Out-of-Sample Extensions for LLE, Isomap, MDS, Eigenmaps, and Spectral Clustering

Yoshua Bengio, Jean-François Paiement and Pascal Vincent
Département d'Informatique et Recherche Opérationnelle
Université de Montréal
Montréal, Québec, Canada, H3C 3J7
{bengioy,paiemeje,vincentp}@iro.umontreal.ca

Technical Report 1238,
Département d'Informatique et Recherche Opérationnelle

July 25, 2003

Abstract

Several unsupervised learning algorithms based on an eigendecomposition provide either an embedding or a clustering only for given training points, with no straightforward extension for out-of-sample examples short of recomputing eigenvectors. This paper provides algorithms for such an extension for Local Linear Embedding (LLE), Isomap, Laplacian Eigenmaps, Multi-Dimensional Scaling (all algorithms which provide lower-dimensional embedding for dimensionality reduction) as well as for Spectral Clustering (which performs non-Gaussian clustering). These extensions stem from a unified framework in which these algorithms are seen as learning eigenfunctions of a kernel. LLE and Isomap pose special challenges as the kernel is training-data dependent. Numerical experiments on real data show that the generalizations performed have a level of error comparable to the variability of the embedding algorithms to the choice of training data.

1 Introduction

In the last few years, many unsupervised learning algorithms have been proposed which share the use of an eigendecomposition for obtaining a lower-dimensional embedding of the data that characterizes a non-linear manifold near which the data would lie: Local Linear Embedding (LLE) (Roweis and Saul, 2000), Isomap (Tenenbaum, de Silva and Langford, 2000) and Laplacian Eigenmaps (Belkin and Niyogi, 2003). There are also many variants of Spectral Clustering (Weiss, 1999; Ng, Jordan and Weiss, 2002), in which such an embedding is an intermediate step before obtaining a clustering of the data that can capture flat, elongated and even curved clusters. The two tasks (manifold

learning and clustering) are linked because the clusters that spectral clustering manages to capture can be arbitrary curved manifolds (as long as there is enough data to locally capture the curvature of the manifold).

2 Common Framework

In this paper we consider five types of unsupervised learning algorithms that can be cast in the same framework, based on the computation of an embedding for the training points obtained from the principal eigenvectors of a symmetric matrix.

Algorithm 1

1. *Start from a data set $D = \{x_1, \ldots, x_n\}$ with n points in some space. Construct a $n \times n$ "neighborhood" or similarity matrix M. Let us denote $K_D(.,.)$ (or K for shorthand) the two-argument function (sometimes dependent on D) which produces M by $M_{ij} = K_D(x_i, x_j)$.*

2. *Optionally transform M, yielding a "normalized" matrix $\tilde{M}$. Equivalently, this corresponds to applying a symmetric two-argument function $\tilde{K}_D$ to each pair of examples (x_i, x_j) to obtain $\tilde{M}_{ij}$.*

3. *Compute the m largest eigenvalues λ'_j and eigenvectors v_j of $\tilde{M}$. Only positive eigenvalues should be considered.*

4. *The embedding of each example x_i is the vector y_i with y_{ij} the i-th element of the j-th principal eigenvector v_j of $\tilde{M}$. Alternatively (MDS and Isomap), the embedding is e_i, with $e_{ij} = \sqrt{\lambda'_j} y_{ij}$. If the first m eigenvalues are positive, then $e_i.e_j$ is the best approximation of $\tilde{M}$ using only m coordinates, in the squared error sense.*

In the following, we consider the specializations of Algorithm 1 for different unsupervised learning algorithms. Let S_i be the i-th row sum of the affinity matrix M:

$$S_i = \sum_j M_{ij}. \tag{1}$$

We say that two points (a, b) are **k-nearest-neighbors of each other** if a is among the k nearest neighbors of b in $D \cup \{a\}$ or vice-versa.

2.1 Multi-Dimensional Scaling

Multi-Dimensional Scaling (MDS) starts from a notion of distance or affinity K that is computed between each pair of training examples. We consider here metric MDS (Cox and Cox, 1994). For the normalization step 2 in Algorithm 1, these distances are converted to equivalent dot products using the "double-centering" formula:

$$\tilde{M}_{ij} = -\frac{1}{2}(M_{ij} - \frac{1}{n}S_i - \frac{1}{n}S_j + \frac{1}{n^2}S_i S_j). \tag{2}$$

The embedding e_{ik} of example x_i is given by $\sqrt{\lambda_k} v_{ik}$ where $v_{.k}$ is the k-th eigenvector of $\tilde{M}$. Note that if $M_{ij} = ||y_i - y_j||^2$ then $\tilde{M}_{ij} = (y_i - \bar{y}).(y_j - \bar{y})$, a centered dot-product, where $\bar{y}$ is the average value of y_i.

2.2 Spectral Clustering

Spectral clustering (Weiss, 1999) can yield impressively good results where traditional clustering looking for "round blobs" in the data, such as K-means, would fail miserably. It is based on two main steps: first embedding the data points in a space in which clusters are more "obvious" (using the eigenvectors of a Gram matrix), and then applying a classical clustering algorithm such as K-means, e.g. as in (Ng, Jordan and Weiss, 2002). Before applying K-means, it has been found useful as discussed in (Weiss, 1999) to normalize the coordinates of each point to have unit norm, i.e. to project the embedded points on the unit sphere, mapping the j-th coordinate of the embedding y_i of the i-th example (i.e. the i-th element of the j-th eigenvector) by $\frac{y_{ij}}{||y_i||}$. The affinity matrix M is formed using a kernel such as the Gaussian kernel. Several normalization steps have been proposed. Among the most successful ones, as advocated in (Weiss, 1999; Ng, Jordan and Weiss, 2002), is the following:

$$\tilde{M}_{ij} = \frac{M_{ij}}{\sqrt{S_i S_j}}. \tag{3}$$

To obtain m clusters, the first m principal eigenvectors of $\tilde{M}$ are computed and K-means applied on the resulting unit-norm coordinates.

2.3 Laplacian Eigenmaps

Laplacian Eigenmaps is a recently proposed dimensionality reduction procedure (Belkin and Niyogi, 2003) that has been proposed for *semi-supervised learning*. The authors use an approximation of the Laplacian operator such as the Gaussian kernel or the matrix whose element (i, j) is 1 if x_i and x_j are k-nearest-neighbors and 0 otherwise. Instead of solving an ordinary eigenproblem, the following generalized eigenproblem is solved:

$$(S - M)y_i = \lambda_i S y_i \tag{4}$$

with eigenvalues λ_i, eigenvectors y_i and S defined in eq. (1). The smallest eigenvalue is left out and the eigenvectors corresponding to the other small eigenvalues are used for the embedding. This is the same embedding that is computed with the spectral clustering algorithm from Shi and Malik (1997) . As noted in (Weiss, 1999) (Normalization Lemma 1), an equivalent result (up to a componentwise scaling of the embedding) can be obtained by considering the principal eigenvectors v_k of the normalized matrix defined in eq. (3).

2.4 Isomap

Isomap (Tenenbaum, de Silva and Langford, 2000) generalizes MDS to non-linear manifolds. They are based on replacing Euclidean distance by an approximation of the geodesic distance on the manifold. We define the *geodesic distance with respect to a data set D, a distance $d(u, v)$ and a neighborhood k* as follows:

$$\tilde{D}(a, b) = \min_p \sum_i d(p_i, p_{i+1}) \tag{5}$$

where p is a sequence of points of length $l \geq 2$ with $p_1 = a$, $p_l = b$, $p_i \in D \; \forall i \in \{2, \ldots, l-1\}$ and (p_i,p_{i+1}) are k-nearest-neighbors. The length l is free in the minimization. The Isomap algorithm obtains the normalized matrix $\tilde{M}$ from which the embedding is derived by transforming the raw pairwise distances matrix as follows: (1) compute the matrix $M_{ij} = \tilde{D}^2(x_i, x_j)$ of squared geodesic distances with respect to the data D and (2) apply to this matrix the distance-to-dot-product transformation (eq. (2)), as for MDS. Like for MDS, the embedding is $e_{ik} = \sqrt{\lambda_k} v_{ik}$ rather than $y_{ik} = v_{ik}$.

2.5 LLE

The Local Linear Embedding (LLE) algorithm (Roweis and Saul, 2000) looks for an embedding that preserves the local geometry in the neighborhood of each data point. First, a sparse matrix of local predictive weights W_{ij} is computed, such that $\sum_j W_{ij} = 1$, $W_{ij} = 0$ if x_j is not a k-nearest-neighbor of x_i and $\sum_j (W_{ij} x_j - x_i)^2$ is minimized. Then the matrix

$$M = (I - W)'(I - W) \tag{6}$$

is formed (possibly with the addition of a small diagonal term for regularization.) The embedding is obtained from the lowest eigenvectors of M, except for the smallest eigenvector which is uninteresting because it is $(1, 1, \ldots 1)$, with eigenvalue 0. Note that the lowest eigenvectors of M are the largest eigenvectors of $\tilde{M} = I - M$ to fit Algorithm 1. The embedding is given by $y_{ik} = v_{ik}$.

3 From Eigenvectors to Eigenfunctions

To obtain an embedding for a new data point, we propose to use the Nyström formula (Baker, 1977), which has been used successfully to speed-up kernel methods computations by focussing the heavier computations (the eigendecomposition) on a subset of examples. The use of this formula can be justified by considering the convergence of eigenvectors and eigenvalues, as the number of examples increases (Baker, 1977; Williams and Seeger, 2000; Shawe-Taylor and Williams, 2003). (Williams and Seeger, 2000) also noted that the Nystrom formula is equivalent to the kernel PCA projection (Schölkopf, Smola and Müller, 1998).

If we start from a data set D, obtain an embedding for its elements, and add more and more data, the embedding for the points in D converges (for eigenvalues that are unique). (Shawe-Taylor and Williams, 2003) give bounds on the convergence error. In the limit, each eigenvector converges to an eigenfunction for the linear operator defined below. It means that the i-th element of the k-th eigenvector converges to the application of the k-th eigenfunction to x_i.

Consider a Hilbert space $\mathcal{H}_p$ of functions with the following inner product:

$$\langle f, g \rangle_p = \int f(x) g(x) p(x) dx$$

with a density function $p(x)$. The kernel K can be associated with a linear operator K_p in $\mathcal{H}_p$:

$$(K_p f)(x) = \int K(x,y) f(y) p(y) dy. \tag{7}$$

We don't know the generating density p but we can approximate the above inner product and linear operator by those defined with the empirical distribution $\hat{p}$. An "empirical" Hilbert space $\mathcal{H}_{\hat{p}}$ is thus defined using the empirical distribution $\hat{p}$ instead of p. Note that the proposition below can be applied *even if the kernel is not positive semi-definite*, although the embedding algorithms we have studied are restricted to using the principal coordinates associated with positive eigenvalues.

Proposition 1

Let $\tilde{K}(a,b)$ be a kernel function, not necessarily positive semi-definite, with a discrete spectrum, that gives rise to a symmetric matrix $\tilde{M}$ with entries $\tilde{M}_{ij} = \tilde{K}(x_i, x_j)$ upon a dataset $D = \{x_1, \ldots, x_n\}$. Let (v_k, λ_k) be an (eigenvector,eigenvalue) pair that solves $\tilde{M} v_k = \lambda_k v_k$. Let (f_k, λ'_k) be an (eigenfunction,eigenvalue) pair that solves $\tilde{K}_{\hat{p}} f_k = \lambda'_k f_k$ with $\hat{p}$ the empirical distribution over D. Let $e_k(x) = y_k(x)\sqrt{\lambda_k}$ or $y_k(x)$ denote the embedding associated with a new point x. Then

$$\lambda'_k = \frac{1}{n}\lambda_k \tag{8}$$

$$f_k(x) = \frac{\sqrt{n}}{\lambda_k} \sum_{i=1}^{n} v_{ik} \tilde{K}(x, x_i) \tag{9}$$

$$f_k(x_i) = \sqrt{n} v_{ik} \tag{10}$$

$$y_k(x) = \frac{f_k(x)}{\sqrt{n}} = \frac{1}{\lambda_k} \sum_{i=1}^{n} v_{ik} \tilde{K}(x, x_i) \tag{11}$$

$$y_k(x_i) = y_{ik}, \qquad e_k(x_i) = e_{ik} \tag{12}$$

If $\tilde{K}(x,y) = \phi(x).\phi(y)$ and $\frac{1}{n}\sum_i \phi(x_i) = 0$ then for $\lambda_k > 0$, $e_k(x)$ is the kernel PCA projection with kernel $\tilde{K}$.

See (Bengio et al., 2003) for a proof and further justifications of the above formulae. The generalized embedding for Isomap and MDS is $e_k(x) = \sqrt{\lambda_k} y_k(x)$ whereas the one for spectral clustering, Laplacian eigenmaps and LLE is $y_k(x)$.

4 Extending to new Points

Using Proposition 1, one obtains a natural extension of all the unsupervised learning algorithms of section 2 that can be mapped to Algorithm 1, provided we can write down a continuous kernel function $\tilde{K}$ that gives rise to the matrix $\tilde{M}$ on D. We consider each of them in turn below.

Once $\tilde{K}$ is defined, it can be used in the equation (12) of Proposition 1 (optionally multiplied by $\sqrt{\lambda_k}$) in order to generalize the embedding to a new data point x.

4.1 Extending MDS

For MDS, a normalized kernel can be defined as follows, using a continuous version of the double-centering eq. (2):

$$\tilde{K}(a,b) = -\frac{1}{2}(d^2(a,b) - E_x[d^2(x,b)] - E_{x'}[d^2(a,x')] + E_{x,x'}[d^2(x,x')]) \quad (13)$$

where $d(a,b)$ is the original distance function and the expectations are taken over the empirical data D. An extension of metric MDS to new points has already been proposed in (Gower, 1968), in which one solves exactly for the coordinates of the new point that are consistent with its distances to the training points, which in general requires adding a new dimension. If $d^2(x_i,x_j) = ||x_i - x_j||^2$, the extension of MDS to a new point x using eq. 12 yields the projection of x on the principal components of D, since $\tilde{K}(x_i,x_j) = (x_i - E[x]).(x_j - E[x])$ (where expectation is over D).

4.2 Extending Spectral Clustering and Laplacian Eigenmaps

Both the version of Spectral Clustering and Laplacian Eigenmaps described above are based on an initial kernel K, such as the Gaussian or nearest-neighbor kernel. An equivalent continuous normalized kernel is:

$$\tilde{K}(a,b) = \frac{1}{n}\frac{K(a,b)}{\sqrt{E_x[K(a,x)]E_{x'}[K(b,x')]}}$$

where the expectations are taken over the empirical data D.

4.3 Extending Isomap

Isomap is a bit more challenging than the previous algorithms because we must be careful in how we define the geodesic distance. It must be done in such a way that the test point is not required in computing the geodesic distance between training points, otherwise we would have to recompute all the geodesic distances. A reasonable solution is to use the definition of $\tilde{D}(a,b)$ in eq. (5), which only uses the training points in the intermediate points on the path from a to b. We obtain a normalized kernel by applying the continuous double-centering from eq. (13) to $\tilde{D}^2$:

$$\tilde{K}(a,b) = -\frac{1}{2}(\tilde{D}^2(a,b) - E_x[\tilde{D}^2(x,b)] - E_{x'}[\tilde{D}^2(a,x')] + E_{x,x'}[\tilde{D}^2(x,x')]). \quad (14)$$

An m-dimensional manifold embedded in $\mathbf{R}^d$ is *isometric* if there is a bijective mapping c between points in m-dimensional Euclidean space and points on the manifold (with d coordinates) such that the geodesic distance on the manifold (in $\mathbf{R}^d$) equals the Euclidean distance of the corresponding points in $\mathbf{R}^m$.

A formula has already been proposed (de Silva and Tenenbaum, 2003) to approximate Isomap using only a subset of the examples (the "landmark" points) to compute the eigenvectors. Using the notation of this paper, that formula is

$$e_k(x) = \frac{1}{2\sqrt{\lambda_k}}\sum_i v_{ik}(E_{x'}[\tilde{D}^2(x',x_i)] - \tilde{D}^2(x_i,x)). \quad (15)$$

where $E_{x'}$ is an average over the data set. The formula is applied to obtain an embedding for the non-landmark examples.

Corollary 2

The embedding proposed in Proposition 1 for Isomap ($e_k(x)$) is equal to formula 15 (Landmark Isomap) when $\tilde{K}(x, y)$ is defined as in eq. 14.

Proof: the proof relies on a property of the Gram matrix for Isomap (and MDS), i.e. that $\sum_i M_{ij} = 0$, by construction. Therefore $(1, 1, \ldots 1)$ is an eigenvector with eigenvalue 0, and all the other eigenvectors v_k have the property $\sum_i v_{ik} = 0$ because of orthogonality with $(1, 1, \ldots 1)$. Writing $(E_{x'}[\tilde{D}^2(x', x_i)] - \tilde{D}^2(x, x_i)) = \tilde{K}(x, x_i) + \frac{1}{2}(E_{x',x''}[\tilde{D}^2(x', x'')] - E_{x'}[\tilde{D}^2(x, x')])$ yields for eq. 15 $\frac{1}{2\sqrt{\lambda_k}}((\sum_i v_{ik}\tilde{K}(x, xi)) + \frac{1}{2}(E_{x',x''}[\tilde{D}^2(x', x'')] - E_{x'}[\tilde{D}^2(x, x')]) \sum_i v_{ik}$ where the last sum is 0, which makes eq. 15 equal to $e_k(x)$.

Corollary 3

If the data D and the test point x come from a convex region on an m-dimensional isometric manifold then, as the number of examples $n \to \infty$, the extension of Isomap to a new point x using eq. (12) (times $\sqrt{\lambda_k}$) yields the projection of x on the principal components of the corresponding low-dimensional data points.

Sketch of the proof: using the main theorem in (de Silva and Tenenbaum, 2003), the geodesic distances estimated by $\tilde{D}$ converge to the geodesic distances on the underlying manifold. In that case $\tilde{K}$ converges to a dot product of centered data, and applying Proposition 1 we obtain the result.

4.4 Extending LLE

The extension of LLE is the most challenging one because it does not fit as well the framework of Algorithm 1: the M matrix for LLE does not have a clear interpretation in terms of distance or dot product, and adding a new point would seem to require re-computing all the weights.

To directly associate a kernel function to the mapping M in eq. (6) would yield a kernel $K(x, y)$ with a singularity at $x = y$ (to account for the identity matrix). Instead we consider the matrix $\tilde{M} = I - M$, which has the same eigenvectors (therefore the same embedding is obtained) but eigenvalues $1 - \lambda$ instead of λ (so we care about the largest eigenvalues, not the smallest ones). To obtain a kernel that generates $\tilde{M}$ we must first associate a function $w(a, b)$ to the matrix W of regression weights such that we obtain $W_{ij} = w(x_i, x_j)$. We first define a local $k \times k$ Gram matrix around an arbitrary point x:

$$C(x)_{ij} = (x - x_{n(i)})(x - x_{n(j)})' 1_{x_{n(i)} \in \mathcal{N}(x)} 1_{x_{n(j)} \in \mathcal{N}(x)}$$

where $\mathcal{N}(x)$ is the subset of k elements from D that are the k nearest neighbors of x and $n(i)$ is the index of the i-th such neighbor of x. Then we define $w(a, b)$ as follows:

$$w(a, b) = 1_{b = x_{n(j)} \in \mathcal{N}(a)} \frac{\sum_q C^{-1}(x)_{jq}}{\sum_{pq} C^{-1}(x)_{pq}}.$$

Note that the above definition makes $\sum_i w(a, x_i) = 1$ and $\sum_i (w(a, b)x_i - a)^2$ minimized, as required. Note also that we are using the training points to predict the test

point x but not vice-versa. Note that for training points we recover $W_{ij} = w(x_i, x_j)$. We thus obtained the equivalent kernel

$$\tilde{K}(a,b) = w(a,b) + w(b,a) - \sum_i w(x_i,a)w(x_i,b).$$

When neither a nor b are in D, $\tilde{K}(a,b) = 0$. When both are in D we obtain $\tilde{K}(x_i,x_j) = \tilde{M}_{ij}$. When only one is in D, we obtain respectively $\tilde{K}(x,x_i) = w(x,x_i)$ or $\tilde{K}(x_i,x) = w(x,x_i)$.

5 Experiments

We want to evaluate whether the precision of the generalizations suggested in the previous section is comparable to the intrinsic perturbations of the embedding algorithms. The perturbation analysis will be achieved by considering splits of the data in three sets, $D = F \cup R_1 \cup R_2$ and training either with $F \cup R_1$ or $F \cup R_2$, comparing the embeddings on F. For each algorithm described in section 2, we apply the following procedure:

1. We choose $F \subset D$ with $m = |F|$ samples. The remaining $n - m$ samples in D/F are split into two equal size subsets R_1 and R_2. We train (obtain the eigenvectors) over $F \cup R_1$ and $F \cup R_2$. When eigenvalues are close, the estimated eigenvectors are unstable and can rotate in the subspace they span. Thus we estimate an affine alignment between the two embeddings using the points in F, and we calculate the Euclidean distance between the aligned embeddings obtained for each $s_i \in F$.

2. For each sample $s_i \in F$, we also train over $\{F \cup R_1\}/\{s_i\}$. We apply the extension to out-of-sample points to find the predicted embedding of s_i and calculate the Euclidean distance between this embedding and the one obtained when training with $F \cup R_1$, i.e. with s_i in the training set.

3. We calculate the mean difference (and its standard error, shown in the figure) between the distance obtained in step 1 and the one obtained in step 2 for each sample $s_i \in F$, and we repeat this experiment for various sizes of F.

The results obtained for MDS, Isomap, spectral clustering and LLE are shown in figure 1 for different values of m. Experiments are done over a database of 698 synthetic face images described by 4096 component that is available at `http://isomap.stanford.edu`. Qualitatively similar results have been obtained over other databases such as Ionosphere (`http://www.ics.uci.edu/~mlearn/MLSummary.html`) and swiss-sroll (`http://www.cs.toronto.edu/~roweis/lle/`). Each algorithm generates a two-dimensional embedding of the images, following the experiments reported for Isomap. The number of neighbors is 10 for Isomap and LLE, and a Gaussian kernel with a standard deviation of 0.01 is used for spectral clustering / Laplacian eigenmaps. 95% confidence intervals are drawn beside each mean difference of error on the figure.

As can be expected, the mean difference between the two distances is almost monotonically decreasing as the number m of overlapping samples grows, mostly because the training set embedding variability decreases as the number of points exchanged ($n - m$) decreases. Furthermore, we find in most cases that the out-of-sample error is less than or comparable to the training set embedding stability. In fact, the out-of-sample error is always less than the variability induced on the training set embedding when more than a few training points are exchanged.

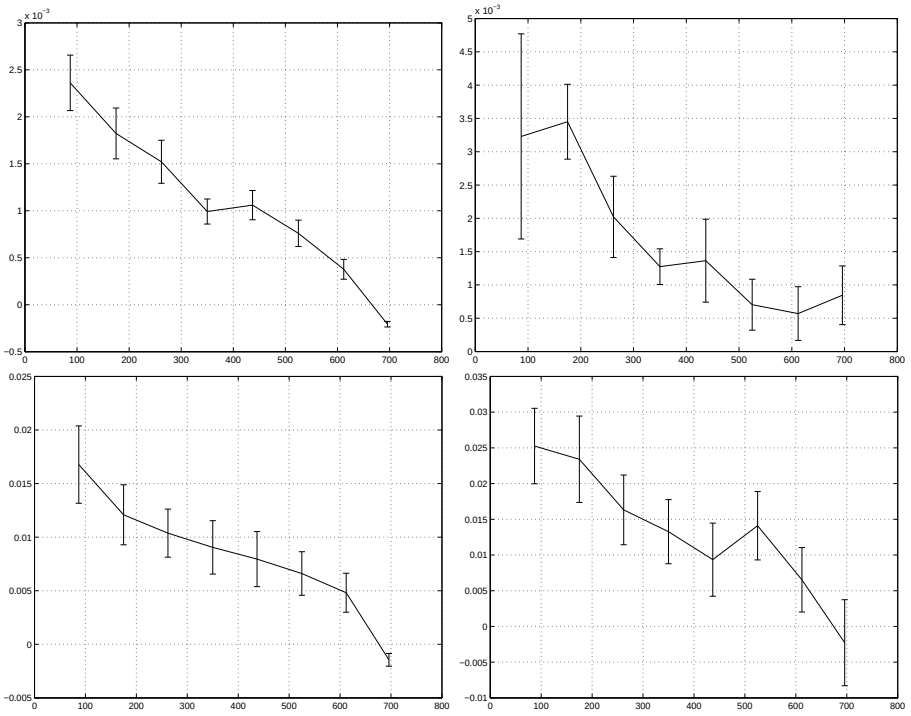

Figure 1: Training set variability minus out-of-sample error, wrt number of training set overlapping points. Top left: MDS. Top right: spectral clustering or Laplacian eigenmaps. Bottom left: Isomap. Bottom right: LLE. Error bars are 95% confidence intervals.

6 Conclusions

In this paper we have presented an extension to four unsupervised learning algorithms based on a spectral embedding of the data: MDS, spectral clustering, Laplacian eigenmaps, Isomap and LLE. This extension allows one to apply a trained model to out-of-sample points without having to recompute eigenvectors. It introduces a practical notion of generalization for these algorithms as well as a new method to measure it. The

experiments on real high-dimensional data show that the average distance between the out-of-sample and in-sample embeddings is comparable or lower than the variation in in-sample embedding due to replacing a few points in the training set.

References

Baker, C. (1977). *The numerical treatment of integral equations*. Clarendon Press, Oxford. 4

Belkin, M. and Niyogi, P. (2003). Laplacian eigenmaps for dimensionality reduction and data representation. *Neural Computation*, 15(6):1373–1396. 1, 3

Bengio, Y., Vincent, P., Paiement, J., Delalleau, O., Ouimet, M., and Roux, N. L. (2003). Spectral clustering and kernel pca are learning eigenfunctions. Technical report, Département d'informatique et recherche opérationnelle, Université de Montréal. 5

Cox, T. and Cox, M. (1994). *Multidimensional Scaling*. Chapman & Hall, London. 2

de Silva, V. and Tenenbaum, J. (2003). Local versus global methods for nonlinear dimensionality reduction. In Becker, S., Thrun, S., and Obermayer, K., editors, *Advances in Neural Information Processing Systems*, volume 15. The MIT Press. 6, 7

Gower, J. (1968). Adding a point to vector diagrams in multivariate analysis. *Biometrika*, 55(3):582–585. 6

Ng, A. Y., Jordan, M. I., and Weiss, Y. (2002). On spectral clustering: Analysis and an algorithm. In Dietterich, T. G., Becker, S., and Ghahramani, Z., editors, *Advances in Neural Information Processing Systems 14*, Cambridge, MA. MIT Press. 1, 3

Roweis, S. and Saul, L. (2000). Nonlinear dimensionality reduction by locally linear embedding. *Science*, 290(5500):2323–2326. 1, 4

Schölkopf, B., Smola, A., and Müller, K.-R. (1998). Nonlinear component analysis as a kernel eigenvalue problem. *Neural Computation*, 10:1299–1319. 4

Shawe-Taylor, J. and Williams, C. (2003). The stability of kernel principal components analysis and its relation to the process eigenspectrum. In Becker, S., Thrun, S., and Obermayer, K., editors, *Advances in Neural Information Processing Systems*, volume 15. The MIT Press. 4

Shi, J. and Malik, J. (1997). Normalized cuts and image segmentation. In *Proc. IEEE Conf. Computer Vision and Pattern Recognition*, pages 731–737.

Tenenbaum, J., de Silva, V., and Langford, J. (2000). A global geometric framework for nonlinear dimensionality reduction. *Science*, 290(5500):2319–2323. 1, 3

Weiss, Y. (1999). Segmentation using eigenvectors: a unifying view. In *Proceedings IEEE International Conference on Computer Vision*, pages 975–982. 1, 3

Williams, C. and Seeger, M. (2000). The effect of the input density distribution on kernel-based classifiers. In *Proceedings of the Seventeenth International Conference on Machine Learning*. Morgan Kaufmann. 4