Multilevel Linear Dimensionality Reduction for Data Analysis using Nearest-Neighbor Graphs[*]

Sophia Sakellaridi
Department of Computer Science and Engineering
University of Minnesota;
Minneapolis, MN 55455
sakell@cs.umn.edu

Haw-ren Fang
Department of Computer Science and Engineering
University of Minnesota;
Minneapolis, MN 55455
hrfang@cs.umn.edu

Yousef Saad
Department of Computer Science and Engineering
University of Minnesota;
Minneapolis, MN 55455
saad@cs.umn.edu

ABSTRACT

Dimension reduction techniques can be time-consuming when the data set is large. This paper presents a multilevel framework to reduce the size of the data set, prior to performing dimension reduction. The algorithm exploits nearest-neighbor graphs. It recursively coarsens the data by finding a maximal matching level by level. Once the coarse graph is available, the coarsened data is projected at the lowest level using a known linear dimensionality reduction method. To obtain the projected data at the highest level, the same linear mapping as that of the lowest level is performed on the original data set, and on any new test data. The methods are illustrated on three applications: manifold mapping, face recognition, and text mining. Experimental results indicate that the multilevel techniques presented in this paper offer a very appealing cost to quality ratio.

Categories and Subject Descriptors

F.2.1 [**Numerical Algorithms and Problems**]: Computations on matrices; G.2.2 [**Graph Theory**]: Graph algorithms; H.3.3 [**Information Search and Retrieval**]: Retrieval models, Relevance feedback

General Terms

algorithms, performance

Keywords

multilevel coarsening, dimensionality reduction, nearest-neighbor graph

1. INTRODUCTION

The goal of dimensionality reduction is to map high dimensional data samples to a lower dimensional space such that certain properties of the data are preserved. When the number of data samples is large, existing methods can be time-consuming.

Several dimensionality reduction methods involve the SVD computation. These include among many others the principal component analysis (PCA), see, e.g., [**?**], the method Locally Linear Embedding, [**?**], the locality preserving projections [**?**], and the Orthogonal Neighborhood Preserving Projections (ONPP) [14, 16]. There were efforts made to bypass the SVD, by using the Lanczos method [3], polynomial filtering [13, 15], or semi-discrete decomposition [17]. This paper considers an alternative that does not rely on the SVD on the whole data set.

The multilevel paradigm has been in use in many applications and was successfully used for graph partitioning (e.g., [11, 12]). It has also been applied to solve various combinatorial optimization problems, such as travelling salesman problem [21, 22]. Inspired by its success in other domains, we propose a multilevel framework for dimensionality reduction.

The multilevel paradigm considered here is graph-based. When it is applied to a set of sampled points, one can use a k-nearest-neighbor (kNN) graph of the data points. (That is, each vertex has k outgoing edges to the k nearest neighbors.)

Given a graph which captures certain information of closeness of the data points (e.g., a kNN graph), we can compute a coarse approximation (i.e., with fewer data points) of the data set using a maximal matching, which is widely used in multilevel graph partitioning (e.g., [11, 12]). Then, we project the coarsened data at the lowest level using an already known linear method for dimensionality reduction, e.g., Principal Component Analysis (PCA).

Since in this paper we consider linear dimensionality reduction methods, in order to refine the data set we may simply embed the original data set into a low dimensional space using the linear transformation from the lowest level.

The rest of this paper is organized as follows. In Section 2 we propose a multilevel framework for linear dimensionality reduction. Applications to manifold mapping (unsupervised learning), face recognition (supervised learning), and

[*] This work was supported by NSF grants DMS 0510131 and DMS 0528492 and by the Minnesota Supercomputing Institute.

text mining (information retrieval) are illustrated in Sections 3, 4, and 5, respectively. All experiments were performed in sequential mode on a PC equipped with two Intel(R) Core(TM)2 @ 2.40GHz processors. A conclusion is given in Section 6.

2. MULTILEVEL DIMENSIONALITY REDUCTION

A multilevel algorithm typically consists of three phases: the coarsening phase, the action phase, and the refining phase. For multilevel dimensionality reduction, the action phase consists of mapping the coarsened data at the lowest level to a lower dimensional space. The multilevel paradigm in this paper relies on a graph $G = (V, E)$ with weighted edges. The vertex set V contains the indices of the data entries. Each edge $(i, j) \in E$ has a weight from certain measurement of the relation between vertex i and vertex j. When it is applied to graph partitioning or travelling salesman problem, an original graph is usually available. In our case we use a kNN graph of the data items.

2.1 The coarsening phase

Consider a data set of n entries represented by a matrix $X = [x_1, x_2, \ldots, x_n] \in \mathbb{R}^{m \times n}$. Suppose we have obtained an affinity graph $G = (V, E)$ of data entries of X, where the vertex set V are indices of data entries, and the edges in E connect neighboring data entries.

The meaning of coarsening a graph is to find a 'coarse' approximation $\hat{G} = (\hat{V}, \hat{E})$ of a given graph $G = (V, E)$ so that $|\hat{V}| < |V|$. By recursively coarsening we obtain a hierarchy of approximations to the original graph.

We use a technique called *maximal matching* for graph coarsening. A *matching* of a graph $G = (V, E)$ is a subset of edges $\hat{E} \subset E$ such that no edge in $\hat{E}$ shares an endpoint. A *maximal matching* is a matching to which no more edges can be added and remain a matching. Algorithm 1 gives a high level description of the algorithm to compute a maximal matching.

Consider Algorithm 1. We do not need in (*) the closest neighbor j of vertex i for a maximal matching. However, the closer the data points are in each pair, the more representative the coarse graph $\hat{G}$ is to the original graph G. This scheme is similar to heavy edge matching (HEM) in multilevel graph partitioning (see, e.g., [11, 12]).

In Algorithm 1, even if the graph G is connected, it is possible that we may not be able to find a matching neighbor for a given vertex. In this case this single vertex is passed to the coarse level without pairing as in (**). The more there are edges in E (e.g., larger k of a kNN graph), the fewer the edges in $\hat{V}$. The number of vertices in $\hat{V}$ is at least half of that in V (i.e., $|\hat{V}| \geq 0.5|V|$). We use a sparse matrix to store distances between neighboring points presented by function d. a large k may slow down the coarsening, since more neighbors need to visit to determine the vertex set $\hat{V}$ and edge set $\hat{E}$.

Algorithm 1 Graph coarsening by a maximal matching.

```
{Given a graph $G = (V, E)$ with $V = \{1, \ldots, n\}$, and
    $d : E \to \mathbb{R}$. Neighboring points are connected by edges
    in $E$. Function $d$ measures the distances of neighboring
    points.}
{Output is a coarse graph $\hat{G} = (\hat{V}, \hat{E})$, where $\hat{V}$ is formed
    by a maximal matching.}
$\hat{V} \leftarrow \emptyset$                         ▷ Maximal matching set
$S \leftarrow \emptyset$                         ▷ Set of matched vertices
for all $i \in V$ do                   ▷ Visit vertices in any order
    if $\exists j \notin S$ such that $(i, j) \in E$ then
        $j = \text{argmin}\{d(i, j) : j \notin S\}$              ▷ (*)
        $\hat{V} \leftarrow \hat{V} \cup \{\{i, j\}\}$
        $S \leftarrow S \cup \{i, j\}$
    else
        $\hat{V} \leftarrow \hat{V} \cup \{\{i\}\}$                        ▷ (**)
    end if
end for
$\hat{E} \leftarrow \emptyset$                 ▷ Edge set of the coarse graph
for all $s, t \in \hat{V}$ do
    if $\exists(i, j)$ such that $i \in s$, $j \in t$ and $(i, j) \in E$ then
        $\hat{E} \leftarrow \hat{E} \cup \{(s, t)\}$
    end if
end for
```

Note that to recursively coarsen the data, we do not need to compute a kNN graph for each level. Each edge is associated with an estimated distance between the two vertices it connects. In the topmost (finest) level the actual distances are used. Following the notation in Algorithm 1, For $(a, b) \in \hat{E}$, we compute

$$\hat{d}(a, b) := \text{mean}\{d(i, j) : i \in a, j \in b\}.$$

Here we take the mean of distances of the fine level that best presents the distance of the coarse level. This is different from the standard multilevel graph partitioning (e.g., [12]), where each edge has a weight, and the weight of an edge of the coarse level are computed as the sum (rather than mean) of the relevant edge weights at the fine level.

By recursively coarsening the graph, we obtain a sequence of graphs $G_1, G_2, \ldots, G_r$, where $G_k = (V_k, E_k)$ is the coarse graph of level k for $k = 1, \ldots, r$, and G_r is the lowest level graph. The corresponding data sets are denoted by matrices $X_i \in \mathbb{R}^{m \times |V_i|}$ for $i = 1, \ldots, r$.

2.2 The dimensionality reduction phase

The purpose of dimensionality reduction is to remove noise and redundancies from the data. Given a data set $X = [x_1, x_2, \ldots, x_n] \in \mathbb{R}^{m \times n}$, a dimensionality reduction method produces $Y = [y_1, y_2, \ldots, y_n] \in \mathbb{R}^{d \times n}$ $(d < m)$, such that Y preserves certain features of X.

In the multilevel framework, we apply a linear dimensionality reduction method to the data set $X_r \in \mathbb{R}^{m \times |V_r|}$ of the lowest level (rth level), and obtain $Y_r \in \mathbb{R}^{d \times |V_r|}$ $(d < m)$. The dimensionality reduction methods used in our experiments are PCA, LPP [9, 10], and ONPP [14, 16], involving SVD computation.

2.3 The refining phase

In the refining phase we have the reduced representation $Y_r \in \mathbb{R}^{d \times |V_r|}$ of data $X_r \in \mathbb{R}^{m \times |V_r|}$ of the lowest level (rth level). The objective is to obtain a reduced representation $Y \in \mathbb{R}^{d \times n}$ of the data $X \in \mathbb{R}^{m \times n}$ at the topmost level ($X = X_1$ and $n = |V_1|$).

When a linear dimensionality reduction method is used, there is a transformation matrix P that maps X_r into Y_r by $Y_r = PX_r$. Therefore, we apply the mapping P to the original data X and obtain $Y = PX$ in the low dimensional space. The procedure is illustrated in Figure 1. The same linear mapping can also be applied to any 'out-of-sample' test data. For example, in face recognition, the same projector is applied to a test image and to the training set of images before a comparison is made.

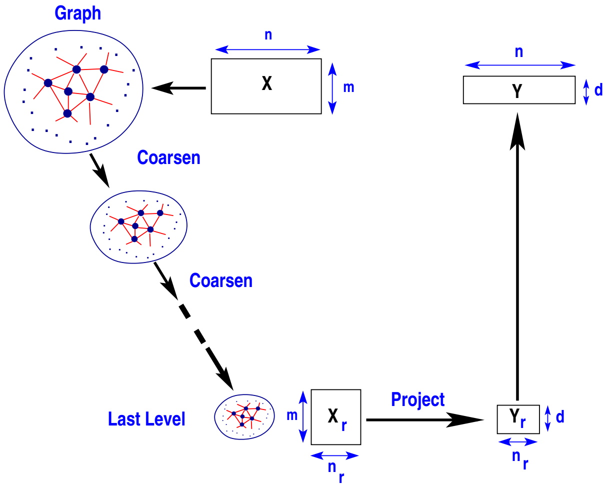
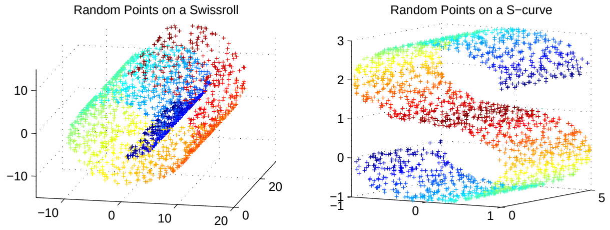


Figure 1: A sketch of the multilevel reduction.

3. APPLICATION TO MANIFOLD MAPPING

In this section we present an evaluation of our multilevel techniques, using as the initial step three dimensionality reduction methods: PCA, LPP [9, 10], and ONPP [14, 16]. These dimensionality reduction methods, used as an unsupervised learning process, can be applied to discover underlying manifold structure.

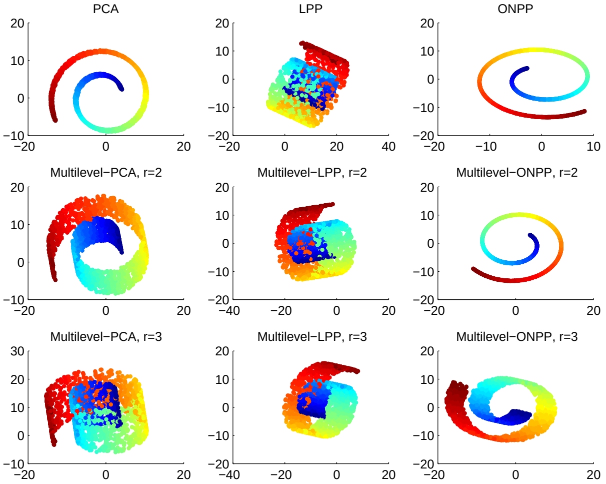

Random Points on a Swissroll

Random Points on a S−curve

Figure 2: Two examples of data points sampled on 3-D manifolds.

Note that both LPP and ONPP need an affinity graph for dimensionality reduction computation. For unsupervised learning, a kNN graph is usually used. In the multilevel framework, we can construct a kNN graph of vertices of the lowest level, or simply use the graph recursively coarsened from the topmost level. In our experiments, the former option was chosen for ONPP ($k = 20$), and the latter was adopted by LPP.

We used two synthetic data sets sampled in three-dimensional space: the `Swissroll` and the `S-curve`, each with 2,000 sample points, as shown in Figure 2. These data sets, though embedded in three-dimensional space, are often used in the extent of two-dimensional submanifolds.


PCA

LPP

ONPP

Multilevel−PCA, r=2

Multilevel−LPP, r=2

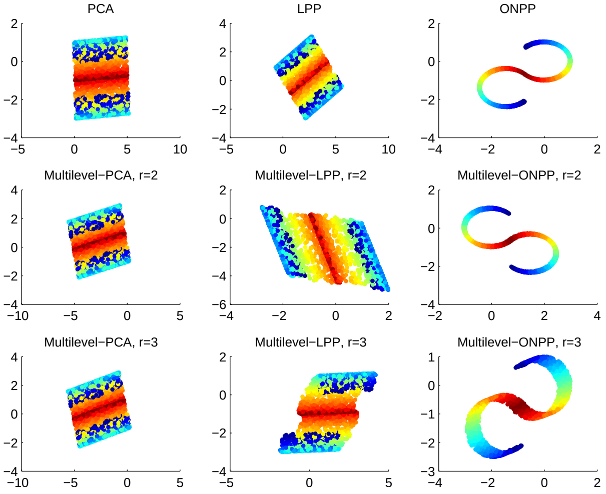
Multilevel−ONPP, r=2

Multilevel−PCA, r=3

Multilevel−LPP, r=3

Multilevel−ONPP, r=3

Figure 3: Two-dimensional projections of `Swissroll` using PCA, LPP, and ONPP, and those with multilevel techniques ($k = 8$).


PCA

LPP

ONPP

Multilevel−PCA, r=2

Multilevel−LPP, r=2

Multilevel−ONPP, r=2

Multilevel−PCA, r=3

Multilevel−LPP, r=3

Multilevel−ONPP, r=3

Figure 4: Two-dimensional projections of `S-curve`, using PCA, LPP, and ONPP, and those with multilevel techniques ($k = 8$).

Figures 3 and 4, illustrate the two-dimensional projection of `Swissroll` and `S-curve` data sets, respectively. They are obtained by PCA, LPP and ONPP, and those with multilevel techniques with the numbers of levels $r = 2, 3$. In the kNN-graph construction, we set the number of the nearest neighbors per sampled point $k = 8$. The result tends to indicate that we gradually lost the cohesiveness but gained the geodesic distance information while the number of levels increased.

4. APPLICATION TO FACE RECOGNITION

Dimensionality reduction methods used in supervised learning (classification), use label information for the samples. This means that each data entry is assigned a class label, and the labels are needed to find the projection. Such methods include for example the Linear Discriminant Analysis (LDA) [1, 6]. The methods LPP, and ONPP, when used in supervised mode, also use class labels to construct a label graph instead of using an affinity graph. However, the label information seems to be partly lost or destroyed in the coarsening phase and for this reason, the multilevel framework is not adequate for these methods.

Figure 5: Sample images from the ORL database.

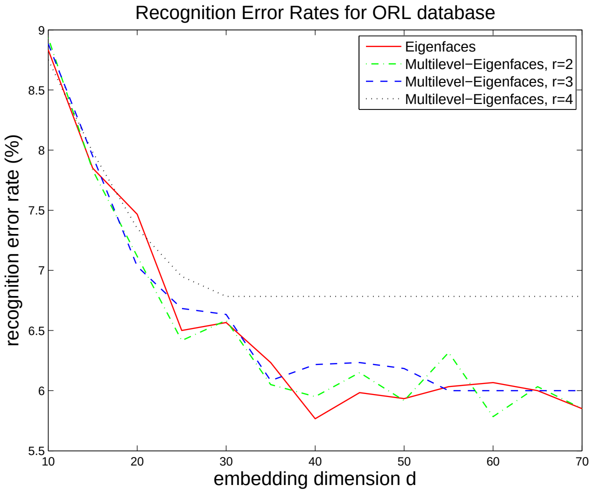


Figure 6: Average recognition error rate using the ORL data set, # training samples/class $t = 5$.

Here, we evaluate PCA (EigenFaces) and multilevel-PCA (multilevel EigenFaces) for face recognition. We used the datasets ORL [20] and UMIST [7]. Both datasets have images of size 112×92. For computational efficiency the images were cropped to size 38×31. For measuring the recognition performance we used a random subset of images from each subject as training set, and the remaining as test set. For the multilevel-PCA, the kNN graph for the training set was constructed with $k = 10$ neighbors for all datasets, and then it was coarsened using up to four levels. In order to ensure that our results are not biased from a specific random realization of the training/test set, we performed 30 different random realizations of the training/test sets.

The ORL database contains images of 40 individuals, each providing 10 images of different facial expressions (smiling/non smiling, open/closed eyes) and facial details. Sample images of two individuals from the ORL database are shown in Figure 5. We formed the training set by a random subset of $t = 5, 6, \ldots, 9$ different images per subject and used the remaining images as the test set, respectively. The average number of vertices of each of the coarsen levels is shown in Table 1. The CPU time used for coarsening is displayed in Table 2, and the best error rates achieved for each value of t are presented in Table 3 along with the corresponding dimension. Figure 6 gives a plot for several dimensions the average error rate for $t = 5$ training samples per class, and the computational savings of multilevel-PCA on computing SVD of a coarsened (smaller) matrix of vectorized images are shown in Figure 7.

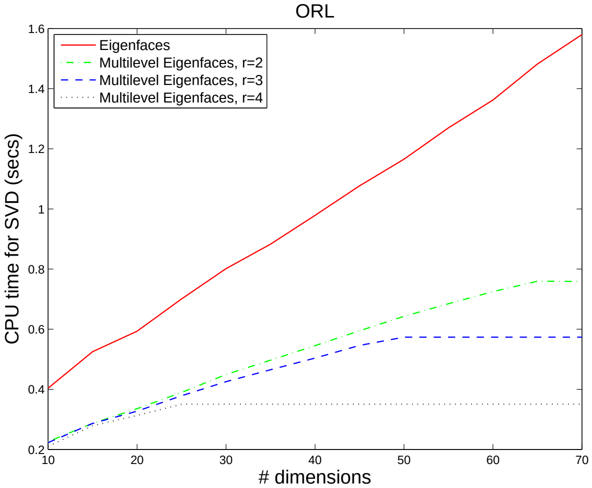


Figure 7: CPU time for truncated SVD using the ORL data set, # training samples/class $t = 5$.

ORL	# train.	5	6	7	8	9
	level #1	200	240	280	320	360
	level #2	105	125	147	166	187
	level #3	56	66	77	87	100
	level #4	30	36	41	46	53
UMIST	# train.	14	15	16	17	18
	level #1	280	300	320	340	360
	level #2	146	156	167	177	188
	level #3	77	83	88	93	99
	level #4	41	44	47	50	53

Table 1: Average number of vertices at all levels.

Figure 8: Sample images from the UMIST database.

The UMIST database contains images of 20 subjects with

	# train.	5	6	7	8	9
ORL	to level #2	0.03	0.03	0.04	0.05	0.04
	to level #3	0.01	0.01	0.02	0.02	0.01
	to level #4	0.01	0.01	0.01	0.01	0.01
	# train.	14	15	16	17	18
UMIST	to level #2	0.04	0.04	0.04	0.04	0.05
	to level #3	0.02	0.02	0.02	0.02	0.02
	to level #4	0.01	0.01	0.01	0.01	0.01

Table 2: CPU time (seconds) for graph coarsening

	# train.	5	6	7	8	9
ORL	level #1	5.77 (40)	3.78 (40)	2.56 (65)	1.63 (45)	1.01 (45)
	level #2	5.78 (60)	3.78 (55)	2.59 (65)	1.68 (45)	0.99 (45)
	level #3	6.00 (55)	4.03 (40)	2.75 (50)	1.75 (65)	1.01 (45)
	level #4	6.78 (30)	4.65 (35)	3.25 (30)	1.76 (45)	1.02 (45)
	# train.	14	15	16	17	18
UMIST	level #1	2.05 (55)	1.81 (50)	1.51 (45)	1.43 (20)	1.16 (65)
	level #2	2.10 (55)	1.82 (55)	1.52 (45)	1.47 (60)	1.21 (65)
	level #3	2.15 (55)	1.83 (55)	1.61 (45)	1.48 (55)	1.29 (65)
	level #4	2.44 (35)	1.92 (45)	1.62 (45)	1.72 (25)	1.32 (20)

Table 3: Best average recognition error rate (%). The values in parentheses denote the optimal dimensions.

19 to 48 images per subject. Figure 8 shows sample images of one individual from the UMIST database. We form the training set by a random subset of $t = 14, 15, \ldots, 18$ different images per subject and use the remaining images as a test set. The average number of vertices of each of the coarsen levels is shown in Table 1. The CPU time used for coarsening is displayed in Table 2, and the best error rates achieved for each value of t are presented in Table 3 along with the corresponding dimension. Figure 9 gives a plot of the average error rate for $t = 14$ training samples per class for various dimensions. The computational savings of multilevel PCA on computing SVD of a coarsened matrix of vectorized images for $t = 14$ are shown in Figure 10.

The experimental results show that in both datasets, multilevel-PCA achieves recognition error rates very close to those of PCA. Although the multilevel eigenfaces spends additional time to perform the graph coarsening, the savings obtained from computing the smaller SVD of the coarsened matrix, outweigh this additional cost.

5. APPLICATION TO TEXT MINING

We illustrate the algorithm described in this paper on an application in information retrieval by *Latent Semantic Indexing* (LSI) [4], a well-established framework for conceptual information retrieval [2, 5].

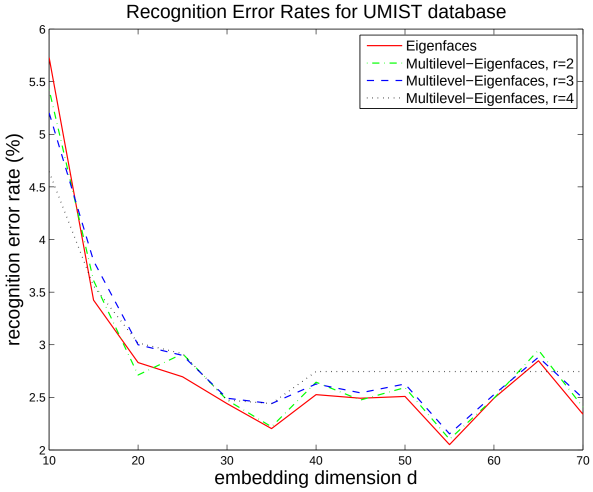


Figure 9: Average recognition rate using the UMIST data set, # of training samples/class $t = 14$.

In the vector space model, a collection of n documents indexed by m terms is represented by a sparse term-document matrix $X \in \mathbb{R}^{m \times n}$. The rows and columns of X are called term vectors and document vectors, respectively. The (i, j) entry of X, denoted by x_{ij}, is the number of occurrences of term i in document j, called *term frequency*.

A term-document matrix $X \in \mathbb{R}^{m \times n}$ is usually normalized/scaled before its usage. In the experiments we used TF-IDF (term frequency-inverse document frequency) scaling [19]. The nearest neighbor graphs are computed based on normalized matrices. The *inverse document frequency* is defined by

$$z_i = \log(n/|\{j : x_{ij} > 0\}|), \qquad (1)$$

where $|\{j : x_{ij} > 0\}|$ is the number of documents with term i occurring in them. A TF-IDF entry is defined by $\tilde{x}_{ij} = x_{ij} z_i$. Finally the TF-IDF scaled matrix $\bar{X}$ is obtained by normalizing the columns to be unit vectors. More precisely, the (i, j) entry of $\bar{X}$ is $\bar{x}_{ij} = \tilde{x}_{ij} / \sqrt{\sum_{k=1}^{m} \tilde{x}_{kj}^2}$ for $i = 1, \ldots, m$ and $j = 1, \ldots, n$.

Query matching is the process of finding the documents most relevant to a given query $q \in \mathbb{R}^m$, an array of term frequencies, which is also called a *pseudo-document* vector. A query is also TF-IDF scaled before the matching process, resulting in the scaled vector $\bar{q}$. Here the inverse document frequencies (IDF), defined in (1), are from the term-document matrix X.

The vector space model measures the similarity of two vectors (a document and a pseudo-document) by the cosine of the acute angle between them. Instead of using the full vector space model, LSI approximates a given term-document matrix by its truncated SVD, denoted by $\bar{X} = [\bar{x}_1, \bar{x}_2, \ldots, \bar{x}_n] \approx U_d \Sigma_d V_d^T$, where d is a certain desired rank. Working in a lower dimensional approximation to X reduces the noise and unravel the underlying semantic structure of the data. The rows of $U_d \in \mathbb{R}^{m \times d}$ are the reduced term vectors. Likewise,

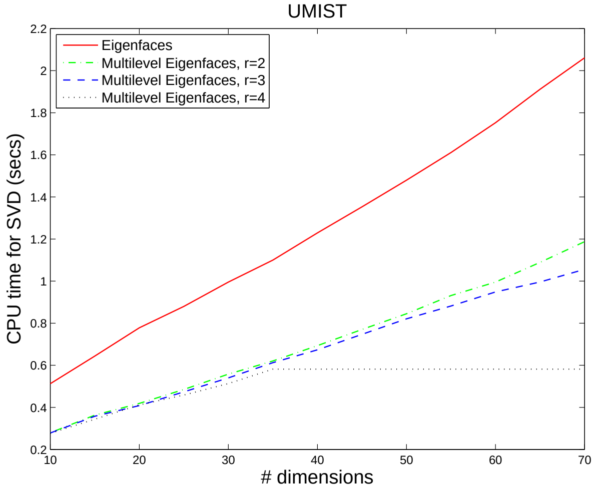


Figure 10: CPU time for truncated SVD using the UMIST data set, # of training samples/class $t = 14$.

the columns of $V_d^T = [\hat{x}_1, \hat{x}_2, \ldots, \hat{x}_n] \in \mathbb{R}^{d\times n}$ are used as reduced representations of document vectors $\bar{x}_1, \bar{x}_2, \ldots, \bar{x}_n \in \mathbb{R}^m$. Given a query $\bar{q} \in \mathbb{R}^m$, it is transformed to a reduced representation $\hat{q} = \Sigma_d^{-1} U_d^T \bar{q} \in \mathbb{R}^d$ in d-dimensional space. Document x_i is considered relevant to the query q if the cosine distance $\langle \hat{q}, \hat{x}_i \rangle / \|\hat{q}\| \|\hat{x}_i\|$ is larger than some pre-defined threshold. When a relevance vector (a boolean string of size n) is provided, the *precision* and *recall* are defined by

$$\text{Precision: } \frac{D_R}{D_T}, \qquad \text{Recall: } \frac{D_R}{N_R}, \tag{2}$$

where D_R, D_T, and N_R are the number of *relevant documents retrieved by the process*, the total number of documents retrieved, and the total number of *relevant* documents in the collection, respectively.

When the term-document matrix X is large, the computation of the SVD factorization can be expensive. The multilevel techniques described in Section 2 will find a smaller set of document vectors, denoted by $X_r \in \mathbb{R}^{m\times n_r}$, to represent $X \in \mathbb{R}^{m\times n}$ ($n_r < n$). We then apply TF-IDF scaling to X_r and obtain $\bar{X}_r$. Like the standard LSI, we compute the truncated SVD of $\bar{X}_r \approx U_d \Sigma_d V_d^T$, where d is the rank. Now the reduced representation of X is $\Sigma_d^{-1} U_d^T \bar{X} = [\hat{x}_1, \hat{x}_2, \ldots, \hat{x}_n] \in \mathbb{R}^{d\times n}$. Each query $q \in \mathbb{R}^m$ is transformed to a reduced representation $\hat{q} = \Sigma_d^{-1} U_d^T \bar{q} \in \mathbb{R}^d$. The similarity of q and x_i are measured by the cosine distance between $\hat{q}$ and $\hat{x}_i$ for $i = 1, \ldots, n$. Note that we have applied TF-IDF scaling to the term-document matrix X and the query q to obtain $\bar{X}$ and $\bar{q}$, but here the inverse document frequencies (IDF) defined in (1) use the coarsened matrix X_r. We call the resulting scheme multilevel-LSI.

Note that the precision and recall defined in (2) depend on the tolerance of the similarity scores in cosine distance measure. To assess the retrieval performance, we also use the average precision [8]. Sorting the similarity scores of query q to documents $x_1, \ldots, x_n$, we consider for $i = 1, \ldots, n$ the first i documents with the highest scores and obtain the precision and recall

$$P_i = r_i/i, \qquad R_i = r_i/r_n,$$

where r_i is the number of relevant documents among the first i documents. The *average precision* is defined by

$$\bar{P} = \frac{1}{n} \sum_{i=0}^{n-1} \hat{P}\left(\frac{i}{n-1}\right), \qquad \hat{P}(x) = \max\{P_i : x \le R_i\}.$$

We used two public data sets in the experiments: Medline and Cranfield[1]. The characteristics of these sets, such as numbers of documents, terms, and queries are listed in Table 4.

Table 4: Characteristics of the test sets.

Data set	Medline	Cranfield
# documents	1033	1398
# terms	7014	3763
sparsity (%)	0.74%	1.41%
# queries	30	225
avg. # rel./query	23.2	8.2

In both tests we coarsened the data down to four levels. Compared with LSI, multilevel-LSI requires additional work to process the graph coarsening. However, it saves time when computing the truncated SVD of the coarsened (smaller) term-document matrix. The CPU time used for coarsening Medline and Cranfield data sets is shown in the second columns of Tables 5 and 6, respectively.

Note that the average precision depends on the dimension used. When the average precision is maximized, we say that the corresponding dimension is *optimal*.

The result of experiment on Medline data set is now discussed. Figure 11 is the resulting plot of average precisions using various dimensions for SVD (ranks of truncated SVD). The number of documents, the optimal dimensions, and the average precision at all levels are displayed in Table 5. Using the optimal dimensions we obtain the precision-recall plot in Figure 12. The savings in CPU time gained by multilevel-LSI for computing truncated SVD are shown in Figure 13.

Table 5: Statistics of Medline data set.

Level	coarsen.	#	optimal	optimal avg.
	time	doc.	# dim.	precision
#1	N/A	1033	30	71.6%
#2	0.37	527	32	72.8%
#3	0.20	271	30	73.1%
#4	0.12	139	33	70.6%

Figure 14 is the resulting plot of average precisions using various dimensions for SVD (ranks of truncated SVD) for the Cranfield data set. The number of documents, optimal dimensions and average precision at all levels are displayed in Table 6. Using the optimal dimensions we obtain the precision-recall plot in Figure 15. The savings in CPU time gained by multilevel-LSI for computing the truncated SVD are shown in Figure 16.

[1] ftp://ftp.cs.cornell.edu/pub/smart

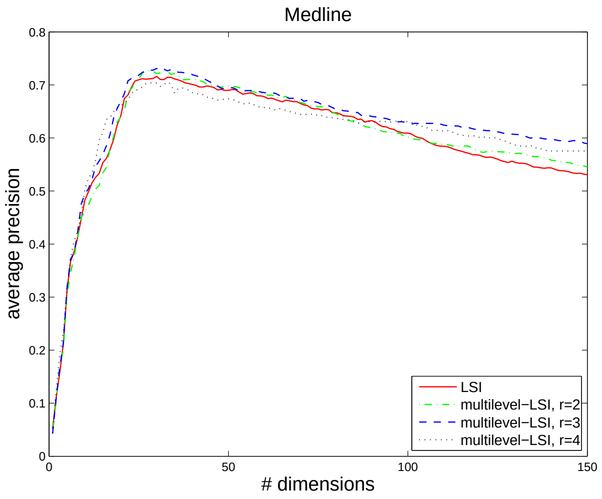


Figure 11: Average precision using the `Medline` data set.

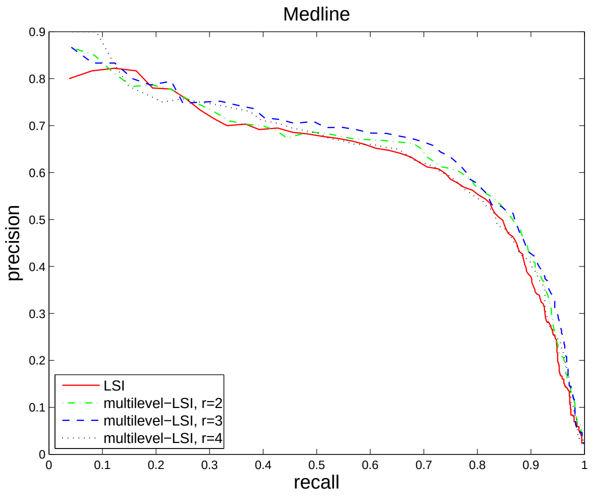


Figure 12: Precision-recall plot using the `Medline` data set.

Table 6: Statistics of `Cran` data set.

Level	coarsen. time	# doc.	optimal # dim.	optimal avg. precision
#1	N/A	1398	95	39.8%
#2	0.95	717	65	39.5%
#3	0.34	375	67	39.1%
#4	0.18	199	52	36.3%

Note that here the savings from the SVD computation are not as pronounced as those obtained for face recognition. This is probably partly due to the fact that the term-document matrices are sparse, and in multilevel coarsening we gradually lose the the sparsity.

Relevance feedback is a common technique in text information retrieval. The assumption is that we know in advance that some document vectors are related to a query. Then the query is added by the sum of these related documents, followed by a standard text mining procedure. More precisely, a query q is replaced by $q + b^T X$, where b is the boolean column vector indicating which documents are known *a priori* related to query q.

We tested relevance feedback for the multilevel framework. The experiments used the vector b defined above as the relevance vector, assuming that the exact information is available. The resulting average precision plots on `Medline` and `Cranfield` data sets are given in Figures 17 and 18, respectively. These show that the multilevel-LSI still worked nicely, but that the average precision was not as good as that of LSI without relevance feed-back.

6. CONCLUSION

A multilevel approach was proposed for dimensionality reduction in data analysis. The main algorithm coarsens an initial graph, and finds a linear projectr based on the data set associated with the coase data. The algorithm worked as expected for the tests we performed. It is generally much faster than the original SVD-based algorithms, and it yields comparable results.

7. REFERENCES

[1] P. N. Belhumeur, J. P. Hespanha, and D. J. Kriegam. Eigenfaces vs. fisherfaces: Recognition using class specific linear project. *IEEE Trans. Pattern Analysis and Machine Intelligence*, 19(7):711–720, 1997.

[2] M. W. Berry and M. Browne. *Understanding Search Engines*. SIAM, 1999.

[3] J. Chen and Y. Saad. Filtered matrix-vector products via the lanczos algorithm with applications to dimension reduction. Technical Report umsi-2007-J1, Minnesota Supercomputer Institute, University of Minnesota, 2007.

[4] S. Deerwester, S. T. Dumais, G. W. Furnas, T. K. Landauer, and R. Harshman. Indexing by latent semantic analysis. *J. Soc. Inf. Sci.*, 41:391–407, 1990.

[5] L. Eldén. *Understanding Search Engines*. SIAM, 2007.

[6] R. A. Fisher. The use of multiple measurements in taxonomic problems. *Annals of Eugenics*, 7:179–188, 1936.

[7] D. B. Graham and N. M. Allinson. Characterizing virtual eigensignatures for general purpose face recognition. *Face Recognition: From Theory to Applications*, 163:446–456, 1998.

[8] D. K. Harman, editor. *The 3rd Text Retrieval Conference (TREC-3)*. NIST Special Publication 500-255, 1995. http://trec.nist.gov/.

[9] X. He and P. Niyogi. Locality preserving projections. In *Advances in Neural Information Processing Systems 16 (NIPS)*, 2003.

[10] X. He, S. Yan, Y. Hu, P. Niyogi, and H.-J. Zhang. Face recognition using laplacianfaces. *IEEE Trans. Pattern Analysis and Machine Intelligence*, 27(3):328–340, 2005.

[11] G. Karypis and V. Kumar. Analysis of multilevel graph partitioning. In *Supercomputing '95:*

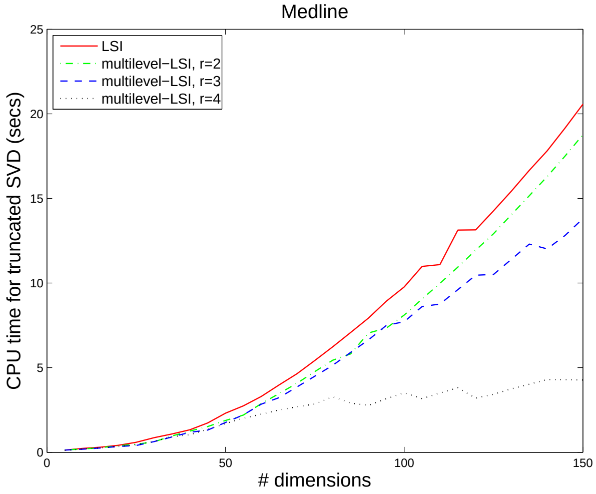


Figure 13: LSI and multilevel-LSI using the `Medline` data set: average precision plot.

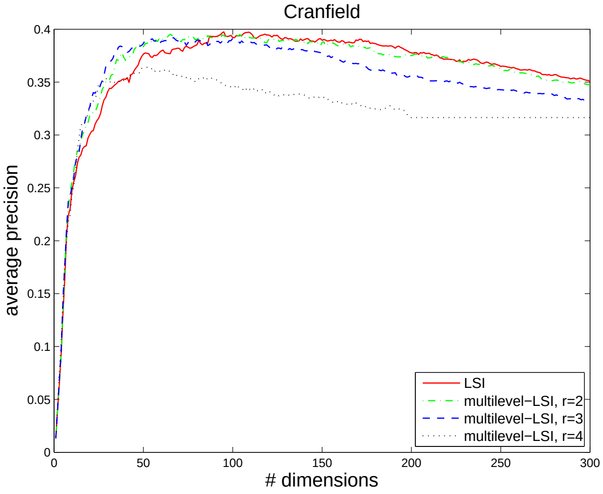


Figure 14: Average precision plot using the `Cranfield` data set.

Proceedings of the 1995 ACM/IEEE conference on Supercomputing (CDROM), New York, NY, USA, 1995. ACM Press. Article No. 29.

[12] G. Karypis and V. Kumar. Multilevel k-way partitioning scheme for irregular graphs. *J. Parallel Distrib. Comput.*, 48(1):96–129, 1998.

[13] E. Kokiopoulou and Y. Saad. Polynomial filtering in latent semantic indexing for information retrieval. In *SIGIR '04: Proceedings of the 27th annual international ACM SIGIR conference on Research and development in information retrieval*, pages 104–111, New York, NY, USA, 2004. ACM.

[14] E. Kokiopoulou and Y. Saad. Orthogonal neighborhood preserving projections. In *Proc. the 5th IEEE International Conference on Data Mining*, pages 234–241, 2005.

[15] E. Kokiopoulou and Y. Saad. PCA and kernel PCA using polynomial filtering: a case study on face recognition. In *SIAM International Conference on Data Mining*, 2005.

[16] E. Kokiopoulou and Y. Saad. Orthogonal Neighborhood Preserving Projections: A projection-based dimensionality reduction technique. *IEEE Trans. Pattern Analysis and Machine Intelligence*, 29(12):2143–2156, 2007.

[17] T. G. Kolda and D. P. O'Leary. A semi-discrete matrix decomposition for latent semantic indexing in information retrieval. *ACM Trans. Information Systems*, 16:322–346, 1998.

[18] Y. Saad. Filtered conjugate residual-type algorithms with applications. *SIAM J. Matrix Anal. and Appl.*, 28(3):845–870, 2006.

[19] G. Salton and C. Buckley. Term-weighting approaches in automatic text retrieval. *Inf. Process. Manage.*, 24(5):513–523, 1988.

[20] F. Samaria and A. Harter. Parameterisation of a stochastic model for human face identiïňAcation. In *2nd IEEE Workshop on Applications of Computer Vision*, Sarasota, FL, December 1994.

[21] C. Walshaw. A multilevel approach to the travelling salesman problem. *Operations Research*, 50(5):862–877, 2002.

[22] C. Walshaw. Multilevel refinement for combinatorial optimisation problems. *Annals of Operations Research*, 131(1–4):325–372, 2004.

[23] P. C. Yuen and J. H. Lai. Face recognition using independent component analysis. *Pattern Recognition*, 35:1247–1257, 2002.

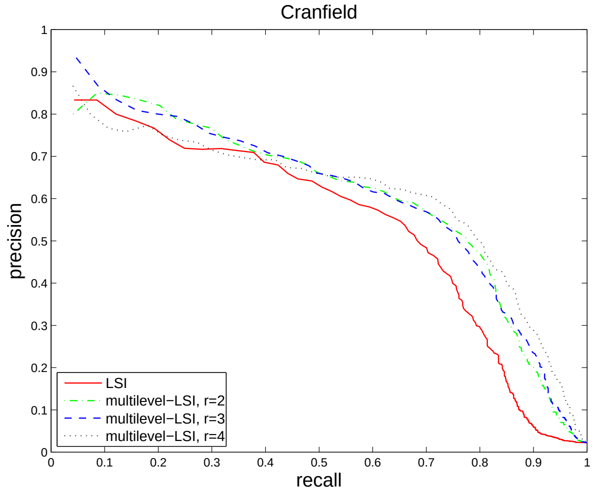


Figure 15: Precision-recall plot using the `Cranfield` data set.

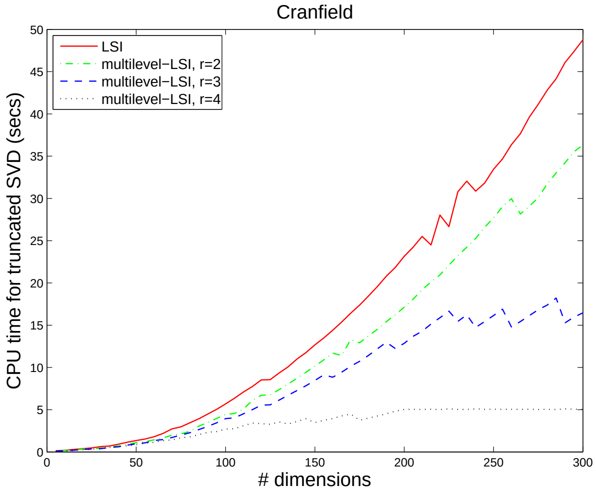


Figure 16: CPU time for truncated SVD using the `Cranfield` data set.

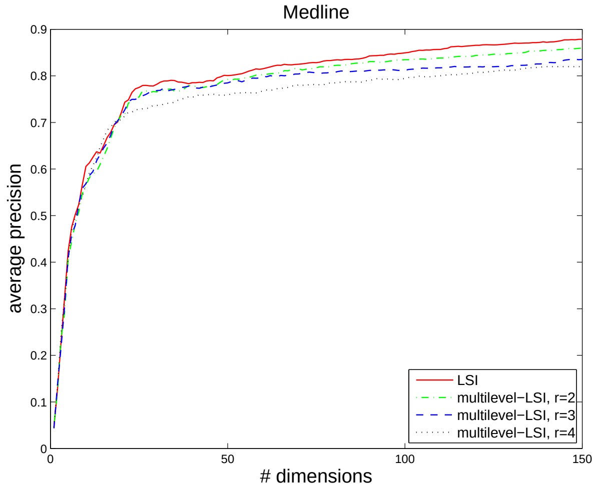


Figure 17: Precision-recall plot using the `Medline` data set with relevance feedback.

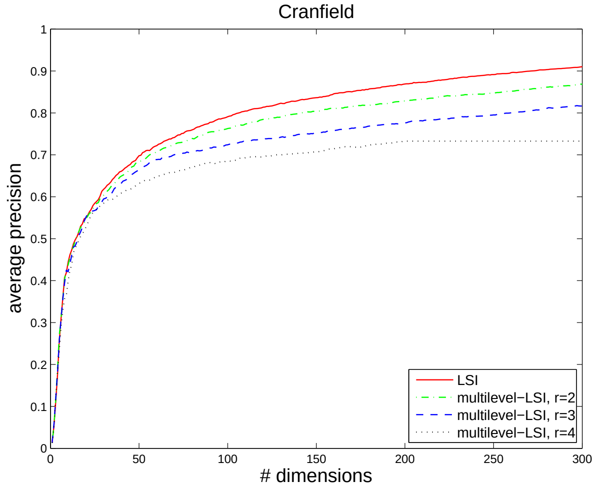


Figure 18: Precision-recall plot using the `Cranfield` data set with relevance feedback.