Non-Linear Dimensionality Reduction Techniques for Classification and Visualization

Michail Vlachos
UC Riverside
mvlachos@cs.ucr.edu

Carlotta Domeniconi
UC Riverside
carlotta@cs.ucr.edu

Dimitrios Gunopulos *
UC Riverside
dg@cs.ucr.edu

George Kollios †
Boston University
gkollios@cs.bu.edu

Nick Koudas
AT&T Labs Research
koudas@research.att.com

ABSTRACT

In this paper we address the issue of using local embeddings for data visualization in two and three dimensions, and for classification. We advocate their use on the basis that they provide an efficient mapping procedure from the original dimension of the data, to a lower intrinsic dimension. We depict how they can accurately capture the user's perception of similarity in high-dimensional data for visualization purposes. Moreover, we exploit the low-dimensional mapping provided by these embeddings, to develop new classification techniques, and we show experimentally that the classification accuracy is comparable (albeit using fewer dimensions) to a number of other classification procedures.

1. INTRODUCTION

During the last few years we have experienced an explosive growth in the amount of data that is being collected, leading to the creation of very large databases, such as commercial data warehouses. New applications have emerged that require the storage and retrieval of massive amounts of data; for example: protein matching in biomedical applications, fingerprint recognition, meteorological predictions, and satellite image repositories.

Most problems of interest in data mining involve data with a large number of measurements (or dimensions). The reduction of dimensionality can lead to an increased capability of extracting knowledge from the data by means of visualization, and to new possibilities in designing efficient and possibly more effective classification schemes. Dimensionality reduction can be performed by keeping only the most important dimensions, i.e. the ones that hold the most information for the task at hand, and/or by projecting some dimensions onto others. These steps will improve significantly our ability to visualize the data (by mapping them in two or three dimensions), and facilitate an improved query time, by refraining from examining the original multi-dimensional data and scanning instead their lower-dimensional "summaries".

For visualization, the challenge is to embed a set of observations into a Euclidean feature-space, that preserves as closely as possible their intrinsic metric structure. For classification, we desire to map the data into a space whose dimensions clearly separate members from different classes.

Recently, two new dimensionality reduction techniques have been introduced, namely Isomap [15] and LLE [26]. These methods attempt to best preserve the local neighborhood of each object, while preserving the global distances "through" the rest of the objects. They have been used for visualization purposes, by mapping data into two or three dimensions. Both methods perform well when the data belong to a single well sampled cluster, and fail to nicely visualize the data when the points are spread among multiple clusters. In this paper we propose a mechanism to avoid this limitation.

Furthermore, we show how these methods could be used for classification purposes. Classification is a key step for many tasks in data mining, whose aim is to discover unknown relationships and/or patterns from large set of data. A variety of methods has been proposed to address the problem. A simple and appealing approach to classification is the K-nearest neighbor method [21]: it finds the K-nearest neighbors of the query point $\mathbf{x}_0$ in the dataset, and then predicts the class label of $\mathbf{x}_0$ as the most frequent one occuring in the K neighbors. However, when applied on large datasets in high dimensions, the time required to compute the neighborhoods (i.e., the distances of the query from the points in the dataset) becomes prohibitive, making answers intractable. Moreover, the curse-of-dimensionality, that affects any problem in high dimensions, causes highly biased estimates, thereby reducing the accuracy of predictions.

One way to tackle the curse-of-dimensionality-problem for classification is to consider locally adaptive metric techniques, with the objective of producing modified local neighborhoods in which the posterior probabilities are approximately constant ([10, 11, 7]). A major drawback of locally adaptive metric techniques for nearest neighbor classification is the fact that they all perform the K-NN procedure multi-

*Supported by NSF CAREER Award 9984729, & NSF IIS-9907477, and the DoD.
†Supported by NSF CAREER Award 0133825

Permission to make digital or hard copies of all or part of this work for personal or classroom use is granted without fee provided that copies are not made or distributed for profit or commercial advantage and that copies bear this notice and the full citation on the first page. To copy otherwise, to republish, to post on servers or to redistribute to lists, requires prior specific permission and/or a fee.
SIGKDD '02 Edmonton, Alberta, Canada
Copyright 2002 ACM 1-58113-567-X/02/0007 ...$5.00.

ple times in a feature space that is tranformed by means of weightings, but has the same number of dimensions as the original one. Thus, in high dimensional spaces these techniques become very costly.

Here, we propose to overcome this limitation by applying K-NN classification in the reduced space provided by locally linear dimensionality reduction techniques such as Isomap and LLE. In the reduced space, we can construct and use efficient index structures (such as [2]), thereby improving the performance of the K-NN technique. However, in order to use this approach, we need to compute an explicit mapping function of the query point from the original space to the reduced dimensionality space.

1.1 Our Contribution

Our contributions can be summarized as follows:

• We analyze the LLE and Isomap visualization power through an experiment, and show that they perform well only when the data are comprised of one, well sampled, cluster. The mapping gets significantly worse when the data are organized in multiple clusters. We propose to overcome this limitation by modifying the mapping procedure, and keeping distances to both closest and farthest objects. We demonstrate the enhanced visualization results.

• To tackle with the curse-of-dimensionality problem for classification we combine the Isomap procedure with locally adaptive metric techniques for nearest neighbor classification. In particular, we introduce two new techniques, WeightedIso and Iso+Ada. By modifying the transformation performed by the Isomap technique to take into consideration the labelling of the data, we can produce homogeneous neighborhoods in the reduced space, where better classification accuracy can be achieved.

• Through extensive experiments using real data sets we demonstrate the efficacy of our methods, against a number of other classification techniques. The experimental findings corroborate the following conclusions:

1. WeightedIso and Iso+Ada achieve performance results competitive to other classification techniques but in significantly lower dimensional space;
2. WeightedIso and Iso+Ada allow to considerably reduce the dimensionality of the original feature space, thereby allowing the application of indexing data structures to perform efficient nearest neighbor search [2].

2. RELATED WORK

Numerous approaches have been proposed for dimensionality reduction. The main idea behind all of them is to keep a lossy representation of the initial dataset, which nonetheless retains as much of the original structure as possible.

We could distinguish two general categories:

1. *Local or Shape preserving*
2. *Global or Topology preserving*

In the first category we could place methods that do not try to exploit the global properties of the dataset, but rather attempt to 'simplify' the representation of each object regardless of the rest of the dataset. If we are referring to time-series, the selection of the k-features should be such that the selected features retain most of the information ("energy") of the original signal. For example, these features could be either the first coefficients of the Fourier decomposition ([1, 9]), or the wavelet decomposition ([5]), or even some piecewise constant approximation of the sequence ([17]).

The second category of methods has mostly been used for visualization purposes, with the objective of discovering a parsimonious spatial representation for the dataset. The most widely used methods are Principal Component Analysis (PCA) [16], Multidimensional Scaling (MDS), and Singular Value Decomposition (SVD). MDS focuses on the preservation of the original high-dimensional distances, for a 2-dimensional representation of objects. The only assumption made by MDS is the existence of a monotonic relationship between the original and projected pairwise distances. Finally, SVD can be used for dimensionality reduction by finding the projection that restores the largest possible original variance, and ignoring those axes of projection which contribute the least to the total variance.

Other methods that enhance the user's visualization abilities have been proposed in [19, 8, 4, 14].

Lately, another category of dimensionality reduction techniques has appeared, namely Isomap [15] and LLE [26]. In this paper we will refer to such category of techniques as *Local Embeddings* (LE). These methods attempt to preserve as well as possible the local neighborhood of each object, while preserving the global distances "through" the rest of the objects (by means of a minimum spanning tree).

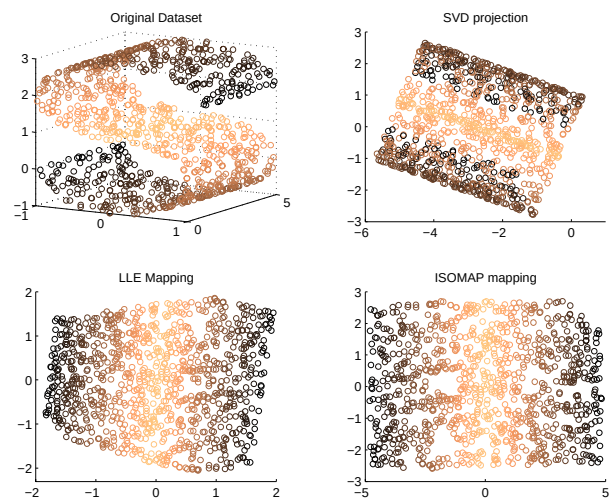

Figure 1: Mapping in 2-dimensions of the SCURVE dataset using SVD, LLE and ISOMAP.

3. LOCAL EMBEDDINGS

Most of the dimensionality reduction techniques fail to capture the neighborhood of data, when points lie on a manifold (manifolds are fundamental to human perception [18]). Local Embeddings attempt to tackle this problem.

Isomap is a procedure that maps high-dimensional objects into a lower dimensional space (usually 2-3 for visualization

purposes), while preserving as well as possible the neighborhood of each object, as well as the 'geodesic' distances between all pairs of objects. Isomap works as follows:

1. *Calculate the K closest neighbors of each object*
2. *Create the Minimum Spanning Tree (MST) distances of the updated distance matrix*
3. *Run MDS on the new distance matrix.*
4. *Depict points on some lower dimension.*

Locally Linear Embedding (LLE) also attempts to reconstruct as close as possible the neighborhood of each object, from some high dimension (q) into a lower dimension. However, while ISOMAP tries to minimize the least square error of the geodesic distances, LLE aims at minimizing the least squares error, in the low dimension, of the neighbors' weights for every object.

We depict the potential power of the above methods with an example. Suppose that we have data that lie on a manifold in three dimensions (figure 1). For visualizations purposes we would like to identify the fact that the data could be placed on a 2D plane, by 'unfolding' or 'stretching' the manifold. Locally linear methods provide us with this ability. However, by using some global method, such as SVD, the results are non-intuitive, and neighboring points get projected on top of each other (figure 1).

4. DATASET VISUALIZATION USING ISOMAP AND LLE

Both LLE and ISOMAP present a meaningful mapping in a lower dimension when the data are comprised of one, well sampled, cluster. When our dataset consists of many well separated clusters, the mapping provided is significantly worse. We depict this with an example. We have constructed a dataset consisting of 6 clusters of equal size in 5 dimensions (GAUSSIAN5D). The dataset if constructed as follows: The center of the clusters are the points $(0,0,0,0,0)$, $(10,0,0,0,0)$, $(0,10,0,0,0)$, $(0,0,10,0,0)$, $(0,0,0,10,0)$, $(0,0,0,0,10)$. The data follow a Gaussian distribution with covariance $\sigma_{i,j} = 0$ for $i \neq j$ and 1 otherwise. In figure 2 we can observe the mapping provided by both methods. All the points of each cluster are projected on top of each other which impedes significantly any visualization purposes. This has also been mentioned in [24]; however the authors only tackle with the problem of recognizing the number of disjoint groups and not how to visualize them effectively.

In addition, we observe that the quality of the mapping changes only marginally, if we sample the dataset and then map the remaining points based on the already mapped portion of the dataset. This is depicted in figure 3. Specifically, using the SCURVE dataset, we map a portion of the original dataset. The rest of the objects are mapped according to the projected sample, so as the distance of the K nearest neighbors is preserved as well as possible in the lower dimensional space. We calculate the residual error of the original pairwise distances and the final ones. The residual error is very small, which indicates that in the case of a *dynamic database*, we don't have to repeat the mapping of *all* the points again. Of course, this holds under the assumption that the sample is representative of the whole database.

The observed *"overclustering"* effect can be mitigated if instead of keeping only the k closest neighbors, we try to

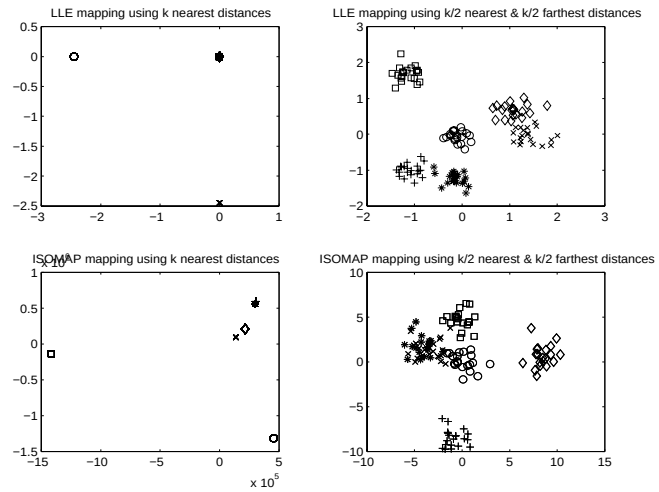

Figure 2: Left:Mapping in 2-dimensions of LLE and ISOMAP using the GAUSSIAN5D dataset. Right: Using our modified mapping the clusters are clearly separated.

reconstruct the distances to the $\frac{k}{2}$ closest objects, as well as to the $\frac{k}{2}$ farthest objects. This is likely to provide us with enhanced visualization results, since not only is it going to preserve the local neighborhood, but also it will retain some of the original global information. This is important and quite different from global methods, where each object's individual emphasis is lost in the average, or in the effort of some global optimization criterion. In figure 2 we can observe that the new mapping clearly separated the clusters of the $GAUSSIAN5D$ dataset.

Therefore, for visualizing large, clustered, dynamic datasets we propose the following technique:

1. Map the current dataset using the $k/2$ closest objects and the $k/2$ farthest objects. This will separate clearly the clusters.

2. For any new points that are added in the database, we don't have to perform the mapping again. The position of every new point in the new space is found by preserving, as well as possible, the original distances of its $\frac{k}{2}$ closest and $\frac{k}{2}$ farthest objects in the new space (using Least-Square fitting). As suggested by the previous experiment the new incremental mapping will be adequately accurate.

5. CLASSIFICATION

In a classification problem, we are given J classes and N training observations. The training observations consist of q feature measurements $\mathbf{x} = (x_1, \cdots, x_q) \in \Re^q$ and the known class labels, y, $y = 1, \ldots, J$. The goal is to predict the class label of a given query $\mathbf{x}_0$. It is assumed that there exists an unknown probability distribution $P(\mathbf{x}, y)$ from which data are drawn. To predict the class label of a given query $\mathbf{x}_0$, we need to estimate the class posterior probabilities $\{P(j|\mathbf{x})_0\}_{j=1}^{J}$.

The K nearest neighbor classification method [13, 20] is a simple and appealing approach to this problem: it finds the

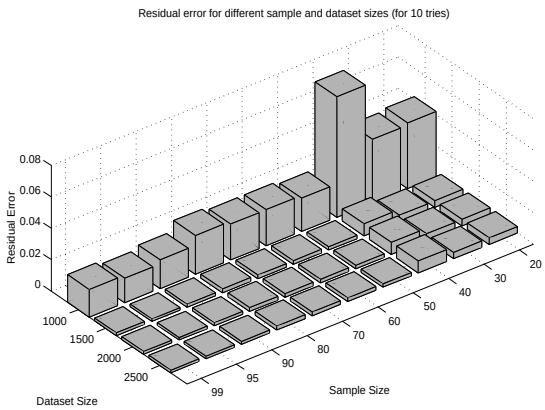

Figure 3: Residual Error when mapping a sample of the dataset; the remaining portion is mapped according to the projected sample.

K nearest neighbors of $\mathbf{x}_0$ in the training set, and then predicts the class label of $\mathbf{x}_0$ as the most frequent one occurring in the K neighbors. K nearest neighbor methods are based on the assumption of smoothness of the target functions, which translates to locally constant class posterior probabilities. It has been shown in [6] that the one nearest neighbor rule has asymptotic error rate that is at most twice the Bayes error rate, independent of the distance metric used.

However, severe bias can be introduced in the nearest neighbor rule in a high dimensional input feature space with finite samples ([3]) . The assumption of smoothness becomes invalid for any fixed distance metric when the input observation approaches class boundaries. One way to tackle this problem is to develop locally adaptive metric techniques, with the objective of producing modified local neighborhoods in which the posterior probabilities are approximately constant. The common idea in these techniques ([10, 11, 7]) is that the weight assigned to a feature, locally at a given query point $\mathbf{q}$, reflects its estimated relevance to predict the class label of $\mathbf{q}$: larger weights correspond to larger capabilities in predicting class posterior probabilities.

A major drawback of locally adaptive metric techniques for nearest neighbor classification is the fact that they all perform the K-NN procedure multiple times in a feature space that is tranformed by means of weightings, but has the same number of dimensions as the original one. In high dimensional spaces, then, these techniques become very costly. Here, we propose to overcome this limitation by applying the K-NN classification in the lower dimensional space provided by Isomap, where we can construct efficient index structures.

In contrast to global dimensionality reduction techniques like SVD, the Isomap procedure has the objective of reducing the dimensionality of the input space while preserving the local structure of the dataset as much as possible. This feature makes Isomap particularly suited for being combined with nearest neighbor techniques, that rely on the queries' local neighborhoods to address the classification problem.

6. OUR APPROACH

The mapping performed by Isomap, combined with the label information provided by the training data, can help us reduce the curse-of-dimensionality effect. We take into consideration the non isotropic characteristics of the input feature space at different locations, thereby achieving more accurate estimations. Moreover, since we will perform nearest neighbor classification in the reduced space, this process will result in a boosted efficiency.

When computing the distance between two points for classification, we desire to consider the two points close to each other if they belong to the same class, and far from each other otherwise. Therefore, we aim to compute a transformation that maps similar observations, in terms of class posterior probabilities, to nearby points in feature space, and observations that show large differences in class posterior probabilities to distant points in feature space. We derive such a transformation by modifying step 1 of the Isomap procedure to take into consideration the labelling of points.

We proceed as follows. We first compute the K nearest neighbors of each data point $\mathbf{x}$ (we set $K = 10$ in our experiments). Let us denote with K_{same} the set of nearest neighbors having the same class label as $\mathbf{x}$. We then "move" each nearest neighbor in K_{same} closer to $\mathbf{x}$ by rescaling their Euclidean distance by a constant factor (set to 1/10 in our experiments). This mapping construction is summarized in Figure 5.

In constrast to visualization tasks, where we wish to preserve the intrinsic metric structure for neighbors as much as possible, here we wish to stretch or constrict such metric in order to derive homogeneous neighborhoods in the transformed space. Our mapping construction aims to achieve this goal. Once we have derived the map into d dimensions, we apply K-NN classification in the reduced feature space to classify a given query $\mathbf{x}_0$. We first need to derive the query's coordinates in d dimensions. To achieve this goal, we learn an explicit mapping $f : \Re^q \rightarrow \Re^d$ using the smooth interpolation technique provided by radial basis function (RBF) networks [12, 23], applied to the known corresponding pairs obtained as output in Figure 5.

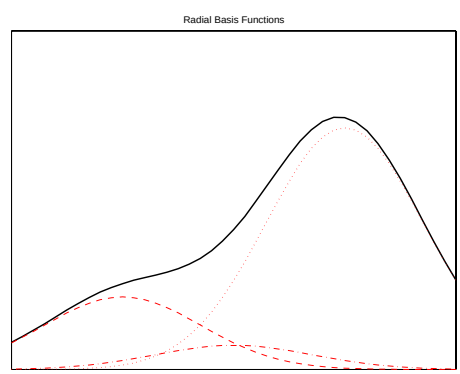

Figure 4: Linear combination of three Gaussian Basis Functions.

An RBF neural network solves a curve-fitting approximation problem in a high-dimensional space. It involves three different layers of nodes. The input layer is made up of source nodes. The second layer is a hidden layer of high enough dimension. The ouput layer supplies the response of the network to the activation patterns applied to the input layer. The transformation from the input space to the hidden-unit space is nonlinear, whereas the transformation

from the hidden-unit space to the output space is linear. Through careful design, it is possible to reduce the dimension of the hidden-unit space, by making the centers and spread of the hidden units adaptive. Figure 4 shows the effect of combining three Gaussian Basis Functions with different centers and spread.

The training phase constitutes the optimization of a fitting procedure to construct the surface f, based on known data points presented to the network in the form of input-output examples. Specifically, we train an RBF network with q input nodes, d output nodes, and nonlinear hidden units shaped as Gaussians. In our experiments, to avoid overfitting, we adapt the centers and spread of the hidden units via cross-validation, and making use of the known corresponding N pairs $\{(\mathbf{x}, \mathbf{x}_d)\}_1^N$. The RBF network construction process is summarized in Figure 6. Figure 7 describes the classification step, that involves mapping the input query $\mathbf{x}_0$ using the RBF network, and then applying the K-NN procedure in the reduced d dimensional space. We call the whole procedure WeightedIso. To summarize, WeightedIso performs three steps as follows:

1. **Mapping Contruction** (Figure 5);
2. **Network Contruction** (Figure 6);
3. **Classification** (Figure 7).

In our experiments we also explore an alternative procedure, with the same objective of reducing the computational cost of applying locally adaptive metric techniques in high dimensional spaces. We call this method Iso+Ada. It combines the Isomap technique with the adaptive metric nearest neighbor technique (ADAMENN) introduced in [7]. Iso+Ada first performs the Isomap procedure (unchanged this time) on the training data, and then applies the ADAMENN technique in the reduced feature space to classify a query point. As for WeightedIso, the coordinates of the query in the d dimensional feature space are computed via an RBF network.

Mapping Construction:

- **Input**: Training data $T = \{(\mathbf{x}, y)\}_1^N$
- Execute on the training data the Isomap procedure modified as follows:
  - Calculate the K closest neighbors $\mathbf{x}_k$ of each $\mathbf{x}$ in T;
  - Let K_{same} be the set of nearest neighbors that have the same class label as $\mathbf{x}$;
    - For each $\mathbf{x}_k \in K_{same}$: scale the distance $dis(\mathbf{x}_k, \mathbf{x})$ by a factor of $1/\alpha$, $\alpha > 1$
  - Use the defined distances to create the Minimum Spanning Tree.
- **Output**: Set of N pairs $\{(\mathbf{x}, \mathbf{x}_d)\}_1^N$, where $\mathbf{x}_d$ corresponds to $\mathbf{x}$ mapped into d dimensions.

Figure 5: The Mapping contruction phase of the WeightedIso algorithm

RBF Network Construction:

- **Input**: Training data $\{(\mathbf{x}, \mathbf{x}_d)\}_1^N$
  1. Train an RBF network NET with q input nodes and d output nodes, using the input training pairs.
- **Output**: RBF network NET.

Figure 6: The RBF network contruction phase of the WeightedIso algorithm

Classification:

- **Input**: RBF network NET, $\{\mathbf{x}_d, y\}_1^N$, query $\mathbf{x}_0$
  1. Use NET to map $\mathbf{x}_0$ into the d dimensional space;
  2. Use the points $\{\mathbf{x}_d, y\}_1^N$ to apply the K-NN rule in the d dimensional space, and classify $\mathbf{x}_0$
- **Output**: Classification label for $\mathbf{x}_0$.

Figure 7: The Classification phase of the WeightedIso algorithm

7. EXPERIMENTS

We compare several classification methods using real data:
• **ADAMENN**-adaptive metric nearest neighbor technique (one iteration) [7]. It uses the *Chi-squared* distance in order to estimate to which extent each dimension can be relied on to predict class posterior probabilities. The estimation process is carried on over a local region of the query. Features are weighted accordingly to their estimated local relevance.
• **i-ADAMENN** - ADAMENN with five iterations;
• **K-NN** method using the Euclidean distance measure;
• **C4.5** decision tree method [25];
• **Machete** [10]. It is an adaptive NN procedure that combines recursive partitioning with the K-NN technique. Machete recursively homes in to the query point by splitting the space at each step along the most relevant feature. Relevance of each feature is measured in terms of the information gain provided by knowing the measurement along that dimension.
• **Scythe** [10]. It is a generalization of the Machete algorithm, in which the input variables influence each split in proportion to their estimated local relevance, rather than the winner-take-all strategy of Machete;
• **DANN** - Discriminant Adaptive Nearest Neighbor Technique. It is a discriminant adaptive nearest neighbor classification technique [11]. It employes a metric that locally behaves as a local linear discriminant metric: larger weights are credited to features that well separates the mean clusters, relative to the within class spread.
• **i-DANN** - DANN with five iterations [11].

Procedural parameters for each method were determined empirically through cross-validation. The data sets used were taken from the UCI Machine Learning Database Repository [22]. They are: Iris, Sonar, Glass, Liver, Lung, Image,

and Vowel. Cardinalities, dimensions, and number of classes for each data set are summarized in Table 1.

Table 1: The datasets used in our experiments

Dataset	♯ data	♯ dims	♯ classes	experiment
Iris	100	4	2	leave 1 out c-v
Sonar	208	60	2	leave 1 out c-v
Glass	214	9	6	leave 1 out c-v
Liver	345	6	2	leave 1 out c-v
Lung	32	56	3	leave 1 out c-v
Image	640	16	15	ten 2fold c-v
Vowel	528	10	11	ten 2fold c-v

8. RESULTS

Tables 2 and 3 show the (cross-validated) error rates for the ten methods under consideration on the seven real data sets. The average error rates for the smaller data sets (i.e., Iris, Sonar, Glass, Liver, and Lung) were based on leave-one-out cross-validation, and the error rates for Image and Vowel were based on ten two-fold-cross-validation, as summarized in Table 1.

In Figure 9 we plot the error rates obtained for the WeightedIso method for different values of reduced dimensionality d (up to 15), and for each data set. We can observe an "elbow" shaped curve for each data set, where the largest improvements in error rates are found when d increases from two to three and four. This means that, through our mapping transformation, we are able to achieve a good discrimination level between classes in low dimensional spaces. As a consequence, it becomes feasible to construct indexing structures that allow a fast nearest neighbor search in the reduced feature space. In Tables 2 and 3, we report the lowest error rate obtained with the WeightedIso technique for each data set. We use the d value that gives the lowest error rate for each data set to run the Iso+Ada technique, and report the corresponding error rates in Tables 2 and 3. We apply the remaining eight techniques in the original q-dimensional feature space.

Different methods give the best performance on different data sets. Iso+Ada gives the best performance on three data sets (Iris, Image, and Lung), and is close to the best performer in the remaining four data sets. A large gain in performance is achieved by both Iso+Ada and WeightedIso for the lung data. The data for this problem are extremely sparse in the original feature space (only 32 points with 56 dimensions). Both the WeightedIso and Iso+Ada techniques reach an error rate of 34.4% in a two-dimensional space.

It is natural to ask the question of robustness. That is, how well a particular method m performs on average in situations that are most favorable to other procedures. We capture robustness by computing the ratio b_m of its error rate e_m and the smallest error rate over all methods being compared in a particular example:

$$b_m = e_m / \min_{1 \le k \le 10} e_k .$$

Thus, the best method m^* for that example has $b_{m^*} = 1$, and all other methods have larger values $b_m \geq 1$, for $m \neq m^*$. The larger the value of b_m, the worse the performance of the mth method is in relation to the best one for that example, among the methods being compared. The distribution of the b_m values for each method m over all the examples, therefore, seems to be a good indicator concerning its robustness. For example, if a particular method has an error rate close to the best in every problem, its b_m values should be densely distributed around the value 1. Any method whose b value distribution deviates from this ideal distribution reflects its lack of robustness.

Figure 8 plots the distribution of b_m for each method over the seven simulated data sets. For each method we stack the seven b_m values. We can observe that the ADAMENN technique is the most robust technique among the methods applied in the original q-dimensional feature space, and Iso+Ada is capable of achieving the same performance. The b values for both methods, in fact, are always very close to 1 (the sum of the values being slightly less for Iso+Ada). Therefore Iso+Ada shows a very robust behavior, achieved in feature spaces much smaller than the original one, upon which ADAMENN has operated. The WeightedIso technique also shows a robust behavior, still competitive with the adaptive techniques that operates in the original feature space. C4.5 is the worst performer. Its poor performance is likely due to estimates with large bias and variance, due to the greedy strategy it employes, and to the partitioning of the input space in disjoint regions.

Table 2: Average classification error rates.

	Iris	Sonar	Glass	Liver	Lung
WeightedIso	4	13.5	30.4	37.1	**34.4**
Iso+Ada	**2.0**	12.0	27.5	34.8	**34.4**
ADAMENN	3.0	9.1	**24.8**	30.7	40.6
i-ADAMENN	5.0	9.6	**24.8**	30.4	40.6
K-NN	6.0	12.5	28.0	32.5	50.0
C4.5	8.0	23.1	31.8	38.3	59.4
Machete	5.0	21.2	28.0	**27.5**	50.0
Scythe	4.0	16.3	27.1	**27.5**	50.0
DANN	6.0	**7.7**	27.1	30.1	46.9
i-DANN	6.0	9.1	26.6	27.8	40.6

Table 3: Average classification error rates.

	Vowel	Image
WeightedIso	17.5	6.7
Iso+Ada	11.4	**4.3**
ADAMENN	**10.7**	5.2
i-ADAMENN	10.9	5.2
K-NN	11.8	6.1
C4.5	36.7	21.6
Machete	20.2	12.3
Scythe	15.5	5.0
DANN	12.5	12.9
i-DANN	21.8	18.1

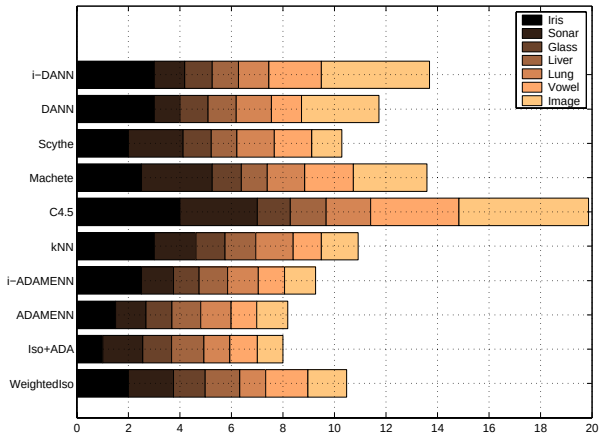

Figure 8: Performance distributions.

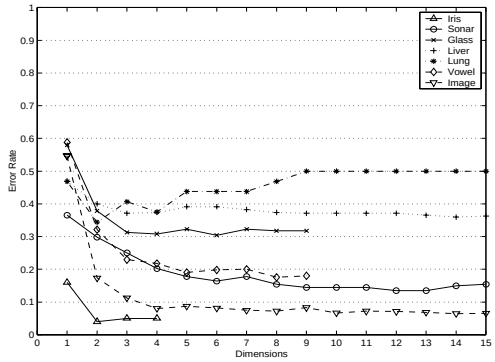

Figure 9: Error rate for the WeightedIso method as a function of the dimensionality d of the reduced feature space.

9. CONCLUSIONS

We have addressed the issue of using local embeddings for data visualization and classification. We have analyzed the LLE and Isomap techniques, and enhanced their visualization power for data scattered among multiple clusters. Furthermore, we have tackled the curse-of-dimensionality problem for classification by combining the Isomap procedure with locally adaptive metric techniques for nearest neighbor classification. Using real data sets we have shown that our methods provide the same classification power as other methods, but in a much lower dimensional space. Therefore, since the proposed methods considerably reduce the dimensionality of the original feature space, efficient indexing data structures can be employed to perform nearest neighbor search.

10. REFERENCES

[1] R. Agrawal, C. Faloutsos, and A. Swami. Efficient Similarity Search in Sequence Databases. *In Proc. of the 4th FODO*, pages 69–84, Oct. 1993.

[2] N. Beckmann, H. Kriegel, and R. Schnei. The r * -tree: an efficient and robust access method for points and rectangles. In *Proceedings of ACM SIGMOD Conference*, 1990.

[3] R. Bellman. *Adaptive Control Processes.* Princeton Univ. Press, 1961.

[4] C. Bentley and M. O. Ward. Animating multidimensional scaling to visualize n-dimensional data sets. In *In Proc.of InfoVis*, 1996.

[5] K. Chan and A. W.-C. Fu. Efficient Time Series Matching by Wavelets. *In Proc. of ICDE*, pages 126–133, Mar. 1999.

[6] T. Cover and P. Hart. Nearest Neighbor Pattern Classification. *IEEE Trans. on Information Theory, pp. 21-27*, 1967.

[7] C. Domeniconi, J. Peng, and D. Gunopulos. An Adaptive Metric Machine for Pattern Classification. *Advances in Neural Information Processing Systems*, 2000.

[8] C. Faloutsos and K.-I. Lin. FastMap: A fast algorithm for indexing, data-mining and visualization of traditional and multimedia datasets. *In Proc. ACM SIGMOD*, pages 163–174, May 1995.

[9] C. Faloutsos, M. Ranganathan, and I. Manolopoulos. Fast Subsequence Matching in Time Series Databases. *In Proceedings of ACM SIGMOD*, pages 419–429, May 1994.

[10] J. Friedman. Flexible Metric Nearest Neighbor Classification. *Tech. Report, Dept. of Statistics, Stanford University*, 1994.

[11] T. Hastie and R. Tibshirani. Discriminant Adaptive Nearest Neighbor Classification. *IEEE Trans. on Pattern Analysis and Machine Intelligence, Vol. 18, No. 6, pp. 607-615*, 1996.

[12] S. Haykin. *Neural Networks: A Comprehensive Foundation.* Macmillan College Publishing Company New York, 1994.

[13] T. Ho. Nearest Neighbors in Random Subspaces. *Lecture Notes in Computer Science: Advances in Pattern Recognition, pp. 640-648*, 1998.

[14] A. Inselberg and B. Dimsdale. Parallel coordinates: A tool for visualizing multidimensional geometry. In *In Proc. of IEEE Visualization*, 1990.

[15] J. C. L. J. B. Tenenbaum, V. de Silva. A global geometric framework for nonlinear dimensionality reduction. *Science v.290 no.5500*, pages 2319–2323, 2000.

[16] I. T. Jolliffe. *Principal Component Analysis.* Springer-Verlag, New York, 1989.

[17] E. Keogh, K. Chakrabarti, S. Mehrotra, and M. Pazzani. Locally adaptive dimensionality reduction for indexing large time series databases. In *Proc. of ACM SIGMOD*, pages 151–162, 2001.

[18] H. S. S. . D. D. Lee. The manifold ways of perception. *Science, v.290 no.5500*, pages 2268–2269.

[19] R. C. T. Lee, J. R. Slagle, and H. Blum. A triangulation method for the sequential mapping of points from N-space to two-space. *IEEE Transactions on Computers*, pages 288–92, Mar. 1977.

[20] D. Lowe. Similarity Metric Learning for a Variable-Kernel Classifier . *Neural Computation,* ***7****(1):72-85*, 1995.

[21] G. McLachlan. *Discriminant Analysis and Statistical Pattern Recognition.* New York: Wiley, 1992.

[22] C. Merz and P. Murphy. UCI Repository of Machine Learning databases. *http://www.ics.uci.edu/mlearn/MLRepository.html*, 1996.

[23] T. Poggio and F. Girosi. Networks for approximation and learning. *proc. IEEE 78, 1481*, 1990.

[24] M. Polito and P. Perona. Grouping and dimensionality reduction by locally linear embedding. In *NIPS*, 2001.

[25] J. Quinlan. *C4.5: Programs for Machine Learning.* Morgan-Kaufmann Publishers, Inc., 1993.

[26] S. R. . L. Saul. Nonlinear dimensionality reduction by locally linear embedding. *Science v.290 no.5500*, pages 2223–2326, 2000.