Compressed Spectral Regression for Efficient Nonlinear Dimensionality Reduction

Deng Cai
The State Key Lab of CAD&CG, College of Computer Science,
Zhejiang University, China
dengcai@cad.zju.edu.cn

Abstract

Spectral dimensionality reduction methods have recently emerged as powerful tools for various applications in pattern recognition, data mining and computer vision. These methods use information contained in the eigenvectors of a data affinity (*i.e.*, item-item similarity) matrix to reveal the low dimensional structure of the high dimensional data. One of the limitations of various spectral dimensionality reduction methods is their high computational complexity. They all need to construct a data affinity matrix and compute the top eigenvectors. This leads to $O(n^2)$ computational complexity, where n is the number of samples. Moreover, when the data are highly non-linear distributed, some linear methods have to be performed in a reproducing kernel Hilbert space (leads to the corresponding kernel methods) to learn an effective non-linear mapping. The computational complexity of these kernel methods is $O(n^3)$. In this paper, we propose a novel nonlinear dimensionality reduction algorithm, called *Compressed Spectral Regression*, with $O(n)$ computational complexity. Extensive experiments on data clustering demonstrate the effectiveness and efficiency of the proposed approach.

Introduction

Dimensionality reduction is one of the most useful tools for data analysis in data mining, pattern recognition and many other research fields. Among various dimensionality reduction approaches, spectral dimensionality reduction methods [Belkin and Niyogi, 2001; He and Niyogi, 2003; Yan *et al.*, 2007; Cai, 2009] have received considerable interests in recent years. These methods use information contained in the eigenvectors of a data affinity (*i.e.*, item-item similarity) matrix to reveal the low dimensional structure of the high dimensional data.

The most popular spectral dimensionality reduction algorithms include Locally Linear Embedding [Roweis and Saul, 2000], ISOMAP [Tenenbaum *et al.*, 2000], and Laplacian Eigenmap [Belkin and Niyogi, 2001]. These algorithms only provide the embedding results of training samples. There are many extensions [He and Niyogi, 2003; Yan *et al.*, 2007] which try to solve the out-of-sample problem (*i.e.*, find the embedding of the test samples [Bengio *et al.*, 2003]) by seeking a projective function in a reproducing kernel Hilbert space. However, a disadvantage of these extensions is that their computations usually involve eigen-decomposition of dense matrices which is expensive in both time and memory [Cai, 2009].

Recently, Cai [2009] proposed an efficient spectral dimensionality reduction framework called *spectral regression* (SR). SR casts the problem of learning a projective function into a regression framework. By avoiding the eigen-decomposition of dense matrices, SR provides an unified efficient solution for many supervised and unsupervised spectral dimensionality reduction algorithms [Cai, 2009]. However, SR in unsupervised setting still needs to construct a data affinity matrix and compute the top eigenvectors of the corresponding Laplacian matrix [Chung, 1997]. For a data set consisting of n data points, these steps have a time complexity of $O(n^2)$. Moreover, when the data are highly nonlinear distributed, SR has to be performed in a reproducing kernel Hilbert space to learn an effective non-linear mapping which leads to kernel SR. As the dense kernel matrix is involved, the time complexity of kernel SR becomes $O(n^3)$. Such a high computational complexity is an unbearable burden for large-scale applications.

Inspired by the recent progresses on scalable semi-supervised learning [Liu *et al.*, 2010] and large scale spectral clustering [Chen and Cai, 2011], we propose an efficient *nonlinear* dimensionality reduction algorithm termed *Compressed Spectral Regression* (CSR) in this paper. Specifically, CSR generates l ($\ll n$) representative points as the landmarks and represent the original data points as the sparse linear combinations of these landmarks. With this landmark-based sparse representation, the spectral embedding of the data as well as the nonlinear projective function can be efficiently computed. The proposed algorithm scales linearly with the data size. Extensive experiments demonstrate the effectiveness and efficiency of proposed approach.

Background

Suppose we have n data points $\{\mathbf{x}_i\}_{i=1}^n \subset \mathbb{R}^m$, $X = [\mathbf{x}_1, \cdots, \mathbf{x}_n]$. In the past decades, many spectral dimensionality reduction algorithms, either supervised or unsupervised,

have been proposed to find a low dimensional representation of $\mathbf{x}_i$. Despite the different motivations of these algorithms, they can be nicely interpreted in a general *graph embedding* framework [Yan *et al.*, 2007; Cai *et al.*, 2007].

These methods first construct an undirected graph $\mathcal{G} = (V, E)$ represented by its adjacency matrix $W = (w_{ij})_{i,j=1}^n$, where $w_{ij} \geq 0$ denotes the similarity (affinity) between $\mathbf{x}_i$ and $\mathbf{x}_j$. The G and W can be defined to characterize certain statistical or geometric properties of the data set [Yan *et al.*, 2007].

Let $\mathbf{y} = [y_1, y_2, \cdots, y_n]^T$ be the one dimensional embedding. The optimal $\mathbf{y}^*$ is given by solving the following optimization problem:

$$\mathbf{y}^* = \operatorname{argmin} \frac{\mathbf{y}^T L \mathbf{y}}{\mathbf{y}^T D \mathbf{y}} = \arg\max \frac{\mathbf{y}^T W \mathbf{y}}{\mathbf{y}^T D \mathbf{y}}, \qquad (1)$$

where the degree matrix D is a diagonal matrix whose entries are column (or row, since W is symmetric) sums of W, $D_{ii} = \sum_j w_{ij}$ and $L = D - W$, which is called graph Laplacian [Chung, 1997]. All the three most popular spectral dimensionality reduction algorithms, LLE [Roweis and Saul, 2000], ISOAMP [Tenenbaum *et al.*, 2000] and Laplacian Eigenmap [Belkin and Niyogi, 2001], can be interpreted in this framework with different choices of W [Yan *et al.*, 2007].

The graph embedding approach in Eq. (1) only provides the mappings for the training data. For classification purpose, a mapping for all the samples, including the new test samples, is required. If we choose a linear projective function, *i.e.*, $y_i = f(\mathbf{x}_i) = \mathbf{a}^T \mathbf{x}_i$, then we have $\mathbf{y} = X^T \mathbf{a}$. Eq. (1) can be rewritten as:

$$\mathbf{a}^* = \arg\max \frac{\mathbf{y}^T W \mathbf{y}}{\mathbf{y}^T D \mathbf{y}} = \arg\max \frac{\mathbf{a}^T X W X^T \mathbf{a}}{\mathbf{a}^T X D X^T \mathbf{a}}. \qquad (2)$$

This approach is called *Linear extension of Graph Embedding* (LGE) [Yan *et al.*, 2007]. With different choices of W, the LGE framework will lead to many popular linear dimensionality reduction algorithms, *e.g.*, Linear Discriminant Analysis, Locality Preserving Projection [He and Niyogi, 2003] and Neighborhood Preserving Embedding [He *et al.*, 2005].

Solving the optimization problem in Eq. (2) involves eigendecomposition of dense matrices XWX^T and XDX^T, which is time consuming for large scale high dimensional data. To tackle this issue, Cai [2009] proposed a spectral regression (SR) approach. SR learns the projective function in two steps. In the first step, SR solves the optimization problem (1) to get $\mathbf{y}^* = [y_1^*, y_2^*, \cdots, y_n^*]^T$. Since W is usually sparse and D is a diagonal matrix, the optimization problem (1) can be efficiently solved. In the second step, SR solves a regression problem to compute the projective function $\mathbf{a}^*$:

$$\mathbf{a}^* = \operatorname*{argmin}_{\mathbf{a}} \left(\sum_{i=1}^n \left(\mathbf{a}^T \mathbf{x}_i - y_i^* \right)^2 + \alpha \|\mathbf{a}\|^2 \right) \qquad (3)$$

which can also be efficiently solved via some iterative algorithms (*e.g.*, LSQR [Paige and Saunders, 1982]). It is important to note that the first step becomes trivial in supervised settings and SR (called Spectral Regression Discriminant Analysis in supervised setting) provides an efficient solution for large scale discriminant analysis [Cai *et al.*, 2008]. Please see [Cai, 2009] for more details.

However in unsupervised settings, similar to all the other spectral dimensionality reduction methods, SR needs to construct a data affinity matrix which leads to $O(n^2)$ computational complexity. Moreover, when the data are highly nonlinear distributed, SR has to be performed in a reproducing kernel Hilbert space (leads to kernel SR) to learn the effective non-linear mapping.

In kernel SR, the linear regression step is replaced by the kernel regression [Cai, 2009]:

$$\min_{f \in \mathcal{F}} \sum_{i=1}^n \left(f(\mathbf{x}_i) - y_i^* \right)^2 + \delta \|f\|_K^2 \qquad (4)$$

where $\mathcal{F}$ is the RKHS associated with Mercer kernel $\mathcal{K}$ and $\| \ \|_K$ is the corresponding norm. By representer theorem [Wahba, 1990], the solution of the optimization problem (4) can be written as

$$f^*(\mathbf{x}) = \sum_{i=1}^n \alpha_i K(\mathbf{x}, \mathbf{x}_i)$$

where $K(\mathbf{x}, \mathbf{x}_i)$ is the kernel function of the corresponding Mercer kernel $\mathcal{K}$. Finally, the optimal $\boldsymbol{\alpha} = [\alpha_1, \alpha_2, \cdots, \alpha_n]^T$ can be computed as $\boldsymbol{\alpha}^* = (K + \delta I)^{-1} \mathbf{y}$, where K is the $n \times n$ kernel matrix.

It is clear that the computational complexity of kernel SR is $O(n^3)$. The high computational costs of the graph construction as well as the kernel approach restrict the applicability of SR for large scale nonlinear problems.

Compressed Spectral Regression

In this section, we introduce our *Compressed Spectral Regression* (CSR) method for large scale nonlinear dimensionality reduction. The basic idea of our approach is compressing the data using the sparse coding technique [Olshausen and Field, 1996]. With the sparse representation, we can construct a special graph with which the spectral embedding can be efficiently computed. Moreover, the sparse representation capture the nonlinear structure of the data and the learned projective function will be nonlinear with respect to the original features.

Data Compression via Landmark-based Sparse Coding

Sparse coding [Olshausen and Field, 1996; Lee *et al.*, 2006] is a matrix factorization technique which tries to "compress" the data by finding a set of *basis* vectors and the *representation* with respect to the basis for each data point. Let $X = [\mathbf{x}_1, \cdots, \mathbf{x}_n] \in \mathbb{R}^{m \times n}$ be the data matrix, sparse coding can be mathematically defined as finding two matrices $U \in \mathbb{R}^{m \times l}$ and $Z \in \mathbb{R}^{l \times n}$ by solving the optimization problem as follows:

$$\min_{U,Z} \|X - UZ\|^2 + \alpha f(Z) \qquad (5)$$

where f is a function measuring the sparsity of each column of Z (*e.g.*, l_1 norm) and α is a parameter controlling the sparsity penalty. Each column of U can be regarded as a basis

vector which captures the higher-level features in the data and each column of Z is the l-dimensional representation of the original inputs with respect to the new basis. Since each basis vector (column vector of U) can be regarded as a concept, a sparse matrix Z indicates that each data point is a combination of *several selected* concepts.

Comparing to some dense matrix factorization techniques (*e.g.*, singular value decomposition and nonnegative matrix factorization), sparse coding has several advantages for data representation. The most significant one is that it yields sparse representations such that each data point is represented as a linear combination of a small number of basis vectors. Thus, the data points can be interpreted in a more elegant way.

However, solving the optimization problem (5) is very time consuming. Most of the existing approaches [Lee *et al.*, 2006] compute U and Z iteratively. Apparently, these methods are not suitable for compressing the data efficiently. Following [Liu *et al.*, 2010; Chen and Cai, 2011], we use a much more simplified coding strategy by treating the basis vectors as the landmark points of the data set. A good set of landmarks should be able to cover the data set well. Thus, we use k-means clustering[1] to generate the landmarks (taking the cluster centers as the landmarks), which gives us the basis matrix U.

To compute the sparse representation matrix Z, we again use a simple yet efficient way. Recall that sparse coding essentially finds two matrices $U \in \mathbb{R}^{m\times l}$ and $Z \in \mathbb{R}^{l\times n}$, whose product can approximate $X \in \mathbb{R}^{m\times n}$,

$$X \approx UZ.$$

For any data point $\mathbf{x}_i$, its approximation $\hat{\mathbf{x}}_i$ can be written as

$$\hat{\mathbf{x}}_i = \sum_{j=1}^{l} z_{ji}\mathbf{u}_j \tag{6}$$

where $\mathbf{u}_j$ is j-th column vector of the basis matrix U and z_{ji} is ji-th element of Z. A natural assumption here is that z_{ji} should be larger if $\mathbf{u}_j$ is closer to $\mathbf{x}_i$. We can emphasize this assumption by setting the z_{ji} to zero as $\mathbf{u}_j$ is not among the r ($\leq l$) nearest neighbors (for all the column vectors of U) of $\mathbf{x}_i$. This restriction naturally leads to a sparse representation matrix Z [Chen and Cai, 2011].

Let $N_{\langle i\rangle}$ denote the index set which consists r indexes of r nearest landmarks of $\mathbf{x}_i$, *i.e.*, $\mathbf{u}_j$ is among the r nearest landmarks of $\mathbf{x}_i$ if $j \in N_{\langle i\rangle}$, we compute z_{ji} as

$$z_{ji} = \begin{cases} \frac{K(\mathbf{x}_i,\mathbf{u}_j)}{\sum_{j'\in N_{\langle i\rangle}} K(\mathbf{x}_i,\mathbf{u}_{j'})}, & j \in N_{\langle i\rangle} \\ 0, & j \notin N_{\langle i\rangle} \end{cases} \tag{7}$$

where $K(\cdot)$ is a kernel function. One can simply choose the most commonly used Gaussian kernel, $K(\mathbf{x}_i,\mathbf{u}_j) = \exp(-\|\mathbf{x}_i - \mathbf{u}_j\|^2/2\sigma^2)$.

[1]There is no need to wait the k-means converge and we can stop the k-means after t iterations, where t is a parameter (5 is usually enough). This will be discussed in the experiments.

Spectral Analysis on Landmark-based Graph

Now we have the compressed sparse representation $Z \in \mathbb{R}^{l\times n}$ for the input $X \in \mathbb{R}^{m\times n}$. Instead of constructing a k nearest neighbor graph to model the geometric structure of the data as most of the spectral dimensionality reduction algorithms [He and Niyogi, 2003; Cai, 2009] do, we simply compute the affinity matrix as

$$W = \hat{Z}^T\hat{Z}, \tag{8}$$

where $\hat{Z} = D^{-1/2}Z$ and D is a $l \times l$ diagonal matrix whose entries are the row sums of Z ($d_{ii} = \sum_j z_{ij}$).

The advantages of using this graph instead of a k-nearest neighbor graph are as follows:

1. Constructing a k-nearest neighbor graph requires $O(n^2)$ while computing W in Eq. (8) only needs $O(nl^2)$. Considering $l \ll n$, this is a significant efficiency boosting.
2. The top eigenvectors of W in Eq. (8) can be efficiently computed as we will discuss next.

Noticing that each column of Z sums up to 1, it is easy to check that

$$\begin{aligned}\sum_{j=1}^{n} w_{ij} &= \sum_{j=1}^{n}\sum_{p=1}^{l} \hat{z}_{pi}\hat{z}_{pj} = \sum_{j=1}^{n}\sum_{p=1}^{l} \frac{z_{pi}z_{pj}}{d_{pp}} \\ &= \sum_{p=1}^{l} z_{pi}\frac{\sum_{j=1}^{n} z_{pj}}{d_{pp}} = \sum_{p=1}^{l} z_{pi} = 1.\end{aligned}$$

Thus the degree matrix of W in Eq. (8) is I, *i.e.*, the affinity matrix W is normalized [Chung, 1997].

Directly computing the top eigenvectors of $W \in \mathbb{R}^{n\times n}$ requires $O(n^2)$ time. By noticing the special structure of W in Eq. (8), we can use an efficient way as follows. Let the Singular Value Decomposition (SVD) of $\hat{Z}$ is as follows:

$$\hat{Z} = A\Sigma B^T,$$

where $\Sigma = diag(\sigma_1, \cdots, \sigma_l)$ and $\sigma_1 \geq \sigma_2 \geq \cdots \geq \sigma_l \geq 0$ are the singular values, $A \in \mathbb{R}^{l\times l}$ and $B \in \mathbb{R}^{n\times l}$ are the left and right singular vector matrices. It is easy to check that the column vectors of B are the eigenvectors[2] of matrix $W = \hat{Z}^T\hat{Z}$ and the column vectors of A are the eigenvectors of matrix $\hat{Z}\hat{Z}^T$. Since the size of matrix $\hat{Z}\hat{Z}^T$ is $l \times l$, we can compute A within $O(l^3)$ time. B can then be computed as

$$B^T = \Sigma^{-1}A^T\hat{Z} \tag{9}$$

The overall time is $O(l^3 + l^2n)$, which is a significant reduction from $O(n^3)$ considering $l \ll n$.

Regression with Landmark-based Sparse Representation

The eigenvectors of W in Eq. (8) (the column vectors of B) can be regarded as an approximation of $\mathbf{y}^*$ in the optimization problem (1) with a k-nearest neighbor graph. We then

[2]It is not hard to verify the the first column of B (*i.e.*, the eigenvector of W corresponding to the largest eigenvalue) is a vector with all ones (without normalization). Thus, we removed this column in real implementation.

follow the idea of spectral regression [Cai, 2009] to learn the projective functions.

For each column vector $\mathbf{b}$ in B, we compute the corresponding projective functions $\mathbf{p}^* \in \mathbb{R}^l$ by solving the regression problem as follows:

$$\mathbf{p}^* = \underset{\mathbf{p}}{\operatorname{argmin}} \left(\sum_{i=1}^{n} \left(\mathbf{p}^T \mathbf{z}_i - b_i \right)^2 + \alpha \|\mathbf{p}\|^2 \right) \tag{10}$$

where b_i is the i-th element in $\mathbf{b}$ and α is the ridge regularization parameter.

It is easy to verify that the optimal $\mathbf{p}^*$ has the close form solution as follows:

$$\mathbf{p}^* = (ZZ^T + \alpha I)^{-1} Z\mathbf{b} \tag{11}$$

If we want to reduce the original data into a c-dimensional subspace, the optimal projective function $P \in \mathbb{R}^{l \times c}$ can be computed as:

$$P = (ZZ^T + \alpha I)^{-1} ZB_{(c)} \tag{12}$$

where $B_{(c)}$ is the first c columns of B.

The proposed method first compresses the data using landmark-based sparse coding; then performs the spectral analysis on the landmark-based graph; finally, the regression is used to learn the projective functions. Thus, we name this approach *Compressed Spectral Regression* (CSR).

The final learned CSR model consists two parts: the landmark matrix $U \in \mathbb{R}^{m \times l}$ and the projective function matrix $P \in \mathbb{R}^{l \times c}$. Given a data point $\mathbf{x} \in \mathbb{R}^m$, CSR model first learns its l-dimensional landmark-base sparse representation $\mathbf{z}$ using Eq. (7), then output the c-dimensional reduced representation as $\hat{\mathbf{x}} = P^T \mathbf{z}$. The computational procedure of CSR is summarized in Algorithm 1.

It is important to note that CSR is a nonlinear dimensionality reduction method. The mapping from $\mathbf{x}$ to $\mathbf{z}$ is nonlinear and the mapping from $\mathbf{z}$ to $\hat{\mathbf{x}}$ is linear. Thus, the overall mapping from $\mathbf{x}$ to $\hat{\mathbf{x}}$ is nonlinear.

Computational Complexity Analysis

Given n data points with dimensionality m, the computational complexity of CSR is as follows:

1. $O(tlnm)$: k-means with t iterations to select l landmarks.
2. $O(lnm)$: learn the landmark-based sparse representation Z.
3. $O(l^3 + l^2 n)$: compute the eigenvectors of W in Eq. (8).
4. $O(l^2 n + l^3 + lnc)$: compute the projection matrix P in Eq. (12).

Considering $l \leq n$, the overall complexity of CSR is $O(mn)$ which is linear with respect to number of training samples.

All the existing spectral dimensionality reduction algorithms (*e.g.*, LPP [He and Niyogi, 2003], SR [Cai, 2009], Laplacian Eigenmap [Belkin and Niyogi, 2001], LLE [Roweis and Saul, 2000], ISOMAP [Tenenbaum *et al.*, 2000]) require to construct the affinity graph which is $O(mn^2)$. Thus, our CSR has a significant computational advantage.

Algorithm 1 Compressed Spectral Regerssion

Input:

n data points $\mathbf{x}_1, \mathbf{x}_2, \ldots, \mathbf{x}_n \in \mathbb{R}^m$; Reduced dimensionality c; Parameters l, t, r, α.

1: Produce l landmark points using k-means with t iterations, leading to the basis matrix U;
2: Learn the landmark-based sparse representation matrix $Z \in \mathbb{R}^{l \times n}$ according to Eq. (7);
3: Compute the first $c+1$ eigenvectors of $\hat{Z}\hat{Z}^T$, denoted by A;
4: Compute B according to Eq. (9) and remove the first column of B;
5: Compute the projection matrix P according to Eq. (12).

Output:

The model: basis matrix U and the projection matrix P;
The dimension reduced data, $\hat{X} = P^T Z \in \mathbb{R}^{c \times n}$.

Experiments

In this section, we conduct the clustering experiment on the MNIST (handwritten digits image) data set to demonstrate the effectiveness of the proposed Compressed Spectral Regression (CSR) approach. Each image is represented as a 784 dimensional vector. It has 60000 training images and 10000 test images.

Compared Methods

The compared methods are listed as follows:

- Principle Component Analysis (**PCA**) [Duda *et al.*, 2000] and Kernel Principle Component Analysis (**KPCA**) [Scholkopf *et al.*, 1998].
- Locality Preserving Projection (**LPP**) and Kernel Locality Preserving Projection (**KLPP**) [He and Niyogi, 2003].
- Spectral Regression (**SR**) and Kernel Spectral Regression (**KSR**) [Cai, 2009].
- Laplacian **Eigenmap** [Belkin and Niyogi, 2001].
- Compressed Spectral Regression (**CSR**) in this paper.

Given a sample set with k clusters, we use all the above methods to reduce the dimensionality to k. Then the k-means is applied on the reduced subspace. All the methods learn the projective function on the training data and embed the test data using the learned function. The Eigenmap can only provide the embedding of the training data, thus, we do not report its performance on test data.

We use the same 5-nearest neighbor graph in LPP (KLPP), SR (KSR) and Eigenmap. For CSR, we empirically set the parameters $l = 1000$ (# landmarks), $t = 5$ (# iterations in k-means), $r = 5$ (# nearest landmarks) and $\alpha = 0.01$ (regression regularization). We use the same Gaussian kernel for all the kernel methods and our CSR in Eq. (7). The kernel width parameter σ is empirically[3] set to be 10.

[3] we estimate σ by randomly choose 3000 samples and let σ equal to the average of the pair-wise distances.

Table 1: Clustering results on the training set (%)

k	Baseline	PCA	KPCA	LPP	KLPP	SR	KSR	CSR	Eigenmap
2	62.7±20.9	64.4±16.7	65.1±16.8	76.1±14.3	**92.6±6.3**	87.1±8.2	**94.1±5.2**	**92.6±6.0**	**94.0±5.1**
3	56.9±16.0	55.6±14.7	58.2±15.5	66.0±17.6	−*	74.7±17.7	**82.8±17.2**	**82.8±14.9**	**85.3±14.1**
4	52.5±10.1	53.2±10.2	−*	55.7±10.1	−*	69.4±9.9	−*	**80.1±10.5**	**82.1±10.6**
5	56.5±8.3	56.0±9.5	−*	60.2±7.9	−*	71.2±10.2	−*	**85.3±8.3**	**86.8±9.5**
6	51.5±7.8	53.1±8.4	−*	53.2±7.1	−*	64.9±7.4	−*	**79.1±7.5**	**81.6±6.9**
7	50.4±5.2	52.0±5.5	−*	53.1±4.3	−*	61.7±4.9	−*	**76.3±6.1**	**79.5±5.0**
8	51.6±4.8	51.6±4.8	−*	53.7±3.3	−*	60.6±5.2	−*	**76.4±5.0**	**79.2±4.5**
9	48.8±3.7	49.7±3.8	−*	50.8±1.6	−*	59.4±3.4	−*	**75.5±3.0**	**78.0±2.8**
10	51.5	49.0	−*	49.6	−*	58.0	−*	**75.6**	**78.2**
Avg.	53.6	53.8		57.6		67.4		**80.4**	**82.7**

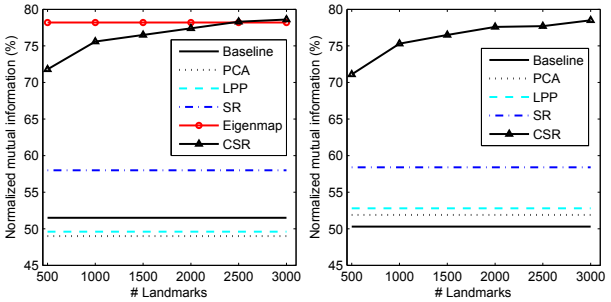
*The kernel methods can not be applied due to the memory limitation

Table 2: Subspace learning time (s)

k	PCA	KPCA	LPP	KLPP	SR	KSR	CSR	Eigenmap
2	**0.4**	18.7	32.9	165.2	41.0	78.3	**4.0**	40.3
3	**0.6**	43.0	79.2	−*	97.0	210.8	**5.6**	96.1
4	**0.7**	−*	146.7	−*	173.2	−*	**6.9**	172.1
5	**0.8**	−*	245.3	−*	281.7	−*	**8.5**	280.3
6	**0.9**	−*	365.0	−*	411.7	−*	**9.8**	409.9
7	**1.1**	−*	514.5	−*	572.5	−*	**11.3**	570.5
8	**1.2**	−*	668.8	−*	737.5	−*	**12.7**	735.1
9	**1.3**	−*	847.3	−*	924.0	−*	**14.2**	921.3
10	**1.6**	−*	1054.8	−*	1144.3	−*	**15.7**	1141.4
Avg	**1.0**		439.4		487.0		**9.9**	485.2

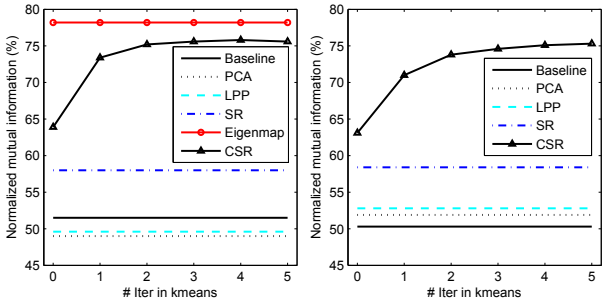
Table 3: Clustering results on the test set (%)

k	Baseline	PCA	KPCA	LPP	KLPP	SR	KSR	CSR
2	66.9±13.5	67.1±13.5	67.6±13.4	76.0±12.6	**92.7±5.8**	86.0±8.1	**94.2±4.7**	**93.4±5.5**
3	60.7±16.0	57.0±14.9	59.8±16.0	67.5±16.8	−*	74.4±17.4	**82.6±17.1**	**83.5±14.7**
4	55.5±10.2	55.5±9.9	−*	56.8±11.3	−*	68.9±9.8	−*	**80.9±10.4**
5	57.0±7.3	59.2±9.0	−*	61.2±7.3	−*	70.1±9.1	−*	**86.2±8.1**
6	55.1±7.7	54.5±7.7	−*	55.1±6.8	−*	65.1±7.1	−*	**80.3±7.3**
7	52.2±4.8	50.6±5.0	−*	53.6±4.5	−*	62.1±4.7	−*	**77.4±6.0**
8	53.1±4.0	52.4±4.5	−*	54.4±3.2	−*	60.9±5.2	−*	**77.7±4.8**
9	51.1±3.7	50.9±3.1	−*	52.4±1.8	−*	59.8±3.4	−*	**75.5±4.3**
10	50.3	51.9	−*	52.8	−*	58.4	−*	**75.3**
Avg	55.8	55.5		58.9		67.3		**81.2**



(a) Training set (b) Test set

Figure 1: The performance of CSR vs. # of landmarks



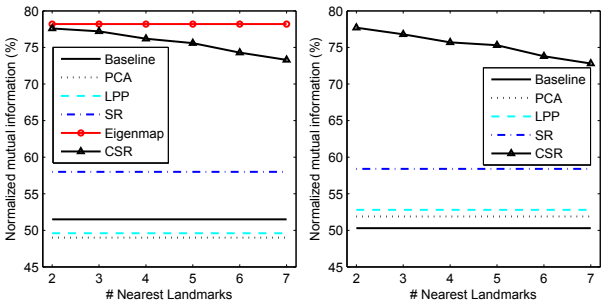
(a) Training set (b) Test set

Figure 2: The performance of CSR vs. # of iterations in kmeans

Evaluation Metric

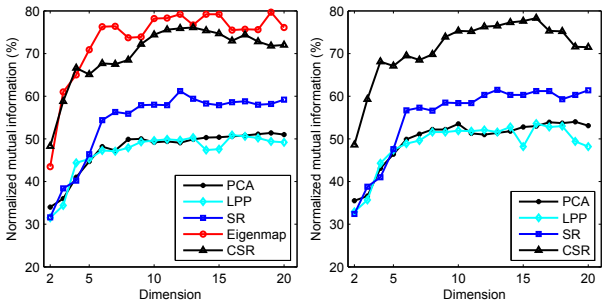
The clustering result is evaluated by comparing the obtained label of each sample with the label provided by the data set. We use the normalized mutual information (NMI) metric to measure the clustering performance. The NMI ranges from 0 to 1 and a higher value indicates a better clustering result. Please see [Cai *et al.*, 2005] for a detailed description on NMI.

We also record the running time of each method. All the codes in the experiments are implemented in MATLAB R2011b and run on a Windows 7 machine with 3.40 GHz i7-2600K CPU, 16GB main memory.



(a) Training set (b) Test set

Figure 3: The performance of CSR vs. # of nearest landmarks.



(a) Training set (b) Test set

Figure 4: The performance of different methods vs. # of reduced dimensions.

Table 4: The performance of CSR (NMI %) vs. the regularization parameter α

α	0	0.01	0.1	1	10	100
Training Set	75.6	75.6	75.6	75.6	75.9	75.7
Test Set	75.2	75.3	75.3	75.2	74.8	75.2

Clustering Results

Table 1, 2 and 3 show the clustering performance on training set, the subspace learning time and the performance on test set, respectively. The evaluations were conducted with the cluster numbers ranging from two to ten. For each given cluster number k (except $k = 10$), 20 test runs were conducted on different randomly chosen clusters and the average performance as well as the standard deviation are reported in the tables. These results reveals a number of interesting points as follows:

- Comparing to the baseline (k-means in the original feature space), PCA and KPCA do not show any significant improvement while all the other methods do. This demonstrates the effectiveness of the spectral dimensionality reduction methods. The nonlinear spectral methods (KLPP, KSR, CSR and Eigenmap) are significantly better than the linear spectral methods (LPP and SR), especially when the cluster number is large. This probably due to that the data are non-linear distributed. However, due to the memory limitation, all the kernel methods can not be applied when k is larger then 3. The proposed CSR approach achieves the similar performance as the Eigenmap does.

- Considering the learning time, our CSR method is linear with respect to the number of samples thus it is very efficient. It only needs 15.7 seconds learning with 10 classes (60000 samples). While all the other spectral methods need to build the affinity graph and need more than 1000 seconds on the same data set.

- Moreover, our CSR model learns the projective function which is defined everywhere (on both training samples and test samples). While the Eigenmap can only provide the embedding results of the training data.

Parameters Selection

Our CSR model has four parameters, # of landmarks (l), # of iterations (t) in k-means to generate the landmarks, # of nearest landmarks (r) for sparse representation and the ridge regularization parameter (α). In this subsection, we will examine how the performance of CSR influenced by these parameters. All the remaining experiments are conducted on the entire MNIST data set (10 classes, 60000 training samples and 10000 test samples). The default settings for CSR are $l = 1000$, $t = 5$, $r = 5$ and $\alpha = 0.01$. When we study the impact of one parameter, the other parameters are fixed as the default.

Figure (1) shows how the performance of CSR changes as the number of landmarks varies. As the number of landmarks increases, both the performance and the learning time of CSR increase. As we use 3000 landmarks, CSR achieves almost the same performance as Eigenmap does while CSR only needs 44.4 seconds which is much less than the learning time of Eigenmap (1141.4 seconds).

Figure (2) shows how the performance of CSR changes as the number of iterations (t) in k-means varies. $t = 0$ indicates we use random sampling to select the landmarks. As the number of iterations increases, it is reasonable to see that both the performance and the learning time of CSR increase. Even with random landmark selection, CSR achieves better performance than LPP and SR.

Figure (3) shows how the performance of CSR changes as the number of nearest landmarks (r) for sparse representation varies. As the number of nearest landmarks increases, the performance of CSR consistently decreases. The best performance is achieved when $r = 2$. Our CSR model is very stable with respect to the regularization parameter α as shown in Table (4).

All the dimensionality reduction algorithms have a critical parameter: the number of reduced dimensions. Figure (4) shows how the performances of different methods vary as the number of reduced dimensions changes. All the algorithms achieve reasonable stable performance as the number of reduced dimensions is between 8 and 20.

Conclusion

In this paper, we have presented a novel efficient nonlinear spectral dimensionality reduction algorithm, called *Compressed Spectral Regression* (CSR). CSR has $O(n)$ computational complexity where n is the number of samples, which is a significant improvement over all the other spectral methods (require at least $O(n^2)$). Extensive experiments show the effectiveness and efficiency of our proposed approach.

Acknowledgments

This work was supported in part by the National Basic Research Program of China(973 Program) under Grant 2013CB336500 and National Nature Science Foundation of China (Grant Nos: 61222207, 91120302).

References

[Belkin and Niyogi, 2001] M. Belkin and P. Niyogi. Laplacian eigenmaps and spectral techniques for embedding and clustering. In *Advances in Neural Information Processing Systems 14*. 2001.

[Bengio *et al.*, 2003] Y. Bengio, J.-F. Paiement, P. Vincent, O. Delalleau, N. Le Roux, and M. Ouimet. Out-of-sample extensions for LLE, isomap, MDS, eigenmaps, and spectral clustering. In *Advances in Neural Information Processing Systems 16*, 2003.

[Cai *et al.*, 2005] Deng Cai, Xiaofei He, and Jiawei Han. Document clustering using locality preserving indexing. *IEEE Transactions on Knowledge and Data Engineering*, 17(12):1624–1637, 2005.

[Cai *et al.*, 2007] Deng Cai, Xiaofei He, and Jiawei Han. Spectral regression for efficient regularized subspace learning. In *IEEE 11th International Conference on Computer Vision*, 2007.

[Cai *et al.*, 2008] Deng Cai, Xiaofei He, and Jiawei Han. SRDA: An efficient algorithm for large scale discriminant analysis. *IEEE Transactions on Knowledge and Data Engineering*, 20(1):1–12, 2008.

[Cai, 2009] Deng Cai. *Spectral Regression: A Regression Framework for Efficient Regularized Subspace Learning*. PhD thesis, Department of Computer Science, University of Illinois at Urbana-Champaign, May 2009.

[Chen and Cai, 2011] Xinlei Chen and Deng Cai. Large scale spectral clustering with landmark-based representation. In *Proceedings of the Twenty-Fifth Conference on Artificial Intelligence*, 2011.

[Chung, 1997] Fan R. K. Chung. *Spectral Graph Theory*, volume 92 of *Regional Conference Series in Mathematics*. AMS, 1997.

[Duda *et al.*, 2000] R. O. Duda, P. E. Hart, and D. G. Stork. *Pattern Classification*. Wiley-Interscience, Hoboken, NJ, 2nd edition, 2000.

[He and Niyogi, 2003] Xiaofei He and Partha Niyogi. Locality preserving projections. In *Advances in Neural Information Processing Systems 16*. 2003.

[He *et al.*, 2005] Xiaofei He, Deng Cai, Shuicheng Yan, and Hong-Jiang Zhang. Neighborhood preserving embedding. In *Proceedings of the 10th IEEE International Conference on Computer Vision*, pages 1208–1213, 2005.

[Lee *et al.*, 2006] Honglak Lee, Alexis Battle, Rajat Raina, , and Andrew Y. Ng. Efficient sparse coding algorithms. In *Advances in Neural Information Processing Systems 19*. MIT Press, Cambridge, MA, 2006.

[Liu *et al.*, 2010] Wei Liu, Junfeng He, and Shih-Fu Chang. Large graph construction for scalable semi-supervised learning. In *Proc. 2010 Int. Conf. Machine Learning (ICML'10)*, 2010.

[Olshausen and Field, 1996] Bruno A. Olshausen and David J. Field. Emergence of simple-cell receptive field properties by learning a sparse code for natural images. *Nature*, 381(6583):607–609, 1996.

[Paige and Saunders, 1982] Christopher C. Paige and Michael A. Saunders. Algorithm 583 LSQR: Sparse linear equations and least squares problems. *ACM Transactions on Mathematical Software*, 8(2):195–209, June 1982.

[Roweis and Saul, 2000] Sam Roweis and Lawrence Saul. Nonlinear dimensionality reduction by locally linear embedding. *Science*, 290(5500):2323–2326, 2000.

[Scholkopf *et al.*, 1998] B. Scholkopf, A. Smola, and K. Muller. Nonlinear component analysis as a kernel eigenvalue problem. *Neural Computation*, 10:1299–1319, 1998.

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

[Wahba, 1990] G. Wahba. *Spline Models for Observational Data*. Society for Industrial and Applied Mathematics, 1990.

[Yan *et al.*, 2007] Shuicheng Yan, Dong Xu, Benyu Zhang, Hong-Jiang Zhang, Qiang Yang, and Steve Lin. Graph embedding and extension: A general framework for dimensionality reduction. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 29(1):40–51, 2007.