Partial Order Embedding with Multiple Kernels

Brian McFee BMCFEE@CS.UCSD.EDU
Department of Computer Science and Engineering, University of California, San Diego, CA 92093 USA

Gert Lanckriet GERT@ECE.UCSD.EDU
Department of Electrical and Computer Engineering, University of California, San Diego, CA 92093 USA

Abstract

We consider the problem of embedding arbitrary objects (e.g., images, audio, documents) into Euclidean space subject to a partial order over pairwise distances. Partial order constraints arise naturally when modeling human perception of similarity. Our partial order framework enables the use of graph-theoretic tools to more efficiently produce the embedding, and exploit global structure within the constraint set.

We present an embedding algorithm based on semidefinite programming, which can be parameterized by multiple kernels to yield a unified space from heterogeneous features.

1. Introduction

A notion of distance between objects can be a powerful tool for predicting labels, retrieving similar objects, or visualizing high-dimensional data. Due to its simplicity and mathematical properties, the Euclidean distance metric is often applied directly to data, even when there is little evidence that said data lies in a Euclidean space. It has become the focus of much research to design algorithms which adapt the space so that Euclidean distance between objects conforms to some other presupposed notion of similarity, e.g., class labels or human perception measurements.

When dealing with multi-modal data, the simplest first step is to concatenate all of the features together, resulting in a single vector space on which metric learning algorithms can be applied. This approach suffers in situations where features cannot be directly concatenated, such as in kernel methods, where the features are represented (implicitly) by infinite-dimensional vectors. For example, if each object consists of mixed audio, video, and text content, entirely different types of transformations may be natural for each modality. It may therefore be better to learn a separate transformation for each type of feature. When designing metric learning algorithms for heterogeneous data, care must be taken to ensure that the transformation of features is carried out in a principled manner.

Moreover, in such feature-rich data sets, a notion of *similarity* may itself be subjective, varying from person to person. This is particularly true of multimedia data, where a person may not be able to consistently decide if two objects (e.g., songs or movies) are similar or not, but can more reliably produce an ordering of similarity over pairs. Algorithms in this regime must use a sufficiently expressive language to describe similarity constraints.

Our goal is to construct an algorithm which integrates subjective similarity measurements and heterogeneous data to produce a low-dimensional embedding. The main, novel contributions of this paper are two-fold. First, we develop the *partial order* embedding framework, which allows us to apply graph-theoretic tools to solve relative comparison embedding problems more efficiently. Our second contribution is a novel kernel combination technique to produce a unified Euclidean space from multi-modal data.

The remainder of this paper is structured as follows. Section 2 formalizes the embedding problem and develops some mathematical tools to guide algorithm design. Section 3 provides algorithms for non-parametric and multi-kernel embedding. Section 4 describes two experiments: one on synthetic and one on human-derived constraint sets. Section 5 discusses the complexity of exact dimensionality minimization in the partial order setting.

1.1. Related work

Metric learning algorithms adapt the space to fit some provided similarity constraints, typically consisting of pairs which are known to be neighbors or belong to the same class (Wagstaff et al., 2001; Xing et al., 2003; Tsang &

Appearing in *Proceedings of the 26th International Conference on Machine Learning*, Montreal, Canada, 2009. Copyright 2009 by the author(s)/owner(s).

Kwok, 2003; Bilenko et al., 2004; Hadsell et al., 2006; Weinberger et al., 2006; Globerson & Roweis, 2007).

Schultz and Joachims (2004) present an SVM-type algorithm to learn a metric from relative comparisons, with applications to text classification. Their formulation considered metrics derived from axis-aligned scaling, which in the kernelized version, translates to a weighting over the training set. Agarwal et al. (2007), motivated by modeling human perception data, present a semidefinite programming (SDP) algorithm to construct an embedding of general objects from paired comparisons. Both of these algorithms treat constraints individually, and do not exploit global structure (e.g., transitivity).

Song et al. (2008) describe a metric learning algorithm that seeks a locally isometric embedding which is maximally aligned to a PSD matrix derived from side information. Although our use of (multiple) side-information sources differs from that of (Song et al., 2008), and we do not attempt to preserve local isometry, the techniques are not mutually exclusive.

Lanckriet et al. (2004) present a method to combine multiple kernels into a single kernel, outperforming the original kernels in a classification task. To our knowledge, similar results have not yet been demonstrated for the present metric learning problem.

1.2. Preliminaries

Let $\mathcal{X}$ denote a set of n objects. Let $\mathcal{C}$ denote a partial order over pairs drawn from $\mathcal{X}$:

$$\mathcal{C} = \{(i,j,k,\ell) : \ i,j,k,\ell \in \mathcal{X}, \ (i,j) < (k,\ell)\},$$

where the *less than* relation is interpreted over dissimilarity between objects. Because $\mathcal{C}$ is a partial order, it can be represented by a DAG with vertices in $\mathcal{X}^2$ (see Figure 1). For any pair (i,j), let $\mathrm{depth}(i,j)$ denote the length of the longest path from a source to (i,j) in the DAG. Let $\mathrm{diam}(\mathcal{C})$ denote the length of the longest (possibly weighted) source-to-sink path in $\mathcal{C}$, and let $\mathrm{length}(\mathcal{C})$ denote the number of edges in the path.

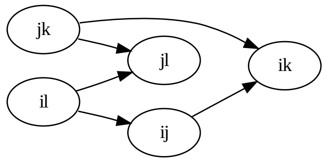


Figure 1. A partial order over similarity in DAG form: vertices represent pairs, and a directed edge from (i,j) to (i,k) indicates that (i,j) are more similar than (i,k).

A *Euclidean embedding* is a function $g : \mathcal{X} \to \mathbb{R}^n$ which maps $\mathcal{X}$ into $n-$dimensional space equipped with the Euclidean (ℓ_2) metric: $\|x-y\| = \sqrt{(x-y)^\mathsf{T}(x-y)}$.

A symmetric matrix $A \in \mathbb{R}^{n\times n}$ has eigen-decomposition $A = V\Lambda V^\mathsf{T}$, where Λ is a diagonal matrix containing the eigenvalues of A in descending order: $\lambda_1 \geq \lambda_2 \geq \cdots \geq \lambda_n$. A is *positive semidefinite* (PSD), denoted $A \succeq 0$, if each of its eigen-values is non-negative. For $A \succeq 0$, let $A^{1/2}$ denote the matrix $\Lambda^{1/2}V^\mathsf{T}$. Finally, for any matrix B, let B_i denote its i^{th} column vector.

2. Partial order embedding

Previous work has considered the setting where similarity information is coded as tuples (i,j,k,ℓ) where objects (i,j) are more similar than (k,ℓ), but treat each tuple individually and without directly taking into consideration larger-scale structure within the constraints. Such global structure can be revealed by the graph representation of the constraints.

If the constraints do not form a partial order, then the corresponding graph must contain a cycle, and therefore cannot be satisfied by any embedding. Moreover, it is easy to localize subsets of constraints which cannot all be satisfied by looking at the strongly-connected components of the graph.

We therefore restrict attention to constraint sets which satisfy the properties of a partial order: transitivity and antisymmetry. Exploiting transitivity allows us to more compactly represent otherwise large sets of independently considered local constraints, which can improve the efficiency of the algorithm.

2.1. Problem statement

Formally, the partial order embedding problem is defined as follows: given a set $\mathcal{X}$ of n objects, and a partial order $\mathcal{C}$ over $\mathcal{X}^2$, produce a map $g : \mathcal{X} \to \mathbb{R}^n$ such that

$$\forall\,(i,j,k,\ell) \in \mathcal{C} : \ \|g(i)-g(j)\|^2 \ < \ \|g(k)-g(\ell)\|^2.$$

For numerical stability, g is restricted to force margins between constrained distances:

$$\forall\,(i,j,k,\ell) \in \mathcal{C} : \ \|g(i)-g(j)\|^2 + e_{ijk\ell} \ < \ \|g(k)-g(\ell)\|^2.$$

Many algorithms take $e_{ijk\ell} = 1$ out of convenience, but uniform margins are not strictly necessary. We augment the DAG representation of $\mathcal{C}$ with positive edge weights corresponding to the desired margins. We will refer to this representation as a *margin-weighted constraint graph*, and $(\mathcal{X}, \mathcal{C})$ as a *margin-weighted instance*.

In the special case where $\mathcal{C}$ is a total ordering over all pairs (i.e., a chain graph), the problem reduces to non-metric multidimensional scaling (Kruskal, 1964), and a

Algorithm 1 Naïve total order construction

```
Input: objects $\mathcal{X}$, margin-weighted partial order $\mathcal{C}$
Output: symmetric dissimilarity matrix $\Delta \in \mathbb{R}^{n\times n}$

for each $i$ in $1 \ldots n$ do
    $\Delta_{ii} \leftarrow 0$
end for
for each $(k, \ell)$ in topological order do
    if in-degree$(k, \ell) = 0$ then
        $\Delta_{k\ell}, \Delta_{\ell k} \leftarrow 1$
    else
        $\Delta_{k\ell}, \Delta_{\ell k} \leftarrow \max_{(i,j,k,\ell)\in\mathcal{C}} \Delta_{ij} + e_{ijk\ell}$
    end if
end for
```

constraint-satisfying embedding can always be found by constant-shift embedding (Roth et al., 2003). In general, $\mathcal{C}$ is not a total order, but a $\mathcal{C}$-respecting embedding can always be produced by reducing the partial order to a total order (see Algorithm 1).

Algorithm 1 produces arbitrary distances for unconstrained pairs, and is therefore unsuitable for applications where the solution must be constrained to somehow respect features of the objects. However, it can be used to construct an upper bound on the diameter of a satisfying solution, which will be useful when designing more sophisticated algorithms in Section 3.

2.2. A diameter bound

Let Δ be the output of Algorithm 1 on an instance $(\mathcal{X}, \mathcal{C})$. An embedding can be found by first applying classical multidimensional scaling (MDS) (Cox & Cox, 1994) to Δ:

$$A = -\frac{1}{2} H \Delta H, \tag{1}$$

where $H = I - \frac{1}{n}\mathbf{1}\mathbf{1}^\mathsf{T}$ is the $n \times n$ centering matrix. Shifting the spectrum of A yields

$$A - \lambda_n(A) I = A^* \succeq 0, \tag{2}$$

where $\lambda_n(A)$ is the minimum eigenvalue of A. The embedding g can be found by decomposing $A^* = V\Lambda^* V^\mathsf{T}$, so that $g(i)$ is the i^{th} column of $(A^*)^{1/2}$; this is the solution constructed by the constant-shift embedding nonmetric MDS algorithm (Roth et al., 2003).

The construction of A^* leads to a bound on the embedding's diameter which depends only on the size of $\mathcal{X}$ and the diameter of the margin-weighted constraint graph.

Lemma 2.1. *For every margin-weighted instance $(\mathcal{X}, \mathcal{C})$, there exists a margin-preserving map $g : \mathcal{X} \to \mathbb{R}^{n-1}$ such that for all $i \neq j$,*

$$1 \leq \|g(i) - g(j)\| \leq \sqrt{(4n+1)(\operatorname{diam}(\mathcal{C}) + 1)}.$$

Proof. Let Δ be the output of Algorithm 1 on $(\mathcal{X}, \mathcal{C})$, and A, A^* as defined in (1) and (2). From (2), the squared-distance matrix derived from A^* can be expressed as

$$\Delta^*_{ij} = \Delta_{ij} - 2\lambda_n(A)\mathbb{1}_{i\neq j}. \tag{3}$$

Expanding (1) yields

$$A_{ij} = \Delta_{ij} - \frac{1}{n}\Delta_i^\mathsf{T}\mathbf{1} - \frac{1}{n}\mathbf{1}^\mathsf{T}\Delta_j + \frac{1}{n^2}\mathbf{1}^\mathsf{T}\Delta\mathbf{1},$$

and since $\Delta_{ij} \geq 0$, it is straightforward to verify the following inequalities:

$$\forall i, j : \ -2\max_{x,y}\Delta_{xy} \leq A_{ij} \leq 2\max_{x,y}\Delta_{xy}.$$

It follows from the Geršgorin circle theorem (Varga, 2004) that

$$\lambda_n(A) \geq -2n\max_{x,y}\Delta_{xy}. \tag{4}$$

Since $\Delta_{ij} \leq 1 + \operatorname{diam}(\mathcal{C})$, substituting (4) into (3) yields the desired result. □

In general, unconstrained distances can lie anywhere in the range $[1, \operatorname{diam}(\mathcal{C}) + 1]$, and Lemma 2.1 still applies. Even within a bounded diameter, there are infinitely many possible solutions, so we must further specify optimality criteria to define a unique solution.

2.3. Constraint graphs

The partial order framework enables pre-processing of the constraint set by operations performed directly on the constraint graph. This has the advantage that redundancies and inconsistencies (i.e., cycles) can be identified and resolved ahead of time, resulting in more robust solutions.

In particular, the transitivity property can be exploited to dramatically simplify a constraint set: any redundant edges may be safely removed from the graph without altering its semantics. We can therefore operate equivalently on the transitive reduction of the constraint graph (Aho et al., 1972).

Additionally, real constraint data can exhibit inconsistencies, in which case we would like to produce a maximal, consistent subset of the constraints. If the data contains sufficient variability, it may be reasonable to assume that inconsistencies are confined to small (local) regions of the graph. This corresponds to subsets of pairs which are too close in perceived similarity to reliably distinguish. Pruning edges within strongly connected components (SCC) can resolve local inconsistencies while retaining the unambiguous constraints. By construction, this always produces a valid partial order.

Some data sets, such as in Section 4.1, exhibit insufficient variability for the SCC method. If it is possible to obtain multiple observations for candidate constraints, the results may be filtered by majority vote or hypothesis testing to generate a robust subset of the constraints.

Although real data typically contains inconsistencies, denoising the constraint set is not the focus of this paper. For the remainder of this paper, we assume that the constraints have already been pre-processed to yield a valid partial ordering.

3. Algorithms

The naïve embedding algorithm in Section 2 gives no principled way to determine unconstrained distances, and may artificially inflate the dimensionality of the data to satisfy the ordering. Previous algorithms address these problems by filling in unconstrained distances according to the optimum of an objective function (Agarwal et al., 2007; Schultz & Joachims, 2004).

Often, the objective function used is a convex approximation to the rank of the embedding. We take a different approach here, and use a *maximum-variance unfolding* (MVU) objective to stretch all distances while obeying the constraints (Weinberger et al., 2004). This choice of objective relates more closely to the form of the constraints (distances), and in practice yields low-dimensional solutions.

3.1. Non-parametric embedding

In the non-parametric embedding problem, we seek a map $g : \mathcal{X} \to \mathbb{R}^n$ which respects $\mathcal{C}$. We do not assume any explicit representation of the elements of $\mathcal{X}$. Our algorithm optimizes the MVU objective: $\max \sum_{i,j} \|g(i) - g(j)\|^2$, subject to $\mathcal{C}$.

To ensure that the objective is bounded, it suffices to bound the diameter of the embedding. Unlike the original MVU algorithm for manifold learning, we do not assume the existence of a neighborhood graph, so we cannot deduce an implicit bound. Instead, we use Lemma 2.1 to achieve an explicit bound on the objective, thereby resolving scale invariance of the solution.

Similarly, without a neighborhood graph, there is no direct way to choose which distances to preserve (or minimize) and which to maximize. However, we can exploit the acyclic structure of $\mathcal{C}$ by constructing margins $e_{ijk\ell}$ which grow with depth in $\mathcal{C}$. This creates nested families of neighborhoods around each constrained point. For certain problems, it is possible to automatically construct margin values to ensure rapid neighborhood growth. Section 4.2 provides one such example.

Algorithm 2 Partial order embedding

```
Input: objects $\mathcal{X}$, margin-weighted partial order $\mathcal{C}$
Output: inner product matrix $A$ of embedded points

Let $d(i,j) = A_{ii} + A_{jj} - 2A_{ij}$.

$\max_A \sum_{i,j} d(i,j)$
$\forall i,j \in \mathcal{X}$        $(4n+1)(\mathrm{diam}(\mathcal{C})+1) \geq d(i,j)$
$\forall (i,j,k,\ell) \in \mathcal{C}$        $d(i,j) + e_{ijk\ell} \leq d(k,\ell)$
        $\sum_{i,j} A_{ij} = 0, \quad A \succeq 0$
```

By combining the MVU objective with the diameter bound, margin constraints, and a centering constraint to resolve translation invariance, we arrive at Algorithm 2.

3.2. Parametric embedding

In the parametric embedding problem, we are provided side information about $\mathcal{X}$ (e.g., vector representations), which parameterizes the map g. The simplest such parameterization is a linear projection M, yielding $g(x) = Mx$. As in (Globerson & Roweis, 2007), this generalizes to nonlinear transformations by using kernels.

In this setting, including a regularization term $\mathrm{Tr}(M^\mathsf{T}M)$ (scaled by a factor $\gamma > 0$) in the objective function allows us to invoke the generalized representer theorem (Schölkopf et al., 2001). It follows that an optimal solution must lie in the span of the training data: if X contains the training set features as column vectors, then $M = NX^\mathsf{T}$ for some matrix N. Therefore, the regularization term can be expressed as

$$\mathrm{Tr}(M^\mathsf{T}M) = \mathrm{Tr}(XN^\mathsf{T}NX^\mathsf{T}) = \mathrm{Tr}(N^\mathsf{T}NX^\mathsf{T}X).$$

Because the embedding function depends only upon inner products between feature vectors, the procedure can be generalized to non-linear transformations by the kernel trick. Let K be a kernel matrix over $\mathcal{X}$, i.e. $K_{ij} = \langle \phi(i), \phi(j) \rangle$ for some feature map ϕ. Then we can replace the inner products in $g(\cdot)$ with the kernel function, yielding an embedding of the form $g(x) = NK_x$, where K_x is the column vector formed by evaluating the kernel function at x against the training set.

Within the optimization, all distance calculations between embedded points can be expressed in terms of inner products:

$$\|g(i) - g(j)\|^2 = (K_i - K_j)^\mathsf{T} N^\mathsf{T} N (K_i - K_j).$$

This allows us to formulate the optimization over a PSD matrix $W = N^{\mathsf{T}}N$. The regularization term can be expressed equivalently as $\mathrm{Tr}(WK)$, and the inner product matrix for the embedded points is $A = KWK$. We can then factor W to obtain an embedding function $g(x) = W^{1/2}K_x$.

As in (Schultz & Joachims, 2004), this formulation can be interpreted as learning a Mahalanobis distance metric $\Phi W \Phi^{\mathsf{T}}$ in the feature space of K, where for $i \in \mathcal{X}$, $\Phi_i = \phi(i)$. If we further restrict W to be a diagonal matrix, it can be interpreted as learning a weighting over the training set which forms a $\mathcal{C}$-respecting metric. Moreover, restricting W to be diagonal changes the semidefinite constraint to a set of linear constraints: $W_{ii} \geq 0$, so the resulting problem can be solved by linear programming.

To allow for a tradeoff between dimensionality and constraint-satisfaction, we relax the distance constraints by introducing slack variables, which are then penalized with a positive scaling parameter β. We assume that K is precentered, allowing us to drop the centering constraint on A. The resulting algorithm can be seen as a special case of Algorithm 3.

3.3. Multiple kernels

For heterogeneous data, we learn a unified space from multiple kernels, each of which may code for different features. We now extend the reasoning of Section 3.2 to construct a multi-kernel embedding algorithm.

Let $K^{(1)}, K^{(2)}, \ldots, K^{(m)}$ be a set of kernel matrices, each having a corresponding feature map $\phi^{(p)}$, for $p \in 1, \ldots, m$. One natural way to combine the kernels is to look at the product space:

$$\phi(i) = \left(\phi^{(p)}(i)\right)_{p=1}^{m},$$

with inner product

$$\langle \phi(i), \phi(j) \rangle = \sum_{p=1}^{m} \left\langle \phi^{(p)}(i), \phi^{(p)}(j) \right\rangle.$$

The *product space kernel matrix* can be conveniently represented in terms of the original kernels: $\widehat{K} = \sum_p K^{(p)}$.

Previous kernel combination algorithms learn a weighting $\sum_p a_p K^{(p)}$ such that informative kernels get higher weight, thereby contributing more to the final prediction (Lanckriet et al., 2004). In general, the discriminative power of each $K^{(p)}$ may vary over different subsets of $\mathcal{X}$, and learning a single transformation of the combined kernel could fail to exploit this effect.

We therefore extend the formulation from the previous section by learning a separate transformation matrix $W^{(p)}$ for each kernel $K^{(p)}$. Each $W^{(p)}$ defines a partial embedding

$$g^{(p)}(x) = \left(W^{(p)}\right)^{1/2} K_x^{(p)},$$

wherein the distance is defined as

$$d^{(p)}(i,j) = \left(K_i^{(p)} - K_j^{(p)}\right)^{\mathsf{T}} W^{(p)} \left(K_i^{(p)} - K_j^{(p)}\right).$$

If we then concatenate the partial embeddings to form a single vector $g(x) = \left(g^{(p)}(x)\right)_{p=1}^{m} \in \mathbb{R}^{nm}$, the total distance is

$$d(i,j) = \sum_p d^{(p)}(i,j) = \|g(i) - g(j)\|^2.$$

We can then add regularization terms $\mathrm{Tr}(W^{(p)}K^{(p)})$ and apply the analysis from Section 3.2 to each portion of the objective and embedding function independently. The final multi-kernel embedding algorithm is given as Algorithm 3.

Algorithm 3 Multi-kernel partial order embedding

```
Input: objects $\mathcal{X}$, margin-weighted partial order $\mathcal{C}$, kernel matrices $K^{(1)}, K^{(2)}, \ldots, K^{(m)}$ over $\mathcal{X}$
Output: matrices $W_1, W_2, \ldots, W_m \succeq 0$.

Let $d(i,j) = \sum_p d^{(p)}(i,j)$.

$\max_{W^{(p)},\xi}$   $\sum_{i,j} d(i,j) - \beta \sum_{\mathcal{C}} \xi_{ijk\ell} - \gamma \sum_p \mathrm{Tr}\left(W^{(p)}K^{(p)}\right)$
$\forall i,j \in \mathcal{X}$   $(4n+1)(\mathrm{diam}(\mathcal{C})+1) \geq d(i,j)$
$\forall (i,j,k,\ell) \in \mathcal{C}$   $d(i,j) + e_{ijk\ell} - \xi_{ijk\ell} \leq d(k,\ell)$
    $\xi_{ijk\ell} \geq 0$
$\forall p \in 1, 2, \ldots, m$   $W^{(p)} \succeq 0$
```

In this context, diagonal constraints on $W^{(p)}$ not only simplifies the problem to linear programming, but carries the added interpretation of weighting the contribution of each (kernel, training point) pair in the construction of the embedding. A large value at $W_{ii}^{(p)}$ corresponds to point i being a landmark for the features compared by $K^{(p)}$.

4. Experiments

The following experiments were conducted with the SeDuMi and YALMIP optimization packages (Sturm, 1999; Löfberg, 2004). In the first experiment, we demonstrate both the parametric and non-parametric algorithms on a human perception modeling task. The second experiment formulates a hierarchical multi-class visualization problem as an instance of partial order embedding, which is then solved with the multi-kernel algorithm.

4.1. Relative image similarity

Our first experiment reproduces the human perception experiment of (Agarwal et al., 2007). The data consists of 55 rendered images of rabbits with varying surface reflectance properties, and 13049 human-derived relative similarity measurements. The constraint set contains numerous redundancies and inconsistencies, which we removed by randomly sampling constraints to produce a maximal partial order. This results in a constraint DAG of 8770 edges and length 55.

The constraint graph was further simplified by computing its transitive reduction (Aho et al., 1972), thereby suppressing redundant constraints, i.e., those which can be deduced from others. This results in an equivalent, but more compact representation of 2369 unique constraints, and a much simpler optimization problem.

Using unit margins, we first constructed a non-parametric embedding (Figure 2(a)) with Algorithm 2. The results are qualitatively similar to those presented in (Agarwal et al., 2007); the x-axis in the figure roughly captures gloss, and the y-axis captures brightness.

We then constructed a parametric embedding with a diagonally constrained W. For features, we used radial basis function (RBF) kernels over image intensity histograms. The embedding, shown in Figure 2(b), exhibits more structure than the non-parametric embedding, but still conforms to the constraints. Unlike in (Agarwal et al., 2007), this embedding extends to new points by applying the kernel function to the novel point and each point in the training set, and then applying the learned transformation to the resulting vector.

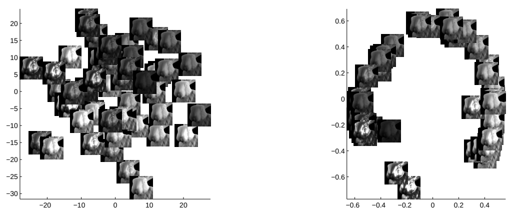

(a) (b)

Figure 2. (a) Non-parametric embedding of the rabbit data set. (b) Parametric embedding of the rabbit data set, using an RBF kernel over intensity histograms.

4.2. Hierarchical embedding

Our second experiment is a hierarchical visualization task. Given a set of labeled objects and a taxonomy over the labels, we seek a low-dimensional embedding which reflects the global structure of the labels. We selected 10 classes from the Amsterdam Library of Object Images (ALOI) (Geusebroek et al., 2005), which were then merged into 3 higher-level classes to form the taxonomy in Table 1.

A label taxonomy can be translated into partial order constraints by following the subset relation up from the leaves to the root of the hierarchy. Let $\mathrm{LCA}(i,j)$ denote the least common ancestor of i and j in the taxonomy tree; that is, the smallest subset which contains both i and j. For each internal node S in the taxonomy tree, we generate constraints of the form:

$$\{(i,j,i,k) : i,j,k \in S, \mathrm{LCA}(i,j) \neq S, \mathrm{LCA}(i,k) = S\}.$$

From each of the 10 classes, we sampled 10 images, each of varying out-of-plane rotation. Each image was full-color with dimensions 192×144. We parameterized the embedding by forming five kernels: $K^{(1)}$ is the inner product between grayscale images, and $K^{(2,\ldots,5)}$ are radial basis function (RBF) kernels over histograms in the red, green, blue channels, and grayscale intensity. All kernels were centered by $K \mapsto HKH$, and then normalized by $K_{ij} \mapsto K_{ij}/\sqrt{K_{ii}K_{jj}}$.

Figure 3(a) illustrates the PCA projection of the product-space kernel for this experiment. Although each of the 10 classes are individually clustered, the product space does not conform to the higher-level structure defined by the taxonomy.

When building the constraint graph, we randomly sampled edges to obtain an equal number of constraints at each depth. We supplied margin weights of the form $e_{ijk\ell} = 2^{\mathrm{depth}(k,\ell)-1}$. This forces tight clusters at lower levels in the taxonomy, but subset diameters increase rapidly at higher levels in the taxonomy. We applied Algorithm 3 with diagonal constraints on each $W^{(p)}$, resulting in the embedding shown in Figure 3(b).

Table 2 shows the fraction of satisfied taxonomy constraints (ignoring margins) for each kernel, before learning (i.e., distances native to the kernel) and after. The first 6 rows were computed from the single-kernel algorithm of Section 3.2, and the last was computed from the multi-kernel algorithm. The metric derived from the dot product kernel outperforms each other individual kernel, indicating that colors are not generally discriminative in this set. The metric learned from the product space kernel can be interpreted as a weighting over the data, with uniform weighting over kernels, and performs much worse than the single best kernel. However, the multi-kernel metric adapts each kernel individually, achieving the highest fraction of satisfied constraints.

Figure 3(c) depicts the weights learned by the algorithm. Although there were 500 weight variables, only 42 received non-zero values, largely eliminating the color features. As

Table 1. The label taxonomy for the hierarchical embedding experiment of Section 4.2.

All		
Clothing	Toys	Fruit
Shoe Hat White shoe	Xmas bear Pink animal Red/yellow block Smurf	Lemon Pear Orange

Table 2. The fraction of taxonomy constraints satisfied before and after learning in each kernel individually, the product-space kernel, and finally the multi-kernel embedding.

Kernel	Before learning	After learning
Dot product	**0.8255**	0.8526
RBF Red	0.6303	0.6339
RBF Green	0.6518	0.6746
RBF Blue	0.7717	0.8349
RBF Gray	0.6782	0.6927
Product	0.7628	0.7674
Multi	—	**0.9483**

Table 2 indicates, the color features are relatively weak when compared to the dot product kernel, but they can be combined to produce a much better embedding.

5. Hardness in $\mathbb{R}^1$

So far, we've focused on algorithms that attempt to produce low-dimensional embeddings, but it is natural to ask if solutions of minimal dimensionality can be found efficiently. As a special case, one may ask if any instance $(\mathcal{X}, \mathcal{C})$ can be satisfied in $\mathbb{R}^1$. However, as Figure 4 demonstrates, not all instances can be satisfied in the line.

Because rank constraints are not convex, convex optimization techniques cannot efficiently minimize dimensionality. This does not necessarily imply other techniques could not work.

However, we show that it is NP-Complete to decide if a given $\mathcal{C}$ can be satisfied in $\mathbb{R}^1$. A satisfying embedding can be verified in polynomial time by traversing the constraint graph, so it remains to show that the $\mathbb{R}^1$ partial order embedding problem (hereafter referred to as *1-POE*) is NP-Hard. We reduce from the following problem:

Definition 5.1 (Betweenness (Opatrny, 1979))**.** Given a finite set Z and a collection T of ordered triples (a, b, c) of distinct elements from Z, is there a one-to-one function $f : Z \to \mathbb{R}$ such that for each $(a, b, c) \in T$, either $f(a) < f(b) < f(c)$ or $f(c) < f(b) < f(a)$?

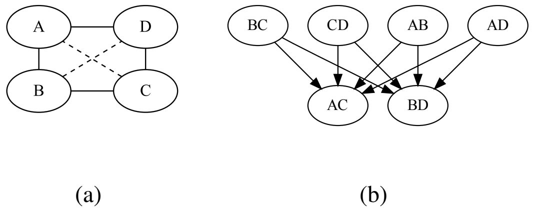


Figure 4. (a) A square in $\mathbb{R}^2$. (b) The partial order over distances induced by the square: each side is less than each diagonal. This constraint set cannot be satisfied in $\mathbb{R}^1$.

Theorem 5.1. *1-POE is NP-Hard.*

Proof. Let (Z, T) be an instance of Betweenness. Let $\mathcal{X} = Z$, and for each $(a, b, c) \in T$, introduce constraints (a, b, a, c) and (b, c, a, c) to $\mathcal{C}$. Since Euclidean distance in $\mathbb{R}^1$ is simply line distance, these constraints force b to lie between a and c. Therefore, the original instance $(Z, T) \in$ Betweenness if and only if the new instance $(\mathcal{X}, \mathcal{C}) \in$ 1-POE. Since Betweenness is NP-Hard, 1-POE is NP-Hard as well. $\square$

6. Conclusion

We have presented a metric learning algorithm which constructs a single unified space from heterogeneous data sources, supplied by multiple kernels. Our kernel combination technique adapts to varying utility of kernels over the training set.

Gearing our algorithm toward subjective similarity motivates the partial order framework, which enables graph-theoretic analysis of the constraints. This leads to more efficient algorithms and more reliable embeddings.

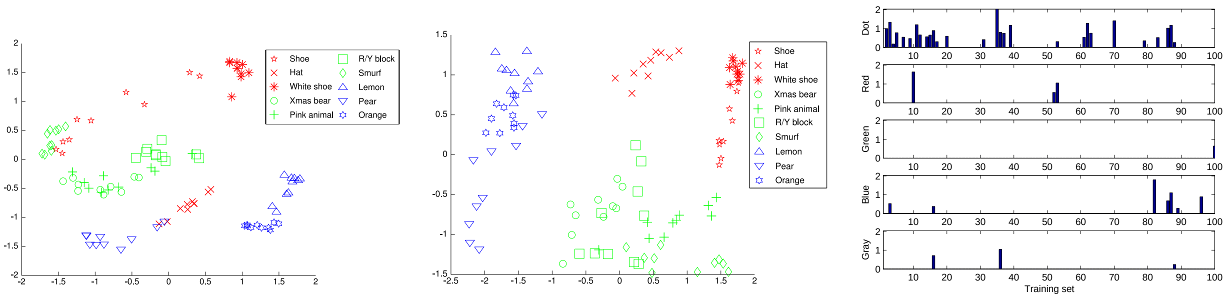
References

Agarwal, S., Wills, J., Cayton, L., Lanckriet, G., Kriegman, D., & Belongie, S. (2007). Generalized non-metric multi-dimensional scaling. *Proceedings of the Twelfth International Conference on Artificial Intelligence and Statistics*.

Aho, A. V., Garey, M. R., & Ullman, J. D. (1972). The transitive reduction of a directed graph. *SIAM Journal on Computing*, *1*, 131–137.

Bilenko, M., Basu, S., & Mooney, R. J. (2004). Integrating constraints and metric learning in semi-supervised clustering. *Proceedings of the Twenty-first International Conference on Machine Learning* (pp. 81–88).

Cox, T. F., & Cox, M. A. (1994). *Multidimensional scaling*. Chapman and Hall.



(a) (b) (c)

Figure 3. (a) PCA of the clothes/toy/fruit set in the native product-space kernel. Higher level classes (clothes, toys, fruit, coded by color) are not clustered. (b) PCA of the same set after learning: high-level structure is now captured in low dimensions. (c) The weighting learned to produce (b). From top to bottom: grayscale inner product, red, green, blue, and grayscale histogram RBF kernels.

Geusebroek, J. M., Burghouts, G. J., & Smeulders, A. W. M. (2005). The Amsterdam library of object images. *Int. J. Comput. Vis.*, *61*, 103–112.

Globerson, A., & Roweis, S. (2007). Visualizing pairwise similarity via semidefinite embedding. *Proceedings of the Twelfth International Conference on Artificial Intelligence and Statistics*.

Hadsell, R., Chopra, S., & LeCun, Y. (2006). Dimensionality reduction by learning an invariant mapping. *Computer Vision and Pattern Recognition*.

Kruskal, J. (1964). Nonmetric multidimensional scaling: a numerical method. *Psychometrika*, *29*.

Lanckriet, G. R. G., Cristianini, N., Bartlett, P., Ghaoui, L. E., & Jordan, M. I. (2004). Learning the kernel matrix with semidefinite programming. *J. Mach. Learn. Res.*, *5*, 27–72.

Löfberg, J. (2004). *Computer Aided Control Systems Design, 2004 IEEE International Symposium on*, 284–289.

Opatrny, J. (1979). Total ordering problem. *SIAM J. Computing*, 111–114.

Roth, V., Laub, J., Buhmann, J. M., & Müller, K.-R. (2003). Going metric: denoising pairwise data. *Advances in Neural Information Processing Systems 15* (pp. 809–816). Cambridge, MA: MIT Press.

Schölkopf, B., Herbrich, R., Smola, A. J., & Williamson, R. (2001). A generalized representer theorem. *Proceedings of the 14th Annual Conference on Computational Learning Theory* (pp. 416–426).

Schultz, M., & Joachims, T. (2004). Learning a distance metric from relative comparisons. *Advances in Neural Information Processing Systems 16*. Cambridge, MA: MIT Press.

Song, L., Smola, A., Borgwardt, K., & Gretton, A. (2008). Colored maximum variance unfolding. *Advances in Neural Information Processing Systems* (pp. 1385–1392). Cambridge, MA: MIT Press.

Sturm, J. F. (1999). Using sedumi 1.02, a matlab toolbox for optimization over symmetric cones. *Optimization methods and software*, *11–12*, 625–653.

Tsang, I. W., & Kwok, J. T. (2003). Distance metric learning with kernels. *International Conference on Artificial Neural Networks* (pp. 126–129).

Varga, R. (2004). *Geršgorin and his circles*. Springer-Verlag.

Wagstaff, K., Cardie, C., Rogers, S., & Schroedl, S. (2001). Constrained k-means clustering with background knowledge. *Proceedings of the Eighteenth International Conference on Machine Learning* (pp. 577–584).

Weinberger, K. Q., Blitzer, J., & Saul, L. K. (2006). Distance metric learning for large margin nearest neighbor classification. *Advances in Neural Information Processing Systems 18* (pp. 451–458). Cambridge, MA: MIT Press.

Weinberger, K. Q., Sha, F., & Saul, L. K. (2004). Learning a kernel matrix for nonlinear dimensionality reduction. *Proceedings of the Twenty-first International Conference on Machine Learning* (pp. 839–846).

Xing, E. P., Ng, A. Y., Jordan, M. I., & Russell, S. (2003). Distance metric learning, with application to clustering with side-information. *Advances in Neural Information Processing Systems 15* (pp. 505–512). Cambridge, MA: MIT Press.