2.2 Parallel Fields and Linear Functions

In geometry and vector calculus, a vector field is a mapping from a manifold $\mathcal{M}$ to tangent spaces [18]. We can think of a vector field on $\mathcal{M}$ as an arrow in the same way as we think of the vector field in Euclidean space, with a given magnitude and direction, attached to each point on $\mathcal{M}$, and chosen to be tangent to $\mathcal{M}$.

As discussed before, we aim to learn a ranking function $f : \mathcal{M} \to \mathbb{R}$ that varies linearly along the manifold $\mathcal{M}$. We first review the definitions of parallel fields and linear functions on the manifold [13].

Definition 1. **Parallel Field** [18]. A vector field X on manifold $\mathcal{M}$ is a parallel field if

$$\nabla X \equiv 0,$$

where ∇ is the covariant derivative on $\mathcal{M}$. ■

Definition 2. **Linear Function** [18]. A continuous function $f : \mathcal{M} \to \mathbb{R}$ is said to be linear if

$$(f \circ \gamma)(t) = f(\gamma(0)) + ct \tag{4}$$

for each geodesic γ. ■

In this paper, a function f is linear means that it varies linearly along the geodesics of the manifold. This definition is a natural extension of linear functions on the Euclidean space.

Then the following proposition reveals the relationship between a parallel field and a linear function on the data manifold:

Proposition 1. [18] Let V be a parallel field on the manifold. If it is also a gradient field for function f, $V = \nabla f$, then f is a linear function on the manifold. ■

3. PARALLEL FIELD RANKING

In this section, we propose the objective function which learns a ranking function that decreases linearly from the query to other points along the data manifold.

3.1 Ensuring the Linearity

As discussed before, we aim to design regularization terms that ensure the linearity of the ranking function with respect to the data manifold, which is equivalent to ensuring the parallelism of the gradient field of the function. However, it is very difficult to design constraints on the gradient field of a function directly [13]. Following the above analysis, we propose to learn a vector field to approximate the gradient field of the ranking function, and require the vector field to be as parallel as possible. Let $C^\infty(\mathcal{M})$ denote smooth functions on $\mathcal{M}$. Following [13], we learn a ranking function f and a vector field V on the manifold simultaneously with two constraints, which correspond to the first two intuitions proposed in Section 1:

- The vector field V should be close to the gradient field ∇f of f, which can be formularized as follows:

$$\min_{f \in C^\infty, V} R_1(f, V) = \int_{\mathcal{M}} \|\nabla f - V\|^2 \tag{5}$$

- The vector field V should be as parallel as possible:

$$\min_{V} R_2(V) = \int_{\mathcal{M}} \|\nabla V\|_F^2 \tag{6}$$

  where ∇ is the covariant derivative on the manifold, and $\|\cdot\|_F$ denotes the Frobenius norm.

∇V measures the change of the vector field V. If ∇V vanishes, then V is a parallel field. Then the following objective function learns a f that varies linearly along the manifold, from the vector field perspective [13]:

$$\begin{aligned} &\arg\min_{f \in C^\infty(\mathcal{M}), V} E(f, V) \\ =\; & R_0(x_q, y_q, f) + \lambda_1 R_1(f, V) + \lambda_2 R_2(V) \end{aligned} \tag{7}$$

where $\lambda_1 > 0$ and $\lambda_2 > 0$ are two regularization parameters. R_0 is a loss function that ensures the predicted ranking score for the query x_q to be close to a constant positive number y_q. If we remove R_0 from Eq. (7), then if f is an optimal solution to $\arg\min_{f \in C^\infty(\mathcal{M}), V} \lambda_1 R_1(f, V) + \lambda_2 R_2(V)$, it is easy to check that $f + c$ is also an optimal solution, where c could be any constant number. Therefore, by adding the R_0 term, we can get a unique solution of our ranking function f. In principle, y_q could be any positive number. Following [28, 13], we set $y_q = 1$, and use the squared loss $R_0(x_q, y_q, f) = (f(x_q) - y_q)^2$ for simplicity.

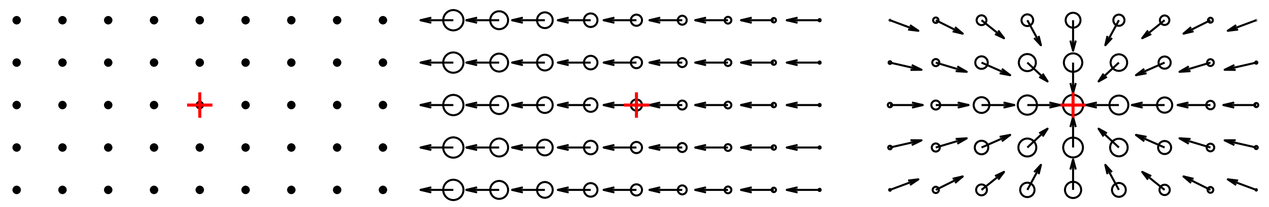
3.2 Discretization

Since the manifold $\mathcal{M}$ is unknown, the function f which minimizes Eq. (7) can not be directly solved. Following [13], we introduce how to discretize the continuous objective function (7) in this subsection.

For simplicity, let $f_i = f(x_i)$, $i = 1, \ldots, n$. Then our goal is to learn a ranking score vector $f = [f_1, \ldots, f_n]^T$. We first build a nearest neighbor graph $\mathcal{G}$ among the data. Let W be the corresponding affinity matrix of $\mathcal{G}$. There are many choices of the affinity matrix W. A simple definition is as follows:

$$w_{ij} = \begin{cases} 1 & \text{if } x_i \in N_k(x_j) \text{ or } x_j \in N_k(x_i) \\ 0 & \text{otherwise.} \end{cases} \tag{8}$$

where w_{ij} denotes the element at the i-th row and j-th column of matrix W. $N_k(x_i)$ denotes the set of k nearest neighbors of x_i. Then for each x_i, we can estimate its tangent space $T_{x_i}\mathcal{M}$ by performing PCA on its local neighborhood. We choose the eigenvectors corresponding to the d largest eigenvalues since $T_{x_i}\mathcal{M}$ is d-dimensional. Let $T_i \in \mathbb{R}^{m \times d}$ be the matrix whose columns constitute an orthonormal basis for $T_{x_i}\mathcal{M}$. It is easy to show that $P_i = T_i T_i^T$ is the *unique* orthogonal projection from $\mathbb{R}^m$ onto the tangent space $T_{x_i}\mathcal{M}$ [7]. That is, for any vector $a \in \mathbb{R}^m$, we have $P_i a \in T_{x_i}\mathcal{M}$ and $(a - P_i a) \perp P_i a$.



(a) Data samples (b) Ranking results without R_3 (c) Ranking results with R_3

Figure 2: A toy example explaining the reason of adding R_3.

Let V be a vector field on the manifold $\mathcal{M}$. For each point x_i, let V_{x_i} denote the value of the vector field V at x_i, and $\nabla V|_{x_i}$ denote the value of ∇V at x_i. According to the definition of vector field, V_{x_i} should be a vector in tangent space $T_{x_i}\mathcal{M}$. Therefore, it can be represented by the local coordinates of the tangent space, $V_{x_i} = T_i v_i$, where $v_i \in \mathbb{R}^d$. We define $\mathbb{V} = \left[v_1^T, \dots, v_n^T\right]^T \in \mathbb{R}^{dn}$. In other words, $\mathbb{V}$ is a dn-dimensional column vector concatenating all the v_i's.

Then according to the analysis in [13], R_1 reduces to the following:

$$R_1(f, \mathbb{V}) = \sum_{i,j=1}^{n} w_{ij} \left((x_j - x_i)^T T_i v_i - f_j + f_i \right)^2 \quad (9)$$

And R_2 can be discretized to the following:

$$R_2(\mathbb{V}) = \sum_{i,j=1}^{n} w_{ij} \left\| P_i T_j v_j - T_i v_i \right\|^2 \quad (10)$$

3.3 Objective Function in the Discrete Form

As discussed in Section 1, a function that varies linearly along the manifold is good for ranking. However, merely ensuring the linearity is not enough. Furthermore, we hope that the ranking function has the highest value at the query point, i.e., no sample in our data set is more relevant to the query than the query itself. Therefore, the ranking function should decrease from the query to neighboring data points, which is equivalent to that the gradient field of the ranking function at the neighborhood around the query should all point to the query. Figure 2 presents a simple toy problem to illustrate this idea. Suppose we are given a set of data points sampled from a 2D rectangle as shown in Figure 2(a). The query is marked by '+'. Figure 2(b) shows the ranking results without employing a third regularizer R_3 which addresses our third intuition, where the arrows denote the gradient field of the ranking function, and the size of the circle at each point denotes the ranking order. Points marked by large circles are ranked higher than those marked by small circles. It is easy to see that the ranking function learned in Figure 2(b) varies linearly, and its gradient field is a parallel field. However, some data points are even ranked higher than the query itself, and points far from the query could be ranked higher than those close to the query, which is not reasonable. By addressing the third intuition via R_3, we force the gradient field at the neighborhood around the query to point to the query while still requiring the gradient field to be parallel along the geodesics, generating good ranking results as shown in Figure 2(c).

Since we approximate the gradient field of the ranking function by the vector field V through R_1, we can easily control the gradient field using V. Therefore, we require V at the neighborhood around the query to point to the query. Let $i \sim j$ denote that x_i and x_j are neighbors. We now propose the third regularization term R_3, which addresses the third intuition proposed in Section 1:

$$\begin{aligned} R_3(\mathbb{V}) &= \sum_{j \sim q} \left\| V_{x_j} - P_j(x_q - x_j) \right\|^2 \\ &= \sum_{j \sim q} \left\| T_j v_j - P_j(x_q - x_j) \right\|^2 \quad (11) \end{aligned}$$

Recall that V_{x_j} denotes the value of the vector field V at x_j, which is a vector in tangent space $T_{x_j}\mathcal{M}$. $(x_q - x_j)$ is a vector pointing from a neighboring point x_j to x_q, and $P_j(x_q - x_j)$ is its projection to $T_{x_j}\mathcal{M}$. Therefore, ensuing V_{x_j} and $P_j(x_q - x_j)$ to be similar forces the vector field at x_j to point from x_j to x_q.

Let $\mathbb{I}$ denote a $n \times n$ matrix where the entry at the q-th row and q-th column is 1, and all the other entries are 0. Let $y \in \mathbb{R}^n$ be a column vector where the q-th entry is $y_q (= 1)$, and all the other entries are 0. Then we have:

$$R_0(x_q, y_q, f) = (f - y)^T \mathbb{I} (f - y) \quad (12)$$

Combining R_0 in Eq. (12), R_1 in Eq. (9), R_2 in Eq. (10) and R_3 in Eq. (11), our final objective function can be formulated as follows:

$$\begin{aligned} J(f, \mathbb{V}) &= R_0(x_q, y_q, f) + \lambda_1 R_1(f, \mathbb{V}) + \lambda_2 R_2(\mathbb{V}) + \lambda_3 R_3(\mathbb{V}) \\ &= (f - y)^T \mathbb{I} (f - y) \\ &\quad + \lambda_1 \sum_{i,j=1}^{n} w_{ij} \left((x_j - x_i)^T T_i v_i - f_j + f_i \right)^2 \\ &\quad + \lambda_2 \sum_{i,j=1}^{n} w_{ij} \left\| P_i T_j v_j - T_i v_i \right\|^2 \\ &\quad + \lambda_3 \sum_{j \sim q} \left\| T_j v_j - P_j(x_q - x_j) \right\|^2 \quad (13) \end{aligned}$$

The trade-off among the three regularization terms is controlled by the parameters λ_1, λ_2 and λ_3 in the range of $(0, +\infty)$.

4. OPTIMIZATION

In this section, we discuss how to solve our objective function (13).

Let D be a diagonal matrix where $D_{ii} = \sum_{j=1}^{n} w_{ij}$. Let $L = D - W$ denote the Laplacian matrix [5] of the graph. Then we can rewrite R_1 as follows:

$$\begin{aligned} R_1(f, \mathbb{V}) &= 2f^T L f + \sum_{i,j=1}^{n} w_{ij} \left((x_j - x_i)^T T_i v_i \right)^2 \\ &\quad - 2 \sum_{i,j=1}^{n} w_{ij} (x_j - x_i)^T T_i v_i s_{ij}^T f \quad (14) \end{aligned}$$

where $s_{ij} \in \mathbb{R}^n$ is a selection vector of all zero elements except for the i-th element being -1 and the j-th element being 1. We further construct a $dn \times dn$ block diagonal matrix G, and a $dn \times n$ block matrix $C = [C_1^T, \dots, C_n^T]^T$. Let G_{ii} denote the i-th $d \times d$ diagonal block of G, and C_i denote the i-th $d \times n$ block of C, we define:

$$G_{ii} = \sum_{j \sim i}^{n} w_{ij} T_i^T (x_j - x_i)(x_j - x_i)^T T_i \tag{15}$$

$$C_i = \sum_{j \sim i}^{n} w_{ij} T_i^T (x_j - x_i) s_{ij}^T \tag{16}$$

With some algebraic transformations, it is easy to check that R_1 in Eq. (14) can be written as follows:

$$R_1(f, \mathbb{V}) = 2f^T L f + \mathbb{V}^T G \mathbb{V} - 2\mathbb{V}^T C f \tag{17}$$

Similarly, in order to rewrite R_2 into a simplified form, we define $Q_{ij} = T_i^T T_j,\ i, j = 1, \dots, n$. Let I denote the identity matrix of size $n \times n$. We further construct a $dn \times dn$ sparse block matrix B. Let B_{ij} denote each $d \times d$ block, $i, j = 1, \dots, n$, then we define

$$B_{ii} = \sum_{j \sim i} w_{ij}(Q_{ij} Q_{ij}^T + I) \tag{18}$$

$$B_{ij} = -2 w_{ij} Q_{ij} \tag{19}$$

Then we can rewrite R_2 as follows:

$$R_2(\mathbb{V}) = \mathbb{V}^T B \mathbb{V} \tag{20}$$

For R_3, we construct a $dn \times dn$ block diagonal matrix D, and a $dn \times 1$ block vector $H = [H_1^T, \dots, H_n^T]^T$. Let D_{jj} denote the j-th $d \times d$ diagonal block of D, and H_j denote the j-th $d \times 1$ block of H. We define:

$$D_{jj} = \begin{cases} I_d, & \text{if } j \sim q \\ 0, & \text{otherwise} \end{cases} \tag{21}$$

$$H_j = \begin{cases} T_j^T (x_q - x_j), & \text{if } j \sim q \\ 0, & \text{otherwise} \end{cases} \tag{22}$$

where I_d is an identity matrix of size $d \times d$. Now we can rewrite R_3 as follows:

$$R_3(\mathbb{V}) = \mathbb{V}^T D \mathbb{V} - 2H^T \mathbb{V} + \sum_{j \sim q} \|P_j(x_q - x_j)\|^2 \tag{23}$$

Combining R_1 in Eq. (17), R_2 in Eq. (20) and R_3 in Eq. (23), we get the following simplified form of our objective function:

$$\begin{aligned} & J(f, \mathbb{V}) \\ = \ & (f - y)^T \mathbb{I} (f - y) \\ & + \lambda_1 (2f^T L f + \mathbb{V}^T G \mathbb{V} - 2\mathbb{V}^T C f) \\ & + \lambda_2 \mathbb{V}^T B \mathbb{V} \\ & + \lambda_3 (\mathbb{V}^T D \mathbb{V} - 2H^T \mathbb{V} + \sum_{j \sim q} \|P_j(x_q - x_j)\|^2) \\ = \ & f^T (\mathbb{I} + 2\lambda_1 L) f - 2y^T \mathbb{I} f - 2\lambda_1 \mathbb{V}^T C f \\ & + \mathbb{V}^T (\lambda_1 G + \lambda_2 B + \lambda_3 D) \mathbb{V} - 2\lambda_3 H^T \mathbb{V} \\ & + y^T \mathbb{I} y + \lambda_3 \sum_{j \sim q} \|P_j(x_q - x_j)\|^2 \end{aligned} \tag{24}$$

From Eq. (24), it is easy to compute the partial derivative of $J(f, \mathbb{V})$ with respect to f and $\mathbb{V}$ as follows:

$$\begin{aligned} & \frac{\partial J(f, \mathbb{V})}{\partial f} \\ = \ & 2(\mathbb{I} + 2\lambda_1 L) f - 2y - 2\lambda_1 C^T \mathbb{V} \end{aligned} \tag{25}$$

$$\begin{aligned} & \frac{\partial J(f, \mathbb{V})}{\partial \mathbb{V}} \\ = \ & -2\lambda_1 C f + 2(\lambda_1 G + \lambda_2 B + \lambda_3 D) \mathbb{V} - 2\lambda_3 H \end{aligned} \tag{26}$$

Requiring that the derivatives vanish, our solution could be obtained by solving the following linear equation system:

$$(\mathbb{I} + 2\lambda_1 L) f - \lambda_1 C^T \mathbb{V} = y \tag{27}$$

$$-\lambda_1 C f + (\lambda_1 G + \lambda_2 B + \lambda_3 D) \mathbb{V} = \lambda_3 H \tag{28}$$

which is further equivalent to solving the following linear system:

$$\begin{pmatrix} \mathbb{I} + 2\lambda_1 L & -\lambda_1 C^T \\ -\lambda_1 C & \lambda_1 G + \lambda_2 B + \lambda_3 D \end{pmatrix} \begin{pmatrix} f \\ \mathbb{V} \end{pmatrix} = \begin{pmatrix} y \\ \lambda_3 H \end{pmatrix} \tag{29}$$

5. EXPERIMENTS

In this section, we empirically evaluate the effectiveness of our proposed Parallel Field Ranking (PFRank) algorithm with comparison to several existing ranking algorithms. Since we focus on learning a ranking function under the manifold assumption, the main comparative method is the Laplacian-based manifold ranking algorithm (MR) [28]. We use the same graph structure according to Eq. (8) and empirically set the number of nearest neighbors to be 15 for both PFRank and MR. The parameter setting of MR is the same as the original paper [28, 9]. We empirically set $\lambda_1 = \lambda_2 = \lambda_3 = 0.01$ in our PFRank algorithm.

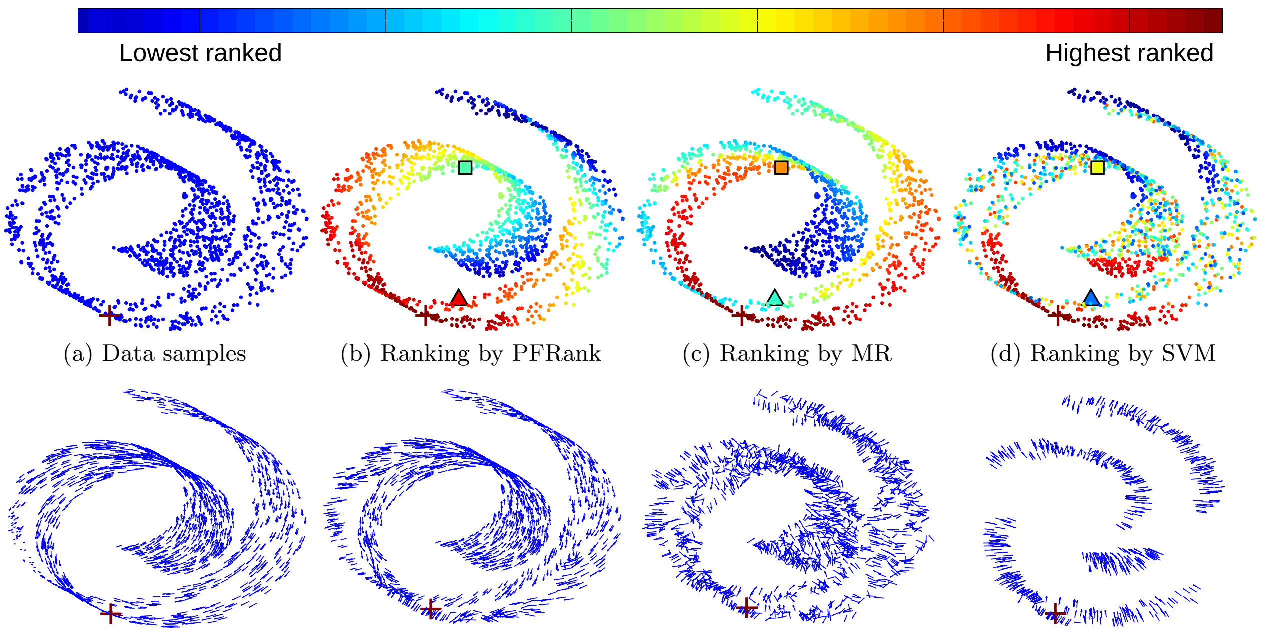
We also compare with the SVM method, which has been successfully applied to image retrieval [24]. With the specified relevant/irrelavant information, a maximal margin hyperplane could be built to separate the relevant data points from the irrelevant ones. Then we can return the data points farthest from the SVM boundary as the relevant ones. We use the LIBSVM toolbox [4] in our experiments. However, with a single query, there are no irrelevant data specified by the user. Following [26], we adopt the pseudo relevance feedback strategy [15]. Specifically, the nearest 10 data points to the query measured by the Euclidean distance are considered to be relevant, and the farthest 10 are treated as irrelevant. Then we can run SVM for ranking.

In the following, we begin with a simple synthetic example to give some intuition about how PFRank works.

5.1 Synthetic Example

A simple synthetic example is given in Figure 3. We randomly sample 1500 data points from a Swiss roll with a hole as shown in Figure 3(a). This data set lies on a 2D manifold in the 3D Euclidean space. The query is marked by '+'.

Figure 3(b)∼(d) show the ranking order generated by PFRank, MR and SVM, respectively, where the data points marked by warmer color (such as red) are ranked higher than those marked by colder color (such as blue). We can see that SVM does not take the manifold structure of the data into consideration. The ranking functions generated by both PFRank and MR vary smoothly along the data manifold, and PFRank better preserves the ranking order of the data points along the geodesics of the manifold. For example, the



(a) Data samples (b) Ranking by PFRank (c) Ranking by MR (d) Ranking by SVM

(e) Vector field by PFRank (f) Gradient field by PFRank (g) Gradient field by MR (h) Gradient field by SVM

Figure 3: Performance comparison of various ranking algorithms on a toy data set. (a) shows the data set sampled from a Swiss roll with a hole, where the query is denote by '+'. (b) shows the ranking order generated by PFRank. We can see that PFRank successfully preserves the ranking order of the data points along the geodesics of the manifold. For example, the geodesic distance between the query and the point marked by '▲' is smaller than that between the query and the point marked by '■'. So '▲' is ranked higher than '■'. (c) shows the ranking order generated by MR. We can see that '▲' is ranked lower than '■', which is counterintuitive. (d) shows the ranking order generated by SVM, which does not take the manifold structure into consideration. (e) and (f) show the vector field and the gradient field of the ranking function learned by PFRank, respectively, which are quite parallel and point to the query. (g) and (h) show the gradient fields of the ranking functions learned by MR and SVM, respectively, which do not vary smoothly along the manifold.

geodesic distance between the query and the triangle point is smaller than that between the query and the square point. Therefore, the triangle point should rank higher than the square point. However, MR ranks the square point higher than the triangle point, which is counterintuitive.

Note that once the ranking function f is obtained, we can estimate its gradient field reversely. Specifically, the gradient field at each point x_i $(1 \leq i \leq n)$, denoted by $\nabla f|_{x_i}$, can be computed by minimizing the following objective function at the local neighborhood of x_i:

$$\begin{aligned}&\nabla f|_{x_i}\\ =\;&\arg\min_{g\in\mathbb{R}^m}\sum_{j\sim i}\big(f(x_j)-f(x_i)-g^TP_i(x_j-x_i)\big)^2 \quad (30)\end{aligned}$$

We further plot the vector field learned by PFRank in Figure 3(e), and plot the gradient fields of the ranking functions learned by PFRank, MR and SVM (computed via Eq. (30)) in Figure 3(f)∼(h), respectively. For SVM, many data points have the same ranking scores (i.e., same distance to the SVM boundary), therefore are ranked randomly. And the gradient field of the ranking function has 0 values at many places, therefore are left blank in Figure 3(h). Note that the value of a ranking function increases towards the direction pointed by its gradient field. Therefore, the ranking orders of data points get higher towards the direction pointed by the arrows of the gradient field. It can be observed that both the vector field and the gradient field of the ranking function learned by PFRank point to the query along the geodesics of the manifold, and are quite parallel. On the contrary, the gradient fields of the ranking functions learned by M-R and SVM do not vary smoothly along the manifold, and therefore cannot preserve the ranking order.

5.2 Image Retrieval

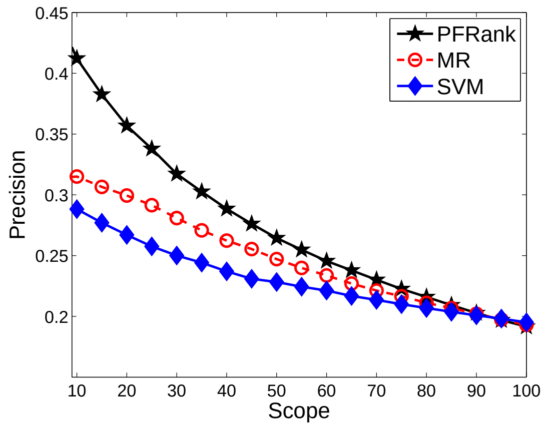
In this subsection, we apply our proposed ranking algorithm to the image retrieval problem in real world image databases. Given an image as a query, we hope to rank the images in our database according to their relevance to the query. We begin with a description of the data preparation.

5.2.1 Data Preparation

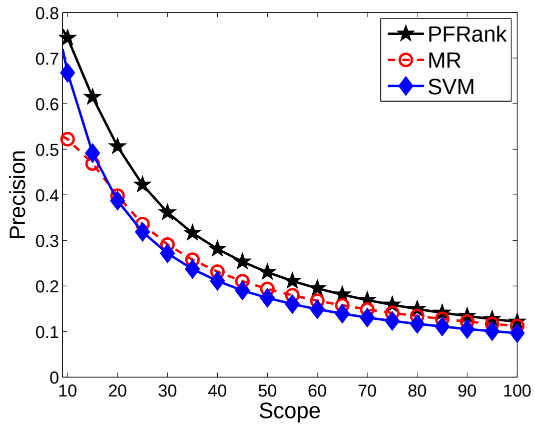
Two real world data sets are used in our experiments. The first one contains 5,000 images of 50 semantic categories, from the COREL database. For each image, we extract a 297-dimensional feature vector which combines the following information:

- Grid Color Moment: Each image is partitioned into 3×3 grids. For each grid, there color moments: mean, variance and skewness are extracted in each color channel (R, G, and B) respectively. Thus, an 81-dimensional grid color moment vector is adopted.

- Edge: The Canny edge detector [3] is used to obtain



(a) COREL



(b) CMU PIE

Figure 4: The average precision-scope curves of different algorithms on the two data sets.

Table 1: Performance comparison using different evaluation metrics on the COREL data set

	NDCG@10	NDCG @20	Recall@10	Recall@20	Recall@50	Recall@100	MAP
SVM	0.297	0.279	0.066	0.121	0.242	**0.404**	0.244
MR	0.312	0.302	0.061	0.116	0.237	0.371	0.237
PFRank	**0.435**	**0.388**	**0.091**	**0.152**	**0.272**	0.394	**0.256**

Table 2: Performance comparison using different evaluation metrics on the CMU PIE data set

	NDCG@10	NDCG @20	Recall@10	Recall@20	Recall@50	Recall@100	MAP
SVM	0.761	0.530	0.503	0.570	0.631	0.695	0.575
MR	0.537	0.446	0.383	0.583	0.698	0.806	0.441
PFRank	**0.786**	**0.605**	**0.572**	**0.750**	**0.847**	**0.889**	**0.729**

the edge map for the edge orientation histogram, which is quantized into 36 bins of 10 degrees each. An additional bin is to count the number of pixels without edge information. Hence, a 37-dimensional vector is used.

- Gabor Wavelets Texture: Each image is first scaled to 64× 64 pixels. The Gabor wavelet transform [11] is then applied on the scaled image with 5 levels and 8 orientations, which results in 40 subimages. For each subimage, 3 moments are calculated: mean, variance and skewness. Thus, a 120-dimensional vector is used.

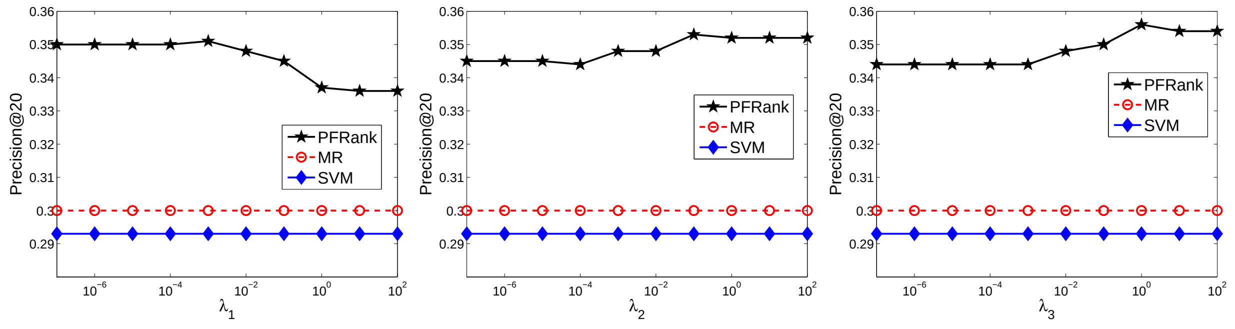
- Local Binary Pattern: The LBP [16] is a gray-scale texture measure derived from a general texture definition in a local neighborhood. A 59-dimensional LBP histogram vector is adopted.

The second data set is from the CMU PIE face database [20]. This database contains 68 subjects with 41,368 face images as a whole. The face images were captured by 13 synchronized cameras and 21 flashes, under varying pose, illumination and expression. In this experiment, we choose the frontal pose (C27) with varying lighting conditions, which leaves us 21 images per subject. Preprocessing to locate the faces were applied. Original images were normalized (in scale and orientation) such that the two eyes were aligned at the same position. Then the facial areas were cropped into the final image for matching. The size of each cropped image in all the experiments is 32 $\times$ 32 pixels, with 256 gray levels per pixel. Therefore, each image can be represented by a 1024-dimensional feature vector in the image space. No further preprocessing is done.

5.2.2 *Experimental Settings*

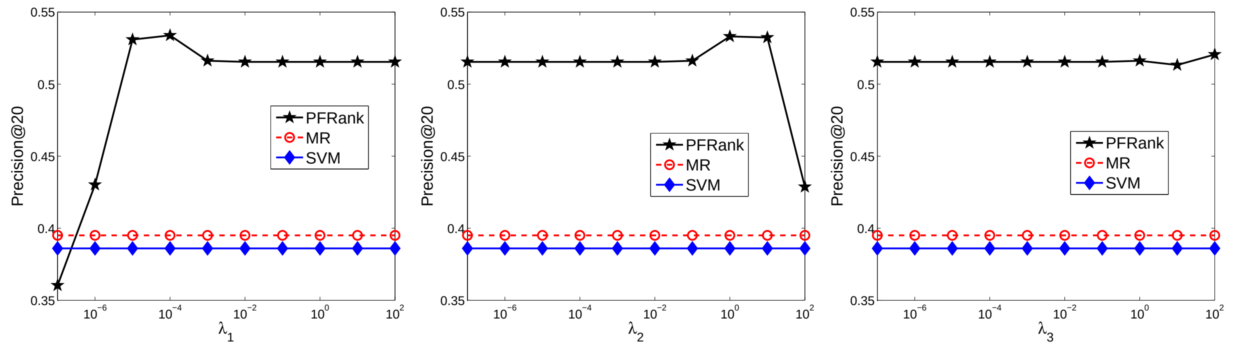
We describe our experimental setup in this subsection. Both of the two image data sets we use have category labels. In the COREL data set, images from the same category belong to the same semantic concept, such as eagle, bird, elephant, etc. In CMU PIE data set, images from the same category belong to the same person (subject). Therefore, given a query, images that belong to the same category as the query are judged relevant. For each data set, we randomly choose 10 images from each category as queries, and average the retrieval performance over all the queries. It is worth noticing that both PFRank and MR need to construct a nearest neighbor graph among the data and invert a matrix whose size is at least $n \times n$, which is time consuming. In order to make the ranking scheme more efficient, given a query image, we first rank all the images according to the Euclidean distance to the query. Then we choose the top 500 images as candidates and use different ranking algorithms to re-rank them.

We use precision, recall, Mean Average Precision (MAP) [15] and Normalized Discount Cumulative Gain (NDCG) to evaluate the ranking results of different algorithms. Precision is defined as the number of relevant presented images divided by the number of presented images. Recall is defined as the number of relevant presented images divided by the total number of relevant images in our database. Given a query, let r_i be the relevance score of the image ranked at position i, where $r_i = 1$ if the image is relevant to the query



(a) Varying λ_1 (b) Varying λ_2 (c) Varying λ_3

Figure 5: Model selection on the COREL data set.



(a) Varying λ_1 (b) Varying λ_2 (c) Varying λ_3

Figure 6: Model selection on the CMU PIE data set.

and $r_i = 0$ otherwise. Then we can compute the Average Precision (AP):

$$\text{AP} = \frac{\sum_i r_i \times \text{Precision@}i}{\#\text{ of relevant images}} \tag{31}$$

MAP is the average of AP over all the queries. And NDCG at position n is defined as:

$$\text{NDCG@}n = Z_n \sum_{i=1}^{n} \frac{2^{r_i} - 1}{\log_2(i+1)} \tag{32}$$

n is also called the scope, which means the number of top-ranked images presented to the user. Z_n is chosen such that the perfect ranking has a NDCG value of 1.

For our PFRank algorithm, the dimensionality of the manifold (d) in the real data is unknown. We perform cross-validation and choose $d = 2$ for the COREL data set, and choose $d = 9$ for the CMU PIE data set, respectively. All the other parameter settings are the same as in the previous experiment.

5.2.3 *Performance Evaluation*

Figure 4 shows the average precision-scope curves of various methods on the two data sets, respectively. The precision-scope curve describes the precision with various scopes, and therefore providing an overall performance evaluation of the algorithms. As can be seen, our proposed PFRank algorithm consistently outperforms the other two algorithms on both data sets. MR generally achieves higher precision than SVM, indicating that considering the manifold structure in the data is useful for image retrieval.

In order to have a comprehensive view of the ranking performance, we present the NDCG, recall and MAP scores of different algorithms on the COREL and CMU PIE data set in Table 1 and Table 2, respectively. Overall, our PFRank method performs the best on both data sets. Although the precision of MR is generally higher than SVM in Figure 4, the NDCG scores of MR are lower than that of SVM on the CMU PIE data set. This is because the precision@n of MR is lower than that of SVM when $n < 20$, and NDCG computes the cumulative relevance scores of the top ranked images. Moreover, the recall and MAP of MR are lower than SVM on the COREL data set. MAP provides a single figure measure of quality across recall levels. Our PFRank achieves the highest MAP on both data sets, indicating reliable performance over the entire ranking list.

5.2.4 *Model Selection*

Model selection is a critical problem in most of the learning problems. In some situations, the learning performance may drastically vary with different choices of the parameters. λ_1, λ_2 and λ_3 are essential parameters in PFRank which control the trade-off among the three regularization terms. We empirically set $\lambda_1 = \lambda_2 = \lambda_3 = 0.01$ in all the previous experiments. In this subsection, we try to study the impact of the parameters on the performance of our PFRank algorithm. Specifically, we fix other parameters the same as before, and let one of $\{\lambda_1, \lambda_2, \lambda_3\}$ vary.

In general, it is appropriate to present 20 images on a screen. Putting more images on a screen might affect the quality of the presented images. Therefore, precision@20 is especially important. Figure 5 and Figure 6 show the precision@20 scores of PFRank with respect to different values of λ_1, λ_2 and λ_3, respectively. As can be seen, our

PFRank algorithm is generally not very sensitive to the parameters, and outperforms the other two methods over a wide range of parameters.

6. CONCLUSIONS

In this paper, we propose a novel ranking algorithm on the data manifold from the vector field perspective. Motivated by recent study about the relationship between vector fields and functions on the manifold, we employ vector fields to ensure the linearity of the ranking function with respect to the manifold, as well as requiring the predicted ranking score of the query to be higher than that of its neighboring data points. In this way, the ranking function learned by our method decreases linearly, and therefore monotonically from the query to other points along the geodesics of the data manifold. Hence the ranking order along the geodesics is well preserved. Experimental results on both synthetic data and real data demonstrate the superiority of our method over existing ranking schemes.

There are several interesting directions for extending this work. One important problem is how to efficiently learn a ranking function from the vector field perspective for large-scale retrieval problems. The algorithm proposed in this paper needs to build a nearest neighbor graph among the data, and involves the matrix inversion operation. Given a query that is out of our database, how to update the nearest neighbor graph and compute the ranking scores efficiently is critical for introducing this technique to real search engines. We may also consider employing distributed computing techniques to further improve the efficiency. Finally, vector fields are very useful to study the geometry and topology of the data manifold. It will be interesting to explore the application of vector fields in many other real-world problems under the manifold assumption, such as face recognition, manifold alignment, recommendation systems and so on.

7. ACKNOWLEDGMENTS

The work was supported in part by the U.S. Army Research Laboratory under Cooperative Agreement No. W911NF-09-2-0053 (NS-CTA), NSF IIS-1017362, U.S. Air Force Office of Scientific Research MURI award FA9550-08-1-0265, and National Basic Research Program of China (973 Program) under Grant 2009CB320801. The views and conclusions contained in this paper are those of the authors and should not be interpreted as representing any funding agencies.

8. REFERENCES

[1] S. Agarwal. Ranking on graph data. In *ICML*, pages 25–32, 2006.

[2] M. Belkin and P. Niyogi. Laplacian eigenmaps and spectral techniques for embedding and clustering. In *NIPS 14*, pages 585–591. 2001.

[3] J. Canny. A computational approach to edge detection. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 8(6):679–698, 1986.

[4] C.-C. Chang and C.-J. Lin. *LIBSVM: a library for support vector machines*, 2001. Software available at http://www.csie.ntu.edu.tw/∼cjlin/libsvm.

[5] F. R. K. Chung. *Spectral Graph Theory*, volume 92 of *Regional Conference Series in Mathematics*. AMS, 1997.

[6] B. Gao, T.-Y. Liu, W. Wei, T. Wang, and H. Li. Semi-supervised ranking on very large graphs with rich metadata. In *KDD*, pages 96–104, 2011.

[7] G. H. Golub and C. F. V. Loan. *Matrix computations*. Johns Hopkins University Press, 3rd edition, 1996.

[8] Z. Guan, J. Bu, Q. Mei, C. Chen, and C. Wang. Personalized tag recommendation using graph-based ranking on multi-type interrelated objects. In *SIGIR*, pages 540–547, 2009.

[9] J. He, M. Li, H.-J. Zhang, H. Tong, and C. Zhang. Manifold-ranking based image retrieval. In *ACM Multimedia*, New York, October 2004.

[10] J. Kleinberg. Authoritative sources in a hyperlinked environment. *Journal of the ACM*, 46(5):604–622, 1999.

[11] M. Lades, J. C. Vorbrüggen, J. M. Buhmann, J. Lange, C. von der Malsburg, R. P. Würtz, and W. Konen. Distortion invariant object recognition in the dynamic link architecture. *IEEE Transactions on Computers*, 42(3):300–311, 1993.

[12] J. Lafferty and L. Wasserman. Statistical analysis of semi-supervised regression. In *NIPS 20*, pages 801–808, 2007.

[13] B. Lin, C. Zhang, and X. He. Semi-supervised regression via parallel field regularization. In *NIPS 24*, pages 433–441. 2011.

[14] D. Liu, X.-S. Hua, L. Yang, M. Wang, and H.-J. Zhang. Tag ranking. In *WWW*, pages 351–360, 2009.

[15] C. D. Manning, P. Raghavan, and H. Schtze. *Introduction to Information Retrieval*. Cambridge University Press, New York, NY, USA, 2008.

[16] T. Ojala, M. Pietikäinen, and D. Harwood. A comparative study of texture measures with classification based on featured distributions. *Pattern Recognition*, 29(1):51–59, 1996.

[17] L. Page, S. Brin, R. Motwani, and T. Winograd. The pagerank citation ranking: Bringing order to the web. Technical report, Stanford University, 1998.

[18] P. Petersen. *Riemannian Geometry*. Springer, New York, 1998.

[19] S. Roweis and L. Saul. Nonlinear dimensionality reduction by locally linear embedding. *Science*, 290(5500):2323–2326, 2000.

[20] T. Sim, S. Baker, and M. Bsat. The cmu pose, illumination, and expression database. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 25(12):1615–1618, 2003.

[21] D. Tao, X. Li, X. Wu, and S. J. Maybank. Geometric mean for subspace selection. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 31(2):260–274, 2009.

[22] D. Tao, X. Tang, X. Li, and Y. Rui. Direct kernel biased discriminant analysis: A new content-based image retrieval relevance feedback algorithm. *IEEE Transactions on Multimedia*, 8(4):716–727, 2006.

[23] J. Tenenbaum, V. de Silva, and J. Langford. A global geometric framework for nonlinear dimensionality reduction. *Science*, 290(5500):2319–2323, 2000.

[24] S. Tong and E. Y. Chang. Support vector machine active learning for image retrieval. In *ACM Multimedia*, pages 107–118, 2001.

[25] X. Wan, J. Yang, and J. Xiao. Manifold-ranking based topic-focused multi-document summarization. In *IJCAI*, pages 2903–2908, 2007.

[26] B. Xu, J. Bu, C. Chen, D. Cai, X. He, W. Liu, and J. Luo. Efficient manifold ranking for image retrieval. In *SIGIR*, pages 525–534, 2011.

[27] X. Yuan, X.-S. Hua, M. Wang, and X. Wu. Manifold-ranking based video concept detection on large database and feature pool. In *ACM Multimedia*, pages 623–626, 2006.

[28] D. Zhou, J. Weston, A. Gretton, O. Bousquet, and B. Schölkopf. Ranking on data manifolds. In *NIPS 16*, 2003.

[29] X. Zhou, M. Belkin, and N. Srebro. An iterated graph laplacian approach for ranking on manifolds. In *KDD*, pages 877–885, 2011.