High-order Tensor Regularization with Application to Attribute Ranking

Kwang In Kim  
University of Bath

Juhyun Park  
Lancaster University

James Tompkin  
Brown University

Abstract

When learning functions on manifolds, we can improve performance by regularizing with respect to the intrinsic manifold geometry rather than the ambient space. However, when regularizing tensor learning, calculating the derivatives along this intrinsic geometry is not possible, and so existing approaches are limited to regularizing in Euclidean space. Our new method for intrinsically regularizing and learning tensors on Riemannian manifolds introduces a surrogate object to encapsulate the geometric characteristic of the tensor. Regularizing this instead allows us to learn non-symmetric and high-order tensors. We apply our approach to the relative attributes problem, and we demonstrate that explicitly regularizing high-order relationships between pairs of data points improves performance.

1. Introduction

Learning tensors from data has many applications in function learning. Regression, classification, and clustering pose the function as a zeroth-order tensor; vector field learning poses the vector as a first-order tensor [41, 35]; and metric or covariance learning pose the metric as a symmetric second-order tensor [39, 37].

The generalization performance of the learned tensor $h$ depends crucially on how it is regularized—how the spatial smoothness of $h$ is enforced. In many problems, data lie on low-dimensional manifolds [29, 25, 32], for which it helps to regularize $h$ with respect to the intrinsic geometry of the data generating manifold $M$: to enforce smoothness along $M$ rather than in the ambient (Euclidean) space on which $M$ is embedded. This has shown improvement for semi-supervised learning, and spectral embedding and clustering [36, 3, 6].

While Stokes’ theorem allows us to perform intrinsic regularization of zeroth-order tensors (functions) on manifolds $M$, extending this idea to higher-order tensors is not straightforward: as $M$ itself is not directly observed, calculating the covariant derivatives—the derivatives along $M$—is not possible. Thus, existing tensor regularization approaches are limited to the special case of Euclidean space [35, 30, 11, 12], the solutions to which cannot be simply applied to general manifold-structured data.

We present a method to intrinsically regularize and learn tensors on Riemannian manifolds. As manifolds are not directly observed in practice, our strategy is to introduce a surrogate object—a kernel function—that encapsulate the geometric characteristic of the tensor. We estimate this kernel function from a point cloud sampled from $M$, and regularize this instead. In contrast to existing approaches for intrinsic tensor regularization which can only learn symmetric positive definite tensors [18], we can learn general non-symmetric tensors and high-order tensors.

To help the novice reader, our supplemental material provides an introduction to regularization on Riemannian manifolds, compares Euclidean and manifold regularization, and discusses the challenge of regularizing tensors directly.

1.1. Application to relative attribute ranking.

We demonstrate our approach by learning a linear ordering, which can be defined by specifying pairwise relations between all data points, and can be represented by a second-order anti-symmetric tensor.

Problem description. Binary labels which describe the presence or absence of image objects or attributes are often insufficient for many tasks [20, 40]. Imagine shopping for shoes: there is no clear boundary between ‘pointy’ and ‘not pointy’ shoes even though it is easy for a human to state that one shoe is ‘pointier’ than another. Thus, measuring relative attributes [28] broadens attribute-based image analysis to abstract and non-categorical labels.

This is accomplished by asking users to describe the relationship between pairs of data points, either as equal or with an order (greater/less than): image $x(i)$ and $x(j)$ share the same amount of attribute $A$; or, $x(i)$ exhibits a stronger/weaker presence of attribute $A$ than $x(j)$. While Parikh and Grauman focus on binary classification [28], the technique can be thought of as implicitly introducing a linear ordering to a dataset for a given attribute: an ordering function $f$ is learned such that $f(x(i)) > f(x(j))$ implies that the rank of $x(i)$ is higher than that of $x(j)$.

Relation to existing techniques. Rank learning relates to the classical data retrieval problem of matching a query to a database (Agresti [1] and Liu [24] survey the field). While data retrieval is framed as the binary attribute problem of splitting matches from non-matches, constructing a perfect binary classifier is challenging. As such, it is commonly solved...
as a ranking problem where each database entry is assigned a continuous rank score representing its relevance to the query. This can be formulated as a regularization problem: Given a set of training labels relating a query to its matches, identify a ranking function \( f \) which trades supervised training error with a regularization energy functional measuring the (inverse) smoothness of \( f \) (a zero-th order tensor). Existing regularization approaches for ranking can be interpreted as saying “if two data points \( x \) and \( y \) are similar, then their rank scores \( f(x) \) and \( f(y) \) with respect to a query data point should be similar.”

In relative attributes problems, no query data point is ever presented as the goal is to learn a linear ordering—that we have learned the attribute ‘shoe pointedness’ through pairwise comparison says nothing about the kind of shoe we desire. Thus, all pairwise comparisons are important. Current relative attribute approaches use standard ranking algorithms from classical data retrieval to learn \( f \), e.g., RankSVM, a support vector machine with a rank loss \[ E \] with a regularization energy functional measuring the (inverse) smoothness of \( f \) (a zero-th order tensor). Their corresponding regularizers only enforce smoothness on the underlying ranking function \( f \), i.e., with respect to a specific query point, and so they do not directly capture/propagate the relative comparisons of all pairwise points. These pairwise relationships can be represented as a second-order tensor, but this requires the ability to regularize such a structure.

Our approach. We explicitly model the full pairwise relationships by learning a second-order anti-symmetric tensor (kernel) that directly expresses the rank relationships. Given the kernel structure, our new regularization energy can be interpreted as saying “if two data points \( x \) and \( y \) are similar, then their rank scores \( f(x) \) and \( f(y) \) with respect to all data points should be similar.” Due to the high time and memory complexities of modeling all pairwise relationships, our approach is not directly applicable to large-scale problems. Therefore, we also present an efficient low-rank approximation of the full kernel-based ranking algorithm. Further, we present a simple algorithm to convert the learned kernel into a linear ordering along with an intuitive explanation based on the ranking of graph-structured data.

While enforcing smoothness on the ranking function \( f \) implicitly enforces smoothness on all pairwise evaluations, our approach conjectures that explicitly enforcing smoothness on pairwise evaluations can help. This is motivated by the effectiveness of high-order derivative-based function regularization: Enforcing the smoothness of \( f \) by penalizing only its first-order derivative norm implicitly penalizes all high-order derivative norms, as the only null space of this norm is constant functions which have zero high-order derivatives. Nonetheless, the use of high-order regularizers is strongly supported by empirical performance (e.g., Thin-plate energy). Our main contribution is to demonstrate that adding this apparently-redundant explicit control over the regularization of all kernel evaluations can improve performance over existing regularizers.

2. Tensor regularization

To begin, we present a general framework for tensor regularization on manifolds, from which we derive our kernel-based ranking algorithm (KR) as its discretization. Our exposition will focus on symmetric and anti-symmetric second-order tensors, as used in metric learning and rank learning applications. Our supplemental material shows how this can be extended to higher-order tensors. We will use standard results from Riemannian geometry; we refer readers to our supplemental material for a brief introduction to vectors and tensors on Riemannian manifolds, and to more substantial texts [17, 22]. Readers interested only in the algorithmic aspects of our approach may jump to Sec. 3.

2.1. Tensor regularization (direct case)

The Harmonic energy of a smooth function \( f \in C^\infty(M) \) (a zeroth-order tensor) on a compact Riemannian manifold \((M, g)\) with metric \( g \) is obtained by integrating the squared norm of the gradient vector \( \nabla g f \) over \( M \):

\[
E_H(f) := \int_M \|\nabla g f(x)\|^2_g p(x) dV(x) \quad (1)
\]

where \( dV \) is the volume form of \( g \), \( p \) is a probability density of \( x \) on \( M \), \( \Delta_p \) is the density \( p \)-weighted Laplace-Beltrami operator \( \Delta_p := \frac{1}{2} (\nabla g)^* p \nabla g \), and Eq. 2 is obtained by applying Stokes’ theorem on \( M \). Here, \( E_H \) measures the first-order variation of \( f \) as weighted by \( p \). This energy is commonly used in regularizing functions, e.g., in semi-supervised learning and spectral clustering and embedding. Once the regularization energy on \( M \) is defined, learning a function \( f \) is facilitated by combining it with the training error functions (e.g. ranking losses \( l_p \) and \( l_{\Omega} \); Eq. 14).

In general, taking a derivative of a tensor increases its order by one: The derivative of function \( f \) is a vector, a first-order tensor. Similarly, the derivative of a second order tensor \( h \) is a third order tensor \( \nabla g h \). Generalizing the norm structure in Eq. 1 to third or higher-order tensors is straightforward given \( g \) (see supplemental Sec. 1). Based on these structures, we can extend the harmonic energy to tensors:

\[
E_H(h) := \int_M \|\nabla g h(x)\|^2_g p(x) dV(x). \quad (3)
\]

Then, we can learn a tensor \( h \) on manifolds by trading \( E_H \) with the training error defined on the tensor evaluations. For instance, for metric learning, the measured distances induced by the metric between a pair of sampled points should be large if the line or geodesic joining them is orthogonal to the class or cluster boundary directions, while the distance should be small when the line is parallel to the boundaries.

When the manifold \((M, g)\) is explicitly given, calculating the tensor harmonic energy \( E_H \) is straightforward (Eq. 3). Calculating \( \nabla g h \) of \( h \) requires estimating the Christoffel symbols, which
requires observing $g$ directly. However, in practical applications, we do not have access to the manifold or $g$. Instead, we obtain a sampled point cloud $X = \{x(1), \ldots, x(n)\}$ as a subset of the ambient space $\mathbb{R}^m$ (i.e., $i(M) \subset \mathbb{R}^m$ with $i$ being an embedding), which does not allow us to explicitly evaluate $\nabla^a h$.

For the special case of the zero-th order tensor $f$, Stokes’ theorem (Eq. 2) allows us to calculate the harmonic energy without having to explicitly evaluate $\nabla^a f$: Calculating the Laplacian $\Delta f$ is sufficient. This facilitates practical applications as the graph Laplacian is available as a consistent estimate of $\Delta$ [3, 13]: As $|X| \to \infty$, $f$ converges to a function $f$ on $M$ ($f := f|_X = [f(x(1)), \ldots, f(x(n))]^T$) and, in this case, the graph Laplacian regularizer corresponds to a sample-based approximation of $E_H(f)$ [3, 13]:

$$C_{(M,g)} F^\top L f \to E_H(f) \text{ as } n \to \infty,$$

where $C_{(M,g)}$ is a positive constant depending only on $(M,g)$. This result provides a theoretical justification of graph Laplacian-based regularization approaches.

For higher-order tensors, even after applying Stokes’ theorem, the resulting object involves tensor derivatives and so calculating the Christoffel symbols is unavoidable.

### 2.2. Tensor regularization (indirect case)

To regularize tensor $h$, we introduce an auxiliary function $H$ to encapsulate the behavior of $h$, then regularize $H$ instead. Roughly, we will construct the function $H(p, q)$ as an integral of $h$ along the arc-length-parameterized geodesic joining $p$ and $q$. Once $H$ is built, we can recover $h$ by taking the derivative of $H$ along the geodesic.

First, we use a local diffeomorphism structure between the tangent space $T_p M$ of $M$ at $p$ and $M$: The exponential map $\exp_p : U_p \subset T_p M \to M$ is defined as:

$$\exp_p(Y) = \gamma_Y(1),$$

where $\gamma_Y$ is a geodesic that agrees with $Y \in T_p M$ at $p$, i.e. $\gamma_Y(0) = p$ and $[\partial \gamma_y / \partial t](0) = Y$. The radius of the domain $U_p$ of $\exp_p$ (called normal neighborhood in which $\exp_p$ is a diffeomorphism) is always positive [22].

Using this diffeomorphism, one can define a distance function that corresponds to the metric $g$. We will develop surrogate functions for other tensors by extension, in particular anti-symmetric tensors for ranking applications.

The squared distance between $p$ and $q \in \exp(U_p)$ can be calculated as the squared length $\|Y\|^2_p$ of vector $Y = \exp^{-1}(q) \in U_p$, which defines our surrogate function $G_p(q) := \|Y\|^2_p$ at $p$. In general, $\|Y\|^2_p$ can be obtained as a second-order Taylor series approximation of $G_p(q)$.

With this identification, each function $G_p$ is defined only at a small neighborhood $\exp(U_p)$. Now, to apply this construction to learn a new tensor $\overline{g}$, we extend the domain of $G_p$ (and $\overline{G}_p$ for $\overline{g}$) to the entire manifold. Actually, the local characterization of $\{G_p\}$ is sufficient to define the corresponding regularizer of $G$ and equivalently $g$, as the regularizers themselves are defined only based on the local derivative evaluations (see Eq. 11). However, we wish to fully exploit the potential supervision information in learning a new tensor $\overline{g}$. For instance, for metric learning, a training label can relate a distinct pair of data points $p$ and $q$, i.e., $q \notin U_p$, e.g., “$p$ and $q$ belong to the same class and should be close with respect to $\overline{g}$”.

To extend the domains of $\{G_p\}$, we use the integration of the metric along the geodesics $\{\gamma\}$ joining $p$ and $q$:

$$G_p(q) = \inf_{\gamma(0)=p, \gamma(1)=q} \mathcal{L}(\gamma)$$

$$\mathcal{L}(\gamma) = \int_0^1 g \left( \frac{\partial \gamma(t)}{\partial t}, \frac{\partial \gamma(t)}{\partial t} \right) \, dt. \quad (7)$$

Now, a general distance function $G : M \times M \to \mathbb{R}^+$ is simply defined as $G(p, q) := G_p(q)$. This is precisely how the Riemannian manifold $M$ becomes a metric space [17, 22], as it can be shown that $G$ satisfies the conditions of non-negativity, symmetry, and triangle inequality. Further, given the distance function $G_p$ and a coordinate $(x^1, \ldots, x^d)$, restoring the metric tensor $g$ at $p$ is straightforward: By evaluating the distance between $p$ and each element in $\{q(1), \ldots, q(k)\} \subset \exp(U)$ (for 0 $\leq d+1/d$, one can calculate the corresponding lengths of vectors $Y(l) = \exp^{-1}(q(l)) = \sum_i y^i(l) \partial / \partial x_i$. This gives a system of equations for the coefficients of $g$ in local coordinates:

$$G_p(q) = \sum_{ij=1}^d g_{ij} y^i(l) y^j(l), \text{ for } l = 1, \ldots, k. \quad (8)$$

A coordinate-independent way of reconstructing $g$ from $G$ is to differentiate $\mathcal{L}(\gamma)$ with respect to $t$ at $a = 0$ for the space of geodesics $\{\gamma\}$. This provides a canonical way of reconstructing the Riemannian structure from the metric space structure [27]. It also demonstrates that there’s no need to explicitly calculate integrals over the geodesics as the metric $g$ is characterized entirely based on local behaviour of $G$. We adopt Eq. 8 to facilitate the $g$ reconstruction when the manifold is only indirectly observed based on a point cloud $X'$.

This system should be solved exactly when $G$ is calculated from $g$ as above. Let us suppose that we are estimating a new metric $\overline{g}$ (or any other symmetric positive definite tensor) on $(M,g)$, e.g., in metric learning applications. In this case, we could construct the corresponding auxiliary function $\overline{G}(p, q)$ and equivalently $\{\overline{G}_p(q) \in C^\infty(M)\}$ as an object to be regularized. As before, using Stokes’ theorem, we can calculate the Harmonic energy of $\overline{G}_p(q)$ by using the Laplacian $\Delta_{\gamma} \overline{G}_p$ instead of explicitly calculating $\nabla^\top \overline{G}_p \overline{G}$ or $\nabla \overline{g}$. Once $\overline{G}$ is estimated, we can restore $\overline{g}$ by constructing a least-square solution of Eq. 8.

Now, we apply this framework to an anti-symmetric tensor $\kappa$ for ranking applications (Fig. 1). First we note that the (squared) distance $G(p, q)$ between two data points $p$ and $q$ are defined as an infimum of the length of geodesics joining $p$ and $q$ (Eq. 6).
For a compact manifold, such a geodesic \( \gamma^* \) always exists. Based on this minimal geodesic \( \gamma^* = \arg\min_{\gamma} (L) \), the surrogate function \( K \) of an anti-symmetric tensor \( k \) is defined as:

\[
K(p, q) = K_p(q) = \int_0^a k\left( \frac{\partial \gamma^*(t)}{\partial t}, -\frac{\partial \gamma^*(t)}{\partial t} \right) dt.
\]  

(9)

Note the minus sign in the second argument of \( K \). By construction, \( K(p, q) \) is antisymmetric. Furthermore, given the antisymmetric function \( K_p \), the corresponding antisymmetric tensor \( k \) at \( p \) can be restored by simply taking the derivatives of \( K_p \) with respect to \( t \). Similarly to metric tensor case, \( k \) can be interpreted as an evaluation of \( K \) along an infinitesimally short path (a vector): By construction, \( K \) is consistent with the local \( k \) evaluation in \( U \):

\[
k(X, Y) = K_p(q),
\]  

(10)

with \( \exp(X) = p, \exp(Y) = q \) for infinitesimal \( X \) and \( Y \).

Applying this to a set of \( q(l) \) in a small neighborhood of \( p \), we obtain a system of equations similar to Eq. 8. As noted, we do not directly observe the manifold \( (M, g) \) but are provided with a sample \( \mathcal{X} \) from \( (M, g) \). These linear equations can be constructed based on Riemannian normal coordinates, as estimated by applying the principal component analysis to a local neighborhood of each data point \( x \in \mathcal{X} \) [9].

Given the construction of a surrogate function \( K_p \) and \( K \) accordingly, we now introduce our (surrogate function-based) tensor harmonic energy:

\[
\mathcal{E}_H(K) = \int_M \int_M ||\nabla^g K_p(q)\|_g^2 dV(p)dV(q).
\]  

(11)

The interpretation of this energy is straightforward. At each point \( q \), the function \( K_p(q) = K(p, q) \) represents the relationship between \( q \) and \( p \). We enforce that, as \( q \) varies, the function \( K_p(q) \) varies smoothly (the outer integral) and this has to be the case for all points \( p \in M \) (the inner integral).

Our tensor harmonic energy is constructed entirely based on tensor evaluations and so it respects the intrinsic geometry of \( M \) (equivalently, it is coordinate independent). An alternative to this construction is to explicitly learn the latent Riemannian structure of the data [34, 2]. These latent variable models enjoy simple Riemannian structure once identified. However, they limit generality as they assume that the data manifold admits a global coordinate representation. In contrast, our discrete approximation has complementary strength of better generality as we assume no global coordinate chart.

3. Kernel-based ranking algorithm

We present a practical algorithm that uses a consistent approximation of this energy. Suppose that we are given a set of data points \( \mathcal{X} = \{x(1), \ldots, x(n)\} \subseteq \mathbb{R}^m \), along with pairwise inequality and equality relationships, respectively, \( \mathcal{P} = \{(i, j)\} \) and \( \mathcal{O} = \{(i, j)\} \), where \((i, j) \in \mathcal{P} \) implies that the rank of \( i \)-th data point is higher than \( j \)-th data point (as denoted as \( \text{Rank}(x(i)) > \text{Rank}(x(j)) \)). Similarity, \((i, j) \in \mathcal{O} \) means \( \text{Rank}(x(i)) = \text{Rank}(x(j)) \).

**RankSVM (RS).** In the original Relative Attributes work [28], the desired ordering is obtained by applying an ordering function \( f : \mathbb{R}^m \rightarrow \mathbb{R} \) to \( \mathcal{X} \):

\[
f(x) = w^\top x,
\]  

(12)

where the parameter vector \( w \) is estimated as the minimizer of the RankSVM (RS) energy functional [5, 28]:

\[
\mathcal{E}_{RS}(w) = \sum_{(i, j) \in \mathcal{P}} l_P(f_i - f_j) + \sum_{(i, j) \in \mathcal{O}} l_O(f_i - f_j) + \lambda ||w||^2;
\]  

(13)

where \( \lambda > 0 \) is a hyper-parameter and \( f_i = f(x(i)) \). The inequality loss \( l_P \) and equality loss \( l_O \) are given respectively as

\[
l_P(a) = \max(0, 1 - a)^2 \quad \text{and} \quad l_O(a) = a^2,
\]  

(14)

while other loss (or inverse likelihood) functions are also possible. For all algorithms discussed in this paper, we use these two loss functions. Therefore, the differences between these algorithms lie only in the respective regularizers. This enables to compare the performance of supervised and semi-supervised learning approaches, and our new regularizer in the kernel-based learning setting. In general, other losses can be adopted in all algorithms compared in this paper (e.g., logistic loss [1]).

**Semi-supervised RankSVM (SSR).** This extension of RankSVM can be obtained by replacing the ambient regularizer \( ||w||^2 \) in Eq. 13 with a manifold regularizer:

\[
\mathcal{E}_{SSR}(w) = \sum_{(i, j) \in \mathcal{P}} l_P(f_i - f_j) + \lambda \sum_{(i, j) \in \mathcal{O}} l_O(f_i - f_j) + \lambda f^\top Lf,
\]  

(15)
assuming that an underlying linear ordering function $f$ exists. Given the kernel function $K$, our new kernel-based ranking energy functional (KR) is obtained as:

$$
E_{KR}(K) = \sum_{(i,j) \in \mathcal{P}} l_P(K_{ij}) + \sum_{(i,j) \in \mathcal{O}} l_O(K_{ij}) + \lambda tr[K^T L K],
$$

where $tr[A]$ is the trace of $A$, and $K_{ij} := K(x(i), x(j))$. We abuse notation and use $K$ to denote a function and a matrix as its sample evaluation. A similar regularizer was used to build a match graph for 3D scene reconstruction [19].

If we adopt the kernel example of Eq. 17, the loss functions (the first two terms in Eq. 18) are the same as in RS, SSR, and TR. Therefore, the main difference of KR from the other algorithms is that we use $K$ instead of $f$ as an object to be learned. This perspective enables us to introduce a new sample-based regularizer $E_H(K) := tr[K^T L K]$. The $i$-th row $K_{[i,:]}$ of the kernel matrix $K$ stores the results of relative comparisons between $x(i)$ and all the other data points in $\mathcal{X}$. Therefore, minimizing this regularization energy enforces the smoothness of all pairwise relationships as weighted by $W$ (Eq. 16): “If $x(i)$ and $x(j)$ are similar, their relative rank comparisons with respect to all other data points should be similar as well”. Furthermore, if required, converting the estimated kernel evaluations $K$ to linear ordering $f$ based on Eq. 17 is straightforward (shown at the end of this section).

Now we provide an interpretation of this energy from the tensor regularization perspective of Sec. 2: Our kernel-based approximate Harmonic energy $E_H(K)$ is a consistent discretization of the tensor Harmonic energy $E_H$ (Eq. 11):

**Proposition 1.** If $M$ is a compact submanifold of $\mathbb{R}^m$ and $\mathcal{X} = \{x(1), \ldots, x(n)\}$ be a sample from a uniform distribution on $M$, then there is a constant $C_M > 0$ such that for $K \in C^\infty(M \times M)$ and $\sigma_2^2(n) = n^{-1/(d+2+\alpha)}$ with $\alpha > 0$ as $n \to \infty$:

$$
\frac{1}{n^3(\sigma^2(u))^{d/2+1}} E_H(K) \xrightarrow{p} C_M E_K(K).
$$

This result combines the convergence properties of two objects: The convergence of graph Laplacian $L$ to Laplacian $\Delta$ and the convergence of kernel evaluation matrix $K$ to the corresponding kernel function $K \in C^\infty(M \times M)$.

The proof is a straightforward application of Theorem 4 by Zhou and Belkin [43]: Since $K \in C^\infty(M \times M)$, $K_p(\cdot) \in C^\infty(M)$ for each $p \in M$. Applying the convergence result of graph Laplacian to $K_{p(i)}(\cdot)$ for a fixed $p(i)$ ($p(i) \sim x(i)$) [3], we have for each $q(j) \in \mathcal{X}$:

$$
\frac{[L_k]_{ij}}{n^2(\sigma^2(n))^{d/2+1}} \xrightarrow{p} \Delta K_{p(i)}(p(j)).
$$

Then Eq. 19 is obtained by applying Eq. 20 to each point $p(j)$ and the Stokes’ identity (Eq. 1).
Low-rank kernel-based ranking. Our preliminary experiments have indicated that the kernel-based ranking (KR) approach (Eq. 18) significantly improves ordering performance over RS, SSR, TR. However, a major drawback of this approach is its high computational and memory complexities: It requires explicitly optimizing an $n \times n$-sized kernel matrix $K$. Therefore, directly applying KR to large-scale problems is infeasible. We overcome this limitation by adopting a low-rank factorized approximation of $K$: Given a factor matrix $B \in \mathbb{R}^{n \times p}$ ($p \ll n$), an antisymmetric kernel matrix $K \in \mathbb{R}^{n \times n}$ of rank $p$ is constructed as:

$$\bar{K} = BQB^\top,$$  \hspace{1cm} (21)

where $Q = R^\top - R$ with $R$ being the lower triangular matrix of ones. By regarding $\bar{K}$ as an approximation of $K$, we take the low-rank matrix $B$ as a new variable to optimize. Unfortunately, reformulating the KR optimization problem (Eq. 18) based on this factorization:

$$\mathcal{E}_{KR}(B) = \mathcal{L}_P(B) + \mathcal{L}_O(B) + \lambda R(B)$$

$$= \sum_{(i,j) \in \mathcal{P}} l_P([BQB^\top]_{ij}) + \sum_{(i,j) \in \mathcal{O}} l_O([BQB^\top]_{ij})$$

$$+ \lambda \|BQB\|^2$$  \hspace{1cm} (22)

renders the energy functional $\mathcal{E}_{KR}$ non-convex with respect to the parameter matrix $B$. However, we empirically observed that when $B$ is initialized with all ones (i.e. $B = [1]_n [1]^\top_p$ with $1 = [1, \ldots, 1]^\top$), the resulting optimized solutions lead to competitive ranking results. In our supplemental material, we further support this factorization and the optimization initialization approach by by evaluating their pure reconstruction capability in image reconstruction as an example.

We minimize $\mathcal{E}_{KR}(B)$ using gradient descent. The derivatives of the regularization energy and the two loss terms are:

$$\frac{\partial \mathcal{R}(B)}{\partial B_{[i,:]}^\top} = -2BQB^\top LBQ - 2LBQ B^\top BQ,$$  \hspace{1cm} (23)

$$\frac{\partial \mathcal{L}_P(B)}{\partial B_{[i,:]}^\top} = \sum_{(i,j) \in \mathcal{P}} \max \left[ 0, 2(T_{ij} - B_{[i,:]} Q B_{[j,:]}^\top) \right] Q B_{[i,:]}^\top$$

$$- \sum_{(i,j) \in \mathcal{P}} \max \left[ 0, 2(T_{ij} - B_{[i,:]} Q B_{[j,:]}^\top) \right] Q B_{[j,:]}^\top$$

$$\frac{\partial \mathcal{L}_O(B)}{\partial B_{[i,:]}^\top} = \sum_{(i,j) \in \mathcal{O}} 2(T_{ij} - B_{[i,:]} Q B_{[j,:]}^\top) Q B_{[j,:]}^\top$$

$$- \sum_{(i,j) \in \mathcal{O}} 2(T_{ij} - B_{[i,:]} Q B_{[j,:]}^\top) Q B_{[i,:]}^\top$$  \hspace{1cm} (24)

where $B_{[i,:]}$ is the $i$-th row of the matrix $B$, and $Q = -Q^\top$.

Reconstruction of $f$ given $K$. While the estimated kernel matrix $K$ may not satisfy the reconstruction constraint of $f$ (Eq. 17)
Semi-supervised RankSVM (SSR)

ETH-80

Kernel-based Ranking (KR)

PubFig

MNIST

25

SVHN

DTD

Transductive Ranking (TR)

CIFAR-10

We applied principal component analysis to reduce attributes to learn. The absolute rank correlation coefficients including the RS results can be found in the supplemental.

Figure 2. Improvement from RS in mean rank correlation coefficients (y-axis) of different ranking algorithms for eight datasets. In all but PubFig, our KR is comparable or better. First row: x-axis shows the number of labels per class. Second row (2–4): x-axis corresponds to the indices of attributes to learn. The absolute rank correlation coefficients including the RS results can be found in the supplemental.

and Grauman and Kovashka et al. [20]. In general, relative attributes can be used when the labels are provided per image or object pairs. However, following Parikh and Grauman, we use the category-level labels as they facilitate objective, quantitative evaluation. For evaluation, we measure Kendall’s rank correlation coefficient on all inequality pairs: It counts the difference between the number of correctly ordered pairs and the number of incorrectly ordered pairs normalized by the number of total pairs.

The ETH-80 dataset contains 3,280 object images from 8 different categories [23]. Each data point is represented based on the HOG (histogram of oriented gradients) descriptors as provided by Ebert et al. [10]. The MNIST training dataset consists of 60,000 isolated digit images of size 28×28. We use the gray-level values as features. The cropped Street View House Numbers (SVHN) dataset contains 26,032 cropped digit images. This dataset has a similar format as MNIST; however, it exhibits large intra-class variations and includes complex photometric distortions that make the learning problem challenging [26]. Each data point is originally presented as a 32×32-sized color image. We reduced the dimensionality of the dataset to 100 using principal component analysis. The CIFAR-10 dataset is a labeled subset of 80-million tiny images datasets [33]. It consists of 60,000 32×32-sized color images in 10 classes [21]. Each image in this dataset is represented based on RGB color values leading to a 3,072-dimensional vector. We applied principal component analysis to reduce the dimensionality to 700 which contains around 99% of total variance. The Describable Textures Dataset (DTD) [7] contains 5,640 texture images arranged based on 47 manually-assigned semantic attributes (e.g., chequered and bumpy; 120 images per attribute). We use the semantic attribute indices as class labels.

Method. To evaluate these eight datasets, we assigned linear ground-truth ranking based on their class labels, e.g., for MNIST, digit 1 has a higher rank than 2. The evaluation criteria is the same as the first three datasets (Kendall’s correlation coefficients). However, the training labels are collected in a different way: Instead of pre-selecting a set of training data points and extracting all possible pairwise labels therein, we randomly selected a prescribed number of pairwise labels from the entire database. For a dataset consisting of c categories, l different labels are selected per class leading to \(c(c-1)/2 \times l\) inequality labels. We report the performances of ranking algorithms with respect to varying number \(l\) of labels per class.
Results. Figure 2 shows the improvement of mean rank coefficients from RankSVM (RS) with corresponding error bars (with length twice the standard deviation). Deep learning algorithm (DR) outperformed RS for all datasets demonstrating the effectiveness of deep learning for ranking problems. Also, except for PubFig, the two transductive learning algorithms TR and KR constantly outperformed RankSVM (RS). This demonstrates the effectiveness of exploiting unlabeled data in relative attribute applications. However, unlike TR and KR, performance of the model-based semi-supervised extension (SSR) is roughly on par with RS (it is better than RS on ETH-80 and DTD, and worse on OSR and PubFig). Our kernel-based ranking algorithm (KR) significantly improves upon the other algorithms including the baseline transductive ranking (TR).

In particular, for MNIST, KR resulted in $\approx 40\%$ higher rank coefficients than other algorithms when the number of labels per class were less than 10. On OSR, DR and KR perform best. The improvement of KR over TR is especially significant when the number of labels $l$ is limited. As $l$ increases, the performance gap between these two algorithms narrows and eventually, they become almost identical as shown in the corresponding results of DTD.

Although the performances of KR and TR on this dataset are roughly equal, their performance variations across different attributes are significantly large. This suggests that, from the performance perspective, DR and KR are complementary. In our supplemental material, we demonstrate that by combining DR and KR we can construct a ranker that frequently outperforms other algorithms.

A notable exception to this tendency is PubFig, where DR is clear winner. This indicates that semi-supervised learning might not be always useful. One possible explanation is that PubFig has insufficient data points to reveal the underlying manifold structure upon which the semi-supervised algorithms build (only 772 data points, while other datasets are of order thousand or ten thousand). Another explanation is simply that the data do not lie on a low-dimensional manifold. Unfortunately, verifying these possibilities is a challenging problem. Furthermore, it is not straightforward to predict which (class of) algorithms would lead to better performances on specific datasets or problems. In practice, users would interact (provide labels) with data and be able to provide feedback on the utility of different algorithms. In this respect, the experiments demonstrate that our kernel-based ranking algorithm provides a good alternative to RankSVM and deep learning.

Hyper-parameters and time complexity. The RS, SSR, TR, and KR compared in the experiments have a regularization hyper-parameter $\lambda$. In addition, all semi-supervised learning algorithms (SSR, TR, KR) require determining the $k$ number of nearest neighbors and the scaling parameter $\sigma^2$ to build the graph Laplacian (Eq. 16). We determined $\sigma^2$ adaptively for each data point $x(i)$ such that $\sigma^2$ becomes half of the mean distance from $x(i)$ to its $k$-NNs [4]. The remaining hyper-parameters $\lambda$ and $k$ were optimized based on a separate validation label sets which have the same size as the corresponding training set for each experiment. For DN, the number of hidden layers was fixed at 6, while the size of each layer (number of units) and the number of epochs were automatically tuned as hyper-parameters. For each run of DN, the network was trained with 5 random initializations, and the one with the smallest validation error was chosen.

The time complexity of our low-rank kernel-based ranking algorithm depends on the number $n$ of data points, the rank $p$ of $K$ factorization (Eq. 21), and the $k$ nearest neighbors used to build the graph Laplacian (Eq. 16). In each gradient calculation step (Eqs. 23-24), $BQ$ can be pre-calculated and $Q^\top B = Q^\top B^\top$. Therefore, the most demanding computation is to calculate $LB$ and $B^\top [BQ]$. The time complexity is to calculate $LB$ and $B^\top [BQ]$. The time complexity of $B^\top [BQ]$ is $O(np^2)$ while $LB$ takes $O(knp)$. In our Matlab implementation, evaluating the gradient of $B$ on 60,000 MNIST points with $p = 50$, $k = 8$, took $\approx 0.004$ seconds on a 3.6GHz machine. The time complexities of RS and TR are $O(mn)$ and $O(n)$, respectively with $m$ being the input space dimensionality. While lower performing, RS is faster when $m$ is smaller than the number of data points $n$. Further, KR is less suitable for interactive applications when $n$ is very large.

5. Discussion and conclusion

We have empirically verified our conjecture on the effectiveness of modeling and regularizing full pairwise rank relationships with second-order tensors (kernels). Our algorithm was obtained as a discrete approximation of tensor regularization framework on manifolds. To cope with large-scale problems, we have proposed sparse factorizations. In supplemental material, we describe how our framework can be applied to learning other tensors, e.g., future work could address how to apply it to learning metric tensors.

Our low-rank factorization approach (Eq. 21) was inspired by approximating the dense kernel matrix from the computational complexity perspective. Therefore, the rank $p$ of the matrix $B$ was prescribed by the expected computational and memory complexities (fixed at 50 throughout the entire experiments) and therefore, we haven’t actively explored the performance effect of varying rank. However, it is well known that low-rank approximation by itself has a regularization effect and it has been actively exploited in estimating matrices, e.g., in compressed sensing [8], photometric stereo and structure from motion [42], and metric learning [39]. Accordingly, future work should analyze the low-rank approximation from the regularization perspective.

References


