Deep Linear Coding for Fast Graph Clustering*

Ming Shao, Sheng Li, Zhengming Ding
Department of ECE
Northeastern University
Boston, MA 02115, USA
{mingshao,shengli,allanding}@ece.neu.edu

Yun Fu
Department of ECE, College of CIS
Northeastern University
Boston, MA 02115, USA
{yunfu}@ece.neu.edu

Abstract

Clustering has been one of the most critical unsupervised learning techniques that has been widely applied in data mining problems. As one of its branches, graph clustering enjoys its popularity due to its appealing performance and strong theoretical supports. However, the eigen-decomposition problems involved are computationally expensive. In this paper, we propose a deep structure with a linear coder as the building block for fast graph clustering, called Deep Linear Coding (DLC). Different from conventional coding schemes, we jointly learn the feature transform function and discriminative codings, and guarantee that the learned codes are robust in spite of local distortions. In addition, we use the proposed linear coders as the building blocks to formulate a deep structure to further refine features in a layerwise fashion. Extensive experiments on clustering tasks demonstrate that our method performs well in terms of both time complexity and clustering accuracy. On a large-scale benchmark dataset (580K), our method runs 1500 times faster than the original spectral clustering.

1 Introduction

Clustering algorithms [Jain *et al.*, 1999] have been widely applied in AI problems: pattern recognition, computer vision, information retrieval, bioinformatics. Like other unsupervised learning algorithms [Barlow, 1989], it is purely data-driven, and therefore, requests least supervised knowledge. There are a wide variety of clustering methods that target at different problems or application scenarios [Jain *et al.*, 1999]. The most popular ones are K-means [Hartigan and Wong, 1979] and spectral clustering [Ng *et al.*, 2002; Von Luxburg, 2007]. K-means keeps updating the cluster centers by current cluster assignments and then updating the assignments by the nearest neighbor rule. Although it only ends up with local solutions, its fast implementation makes it a fundamental tool in data analysis. However, K-means suffers from strong assumptions on the data, and local optimum.

Differently, spectral clustering has a solid theoretical foundation [Chung, 1997], and is able to reach a global solution. Besides, it has close relations with other methods such as graph cut [Shi and Malik, 2000], random walk [Meila and Shi, 2001], and thus is more intuitive and understandable. In addition, the non-parametric property and many fast implementations make it very suitable for large-scale data analysis under complex distributions [Fowlkes *et al.*, 2004]. However, the time-consuming graph construction and eigen-decomposition make it still inferior to K-means in terms of speed. Note there are also variations of K-means such as kernel K-means that can identify non-linear structure; however, they are proved equal to spectral clustering under mild conditions [Dhillon *et al.*, 2004].

Recently, deep structure has been widely applied in learning problems [Hinton *et al.*, 2006; Bengio, 2009] as it can extract better representations for learning tasks [Bengio *et al.*, 2013]. It is especially useful for clustering tasks since it can disentangle explanatory factors in a low-dimensional non-linear feature space in unsupervised fashion. In addition, the greedy layerwise training strategy [Bengio *et al.*, 2007] and improvement of computing hardware make the fast implementation of deep methods possible [Dean *et al.*, 2012]. Notably, there are a few deep methods proposed recently for clustering tasks [Song *et al.*, 2013; Huang *et al.*, 2014; Tian *et al.*, 2014]. In this paper, we propose a novel Deep Linear Coding (DLC) scheme for fast data clustering. Inspired by the common objective of K-means and spectral clustering methods, we jointly learn a linear feature transform and codings for the test data under single-layer marginalized denoising autoencoder framework, which can minimize the reconstruction error of the data/similarity graph. In addition, we propose to refine the learned representations by feeding them to the next layers and repeat the joint learning procedure. Experimental results on clustering tasks demonstrate the superiority of our method in terms of speed and accuracy.

2 Related Work

Spectral clustering utilizes both pairwise similarity graph and spectral theory to cut the graph with the minimal loss, and therefore is usually referred as graph clustering. The advantage over conventional K-means is the locality awareness due

*This research is supported in part by the NSF CNS award 1314484, ONR award N00014-12-1-1028, ONR Young Investigator Award N00014-14-1-0484, and U.S. Army Research Office Young Investigator Award W911NF-14-1-0218.

to the non-parametric formulation of graph. In the original formulation of spectral clustering, a pairwise sparse graph is built first based on the kNN rule, where non-zero values in the graph are Gaussian similarities between two samples. Then the graph is normalized and its first several smallest eigenvectors's row space is considered as the new representations in the embedding space. Finally K-means is implemented on the new representations. Although it can identify the Non-Gaussian data distributions, the graph construction and followed eigen-decomposition are not cheap, which have the time complexity of $\mathcal{O}(kn^2)$ and $\mathcal{O}(n^3)$, respectively, where k is the number of neighbors, and n is the number of samples.

A straightforward way to speed up the problem above is to exploit fast kNN implementation and sparse eigen-decomposition [Chen *et al.*, 2011]. Differently, Nystrom method [Fowlkes *et al.*, 2004] samples a few data, and builds a dense graph containing the connections only between sampled data and all data. Then eigenvectors of the original graph can be approximated by this substantially small graph. Recently, landmarks based methods either build a landmarks-only graph [Yan *et al.*, 2009] or a factor matrix of the original graph [Chen and Cai, 2011], and significantly save the running time of eigen-decomposition. However, none of them can get rid of eigen-decomposition, and some of them may even sacrifice certain performance due to the approximation operations [Yan *et al.*, 2009]. Different from them, our method enjoys the formulation of spectral clustering, but excludes the time-consuming eigen-decomposition operations.

Recently, deep learning methods have been applied on data clustering. One of the pioneer work adds the pairwise constraint between data samples and current affiliated centers to the loss function of the deep autoencoders [Song *et al.*, 2013]. A more recent work uses the deep sparse autoencoders to explore the representations of the normalized similarity graph, and proves that the deep model works similarly to the spectral clustering on the graph reconstruction [Tian *et al.*, 2014]. Finally, both locality constraint and group sparsity are introduced to RBMs for the purpose of clustering [Huang *et al.*, 2014]. However, all these methods do not explicitly consider the efficiency, especially for large-scale data. Thus, they may suffer from extremely long running time in practice.

3 Methodology

3.1 Background and Motivations

The common idea of K-means and spectral methods is to reconstruct the data/graph and use the indicator vector/matrix of the reconstruction as the new feature for clustering:

$$\min_{\mathbf{A}} \|\mathbf{X} - \Phi(\Sigma, \mathbf{A})\|_F^2, \tag{1}$$

where $\mathbf{X}$ is the data samples (K-means) or normalized affinity graph (spectral clustering), Φ is the reconstruction function, Σ is approximate cluster centers (K-means), or singular values diagonal matrix (spectral clustering), and $\mathbf{A}$ is the indicator vector/matrix. In K-means, the objective function essentially minimize the average square distance between samples and their affiliated center, where indicators play the roles of reconstruction coefficients. On the other hand, spectral clustering manages to find a low-rank approximation of the normalized affinity graph, and the eigenvectors are the indicators.

In our model, we also propose to find a new representation of the data with minimal reconstruction loss that can be used as the indicators for clustering. Recently, linear coding has been widely used in high level feature extraction and data representation [Lee *et al.*, 2006; Yang *et al.*, 2009]. Specifically, it has been adapted in the decomposition of affinity graph for fast spectral clustering [Chen and Cai, 2011]. Typically, the objective function of linear coding aims at minimizing the following reconstruction loss:

$$\min_{\mathbf{D},\mathbf{A}} \|\mathbf{X} - \mathbf{D}\mathbf{A}\|_F^2 + \mathscr{R}(\mathbf{A}), \tag{2}$$

where $\mathbf{D}$ is the dictionary, $\mathbf{A}$ is the learned codes, and $\mathscr{R}(\mathbf{A})$ is a regularizer that encourages small magnitude or sparse solutions of $\mathbf{A}$. A popular choice of $\mathscr{R}(\mathbf{A})$ is l_1 norm that induces sparsity on the feature vectors. However, for data clustering, especially large-scale problems, the weakness is obvious: 1) the solution to Eq. (2) is computationally expensive due to the non-smooth property of the objective function; 2) the sparse kernel approximation introduced in [Chen and Cai, 2011] trades the approximation accuracy for speed.

In the proposed model, we manage to address the two problems above, and provide a fast coding scheme appropriate for large-scale data clustering. We introduce a linear transform matrix $\mathbf{W}$ for the coding part to compensate for the large reconstruction loss between $\mathbf{X}$ and $\mathbf{DA}$ due to the induced sparsity. In addition, our method is still able to provide low-rank approximation of the affinity graph without time consuming eigen-decomposition, which significantly saves running time. Next, we detail the single-layer linear coding scheme.

3.2 Method Details

Given a set of data $\mathbf{X} \in \mathbb{R}^{d\times n}$, where d is the dimensionality of the data and n is the number of data samples, the proposed Single-layer Linear Coding (SLC) scheme can be written as:

$$\min_{\mathbf{W},\mathbf{A}} \|\mathbf{X} - \mathbf{W}\mathbf{D}\mathbf{A}\|_F^2 + \lambda \|\mathbf{A}\|_F^2, \tag{3}$$

where $\mathbf{W} \in \mathbb{R}^{d\times d}$ is the linear transform function, $\mathbf{D} \in \mathbb{R}^{d\times m}$ is the dictionary, $\mathbf{A} \in \mathbb{R}^{m\times n}$ is the codes for dataset $\mathbf{X}$, λ is the weight of the regularizer, and in most cases $n > m, n \gg d$. In our problem, $\mathbf{D}$ is also referred as a set of m landmarks, and fixed during the optimization. Therefore, the proposed model manages to solve two variables at the same time; however, to the best of our knowledge, such problem does not have a closed-form solution. In fact, the objective in Eq. (3) is convex in either $\mathbf{W}$ or $\mathbf{A}$, but not in both. Therefore, we resort to an iterative solution that is able to solve one unknown variable at a time by fixing the other.

First, suppose we have already obtained a dictionary $\mathbf{D}$ and corresponding codes $\mathbf{A}$ for each data sample in $\mathbf{X}$, then the original SLC model is converted to:

$$\min_{\mathbf{W}} \|\mathbf{X} - \mathbf{W}\tilde{\mathbf{X}}\|_F^2, \tag{4}$$

where $\tilde{\mathbf{X}} = \mathbf{DA}$ represents the reconstructed data samples through learned dictionary and codes. Intuitively, Eq. (4)

```
Input: Dataset $\mathbf{X}$, number of landmarks $m$, number of
       nearest neighbor $k$, threshold $\epsilon$.
Output: Codes $\mathbf{A}$ for $\mathbf{X}$
1 Find $m$ landmarks of $\mathbf{X}$ by K-means, and assign to $\mathbf{D}$.
2 Compute the sparse codings by kernel approximations:
  $[\mathbf{A}]_{ij}(1 \le i \le m, 1 \le j \le n) = \frac{\phi(\mathbf{x}_i,\mathbf{x}_j)}{\sum_u \phi(\mathbf{x}_u,\mathbf{x}_j)}$, $\mathbf{x}_u$ is
  within the first $k$ nearest neighbors of $\mathbf{x}_j$ in $\mathbf{D}$,
  $\mathbf{x}_i \in \{\mathbf{x}_u\}$, and $\phi(\cdot,\cdot)$ is the Gaussian kernel with
  bandwidth $\sigma$.
3 for $t = 1$ to $T$ do
4     Fix dictionary $\mathbf{D}$ and codes $\mathbf{A}$, and update linear
      transform $\mathbf{W}$ by Eq. (4).
5     Fix linear transform $\mathbf{W}$ and dictionary $\mathbf{D}$, and
      update codes $\mathbf{A}$ by Eq. (6).
6     If $\|\mathbf{X} - \mathbf{WDA}\|_F^2 < \epsilon$, terminate the algorithm.
7 end
```

Algorithm 1: Algorithm of Single-layer Linear Coding.

manages to find a linear transform (rotation) by which the reconstructed data $\tilde{\mathbf{X}}$ can match the original data, despite of the limit basis in $\mathbf{D}$ and sparse coefficients from $\mathbf{A}$. Obviously, this problem can be converted to ordinary least square problems [Hastie *et al.*, 2009] with closed-form solutions as:

$$\mathbf{W} = \mathbf{P}\mathbf{Q}^{-1} \text{ where } \mathbf{P} = \mathbf{X}\tilde{\mathbf{X}}^\top, \mathbf{Q} = \tilde{\mathbf{X}}\tilde{\mathbf{X}}^\top, \tag{5}$$

Second, since we have already found the linear transform $\mathbf{W}$, we could achieve the new representation of the dictionary $\tilde{\mathbf{D}} = \mathbf{WD}$ and update the coding in the following way:

$$\min_{\mathbf{A}} \left\|\mathbf{X} - \tilde{\mathbf{D}}\mathbf{A}\right\|_F^2 + \lambda \|\mathbf{A}\|_F^2, \tag{6}$$

which is essentially a ridge regression problem, with λ as the balancing parameter. The closed-form solution for this problem is:

$$\mathbf{A} = (\tilde{\mathbf{D}}^\top\tilde{\mathbf{D}} + \lambda\mathbf{I})^{-\mathbf{1}}\tilde{\mathbf{D}}^\top\mathbf{X}, \tag{7}$$

where $\mathbf{I}$ is a $m \times m$ identity matrix. The introduced regularizer is able to suppress arbitrarily large coefficients in $\mathbf{A}$ and avoid over-fitting. The updated coding and dictionary can be used again to learn the new linear transform. We name this algorithm as Single-layer Linear Coding (SLC), which is summarized in Algorithm 1.

3.3 Relations to Existing Methods

Marginalized Denoising Autoencoder

People might be curious why the proposed SLC model is appropriate for feature learning and data clustering. Here we elaborate a few connections with existing methods, and therefore highlight the advantage of SLC.

Recall in the first step of SLC algorithm, we solve Eq. (4) to find a linear transform, which is able to reconstruct the original input by perturbed samples $\mathbf{DA}$. This is essentially a single-layer denoising autoencoder using $\mathbf{DA}$ as the corrupted data. Arbitrary noises can not guarantee a stable performance, but in general, more reasonable noisy samples yield a much better result. However, this will add tremendous computational burden when the noisy patterns go to infinity.

Recently, a marginalized denoising autoencoder has been proposed to solve the computation problem above [Chen *et al.*, 2012]. Instead of exhaustedly sampling contaminated data, they compute the empirical expectation over the original data $\mathbf{X}$, and find the linear transform in the following way:

$$\begin{aligned} &\mathbf{W} = \mathbb{E}[\mathbf{P}]\mathbb{E}[\mathbf{Q}]^{-1}, \text{ where} \\ &\mathbb{E}[\mathbf{P}] = \sum_i \mathbf{x}_i\mathbf{x}_i^\top \otimes \Lambda_{\mathbf{P}}, \mathbb{E}[\mathbf{Q}] = \sum_i \mathbf{x}_i\mathbf{x}_i^\top \otimes \Lambda_{\mathbf{Q}}, \end{aligned} \tag{8}$$

where $\Lambda_{\mathbf{P}}$ ($\Lambda_{\mathbf{Q}}$) is $d \times d$ matrix encoding joint survival ratios of two single features from $\mathbf{x}_i$ and $\tilde{\mathbf{x}}_i$ (both from $\tilde{\mathbf{x}}_i$), and $\otimes$ is the element-wise multiplication. Although the proposed SLC does not explicitly model the marginalization procedure, the following the theorem demonstrates that SLC is also a single-layer marginalized denoising autoencoder:

Theorem 1 *Problem stated in Eq.* (4) *is a single-layer marginalized denoising autoencoder, where factor matrix* $\mathbf{Q}$ *is the empirical expectation of the covariance matrix of landmarks up to a constant c, and* $\mathbf{P}$ *is the empirical expectation of the covariance matrix of landmarks up to a constant* $c\mathbf{W}$.

Proof. For $\mathbf{Q}$ we can easily check that

$$\mathbf{Q} = \mathbf{DAA}^\top\mathbf{D}^\top = \sum_i \mathbf{D}\mathbf{a}_i\mathbf{a}_i^\top\mathbf{D}^\top \approx c\mathbb{E}[\mathbf{DD}^\top], \tag{9}$$

where $\mathbf{a}_i$ is the i-th column vector in $\mathbf{A}$, and c is a constant compensating for different column sizes between $\mathbf{D}$ and $\mathbf{DA}$. Similar to Eq. (8), Eq. (9) is the empirical expectation of $\mathbf{DD}^\top$ with $\mathbf{AA}^\top$ encoding the joint survival probability of every two features. Different from Eq. (8) that uses different survival probability for features from the same data sample, $\mathbf{a}_i$ assigns the same probability to all features from a single data sample. For $\mathbf{P}$ we have the following deductions:

$$\mathbf{P} = \mathbf{XA}^\top\mathbf{D}^\top = \mathbf{WDAA}^\top\mathbf{D}^\top \approx c\mathbf{W}\mathbb{E}[\mathbf{DD}^\top], \tag{10}$$

where we take advantage of the fact that $\mathbf{X} \approx \mathbf{WDA}$ in the inner loop of Algorithm 1. Similar to $\mathbf{Q}$, $\mathbf{P}$ is the empirical expectation of covariance matrix of $\mathbf{D}$ up to a constant factor matrix $c\mathbf{W}$. This completes the proof. □

Spectral Clustering

From the reconstruction point of view, spectral clustering attempts to minimize the following graph approximation loss with a low-rank constraint:

$$\min_{\tilde{\mathbf{G}}} \|\tilde{\mathbf{G}} - \mathbf{G}\|_F^2, \text{ s.t. rank}(\tilde{\mathbf{G}}) \le r, \tag{11}$$

where $\tilde{\mathbf{G}}$ is the approximate graph, r is a predefined matrix rank. According to Eckart-Young-Mirsky theorem [Eckart and Young, 1936], its solution is $\tilde{\mathbf{G}} = \mathbf{U}\Sigma\mathbf{U}^\top$, where $\mathbf{U} \in \mathbb{R}^{n\times r}$ are the first r eigenvectors of graph $\mathbf{G}$, and Σ is a matrix with corresponding eigenvalues on the diagonal. Recall the loss function in Eq. (3), we also try to minimize the reconstruction error, but on the original data $\mathbf{X}$. Next proposition shows that the proposed SLC also seeks a low-rank approximation of the linear similarity graph $\mathbf{X}^\top\mathbf{X} \in \mathbb{R}^{n\times n}$.

Proposition 1 *The objective function in Eq.* (3) *seeks for a low-rank approximation of linear similarity graph* $\mathbf{X}^\top\mathbf{X}$ *in the sense of* $\mathbf{X}^\top\mathbf{X} \approx \mathbf{A}^\top\tilde{\mathbf{\Sigma}}\mathbf{A}$, *and rank*$(\mathbf{A}^\top\tilde{\mathbf{\Sigma}}\mathbf{A}) \leq \min\{m, d, n\}$, *where* $\tilde{\mathbf{\Sigma}} = \mathbf{D}^\top\mathbf{W}^\top\mathbf{W}\mathbf{D}$.

We can see that the solution of SLC is also a low-rank approximation of linear kernel similarity graph $\mathbf{X}^\top\mathbf{X}$, and its rank depends on the values of m, d and n. In most cases, n is the largest one, especially in the large-scale case. Therefore, SLC essentially provides a rank $(\leq \min\{d, m\})$ approximation of $\mathbf{X}^\top\mathbf{X}$, which works in a similar way as spectral clustering.

3.4 Deep Layerwise Feature Learning

Recent works on representation learning take advantage of deep structure to build discriminative features [Bengio *et al.*, 2013]. They usually use RBMs or autoencoders as the basic building blocks to learn a deep structure, and employ greedy layerwise training to obtain a better initializations for fine tuning. Finally either hidden layers or supervised layer on top will be used for classification tasks. The insight behind is deep structure can gradually refine the features from coarse to fine, and layerwise training helps to find better local solutions for the non-convex objective [Bengio *et al.*, 2007].

In the proposed SLC model, features can be refined in a similar layerwise way, meaning the learned features $\mathbf{A}$ can be considered as the input features of the second layer, and is able to build new landmarks. Suppose the l-th layer's coding is $\mathbf{A}_l$, dictionary is $\mathbf{D}_l$, and linear transform is $\mathbf{W}_l$, then we first update the dictionary in the $(l+1)$-th layer by $\mathbf{D}_{l+1} = \text{K-means}(\mathbf{A}_l)$. Second, we solve $\mathbf{W}_{l+1}$ by $\mathbf{W}_{l+1} = \Psi_{\mathbf{W}}(\mathbf{D}_{l+1}, \mathbf{A}_l)$, where $\Psi_{\mathbf{W}}$ indicates Eq. (5). Finally, we update the $(l+1)$-th layer's coding $\mathbf{A}_{l+1} = \Psi_{\mathbf{A}}(\mathbf{W}_{l+1}, \mathbf{D}_{l+1}, \mathbf{A}_l)$, where $\Psi_{\mathbf{A}}$ indicates Eq. (7). We summarize such layerwise feature learning procedure in Algorithm 2, and name it Deep Linear Coding (DLC) in this paper.

```
Input: Dataset $\mathbf{X}$, and the number of clusters.
Output: Codes $\mathbf{A}_L$ for dataset $\mathbf{X}$, and cluster labels of
        each sample.
1 for $l = 1$ to $L$ do
2     Use Algorithm 1 to learn $\mathbf{W}_l, \mathbf{D}_l$ and codes $\mathbf{A}_l$
3     Set $\mathbf{X}_{l+1} = \mathbf{A}_l$
4 end
5 Run K-means on the column space of $\mathbf{A}_L$, and obtain the
  cluster labels.
```

Algorithm 2: Algorithm of Deep Linear Coding (DLC).

3.5 Time Complexity

The time cost of DLC mainly includes four parts:

1. Find landmarks $\mathbf{D}$ through K-means,
2. Find k nearest neighbors from $\mathbf{D}$ for each sample,
3. Solve the ordinary least square problem in Eq. (4),
4. Solve the ridge regression problem in Eq. (6).

Table 1: Dataset details.

Name	Type	#Samples	#Features	#Classes
Corel	Image	2074	144	18
Coil20	Object	1440	1024	20
YaleB	Face	5850	1200	10
Pendigit	Number	10992	16	10
Letter	Letter	15000	16	26
Mnist	Number	70000	784	10
Covtype	Scientific	581012	54	7

First, K-means usually takes time of $\mathcal{O}(sdmn)$, where s is the number of iterations of K-means. In our experiments, we found that a small number of iterations, e.g., 10, will provide good results. Therefore, it is less expensive compared to conventional K-means algorithms for clustering purpose. Second, the brute-force way of kNN search usually takes time of $\mathcal{O}(kmn)$; however, many off-the-shelf algorithms can substantially speedup this process, e.g., FLANN library [Muja and Lowe, 2014]. Third, the ordinary least square problem includes a few matrix multiplications, and a matrix inverse operation, and the total running time is $\mathcal{O}(d^2(d+n))$. Finally, similar to the last step, the time complexity of ridge regression is $\mathcal{O}(m(2md + m^2 + dn))$. Therefore, the total time complexity for SLC is approximately equal to:

$$t_{\text{SLC}} = \mathcal{O}(T(mn(d + sd + k) + d^2n)), \tag{12}$$

where T is the number of iterations in SLC. Then, total time cost of DLC is: Lt_{SLC}, where L is the number of layers.

4 Experimental Results

In this section, we evaluate the proposed DLC method through data clustering on seven benchmark datasets in terms of both accuracy and running time. Dataset details are listed in Table 1.

4.1 Datasets and Configurations

Corel The dataset has been widely used in computer vision and image processing. We use its subset from [Chen *et al.*, 2011] as our test, where 2074 images, 144 features including shape, texture, color, are chosen for evaluations.

Coil20 An object image database with 20 different objects. Each of them has 72 images under different view point, with 5 degree as the interval. All images are resized to 32×32 pixels and we use the raw pixels as our features.

YaleB This database is popular in face recognition algorithms evaluations, including 38 people, 64 lighting conditions, and 9 poses. A subset of 10 people, 5850 faces are adopted for data clustering. 1200 features are extracted for each image.

Pendigit This is a handwritten digit data set including 10992 samples from 44 writers. A sampled coordination information (16 features) is taken for the feature.

Letter The dataset consists of 26 capital letters in the English alphabet and 16 character image features are selected for clustering task.

Mnist Another handwritten digits benchmark dataset widely used in clustering evaluations. We use raw pixels including 784 features in the evaluations.

Table 2: Comparisons with fast spectral clustering methods on accuracy. "Ours-L1" means the SLC method and "Ours-L2" means a two-layer DLC.

Acc(%)	Corel	Pendigit	Letter	Mnist	Covtype
Spectral	37.62	76.55	31.04	**72.46**	**44.24**
Nystrom	36.89	73.94	30.11	53.70	22.31
KASP	34.32	72.47	29.49	56.51	22.42
LSC-K	35.48	79.27	30.33	67.04	25.50
Ours-L1	39.41	79.34	33.98	65.34	43.11
Ours-L2	**40.28**	**80.18**	**35.15**	66.48	44.19

Table 3: Comparisons with deep clustering methods on accuracy(%), NMI, and running time(s).

Method	COIL20			YaleB		
	Acc	NMI	Time	Acc	NMI	Time
AEC	70.83	0.845	8.78	90.20	0.923	40.84
LDR	69.44	0.791	5.27	79.49	0.901	29.65
DEN	72.40	0.870	-	81.73	0.920	-
Ours-L1	71.04	0.853	**0.31**	92.17	0.924	**2.39**
Ours-L2	**73.61**	**0.884**	0.64	**94.47**	**0.941**	4.21

Covtype A large scale scientific dataset containing cartographic variables for predicting forest cover type. Each sample has 54 attributes, and in total there are over 580k samples.

Note for all datasets, each data sample is normalized to have unit length. We set the number of neighbors in kNN search at 5, and the number of landmarks in the first and second layers at 1000 unless otherwise specified. In addition, we set both the balancing parameter λ and Gaussian kernel bandwidth σ at 1. To balance the performance and speed, the number of iterations in each layer is set to $T = 5$. In all tests, we run our algorithm 20 times, and average results are reported. For all compared methods, we strictly follow their recommended configurations. Finally, we introduce two performance measurement in our evaluations: clustering accuracy and normalized mutual information (NMI).

Clustering accuracy is the average performance of label matching results between resulted labels and ground truth labels, which can formulated as: $\sum_i (y_i == f(l_i))/n$, where f is a mapping function that maps category label to different cluster labels. Since clustering is an unsupervised process, we need f to work as a permutation operation, and maximize this fraction as the final clustering accuracy.

Normalized mutual information (NMI) measures the mutual information entropy between resulted cluster labels and ground truth labels, followed by a normalization operation which guarantees that NMI ranges from 0 to 1. Mathematically, it can be written as:

$$\text{NMI} = \frac{\sum_i \sum_j n_{i,j} \log(\frac{m \cdot n_{i,j}}{n_i \cdot n_j})}{\sqrt{(\sum_i n_i \log \frac{n_i}{m})(\sum_j n_j \log \frac{n_j}{m})}}, \tag{13}$$

where n_i and n_j denotes the number of data in cluster i and category j, and $n_{i,j}$ denotes the number of data in both cluster i and category j. Therefore, if the data are randomly partitioned, NMI is inclined to 0.

Table 4: Comparisons with fast spectral clustering methods on running time. "Ours-L1" means the SLC method and "Ours-L2" means a two-layer DLC.

Time(s)	Corel	Pendigit	Letter	Mnist	Covtype
Spectral	2.12	60.48	195.63	3654.9	1.8×10^5
Nystrom	1.05	11.49	24.43	48.88	258.25
KASP	1.12	22.15	66.65	416.66	360.07
LSC-K	1.27	28.58	61.59	468.17	615.84
Ours-L1	**0.78**	**3.11**	**4.64**	**17.41**	**80.2**
Ours-L2	1.26	6.83	8.39	28.62	116.1

4.2 Experimental Results

Comparisons with Spectral Methods

First, we compare our method with state-of-the-art graph clustering methods, as ours enjoys the similar formulation of these methods; however, the linear coding scheme is able to significantly speed up the process. We compare with the following methods in the experiments:

Spectral Original spectral clustering method with Gaussian similarity kernel. Sparse kNN graph is used for graph construction, and k is set to a small number, following the conventional configuration [Von Luxburg, 2007].

KASP A divide and conquer based fast spectral clustering method proposed in [Yan *et al.*, 2009], which uses K-means for landmarks. For efficiency and a fair comparison, we take a Matlab implementation of the multi-way partition version in the evaluations.

Nystrom It uses sampled data to approximate the original graph, and the eigen-decomposition, which saves substantial time. We choose the Matlab implementation with orthogonalization as our evaluation method [Chen *et al.*, 2011].

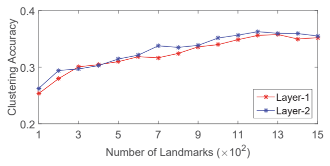
LSC-K Landmark-based spectral clustering using K-means for landmark-selection [Chen and Cai, 2011]. It takes advantage of graph factorization for fast eigen-decomposition.

Results from Table 2 and 4 demonstrate that our method provides comparable accuracy (Pendigit, Mnist, Covtype) or even better accuracy (Corel, Letter) with reduced running time. The advantage of our method lies in the flexible feature dimensions and linear coding scheme instead of time consuming eigen-decomposition. Specifically, the relative high dimension features fed to K-means in the last step are favored by the natural scene/color images. However, in most of fast implementations of spectral methods, the final feature dimensions for K-means usually depend on the number of clusters. In addition, the linear coding scheme substantially reduces the time complexity and makes large-scale clustering easy. Compared to the original spectral clustering, our method is 1500 times faster, and compared to Nystrom, ours only needs less than half of its running time, but with large improvement.

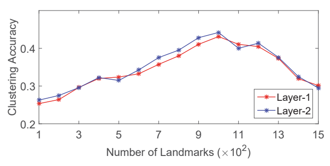
Comparisons with Deep Methods

Second, since our method also utilizes the deep structure to further refine the representation for clustering tasks, we compare with the most recent state-of-the-art deep clustering methods. The details of them are listed below:

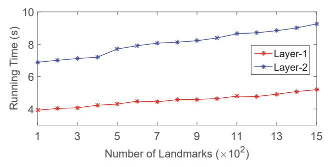
AEC A deep autoencoder with compactness constraint added on the top layer, which is specifically tailored for clustering tasks [Song *et al.*, 2013].



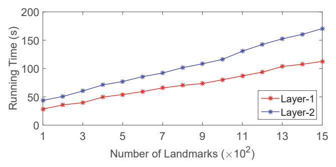
(a) Letter.



(b) Covtype.



(c) Letter.



(d) Covtype.

Figure 1: Clustering accuracy vs. number of landmarks on (a) Letter, (b) Covtype datasets; Running time vs. number of landmarks on (c) Letter, (d) Covtype datasets.

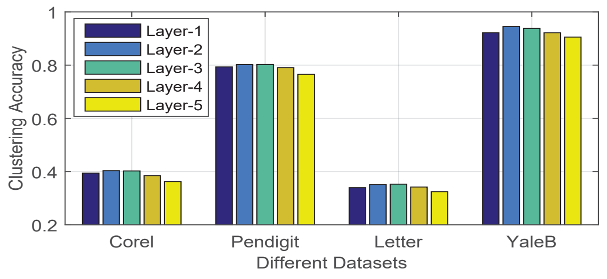
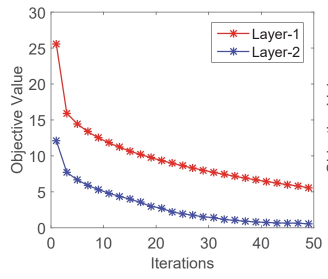
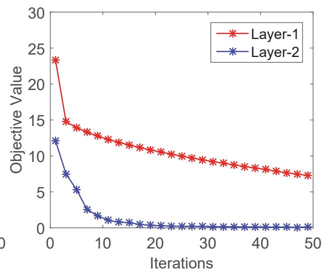


Figure 2: Number of layers vs. clustering accuracy.



(a) Corel.



(b) YaleB.

Figure 3: Convergence of the proposed objective function.

LDR A deep sparse autoencoder to refine the normalized affinity graph, followed by a K-means clustering [Tian *et al.*, 2014]. We evaluate on a Matlab implementation with recommended configurations.

DEN A deep structure with RBM as building blocks, and takes advantage of locality preserving constraint and group sparsity for clustering purpose [Huang *et al.*, 2014].

From Table 3, we can see that the proposed method is significantly faster than other competitors, while achieves comparable performance. The main reason is our method does not need time-consuming back propagation, pre-training and fine tuning techniques that are frequently used in the deep approaches. Admittedly, those learning tricks are critical in learning deep non-linear features; however they are too expensive in fast clustering tasks. In addition, the clustering tasks usually prefer transductive learning fashion, and the inductive model learned from deep methods seems too "heavy" for the problem.

Model Discussions

There are a few model parameters to be tuned in DLC: number of landmarks and number of layers in the deep structure. To find the appropriate settings, we conduct two experiments by setting one parameter fixed while changing the other.

First, we experiment by Single-layer Linear Coding (SLC) scheme and vary the number of landmarks from 100 to 1500, which is shown in Figure 1. In each subfigure, "Layer-1" means the results of SLC, and the Y-Axis illustrates the clustering accuracy or running time. Apparently, more landmarks takes more running time, but it also improves the final clustering performance. Without loss of generality, we set the number of landmarks in the first layer at 1000, which also provides acceptable performance, and add the second layer on top to further refine the feature. In four subfigures, we use "Layer-2" to indicate such settings. It can be seen that a moderate number of landmarks leads to better results. Thus, we still set the number of landmarks in this layer at 1000. Although we may obtain better results by other settings, the current ones have already been comparable with existing methods.

Second, we conduct experiments to find out the appropriate number of layers used in DLC. Similar to last experiment, we set the the number of landmarks in each layer at 1000, and gradually add the layer of the deep structure. Clustering accuracy results on a few datasets are shown in Figure 2. From this result, we can conclude that a two-layer DLC is a good choice in terms of both accuracy and time cost.

Finally, since DLC is iteratively solved in two subproblems, and there are no closed-form solutions, we need to guarantee the objective is monotonously decreased. To this end, we experiment on Corel and YaleB datasets with SLC and a two-layer DLC, and show the objective values in different iterations in Figure 3. Note in this experiment value in the X-Axis indicates the number of iterations for both SLC and DLC. Clearly, the objective values are gradually decreased, and a deeper model can significantly speed up this process.

5 Conclusions

In this paper, we proposed a novel deep linear coding scheme for fast graph clustering. Different from previous methods either relying on eigen-decomposition or time consuming deep autoencoders, our method jointly learned a linear transform and codings at the same time, and utilized deep structure to further refine the discriminative features. Extensive evaluations on seven benchmark datasets demonstrated the effectiveness of our methods, especially for large-scale data.

References

[Barlow, 1989] Horace Barlow. Unsupervised learning. *Neural Computation*, 1(3):295–311, 1989.

[Bengio *et al.*, 2007] Yoshua Bengio, Pascal Lamblin, Dan Popovici, Hugo Larochelle, et al. Greedy layer-wise training of deep networks. *Advances in Neural Information Processing Systems*, 19:153, 2007.

[Bengio *et al.*, 2013] Yoshua Bengio, Aaron Courville, and Pascal Vincent. Representation learning: A review and new perspectives. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 35(8):1798–1828, 2013.

[Bengio, 2009] Yoshua Bengio. Learning deep architectures for ai. *Foundations and Trends® in Machine Learning*, 2(1):1–127, 2009.

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

[Chen *et al.*, 2011] Wen-Yen Chen, Yangqiu Song, Hongjie Bai, Chih-Jen Lin, and Edward Y Chang. Parallel spectral clustering in distributed systems. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 33(3):568–586, 2011.

[Chen *et al.*, 2012] Minmin Chen, Zhixiang Xu, Kilian Weinberger, and Fei Sha. Marginalized denoising autoencoders for domain adaptation. In *International Conference on Machine Learning*, 2012.

[Chung, 1997] Fan RK Chung. *Spectral graph theory*, volume 92. American Mathematical Soc., 1997.

[Dean *et al.*, 2012] Jeffrey Dean, Greg Corrado, Rajat Monga, Kai Chen, Matthieu Devin, Mark Mao, Andrew Senior, Paul Tucker, Ke Yang, Quoc V Le, et al. Large scale distributed deep networks. In *Advances in Neural Information Processing Systems*, pages 1223–1231, 2012.

[Dhillon *et al.*, 2004] Inderjit S Dhillon, Yuqiang Guan, and Brian Kulis. Kernel k-means: spectral clustering and normalized cuts. In *ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, pages 551–556. ACM, 2004.

[Eckart and Young, 1936] Carl Eckart and Gale Young. The approximation of one matrix by another of lower rank. *Psychometrika*, 1(3):211–218, 1936.

[Fowlkes *et al.*, 2004] Charless Fowlkes, Serge Belongie, Fan Chung, and Jitendra Malik. Spectral grouping using the nystrom method. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 26(2):214–225, 2004.

[Hartigan and Wong, 1979] John A Hartigan and Manchek A Wong. Algorithm as 136: A k-means clustering algorithm. *Applied Statistics*, pages 100–108, 1979.

[Hastie *et al.*, 2009] Trevor Hastie, Robert Tibshirani, Jerome Friedman, T Hastie, J Friedman, and R Tibshirani. *The elements of statistical learning*, volume 2. Springer, 2009.

[Hinton *et al.*, 2006] Geoffrey Hinton, Simon Osindero, and Yee-Whye Teh. A fast learning algorithm for deep belief nets. *Neural Computation*, 18(7):1527–1554, 2006.

[Huang *et al.*, 2014] Peihao Huang, Yan Huang, Wei Wang, and Liang Wang. Deep embedding network for clustering. In *International Conference on Pattern Recognition*, pages 1532–1537. IEEE, 2014.

[Jain *et al.*, 1999] Anil K Jain, M Narasimha Murty, and Patrick J Flynn. Data clustering: a review. *ACM Computing Surveys (CSUR)*, 31(3):264–323, 1999.

[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*, pages 801–808, 2006.

[Meila and Shi, 2001] Marina Meila and Jianbo Shi. A random walks view of spectral segmentation. In *International Workshop on Artificial Intelligence and Statistics*, 2001.

[Muja and Lowe, 2014] Marius Muja and David Lowe. Scalable nearest neighbour algorithms for high dimensional data. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 36(11):2227 – 2240, 2014.

[Ng *et al.*, 2002] Andrew Y Ng, Michael I Jordan, Yair Weiss, et al. On spectral clustering: Analysis and an algorithm. *Advances in Neural Information Processing Systems*, 2:849–856, 2002.

[Shi and Malik, 2000] Jianbo Shi and Jitendra Malik. Normalized cuts and image segmentation. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 22(8):888–905, 2000.

[Song *et al.*, 2013] Chunfeng Song, Feng Liu, Yongzhen Huang, Liang Wang, and Tieniu Tan. Auto-encoder based data clustering. In *Progress in Pattern Recognition, Image Analysis, Computer Vision, and Applications*, pages 117–124. Springer, 2013.

[Tian *et al.*, 2014] Fei Tian, Bin Gao, Qing Cui, Enhong Chen, and Tie-Yan Liu. Learning deep representations for graph clustering. In *AAAI Conference on Artificial Intelligence*, 2014.

[Von Luxburg, 2007] Ulrike Von Luxburg. A tutorial on spectral clustering. *Statistics and Computing*, 17(4):395–416, 2007.

[Yan *et al.*, 2009] Donghui Yan, Ling Huang, and Michael I Jordan. Fast approximate spectral clustering. In *ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, pages 907–916. ACM, 2009.

[Yang *et al.*, 2009] Jianchao Yang, Kai Yu, Yihong Gong, and Thomas Huang. Linear spatial pyramid matching using sparse coding for image classification. In *IEEE Conference on Computer Vision and Pattern Recognition*, pages 1794–1801. IEEE, 2009.