Learning a kernel for discriminative, low-dimensional embedding of partially labeled data

Paul Vernaza, Daniel D. Lee, and Ben Taskar
GRASP Laboratory
University of Pennsylvania

April 28, 2009

Abstract

We describe a novel way of producing a low-dimensional, topology-respecting embedding of partially labeled data that also admits linear separation between classes. Our method draws upon concepts popularized in the classic Linear Discriminant Analysis algorithm as well as the more recent development of Maximum Variance Unfolding. The algorithm is formulated as a simple convex optimization problem that can be solved exactly and in polynomial time with commonly available software. Additionally, the convex formulation is very flexible, admitting a variety of useful variations. Furthermore, we show how our algorithm can be interpreted as a powerful data-driven kernel learning method that is able to leverage application-specific distance information to produce discriminative kernels that generalize well without much hand-tuning. Results are shown that demonstrate the method's ability to produce embeddings that are of high quality, in a qualitative as well as a quantitative sense.

1 Introduction

Dimensionality reduction and class discrimination via kernel machines are two concepts that may superficially seem to be at odds with one another. On the one hand, dimensionality reduction usually entails embedding the data in a space of reduced dimension that preserves most of its interesting details. On the other hand, supervised kernel machines implicitly embed the data in a higher dimensional feature space in order to aid discrimination. Thus, while recent unsupervised nonlinear dimensionality reduction techniques are able to produce low-dimensional views of high-dimensional data that seem qualitatively "natural" to the human eye, we usually have little reason to believe that such views are useful for discrimination via kernel machines [13].

It is therefore perhaps a natural question to wonder whether these competing goals could be unified in some common framework. Our method, which we refer henceforth as Discriminative Variance Unfolding (DVU), can be thought of as a kernel machine that first learns a low-dimensional view of the data—or equivalently, the kernel—that is salient for classification, and then classifies the data with it. Afterwards, we can visualize the data via the equivalent of kernel PCA [11] to take a look inside the mind of the kernel machine. We will argue and demonstrate that the resulting view is a very natural view of the data that also lends itself well to discrimination.

1.1 Relation to Linear Discriminant Analysis

We draw our inspiration principally from two sources. The first is Linear Discriminant Analysis (LDA), published in its first form by Fisher [5]. LDA attempts to find linear projections of the data that are optimal for discrimination, in the case of normally distributed data. Specifically, it finds functions of the form $f_i(x) = w_i^T x$ such that the data look maximally separated after mapping by these functions. This is accomplished by finding a basis of functions w_i that each maximize the projected-variance ratio given by

$$\frac{w_i^T \Sigma_{bc} w_i}{w_i^T \Sigma_{ic} w_i} \tag{1}$$

where Σ_{bc} is the "between-class" covariance of class centroids, and Σ_{ic} is "in-class" covariance of the classes (assuming homoscedasticity; i.e., all classes are Gaussian with a common covariance matrix). This optimization is easily solved as a generalized eigenvalue problem. If we choose not to find a full basis of functions, but instead only the most salient ones, LDA serves as a sort of dimensionality reduction that is class-aware; i.e., it finds an alternate view of the data in which we hope that the data appear separable in some simple way.

Of course, we are not always so lucky as to have a simple linear transformation that makes the data look separable in a low-dimensional space. Accordingly, there exist various nonlinear extensions of LDA, such as Flexible Discriminant Analysis [6], which views FDA as an optimal scoring method and replaces the linear functions mentioned above with arbitrary predictors.

Another straightforward extension of LDA consists of applying the venerable kernel trick [9], where we perform a change of basis into some notionally high-dimensional linear space, where we hope that the data are more separable. In this case, we usually choose the kernel function a-priori, comforted by the notion that using the kernel trick with this function actually corresponds to performing some mapping into another linear "feature" space. The form of this functional mapping to feature space is fixed by the kernel function.

The idea we propose in this paper might be considered a way to learn the kernel function with a goal that is similar in spirit to LDA. Equivalently, it may be thought of a method to construct a sensible mapping into feature space that accomplishes some set goals. We begin by enumerating some of these desiderata.

1. First, as in LDA, the mapping should have the goal of making the data look separable to facilitate classification and visualization.

2. However, the mapping should be plausible with respect to the original features in some way; i.e., it should be appropriately regularized. Otherwise, a trivial mapping could be constructed to perfectly classify the data. It is reasonable to expect (in most settings) that the information pertinent to classification is encoded in the topology of the data, as opposed to the specific embedding. Borrowing an idea from the transductive setting [1], we use the idea of having a sort of topological prior as regularization in finding a plausible map.

3. The mapping should also be regularized in the sense of producing a relatively low-dimensional mapping into feature space. Again, otherwise, we could construct a mapping into an extremely high-dimensional space where the data are separable by virtue of overwhelming dimensionality.

4. Simultaneously, we do not want to enforce too strong a prior on the class of allowable mappings—such as is the case with fixed kernel functions, where our mapping is implicitly chosen from a certain class of mappings. Ideally, the map should be highly nonparametric to allow it to fit any conceivable data distribution.

In the following, we will show how these issues can be addressed by solving a suitable convex optimization problem. We begin by discussing a related unsupervised learning method.

2 Relation to Maximum Variance Unfolding

Recent years have seen an ever-increasing appreciation for convex optimization in machine learning. In particular, we were inspired by Weinberger and Saul's application of convex optimization to the nonlinear dimensionality reduction problem, in a method now commonly known as Maximum Variance Unfolding (MVU) [13].

Our method was originally conceived as an extension of MVU to the case of labeled data. In this sense, it is similar to [12], where a "colored" variant of MVU was developed to bias the MVU solution towards

prior data. Our method is primarily different in that it explicitly attempts to model class differences, and optimizes separability, as opposed to statistical dependence on side-information.

To develop the method, it will be useful to describe the general idea of MVU. MVU optimizes the following problem over the embedded vectors $x_i \in \mathbb{R}^n$:

$$\begin{aligned} \max_{x} \quad & \sum_{i \in \mathcal{N}} \|x_i\|^2 \\ \text{subject to} \quad & \|x_i - x_j\| = d_{ij},\ (i,j) \in \mathcal{E} \\ & \sum_{i \in \mathcal{N}} x_i = 0 \end{aligned}$$

We are given a graph with edge set $\mathcal{E}$ whose nodes $i \in \mathcal{N}$ represent observed data points and edges represent connections between local neighbors with associated distances d_{ij}. The optimal x_i constitute an embedding of the data respecting these distances while maximizing a measure of the variance of the embedding. This problem is not convex, since it has nonconvex constraints and a nonconcave objective. To circumvent this issue, MVU operates in the space of the Gram matrix $X_{ij} = \langle x_i, x_j \rangle$, reformulating the problem as

$$\begin{aligned} \max \quad & \operatorname{tr}(X) \\ \text{subject to} \quad & X_{ii} + X_{jj} - 2X_{ij} = d_{ij}^2,\ (i,j) \in \mathcal{E} \\ & \sum_{ij} X_{ij} = 0 \\ & X \succeq 0 \end{aligned}$$

where $X \succeq 0$ means that X is constrained to be positive semidefinite and symmetric. This problem is concave with convex constraints, and is completely equivalent save for one subtle difference. The x_i are recovered by an eigendecomposition of X. The resulting x_i will therefore be of dimensionality equal to the number of observations instead of n. The number of unused dimensions is equal to the dimensionality of the null space of X. The *affine dimension* of the set of x_i is therefore equal to the rank of the Gram matrix X.

In practice, the embedding is obtained by truncating the spectrum in the expansion of the x_i, embedding the data in $\mathbb{R}^d$, where $d \ll n$. In order for this approximation to be valid, the spectrum of X should be as sparse as possible. This notion plays an important part in the development of our own method, which follows.

3 Formulation as a convex program

The same Gram matrix trick used in MVU forms the foundation of our formulation, as described shortly. In the following discussion, we assume we are given vectors x_i, some of which have labels in the set $\{1 \dots K\} \in \mathcal{K}$. We wish to find an embedding of the points in a similar manner as MVU, but that also optimizes class separation. We will accomplish this by solving a convex optimization problem that does this and addresses the other aforementioned desiderata.

3.1 Class separation objective

To address our first desideratum, we use an objective similar to LDA; i.e., we wish to maximize separability by minimizing in-class variance while maximizing between-class variance. To make these terms precise, we consider the act of decomposing the Gram matrix via eigendecomposition, as described above. This is equivalent to principal component analysis (PCA), which is equivalent to finding an ordered basis of $\mathbb{R}^n$

consisting of vectors that successively maximize one-dimensional projections of variance. In other words, let us write

$$X = Z^T Z = \begin{bmatrix} x_1^T \\ \vdots \\ x_n^T \end{bmatrix} \begin{bmatrix} x_1 & \cdots & x_n \end{bmatrix} \tag{2}$$

Assuming that the points are centered, the empirical covariance matrix of the data is then given by $\Sigma = ZZ^T$. A scalar measure of the covariance matrix is then given by its trace norm, $\|\Sigma\|_{tr} = \text{trace}(\Sigma) = \text{trace}(X) = \sum_{i=1}^n X_{ii}$, which is equal to the sum of the distances of the points x_i from their mean. When we speak of the variance of a set of points in $\mathbb{R}^d$, we mean precisely this definition. Therefore, assuming that a set of points and its mean are represented in the Gram matrix, we can compute the variance of that set of points with a similar formula.

In particular, consider all x_i whose label is k. Assume that there is a point $x_{\hat{k}}$ represented in the Gram matrix such that $\langle x_i, x_{\hat{k}} \rangle = X_{i\hat{k}}$. If $x_{\hat{k}}$ is constrained to be the mean of all x_i with label k, we can write the *in-class variance* of the kth class, $\sigma_k^2(X)$, as follows:

$$\sigma_k^2(X) = \frac{1}{|\mathcal{K}_k|} \sum_{i \in \mathcal{K}_k} \|x_i - x_{\hat{k}}\|_2^2 = \frac{1}{|\mathcal{K}_k|} \sum_{i \in \mathcal{K}_k} X_{ii} + X_{k\hat{k}} - 2X_{i\hat{k}} \tag{3}$$

where $\mathcal{K}_k$ is the set of i such that the label of $x_i = k$. The overall in-class variance is then defined to be this quantity averaged over all the classes, i.e.:

$$\sigma_{icv}^2(X) = \frac{1}{|\mathcal{K}|} \sum_{k \in \mathcal{K}} \sigma_k^2 \tag{4}$$

Now assume there is a point x_μ that is constrained to be the mean of the $x_{\hat{k}}$. We then define the *between-class variance* σ_{bc}^2 as

$$\sigma_{bcv}^2(X) = \frac{1}{|\mathcal{K}|} \sum_{k \in \mathcal{K}} X_{kk} + X_{\mu\mu} - 2X_{k\mu} \tag{5}$$

To use these definitions in a convex program, we need to enforce the constraints that the special points defined above be centroids of their respective classes. These centering constraints are analogous to the global centering constraint $\sum_{ij} X_{ij} = 0$ in MVU.

With $X = Z^T Z$ as before, the centering constraint for class k can be written as $Zc_k = 0$, for a suitable vector c_k, which is equivalent to the condition that $\|Zc_k\|^2 = 0 = c_k^T Z^T Z c_k = \text{tr}(c_k c_k^T X) = 0$ (assuming X is positive semidefinite), which is a linear constraint in X.

A LDA-like objective would then be to maximize the linear-fractional function $\sigma_{bc}^2(X)/\sigma_{ic}^2(X)$. However, this is not a concave function of X. Although it is quasiconvex and can be optimized by solving a sequence of convex feasibility problems [2], for simplicity and other considerations, we choose to minimize the following simpler term in the objective:

$$\sigma_{icv}^2(X) - \sigma_{bcv}^2(X) \tag{6}$$

3.2 Regularization

3.2.1 Topological regularization

By "topological regularization", we mean a kind of regularization that ensures our mapping into feature space is in some sense sufficiently smooth with respect to the topology of the data, as in [1]. In our case, this means encouraging a mapping that does not overly distort the intrinsic distances among exemplars. Since our mapping is completely nonparametric, this is of the utmost importance.

A very important issue is therefore how to generate these "intrinsic distances", and how many of them to include in the optimization. We will not dwell on it, but only underscore the importance of this issue.

It is expected that preservation of only local distances will not always be sufficient to preserve other distances after embedding. This suggests sampling longer distances and preserving those as well. Clearly, we could add all pairwise distances, but this would adversely impact the computational complexity of the optimization. It seems reasonable that local distances in addition to sparsely-sampled longer distances should be sufficient for the purpose of regularization, since much of the distance information is redundant.

To give a concrete example, we found that we were able to obtain good results on one dataset by preserving distances to the two nearest neighbors per example, in addition to a randomly chosen example. This is also a simple way of ensuring that the graph is connected with high probability; if it is not, we may end up with a unbounded objective.

As for the generation of these distances, they should reflect the known structure of the data as much as possible. In the results section, we will discuss a concrete example where distances can be constructed using prior knowledge of invariances that should be reflected in the embedding.

Given a graph with distances, there is now the matter of how to add constraints to enforce them. In some cases, it may be sufficient to enforce hard distance constraints, as in MVU:

$$D_{ij} = X_{ii} + X_{jj} - 2X_{ij} = \bar{D}_{ij} \tag{7}$$

where D_{ij} is the squared distance between x_i and x_j expressed as a linear function of X and $\hat{D}_{ij}$ is the original distance between x_i and x_j. This would be desirable from the perspective of ensuring a faithful embedding. However, there are certain cases where our distance is not actually a distance metric; i.e., the distance function might not satisfy triangle inequality. For example, this would appear to be the case with the tangent distances used in our experiments. In this case, the distances are not embeddable in Euclidean space of any dimension, and we are therefore forced to relax the distance constraints a bit.

One way to achieve this is via simple interval constraints. Introducing new decision variables ϵ_{ij} yields the constraints

$$D_{ij} \geq (1 - \epsilon_{ij})\bar{D}_{ij} \tag{8}$$
$$D_{ij} \leq (1 + \epsilon_{ij})\bar{D}_{ij} \tag{9}$$

A penalty of the form $\sum_{i,j} \epsilon_{ij}$ is then added to the objective.

3.2.2 Rank regularization

As mentioned earlier, low-dimensional embeddings are associated with low-rank Gram matrices. We should therefore have a sort of rank regularization term in the objective. The core concept of MVU is that maximizing variance subject to distance constraints tends to produce low-rank solutions.

We have observed that minimizing $\sigma^2_{icv} - \sigma^2_{bcv}$ has a similar effect. It also has the benefit that it does not affect the placement of the unlabeled data, whereas variance maximization slightly biases the placement of the unlabeled points away from the origin.

3.3 Summary

Based on the preceding discussion, we therefore propose the following convex optimization problem:

$$\begin{aligned}
\min \quad & \sigma^2_{icv}(X) - \sigma^2_{bcv}(X) + \lambda \sum_{ij} \epsilon_{ij} \\
\text{subject to} \quad & D_{ij} = X_{ii} + X_{jj} - 2X_{ij},\ \forall (i,j) \in \mathcal{E} \\
\text{(distance constraints)} \quad & D_{ij} \geq (1 - \epsilon_{ij})\bar{D}_{ij},\ \forall (i,j) \in \mathcal{E}
\end{aligned}$$

$$
\begin{aligned}
& D_{ij} \leq (1+\epsilon_{ij})\bar{D}_{ij},\ \forall (i,j) \in \mathcal{E} \\
\text{(centering constraints)} \quad & \mathrm{tr}(c_k c_k^T X) = 0,\ k \in \mathcal{K} \cup \mu \\
& \mathrm{tr}(11^T X) = 0 \\
\text{(positive semidefiniteness)} \quad & X \succeq 0
\end{aligned}
$$

with σ^2_{icv}, σ^2_{bcv}, and c_k as defined previously. Note that this optimization allows us to embed both labeled and unlabeled data. In order to use DVU for supervised classification, it is necessary to embed the test points at training time. Supervised classification can then be performed using the DVU representation of the data.

3.4 Variations

We would like to briefly point out that there are many potentially useful variations on this program that maintain convexity. Convexity encompasses a surprisingly wide range of constraints and objectives, some of which we mention here.

3.4.1 Trace minimization vs. maximization

One might choose to add an explicit trace maximization term, as in MVU. In this case, some scaling issues need to be accounted for in order to ensure the solution is bounded (technically, this is also an issue with the formulation above, though the scaling issue is less dramatic).

Another way to handle the unboundedness issue is to minimize the trace of the Gram matrix. Surprisingly (or not), this is also a sensible regularization term that has been shown to produce low-rank solutions in other problems [4].

3.5 Quadratic distance penalties

MVU applications elsewhere [14][12] have suggested the use of distance penalties of the form $(D_{ij} - \bar{D}_{ij})^2$. We note that this penalty overapproximates the quadratic penalty $(d_{ij} - \bar{d}_{ij})^2$ by a factor of $(d_{ij} + \bar{d}_{ij})^2$; i.e., violations of longer distances are penalized far more than short distances.

We note that the quadratic penalty may be written as $D_{ij} + \bar{D}_{ij} - 2\sqrt{D_{ij}\bar{D}_{ij}}$, which is the sum of three convex terms, (the last being the negative geometric mean), and is therefore convex.

3.6 Signal-to-noise ratio

We may also explicitly set a desired minimum "signal-to-noise" ratio by adding the simple linear constraint

$$\sigma^2_{bcv} \geq \epsilon \sigma^2_{icv} \tag{10}$$

Alternatively, it is possible to use a more LDA-like variance ratio penalty without too much trouble, by enforcing the following constraint:

$$\frac{(\sigma^2_{icv})^2}{\sigma^2_{bcv}} \leq \epsilon \sigma^2_{icv} + \xi,\ \xi \geq 0 \tag{11}$$

Here ϵ is a desired nominal noise-to-signal ratio, and ξ is a nonnegative slack variable that is penalized in the objective.

4 Results

We implemented DVU using the optimization modeling tool YALMIP [8] and the CSDP solver. We chose to test on the challenging USPS digits dataset. The USPS dataset is additionally interesting in our case because it lends itself well to the use of a particular distance measure, known as the tangent distance [10].

M-SVM (pre)	M-SVM (post)	3-NN (pre, Euc. distance)	3-NN (post)	3-NN (pre, tangent distance)
88.8	96.9	91.7	97.7	97.1

Table 1: Results of supervised classification experiments (percent accuracy). "pre" indicates that classifier was trained/evaluated on original input space, "post" indicates it was trained/evaluated on DVU components.

Briefly, the tangent distance is a distance measure that is invariant to certain known transformations such as translation, rotation, scaling, and dilation. Using the tangent distance in conjunction with a simple nearest neighbor classifier yields state-of-the art error rates on this difficult dataset. By using tangent distances during the embedding process, we are essentially learning a kernel that builds in these invariances.

We first preprocessed the USPS digits by aligning (deslanting) them. We then embedded 2500 digits using DVU, hiding 20% of the labels to use as test data. As seen in Figure 1, DVU finds an embedding that very effectively discriminates between classes in a low-dimensional space, but is it a faithful representation?

To answer this question, we performed supervised classification experiments using the DVU representation of the data. We implemented both a multiclass SVM [3] and a simple k-nearest neighbor classifier. We note that using the learned kernel in conjunction with the M-SVM was as simple as using a linear kernel on the DVU components. Both the SVM and nearest neighbor (post-embedding) were trained on the top 10 dimensions of the embedding. Results are given in Table 1.

Clearly the DVU components are extremely useful for classification, as a linear kernel machine trained on them performs nearly as well as the nearest-neighbor tangent distance classifier on the original high-dimensional space. Without the benefit of the embedding, the SVM performs relatively poorly. Furthermore, the best performance was achieved by using a 3-NN classifier on the DVU components.

5 Conclusions

We have demonstrated a novel algorithm that unifies and extends concepts from both LDA and MVU. The algorithm is formulated as a convex problem that is readily solved via widely-available tools. We have also shown how the convex formulation is fairly flexible, allowing a wide range of expressivity for the constraints and objective. Finally, we have shown that it is able to achieve classification performance that is on par with state-of-the-art methods with little effort.

Perhaps the most pressing area of future investigation on DVU is in regard to computational efficiency. Though the SDP can be solved in polynomial time, its solution is not feasible for very large-scale problems. We are investigating the use of specialized solvers, such as spectral bundle methods [7], in addition to exploring ways to reduce the size of the formulation using graph Laplacian regularization [14].

We are also investigating ways to produce an "out-of-sample" extension.

References

[1] Mikhail Belkin, Partha Niyogi, and Vikas Sindhwani. On manifold regularization. In *Proceedings of the Tenth International Workshop on Artificial Intelligence and Statistics (AISTATS)*, 2005.

[2] S. Boyd and L. Vandenberghe. *Convex Optimization*. Cambridge University Press, 2004.

[3] Koby Crammer and Yoram Singer. On the algorithmic implementation of multiclass kernel-based vector machines. *Journal of Machine Learning Research*, 2001.

[4] Maryam Fazel, Haitham Hindi, and Stephen P. Boyd. A rank minimization heuristic with application to minimum order system approximation. In *Proceedings of the American Control Conference*, June 2001.

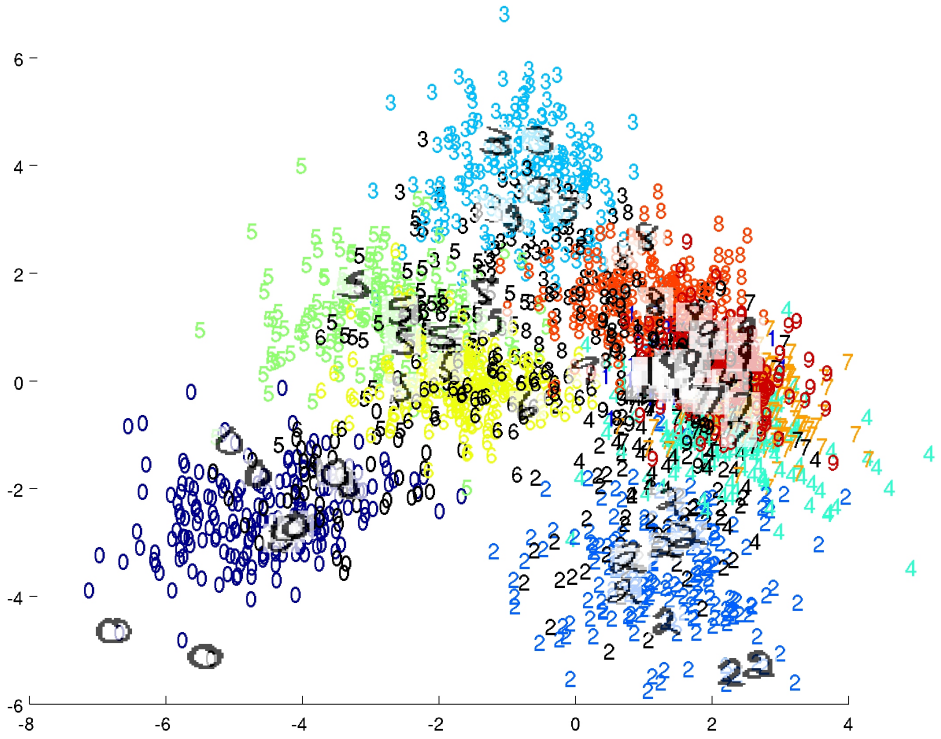

Figure 1: DVU embedding of 2500 USPS digits (deslanted), 80% labeled

[5] R. A. Fisher. The use of multiple measures in taxonomic problems. *Annals of Eugenics*, 1936.

[6] T. Hastie, R. Tibshirani, and A. Buja. Flexible discriminant analysis by optimal scoring. *J. Amer. Statist. Assoc.*, 1994.

[7] C. Helmberg and F. Rendl. A spectral bundle mehtod for semidefinite programming. *SIAM Journal on Optimization*, 2000.

[8] J. Lfberg. YALMIP : A toolbox for modeling and optimization in MATLAB. In *Proceedings of the CACSD Conference*, Taipei, Taiwan, 2004.

[9] S. Mika, G. Ratsch, J. Weston, and K. R. Mullers. Fisher discriminant analysis with kernels. In *Neural Networks for Signal Processing IX*, 1999.

[10] J. S. Denker P. Y. Simard, Y. A. LeCun and B. Victorri. Transformation invariance in pattern recognition: tangent distance and propagation. *Neural Networks: Tricks of the Trade*, 1998.

[11] B. Scholkopf, A. Smola, and K.R. Muller. *Advances in kernel methods–Support vector learning*, chapter Kernel principal component analysis. MIT Press, 1998.

[12] Le Song, Alex Smola, Karsten Borgwardt, and Arthur Gretton. Colored maximum variance unfolding. In *Neural Information Processing Systems*, 2007.

[13] K. Q. Weinberger, F. Sha, and L. K. Saul. Learning a kernel matrix for nonlinear dimensionality reduction. In *Proceedings of the Twenty First International Conference on Machine Learning (ICML-04)*, pages 839–846, Banff, Canada, 2004.

[14] Kilian Q. Weinberger, Fei Sha, Qihui Zhu, and Lawrence K. Saul. Graph laplacian regularization for large-scale semidefinite programming. In *Neural Information Processing Systems*, 2007.