Scalable Nearest Neighbor Sparse Graph Approximation by Exploiting Graph Structure

Ming Shao, *Member, IEEE*, Xindong Wu, *Fellow, IEEE*, and Yun Fu, *Senior Member, IEEE*

Abstract—We consider exploiting graph structure for sparse graph approximations. Graphs play increasingly important roles in learning problems: manifold learning, kernel learning, and spectral clustering. Specifically, in this paper, we concentrate on nearest neighbor sparse graphs which are widely adopted in learning problems due to its spatial efficiency. Nonetheless, we raise an even challenging problem: can we save more memory space while keep competitive performance for the sparse graph? To this end, first, we propose to partition the entire graph into intra- and inter-graphs by exploring the graph structure, and use both of them for graph approximation. Therefore, neighborhood similarities within each cluster, and between different clusters are well preserved. Second, we improve the space use of the entire approximation algorithm. Specially, a novel sparse inter-graph approximation algorithm is proposed, and corresponding approximation error bound is provided. Third, extensive experiments are conducted on 11 real-world datasets, ranging from small- to large-scales, demonstrating that when using less space, our approximate graph can provide comparable or even better performance, in terms of approximation error, and clustering accuracy. In large-scale test, we use less than 1/100 memory of comparable algorithms, but achieve very appealing results.

Index Terms—Scalable graph approximation, nearest neighbor graph, intra-graph, inter-graph, approximation error

1 Introduction

Graphs play increasingly important roles in learning problems: (1) data clustering, by building a pairwise similarity graph [1], [2], [3], [4], [5], [6]; (2) subspace learning, including both linear dimensionality reduction methods, e.g., PCA [7], LDA [8], LPP [9], NPE [10], non-linear ones, e.g., LEM [11], LLE [12], ISOMAP [13], and a unified graph embedding framework [14], [15]; (3) kernel learning [16], [17], [18], [19]; (4) feature extraction [20], [21], and networks issues, e.g., privacy [22], robustness [23], heterogeneity [24]. A typical graph consists of two parts: nodes and edges. In most of graph based algorithms, a node represents a data sample, and an edge encodes the pairwise relation between two attached nodes, e.g., similarity. In learning community, there are two typical similarity graphs: dense graph, and t nearest neighbor (t-NN) sparse graph, which are frequently used in kernel learning and graph Laplacian [25], respectively. While dense graph computes all pairwise similarities and therefore requires more memory space, t-NN sparse graph and its fast implementations [26], [27], [28] usually take much less memory space due to the characteristics sparsity. A t-NN graph can be formally defined as:

Definition 1. *A t-NN sparse graph is a directed graph in general. There is an edge from node x_i to x_j if and only if $\rho(x_i, x_j)$ is among the t smallest elements of the sets $\{\rho(x_i, x_k)|k = 1, 2, \ldots, i-1, i+1, \ldots, m\}$, where m is the total number of nodes, and $\rho(\cdot, \cdot)$ is the distance metric.*

While a t-NN graph uses tm storage units, a dense graph will occupy m^2 units, and in most real-world application $m \gg t$. This becomes prohibitively large when m is large. To address this problem, substantial efforts have been made to find compact representations for graph matrices and low-rank matrix approximation is one of the feasible approaches. In low-rank matrix approximation, original matrix is factorized into a product of two low-rank matrices. Namely, given a $m \times m$ matrix A, we try to approach a factorization: $A \approx B \times C$ where B and C are $m \times r$ and $r \times m$ matrices, and in general $r \ll m$. Theoretically, the optimal low-rank approximation (w.r.t. spectral or Frobenius norm) can be computed through truncated SVD. For a typical $m \times m$ graph, however, the time complexity of SVD is cubic in m, which is prohibitively expensive given a large m. To address this problem, many algorithms have been proposed [29], [30], [31]. Among them, randomized algorithms [32], [33] are popular due to their simplicity and effectiveness. On the other hand, truncated SVD is feasible and tractable for large-scale t-NN sparse graphs thanks to the quick solver, e.g., ARPACK [34], which works as a complement when graph is not symmetric, or dense. However, the two approaches above perform poorly given limited memory [35], [36].

"Divide and conquer" (DC) [37] plays an important role in algorithms design when problems are not tractable due to large size. For instance, in data mining, it has been adopted to boost the subspace learning [38], [39], and kernel machines [40], [41] in two steps: local optimization (divide), and global alignment (conquer). Recently, a general graph approximation method with DC strategy is proposed [36].

- *M. Shao is with the Department of Computer and Information Science, University of Massachusetts Dartmouth, Dartmouth, MA 02747. E-mail: mshao@umassd.edu.*
- *X. Wu is with the School of Computing and Informatics, University of Louisiana at Lafayette, Lafayette, Louisiana 70503. E-mail: xwu@louisiana.edu.*
- *Y. Fu is with the Department of Electrical and Computer Engineering and the College of the Computer and Information Science, Northeastern University, Boston, MA 02115. E-mail: yunfu@ece.neu.edu.*

Manuscript received 8 Mar. 2016; revised 10 July 2016; accepted 7 Aug. 2016. Date of publication 17 Oct. 2016; date of current version 29 Dec. 2016. Recommended for acceptance by W. Fan.
For information on obtaining reprints of this article, please send e-mail to: reprints@ieee.org, and reference the Digital Object Identifier below.
Digital Object Identifier no. 10.1109/TBDATA.2016.2617883

It first explores the internal structure, and then approximates each local graph with random projections. Finally, local graphs are extrapolate to build a global graph.

Although a t-NN sparse graph occupies less space than a dense graph, it still needs to be compressed for large-scale problems. Existing methods such as truncated SVD for sparse graph do not perform well with limited memory [36], and approximation methods for dense graph do not always fit for the sparse graph, especially when it is asymmetric. To our best knowledge, such t-NN sparse graph approximation problem has not been widely discussed before.

To address these issues, we propose a *memory* efficient approximation approach for a *given* t-NN graph.[1] Different from previous DC methods, we are the first to employ different graph approximation strategies for large-scale t-NN sparse similarity graph[2] by exploring the graph structure, i.e., the induced intra-graph [36] and inter-graph learned from underlying data distribution after data clustering, which, to our best knowledge, has not been discussed before. The novelty of this paper can be summarized as:

- *First*, we propose to partition the entire graph into intra- and inter-graphs by exploring the underlying graphs structure, and use both of them for graph approximation. Thus, similarities within each cluster and between different clusters are well preserved.
- *Second*, we improve the space use of our approximation algorithm. Specially, a novel sparse inter-graph approximation algorithm is proposed, and corresponding approximation error bound is provided.
- *Finally*, extensive experiments demonstrate that our method provides comparable or even better graph approximation given limited time or storage, especially on large-scale datasets: Mnist, RCV1, Covtype.

2 Related Work

An intuitive memory efficient way is to sparsify the graph and keep as few nonzero entries as possible. To this end, researchers propose to learn a distribution over which the expected value of each entry in the approximate graph is equal to the true value of the graph entry, and meanwhile ensure the approximate graph is sparse compared to the original one [42], [43]. In brief, the sparsity is generated by the original graph in addition to a sparse prior, with good theoretical guarantees. However, it has not been widely adopted in practice for graph approximation so far, especially for the purpose of clustering.

Randomly selecting columns/rows also draws considerable attention [44], [45] recently. Matrix built by selected columns assures the reconstruction error through r selected columns is upper-bounded by the residue between A and rank-r approximation $A_{(r)}$ [44], which guides the process of automatic selection of columns. One instance uses greedy algorithms for solutions [31], [46], [47]. Another instance is incomplete Cholesky decomposition (ICD) [29], [48] which is widely adopted for the matrix decomposition. Different from greedy algorithms, Nyström method [49] uses sampled landmarks to directly approximate the eigensystem of the original matrix [35], [49], [50], showing enormous practical values in image segmentation and spectral clustering. Different from ours, all the above methods do not exploit the underlying structure of the data, and may not be appropriate for sparse matrix approximation. Notably, a recent Nyström method uses k-means to discover landmarks [51], which seems similar to our work; however, it did not consider the intra-graph and can only work on dense graph.

Sub-matrix methods provide a more general formulation assuming matrix is decomposable by sub-matrices and coefficient matrices [52], [53]. A typical approach named CUR decomposition using both sub-collection of rows and columns has been proposed in [54] with the form: $A \approx CUR$ where C and R are sub-matrices built by columns and rows, and U is the linkage matrix. A randomized algorithm for this problem can be found in [55]. Similar to columns/rows sampling methods, the underlying structure of data is not considered, leading to an inferior performance.

Another highly related work by random structure is dimensionality reduction [33], which uses the range of data in the randomized column space to reduce the size of the matrix, but keeps its most geometrical structure [56]. A Gaussian random matrix with over sampling is used in [57] to perform dimensionality reduction and proved to yield better results. Different from applying dimensionality reduction to the whole graph, we adopt it in the local structure, giving rise to a better approximation by a factor of k, where k is the number of local structures. This is particularly useful given a large graph with limited memory.

"Divide and conquer" is an efficient approach towards large-scale graph approximations [36], [40], [41], [50], [58]. In graph approximations, clustering methods are usually adopted for the "divide" purpose, e.g., k-means [59], random projection tree [60]. Afterwards, original large graph is partitioned into a set of small subgraphs, whose size is reduced by a factor of k. Finally, the original graph is approximately recovered in the "conquer" step by extrapolating intra-graphs. Different from dividing the graph itself, method in [50] reduces the computation burden through parallel techniques, i.e., Map-Reduce. Note that our method also differs from [36], [40], [61] since we explicitly consider both intra and inter relations of a sparse graph, and develop a specific algorithm for the inter-graph approximation and corresponding error bound.

3 Framework

We first briefly explain how to use data's underlying structure for better approximations, and then detail the intra- and inter-graph approximations. Illustration of the entire framework can be found in Fig. 1. In addition, we summarize all the variables discussed in this paper in Table 1.

3.1 Graph Partition and Representation

Given a t-NN sparse graph and corresponding data samples, we first find k clusters and therefore k^2 graph partitions in the "divide" step. In our problem, there are two

1. In this paper, we assume the t-NN sparse graph is known at the beginning, and we only consider how to better approximate such a sparse graph, in terms of approximation error and clustering accuracy. Detailed discussion of nearest neighbor graph construction is already beyond the scope of this paper.

2. There might be other NN sparse graphs; however, in this paper, we are more interest in NN similarity graph for learning problems.

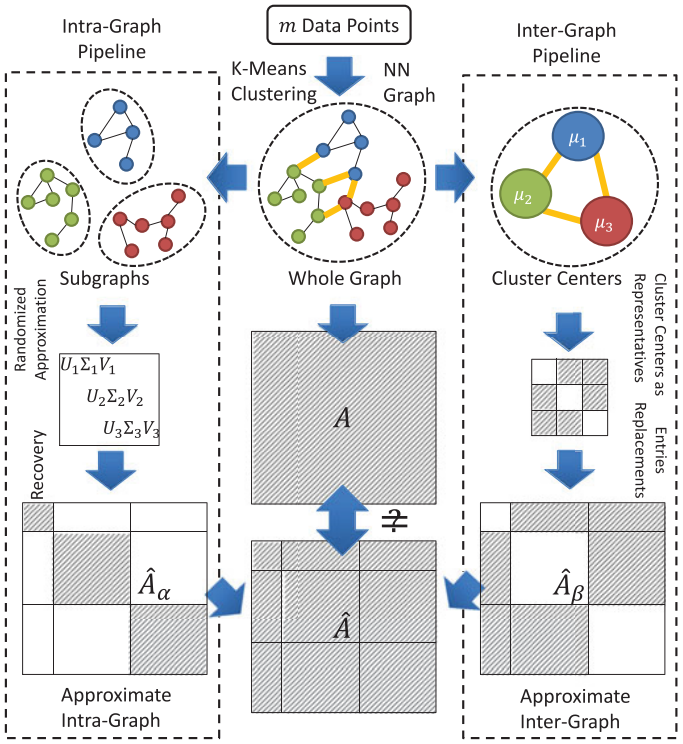

Fig. 1. Framework of the proposed method. For a given t-NN graph of m samples, we first use k-means clustering to do the graph partition (Section 3.1), and obtain intra- and inter-graphs on the left and right hand side, respectively. Then for each of them, we follow algorithms introduced in Sections 3.2 and 3.3 to achieve graph approximation. Finally, approximate intra- and inter-graphs are combined together.

popular ways to group the data samples: (1) vector clustering method, e.g., k-means [59]; (2) graph clustering algorithms, e.g., Louvain algorithm [62], Constrained Laplacian Rank algorithm [63]. Notably, although vector clustering usually uses Euclidean distance as the metric assuming Gaussian as the underlying distribution, e.g., k-means, which may conflict with the data distribution, it is still helpful for affinity graph with shift invariant kernel [40], e.g., Gaussian kernel.[3] Given a dataset $X \in \mathbb{R}^{d\times m}$, clustering algorithms collapse all the data to k clusters $\{X_i \in \mathbb{R}^{d\times m_i} | i = 1, 2, \ldots, k\}$. Each sample is assigned to a cluster represented by its center:

Definition 2. *Cluster center μ_i is the representative of the cluster X_i computed by: $\mu_i = \sum_{x\in X_i} x/m_i$.*

We then re-arrange data samples into the order of $[X_1, X_2, \ldots, X_k]$ according to the learned cluster indices, and therefore can build the re-ordered graph matrix A including k^2 blocks. As we do not change the neighborhood of each sample, the original t-NN sparse graph is still valid. We only need to shift the row/column of the original graph according to the order of $[X_1, X_2, \ldots, X_k]$. The re-arranged A can be either an affinity graph or a normalized graph Laplacian, according to application scenarios. Among k^2 blocks, k diagonal blocks encode the intra relations within each cluster, while the $k(k-1)$ off-diagonal blocks encode the inter relations between two clusters.

3. Unless specified otherwise, we will use k-means for the data clustering through this paper due to its simplicity.

TABLE 1
Notations and Descriptions

Variable	Description
A	Affinity graph or normalized graph Laplacian
A_α, A_β	Intra- and inter-graph
W	Affinity graph, if A indicates graph Laplacian
d_i	Degree of the ith node in the graph
D	Diagonal matrix with $d_1, \ldots, d_m$ on the diagonal
U, Σ, V	Factor matrices of SVD
$\hat{A}, \hat{W}, \hat{D}$	Approximate matrices of A, W, D
Δ_D	Difference between $\hat{D}$ and D, i.e., $\hat{D} - D$
Λ	Logical index matrix for inter-graph
P	Projection matrix
Ω	Gaussian random matrix
Y	Low-dimensional representation of data
m	Number of data samples
k	Number of clusters for k-means
r	Target rank for low-rank approximation
p	Over sampling factor
t	Number of neighbors in NN graph
γ	bandwidth of Gaussian kernel

Definition 3. *Intra-graph A_α is the following diagonal fractions of the original graph A:*

$$A_\alpha = \text{diag}(A_1, A_2, \ldots, A_k) = \begin{bmatrix} A_1 & 0 & \ldots & 0 \\ 0 & A_2 & \ldots & 0 \\ \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & \ldots & A_k \end{bmatrix}, \quad (1)$$

where A_i is the induced subgraph of $X_i, 1 \le i \le k$.

Although the original graph has been divided into $k \times k$ subgraphs, the total size of them may still be large if they are stored directly in memory. A popular approach would be low-rank or rank-r matrix approximation. In this paper, for efficiency, we further compress them using randomized low-rank approximation algorithm [33]. Although a direct randomized low-rank approximation on the original graph matrix is also feasible, the low-rank approximation after graph partition substantially improves the performance [36]. This is because a rank-r approximation on each subgraph yields a rank-kr approximation on the entire intra-graph.

Definition 4. *Inter-graph A_β is the following off-diagonal fractions of the original graph A:*

$$A_\beta = \begin{bmatrix} 0 & A_{12} & \ldots & A_{1k} \\ A_{21} & 0 & \ldots & A_{2k} \\ \vdots & \vdots & \ddots & \vdots \\ A_{k1} & A_{k2} & \ldots & 0 \end{bmatrix}, \quad (2)$$

where $A_{ij}, 1 \le i, j \le k$ reflects the pairwise relationships between data samples from X_i and X_j.

In a dense graph, entries in the inter-graph dominate the entire graph, which is impossible to store on a common PC given a large m. In a sparse graph, however, either intra- or inter-graph could dominate the entire graph, since the dominance is affected by many factors: number of neighbors, number of clusters, etc. To demonstrate the impacts of these factors, we show experimental results from four datasets: Corel, Satimage, Pendigit, Mnist, and use "IOA", i.e., the

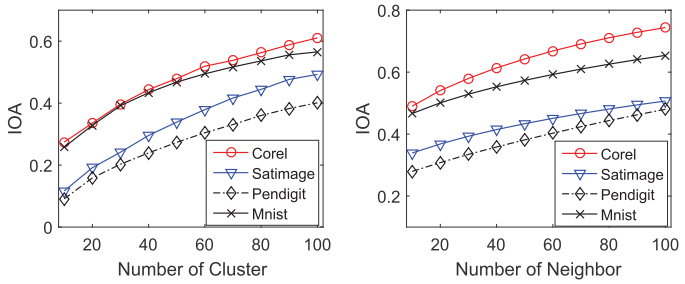

Fig. 2. Statistical Information of graph entries. Here "IOA" means the number of entries from inter-graph over that from the entire graph. We experiment on four datasets with varied number of clusters/neighbors to show the importance of both intra- and inter-graphs.

number of entries in inter-graph over the number of all entries to quantitatively measure the entries' distribution in Fig. 2, where the number of neighbors is set to 30 in the left figure, and the number of clusters is set to 50 in the right figure.

From Fig. 2, we can conclude that both intra- and inter-graph are essential to a sparse graph. In addition to the block-wise randomized low-rank approximation for intra-graph, we propose a new inter-graph approximation method according to the inter-graph's characteristics, which substantially reduces the space requirement. Both intra-graph and inter-graph construction details will be revealed in the following sections.

3.2 Intra-Graph Approximation

According to *Definition* 2, intra-graph consists of k subgraphs and they are usually dense. The reason for being dense is, for a typical t-NN graph, the clustering process in the graph partition may already group the data and their t nearest neighbors together. This means intra-graph covers most of the pairwise connections of the entire graph if the number of clusters is not large. Under this situation, using sparse eigen-decomposition method, e.g., ARPACK [34] for computational efficiency is less reasonable. For that reason, we treat the blocks of intra-graph as dense graphs and use rank-r matrix approximation to save space.

In general, rank-r approximation can be formulated by

$$\min_{\mathrm{rank}(\hat{A})=r} \|A-\hat{A}\|_F^2 \rightarrow \min_{\substack{U,V\in\mathbb{R}^{m\times r},\\ U^\top U=I}} \|A-UV^\top\|_F^2, \tag{3}$$

where $\hat{A}=UV^\top$ is the rank-r decomposition of A. According to the facts revealed in [64], we learned that the solution to Eq. (3) can be found by the eigen-decomposition of $AA^\top$ after A being centralized. The above decomposition can enjoy further speed-up via randomized low-rank approximation [33]. "Randomized" here means we impose a random projection on the original matrix to form a reduced matrix, whose range can be used to approximate the range of the original input. Then the SVD on the reduced matrix can help find the low-rank approximation of the original matrix with a lower time complexity. Suppose the target rank is r, then time consuming SVD in the original low-rank approximation approach can be reduced from $O(m^3)$ to $O(rm^2)$. We detail this process in Algorithm 1.

In Algorithm 1, first, a rank-$(r+p)$ Gaussian random matrix is generated where p is a over-sampling number in step 1. Then we project the graph matrix to the random space and find its range P through QR decomposition in steps 2 and 3. Note that we use $(AA^\top)^qA$ $(q\geq 1)$ instead of A to mitigate the slow spectral decay of graph matrix A [33, Section 4.5]. Finally, we take the orthogonal projection $PP^\top$ on A as the approximation of A. Therefore, A can be decomposed through the factorization of $PP^\top A$. In brief, we can see that the randomized low-rank decomposition uses approximately $O(mr)$ space, where $r\ll m$ in practice.

Algorithm 1. Randomized Low-Rank Graph Approximation

Input: An $m\times m$ matrix A, target rank r, over-sampling parameter $p\geq 1$, $l=r+p$
Output: Low-rank approximation of A, so that $A\approx U\Sigma V^\top$
1: Generate an $m\times l$ Gaussian random matrix Ω.
2: Form the $m\times l$ matrix $Y=(AA^\top)^qA\Omega$.
3: Construct an $m\times l$ matrix P whose columns are orthogonal basis of the span of Y.
4: Apply SVD on $P^\top A$ and obtain decomposed components $\tilde{U}$, Σ, and V.
5: Construct the low-rank decomposition for A with U, Σ, and V, where $U=P\tilde{U}$

Next we implement randomized low-rank approximation on each subgraph of intra-graph, leading to a time complexity of $O(r\sum_i m_i^2)$ on SVD itself, where m_i is the number of samples in each cluster. In addition, we achieve a rank-rk rather than rank-r approximation by using the same amount of memory

$$\underbrace{\begin{bmatrix}U\end{bmatrix}}_{\text{rank-}r}\begin{bmatrix}\Sigma\end{bmatrix}\underbrace{\begin{bmatrix}V\end{bmatrix}^\top}_{\text{rank-}r}\Rightarrow\underbrace{\begin{bmatrix}U_1\Sigma_1V_1^\top & 0 & 0\\ 0 & \ddots & 0\\ 0 & 0 & U_k\Sigma_kV_k^\top\end{bmatrix}}_{\text{rank-}kr},$$

where the left formulation represents randomized low-rank approximation on the entire graph with $U\sim m\times r$, $\Sigma\sim r\times r$, $V\sim m\times r$, while the right one represents that on each single subgraph with $U_i\sim m_i\times r$, $\Sigma_i\sim r\times r$, $V\sim m_i\times r$[4].

From Algorithm 1, we can see rank-r approximation for the ith subgraph is $P_iP_i^\top A_i$ where P_i is the orthogonal basis of the ith block. Therefore, the block-wise intra-graph's randomized low-rank approximation can be illustrated as

$$A_\alpha\approx \mathrm{diag}(P_1P_1^\top A_1,\ldots,P_kP_k^\top A_k)$$
$$=\underbrace{\begin{bmatrix}P_1 & 0 & 0\\ 0 & \ddots & 0\\ 0 & 0 & P_k\end{bmatrix}}_{P}\underbrace{\begin{bmatrix}P_1^\top & 0 & 0\\ 0 & \ddots & 0\\ 0 & 0 & P_k^\top\end{bmatrix}}_{P^\top}\underbrace{\begin{bmatrix}A_1 & 0 & 0\\ 0 & \ddots & 0\\ 0 & 0 & A_k\end{bmatrix}}_{A_\alpha},$$

and corresponding block-wise approximation error w.r.t Frobenius norm can be immediately derived as

$$\|A_\alpha-\hat{A}_\alpha\|_F^2=\|A_\alpha-PP^\top A_\alpha\|_F^2$$
$$=\|\,(I-PP^\top)A_\alpha\,\|_F^2=\sum_{i=1}^{k}\|\,(I_i-P_iP_i^\top)A_i\,\|_F^2\,.$$

4. Note U and $[U_1;\ldots;U_k]$ are different.

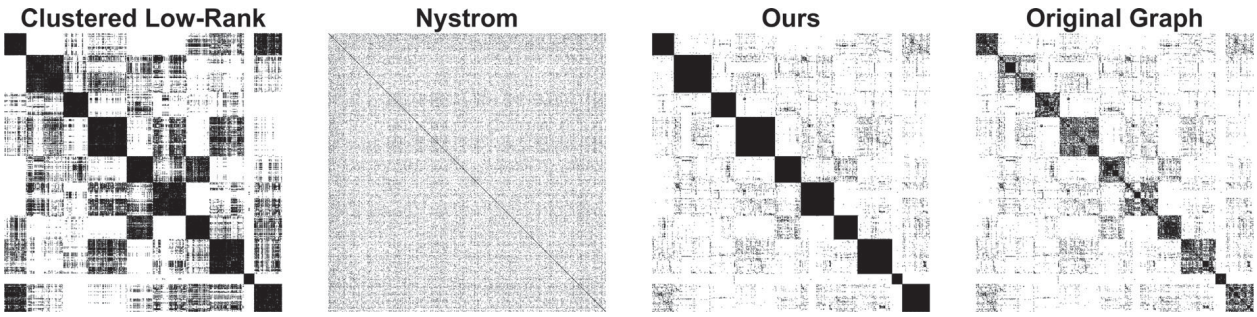

Fig. 3. Illustration of approximate graphs by (from left to right): (1) Clustered Low-Rank [36], (2) Nyström method [49], (3) Ours, and (4) Original Graph on Corel dataset. Numbers of clusters and neighbors are set to 10 and 30, and the target rank for each block is set to 10. Note, Clustered Low-Rank and Nyström methods render a very dense inter-graph.

3.3 Inter-Graph Approximation

Inter-graph has a significantly different underlying structure compared to that of intra-graph, since it is almost always sparse if the entire t-NN graph is sparse. This motivates us to find different approximation strategies. First, dense graph approximation methods are obviously inappropriate for the sparse one. Second, existing sparse graph approximation methods may fail as they usually provide a dense approximate graph for the blocks from the inter-graph. When the ground truth inter-graph is sparse, it will substantially increase the approximation error. To make this clear, we show approximate graphs by existing sparse graph approximation methods in Fig. 3. As we can see, Clustered Low-Rank [36] generates a very dense inter-graph, which is significantly different from the original graph shown on the right.

To address this problem, we propose a novel inter-graph approximation method by preserving the underlying structure of the inter-graph while approximating each block therein by a single representative. First, to preserve the underlying structure, we convert the entire inter-graph to a binary graph, using "1" to mark the connected pairs in the inter-graph, and "0" to mark the disconnected pairs. Second, we use the center of each cluster to represent the entire cluster, and the simplified relation between the ith and jth clusters can be computed as: $a_{ij} = \exp(-\frac{(\mu_i-\mu_j)^2}{2\gamma^2})$, where μ_i and μ_j are centers of the ith and jth clusters, respectively. The resulting approximate subgraph $\hat{A}_{ij}(i \neq j)$ in off-diagonal blocks can be written as

$$\hat{A}_{ij} = a_{ij}\Lambda_{ij}, \tag{4}$$

where $|\cdot|$ denotes the cardinality of a set, Λ_{ij} is a $|X_i| \times |X_j|$ 0-1 logical matrix indicating the existence of an entry in a sparse graph. Therefore, the approximation error of the proposed method for inter-graph is

$$\|A_\beta - \hat{A}_\beta\|_F^2 = \sum_{i,j,i\neq j} \|A_{ij} - a_{ij}\Lambda_{ij}\|_F^2. \tag{5}$$

We illustrate an approximate graph by our proposed approximation algorithm in the third sub-figure of Fig. 3. Compared to Clustered Low-Rank [36] and Nyström [49] methods, ours yields a more sparse inter-graph. Additional advantage of using this inter-graph approximation is obvious: the space requirement is reduced substantially. Currently, we only need to store a single real number for each off-diagonal block, plus a highly sparse logical matrix which in general is very small. Although the approximation method is intuitive, it is not theoretically clear on how good this inter-graph approximation will be. This is because we throw away many details after all and only preserve the underlying structure and key similarities. Next section will detail the error bounds of both intra- and inter-graphs.

4 Theoretical Analysis

We present theoretical analysis of the approximation error and time/space use. As the proposed approximate graph includes two separated parts: A_α and A_β, we will discuss them separately. Total error is thus bounded by the sum of them. Without loss of generality, we discuss the average case of the error, i.e., the expectation of $\|A - \hat{A}\|_F^2$

$$\begin{aligned}
&\mathbb{E}[\|A - \hat{A}\|_F^2] \\
&= \mathbb{E}[\|A_\alpha + A_\beta - \hat{A}_\alpha - \hat{A}_\beta\|_F^2] \\
&= \mathbb{E}[\|A_\alpha - \hat{A}_\alpha + A_\beta - \hat{A}_\beta\|_F^2] \\
&= \mathbb{E}[\|A_\alpha - \hat{A}_\alpha\|_F^2] + \mathbb{E}[\|A_\beta - \hat{A}_\beta\|_F^2] \\
&= \Sigma_i^k \mathbb{E}[\|(I - P_i P_i^\top)A_i\|_F^2] + \mathbb{E}[\|A_\beta - \hat{A}_\beta\|_F^2].
\end{aligned} \tag{6}$$

4.1 Error Bound of Intra-Graph

From Eq. (6), it can be learned that the approximation error from intra-graph A_α is the sum over k components from diagonal blocks. The error expectation of each is bounded by the conventional randomized low-rank approximation theorem stated in [33]:

Theorem 1 (Theorem 10.5 in [33]). *A is a real $m \times n$ matrix with decrease order singular values $\sigma_1 \geq \sigma_2 \geq \sigma_3, \ldots$, r is the target rank $(r \geq 2)$, and p is the over-sampling parameter, $p \geq 2$, where $r + p \leq \min(m, n)$. Executing Algorithm 1, we can obtain an $m \times (r + p)$ matrix P with orthogonal columns. Then we have*

$$\mathbb{E}[\| A - PP^\top A \|_F^2] \leq \left(1 + \frac{r}{p-1}\right) \sum_{j>r} \sigma_j^2, \tag{7}$$

where σ_{r+1} is the $(r+1)$th singular value of A.

Remarks. From theorem above, we can see the approximation error is bounded by the sum of squares of small singular values of the original graph up to a constant factor: $1 + r/(p-1)$. In general, higher target rank will yield better approximation performance. When we use higher

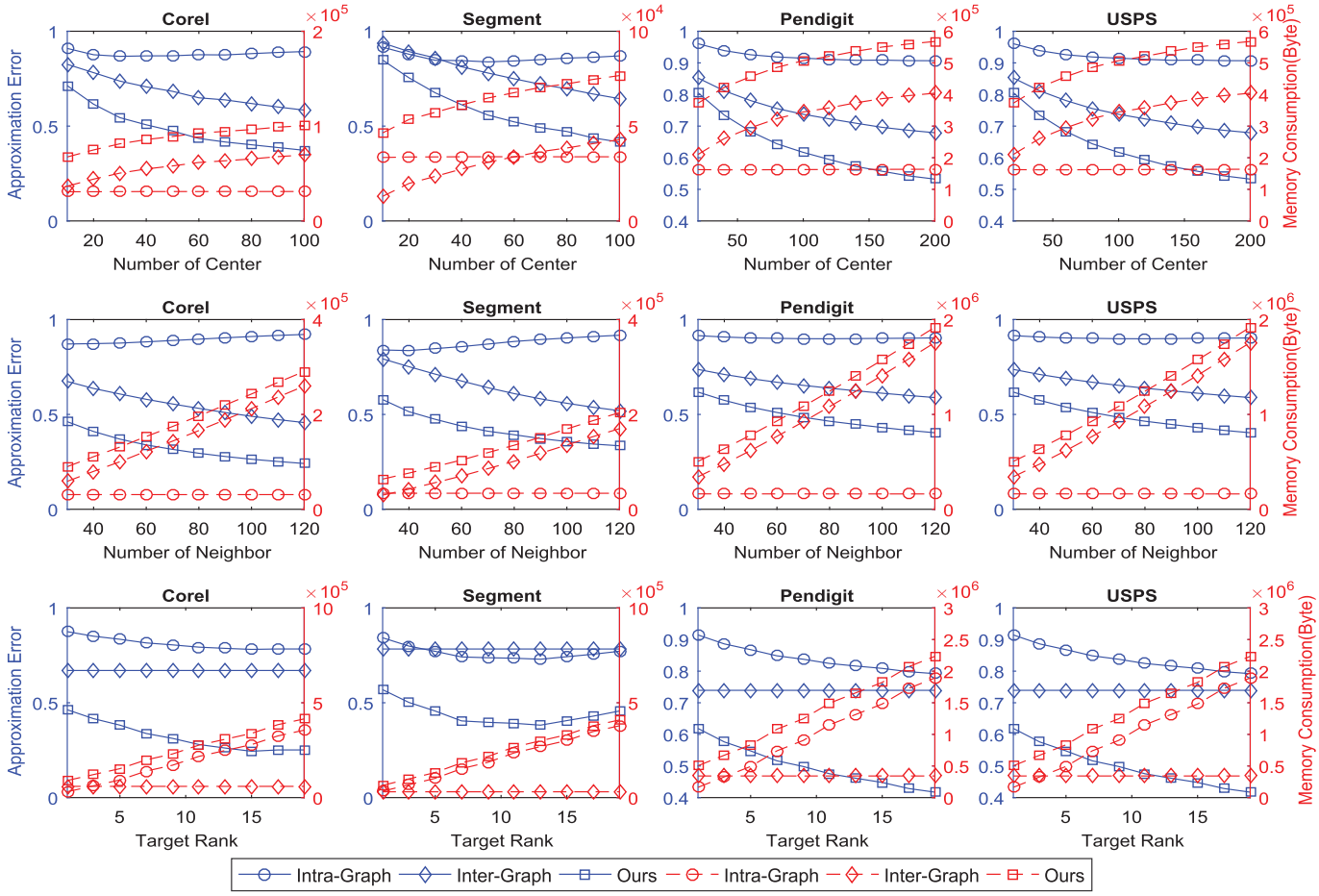

Fig. 4. Experimental results of graph approximation error (marked in blue, ↓ better) and memory consumption (marked in red, ↓ better) on four datasets given different number of clusters (1st row), neighbors (2nd row), and ranks (3rd row). The *X*-axis represents the number of clusters/neighbors, and ranks, and the *Y*-axis represents the approximation error and memory consumption. We compare three different methods: intra-graph, inter-graph, and our method.

rank to approach A_α, however, it is not clear whether this upper bound will increase or decrease, because the right part of Eq. (7) is not necessarily a monotonic function w.r.t. r. This is also demonstrated by experiments on Segment dataset from the 3rd row of Fig. 4. Finally, we obtain the expected error bound for the entire intra-graph A_α

$$\sum_{i=1}^{k}\mathbb{E}[\| A_i - P_i P_i^\top A_i \|_F^2] \le \left(1+\frac{r}{p-1}\right)\sum_{i=1}^{k}\sum_{j>r}\sigma_{ij}^2,$$

where σ_{ij} denotes the jth singular value of graph A_i.

4.2 Error Bound of Inter-Graph

The approximation error of inter-graph is due to the entry replacement in A_β. Namely, we use the center of a cluster to replace all members in this cluster. First, to derive a general error bound for the inter-graph, we use *perturbation theory* [65] to analyze the error caused by the approximation. Similar idea has been adopted in the theoretical analysis of spectral clustering [2]. Second, we will discuss the impacts of different model parameters: k, m, t, and the connections between k-means and the inter-graph error bound.

Let $\hat{x}_i = x_i + \epsilon_i$ be the perturbed samples that we use in the approximation, where ϵ_i is a small additive noise hinged to a specific cluster if any. To make this analysis tractable, some restrictions are added to the distributions of ϵ_i for each cluster: (1) ϵ_i is independent of x_i; (2) ϵ_i is i.i.d. according to a symmetric distribution with zero mean and bounded support; (3) the variance of ϵ_i is small compared to the original data. Next theory shows the difference between $\hat{A}_\beta$ and A_β due to perturbation is upper bounded by the variance of ϵ_i times some constant.

Theorem 2. *Assuming that: (1) $x_1, \ldots, x_m \in \mathbb{R}^d$ are generated i.i.d. from a mixture of distributions such that the degree of node in the built sparse graph is large enough, namely, $d_i/m > c_0$ holds for some constant $c_0 > 0$, (2) for each cluster, additive noise ϵ_i has zero mean, and an unique, yet bounded support, and (3) $\|\Delta_D D^{-1}\|_2 = o(1)$, then*

$$\begin{aligned}&\mathbb{E}\|A_\beta - \hat{A}_\beta\|_F^2 \\ &\le \left(\frac{2}{c_0^2 m^2} + \frac{2tc_1}{c_0^4 m^3}\right)\theta^2 \sum_{i,j}\lambda_{ij}\big(\mathrm{Var}(\epsilon_i) + 2\mathrm{Var}(\epsilon_j)\big),\end{aligned} \tag{8}$$

where Var(·) is the variance of the random variable, θ and c_1 are some constants, λ_{ij} is an entry from the logical index matrix Λ, and i, j index the data samples.

Proof. To prove the theory, we need to find an appropriate formulation for "$\|A_\beta - \hat{A}_\beta\|_F^2$". Here, we are more interested in graph Laplacian than Gaussian graph since the latter one can be proved similarly by setting matrix D as an identity matrix. In the followings, we denote W as the affinity graph, A as the graph Laplacian, and remove the subscript β for simplicity. The following lemma gives an approximation of this square Frobenius norm:

Lemma 1. *Assume for each data sample, the sum of addictive noises of connected nodes are not large compared to its degree, namely,* $\|\Delta_D D^{-1}\|_2 = o(1)$*, then we have the following results:*

$$\begin{aligned}&\|A_\beta - \hat{A}_\beta\|_F \\ &\le \|D^{-\frac{1}{2}}\Delta_W D^{-\frac{1}{2}}\|_F + (1+o(1))\|\Delta_D D^{-\frac{3}{2}} W D^{-\frac{1}{2}}\|_F,\end{aligned}$$

where $\Delta_W = \hat{W} - W$*, and* $\Delta_D = \hat{D} - D$*.*

The proof of *Lemma 1* can be found in the supplementary file, which can be found on the Computer Society Digital Library at http://doi.ieeecomputersociety.org/10.1109/TBDATA.2016.2617883, which is similar to the proof of [58, Lemma 4]. Next, we derive upper bounds for $\|D^{-\frac{1}{2}}\Delta_W D^{-\frac{1}{2}}\|_F^2$, and $\|\Delta_D D^{-\frac{3}{2}} W D^{-\frac{1}{2}}\|_F^2$, respectively.

First, we rewrite $\|D^{-\frac{1}{2}}\Delta_W D^{-\frac{1}{2}}\|_F^2$ in an entry-wise way

$$\|D^{-\frac{1}{2}}\Delta_W D^{-\frac{1}{2}}\|_F^2 = \sum_{i,j}\frac{\lambda_{ij}\delta_{ij}^2}{d_i d_j} \le \frac{1}{c_0^2 m^2}\sum_{i,j}\lambda_{ij}\delta_{ij}^2,$$

$$\text{where } \delta_{ij} = \exp\left(-\frac{\|x_i+\epsilon_i-x_j-\epsilon_j\|^2}{2\gamma^2}\right) - \exp\left(-\frac{\|x_i-x_j\|^2}{2\gamma^2}\right),$$

and $i, j \in [1, m]$ index two different data samples. However, since the exponential term in the formulation above is not easy to bound, we convert it to a more explicit one w.r.t. ϵ by Cauchy Mean Value Theorem. Suppose the function in Cauchy Mean Value Theorem is: $f(x) = \exp\left(-\frac{\|x\|^2}{2\gamma^2}\right)$, and two variables are $x_i + \epsilon_i - x_j - \epsilon_j$ and $x_i - x_j$, respectively. Then we trivially have

$$\delta_{ij} = f'(\xi_{ij})(\epsilon_i - \epsilon_j), \quad \xi_{ij} \in (x_i+\epsilon_i-x_j-\epsilon_j, x_i - x_j). \tag{9}$$

Since $f(x)$ is symmetric in y-axis, $f'(x)$ is bounded everywhere. Assume $\max\{\|f'(x)\|\} = \theta$, then we have $\delta_{ij}^2 \le \theta^2\|\epsilon_i - \epsilon_j\|^2$. In the following parts, we discuss how to bound the expectation of $\|\epsilon_i - \epsilon_j\|^2$, over both ϵ_i and ϵ_j.

Since there are only off-diagonal blocks in the inter-graph, it is reasonable to assume ϵ_i and ϵ_j are additive noises from two different clusters. The following lemma gives a upper bound on $\mathbb{E}(\|\epsilon_i - \epsilon_j\|^2)$.

Lemma 2. *Assume X and Y are random variables sampled from two different distributions, then we have the following conclusions:*

$$\begin{aligned}&\mathbb{E}_{XY}(\|X-Y\|^2) \\ &\le \mathrm{Var}(X) + 2\mathrm{Var}(Y) + 2\|\mathbb{E}(X) - \mathbb{E}(Y)\|^2.\end{aligned} \tag{10}$$

The proof of *Lemma 2* can be found in the supplementary file, available online. From *Lemma 2*, it can be learned that the expectation of $\|\epsilon_i - \epsilon_j\|^2$ can be bounded by the variance of ϵ_i and ϵ_j, plus a square of the difference of their expectations. Given the fact that $\forall i$, $\mathbb{E}(\epsilon_i) = 0$, and considering all the off-diagonal entries, we have the following conclusions:

$$\mathbb{E}(\|D^{-\frac{1}{2}}\Delta_W D^{-\frac{1}{2}}\|_F^2) \le \frac{\theta^2}{c_0^2 m^2}\sum_{i,j}\lambda_{ij}\big(\mathrm{Var}(\epsilon_i) + 2\mathrm{Var}(\epsilon_j)\big). \tag{11}$$

Second, for $\|\Delta_D D^{-\frac{3}{2}} W D^{-\frac{1}{2}}\|_F^2$, we have the similar formulation

$$\|\Delta_D D^{-\frac{3}{2}} W D^{-\frac{1}{2}}\|_F^2 = \sum_{i,j}\frac{w_{ij}\delta_i^2}{d_i^3 d_j} \le \frac{1}{c_0^4 m^3}\sum_i \delta_i^2, \tag{12}$$

where we adopt the fact that $\forall i, j,\ w_{ij} < 1$, and $\delta_i = \sum_j \lambda_{ij}\left(\exp\left(-\frac{\|x_i+\epsilon_i-x_j-\epsilon_j\|^2}{2\gamma^2}\right) - \exp\left(-\frac{\|x_i-x_j\|^2}{2\gamma^2}\right)\right)$. (13)

Similar to the previous proof, we use Cauchy Mean Value Theorem to prove the upper bound where we have the following deductions:

$$\mathbb{E}(\delta_i^2) \le \theta^2\mathbb{E}\Big(\textstyle\sum_j \lambda_{ij}\|\epsilon_i - \epsilon_j\|\Big)^2 \le t\theta^2\textstyle\sum_j \lambda_{ij}\mathbb{E}(\|\epsilon_i - \epsilon_j\|^2).$$

We sum over i and have the following conclusions:

$$\begin{aligned}&\mathbb{E}\Big(\|\Delta_D D^{-\frac{3}{2}} W D^{-\frac{1}{2}}\|_F^2\Big) \\ &\le \frac{t\theta^2}{c_0^4 m^3}\sum_{i,j}\lambda_{ij}\big(\mathrm{Var}(\epsilon_i) + 2\mathrm{Var}(\epsilon_j)\big).\end{aligned} \tag{14}$$

Combining Eqs. (11) and (14) completes the proof. □

Remarks. From Theorem 2, we can conclude that followed variables play key roles in the inter-graph approximation

1) the variance of additive noise Var (ϵ_i) of each sample
2) the total number of samples m
3) the number of nearest neighbor t of the graph.

First, if we assume more clusters for the graph partition, then there are fewer data in each cluster, and their variance Var (ϵ_i) in each cluster might be reduced. Consequently, the variance of additive noise and the upper bound of inter-graph approximation will be reduced too. *Second*, more data samples, i.e., a larger m will decrease the value of the first factor of Eq. (8), but will potentially increase the variance of samples in each cluster. Thus its impact on the upper bound is unclear. *Third*, if more neighbors are considered, i.e., t is larger, then the third factor in Eq. (8) will increase too. However, the first factor may decrease, because d_i becomes larger (c_0 becomes larger as well.) as more neighbors are discovered. In fact, the number of NN t in the first factor makes it even complex. Although it is non-trivial to quantitatively measure the relation between the inter-graph approximation error and the number of clusters/neighbors, we experimentally demonstrate the conclusions above in Fig. 4 w.r.t. the number of clusters/neighbors.

In addition, on the strength of the *Theorem 2*, we could have a better understanding of "graph partition" in the first step of our framework. Suppose in each cluster, $\hat{x}_i$ is generated by a unknown vector, plus an additive noise ϵ_i hinged to a specific data cluster, then the upper bound in Eq. (8), specifically, Var (ϵ_i), can be reduced by minimizing $\mathbb{E}_{x_i}\|x_i - \hat{x}_i\|^2$ for each cluster. This is empirically identical to "vector quantization" problem in signal processing [66] with least square as the loss measurement,

$$\{c_1, c_2, \ldots, c_k; \mathbb{Q}\} = \min \frac{1}{m} \sum_{x \in X} \|x - \mathbb{Q}(x)\|^2, \quad (15)$$

where $\mathbb{Q}$ is a quantizer that quantizes the input signals into k different codewords $c_1, c_2, \ldots, c_k$. While solving this problem is NP-Hard, there is promising local minima by k-means where c_i indicates the cluster center [67]. This fact supports our using of k-means for graph partition in the first step.

4.3 Time Complexity

Time cost of the proposed method includes three parts: (1) graph partition, (2) intra-graph approximation, and (3) inter-graph approximation.

Graph Partition. Suppose the dataset has m samples, k clusters, and the maximal iteration for k-means is set to n, then the time complexity graph partition is $O(kmn)$. In practice, a relatively small n works well for the final graph approximation performance. Thus, we set $n = 50$ in our implementation.

Intra-Graph Approximation. Suppose the target rank is r, the number of samples in the ith cluster is m_i, over sampling factor is p, and $l = r + p < m_i$. We list the detailed time complexity of each step of Algorithm 1 below:

- Step-1: $\sum_i lm_i T_{\mathrm{G}}$. This step generates $\sum_i lm_i$ Gaussian samples, and each of them takes T_{G} time.
- Step-2: $\sum_i O(2qm_i^{2.373} + lm_i^2)$. This step includes two parts: complexity of $(A_i A_i^\top)^q A_i$, and $(A_i A_i^\top)^q A_i \Omega$. The first is the product of $(2q+1)$ $m_i \times m_i$ matrices, and each of it takes $m_i^{2.373}$ steps [68]. The second part involves the product of a $m_i \times m_i$ and a $m_i \times l$ matrix.
- Step-3: $l^2 \sum_i m_i$. This is for the QR decomposition on $Y \in \mathbb{R}^{m_i \times l}$.
- Step-4: $2l \sum_i m_i^2$. This is for the product $P^\top A_i$ and the SVD on $P^\top A_i$.
- Step-5: $r^2 \sum_i m_i$. Complexity for product of $P\tilde{U} = U$.

Remarks. (1) We use power method to mitigate the slow decay of matrix spectral of A_i, and it will introduce additional $2qm_i^{2.373}$ flops in Step 2. In most cases, $q = 1$ is adequate to address the above spectral problem. (2) The product $A_i\Omega$ in step-2 can be further reduced to $O(m_i^2 \log l)$ given a structured Gaussian, e.g., subsampled random Fourier transform [69].

Inter-Graph Approximation. We first compute the small graph with cluster centers as nodes, which takes k^2 steps. Then, all non-zero entries in the block $A_{ij}(i \neq j)$ are replaced by the similarity between representatives μ_i and μ_j, which needs $\sum_{i,j} \lambda_{ij}$ steps. Therefore, the total steps of the proposed inter-graph approximation is: $\sum_{i,j} \lambda_{ij} + k^2$.

4.4 Space Consumption

Next, we discuss the space use of both intra- and inter-graphs. For a specific A_i from intra-graph, its randomized low-rank approximation takes up $2m_i r + r^2$ memory space. Since there are k clusters, the total size is: $\sum_{i=1}^k 2m_i r + kr^2$. For inter-graph, we need to store a $k \times k$ compact graph, and a $m \times m$ logical index matrix Λ indicating the existence of an entry in graph A_β. Compared to the dense double precision matrix, Λ only needs to store the coordinates of off-diagonal blocks represented by integers, which reduces the memory use. Next we compare the space consumption of our method with related methods in Table 2.

TABLE 2
Space Consumption of Related Methods

Method	Space Consumption
Original	mt (double)
Nyström [49]	$m\times$ # selected columns (double)
Meka [40]	$O(mr + (kr)^2)$ (double)
LSC [70]	$m\times$ # of landmarks (double)
Ours	m^2(logical) + $\left(k^2 + \sum_{i=1}^k 2m_i r + kr^2\right)$(double)

5 Experiments

In this section, we evaluate our sparse graph approximation method on 11 real-world datasets, and show the experimental results of *approximation error* and *spectral clustering* based on the approximate graph. In our experiments, we compare with the most recent state-of-the-art works, e.g., [36], [40], [49], [50], [51], [70]. In addition, we discuss time and space cost of both the proposed method and comparison methods.

5.1 Dataset Descriptions

We use four small-scale, and four mid-scale real-world datasets to demonstrate our methods can work well on a wide range of datasets, and use three additional large-scale real-world datasets to test the scalability of the proposed method. The details of 11 datasets are elaborated in Table 3.

Corel[5] The dataset has been widely used in computer vision and image processing. We use its subset from [50] for our test, where 2,074 images, 144 features including shape, texture, color, are chosen for evaluations.

RCV1[6] Reuters Corpus Volume I (RCV1) is an archive of 804,414 manually categorized newswire stories from Reuters Ltd. [71], which includes three controlled vocabularies: industries, topics, and regions. There are 23,149 training documents and 781,256 test documents. In our evaluations, similar to [50], only categories with more than 500 instances are selected, which constructs a dataset with 193,844 samples in 103 categories.

Segment, DNA, Satimage, Pendigit, USPS, Letter, Mnist, Covtype are popular machine learning datasets that are widely used in clustering and multi-class classification tasks. These datasets cover a broad range of data type from images, DNA, to digital numbers or letters. For their detailed descriptions, please refer to LIBSVM Data.[7]

5.2 Experiment Configurations

To evaluate the proposed method and make a fair comparison with existing methods, we conduct several groups of experiments. In all experiments, we consider normalized graph Laplacian with Gaussian kernel as our objective

5. https://sites.google.com/site/dctresearch/Home/content-based-image-retrieval
6. http://www.jmlr.org/papers/volume5/lewis04a/lyrl2004_rcv1v2_README
7. https://www.csie.ntu.edu.tw/~cjlin/libsvmtools/datasets/multiclass.html

TABLE 3
Details of the Datasets Used in the Experimental Section

Details/ Name	Small-Scale				Mid-Scale				Large-Scale		
	Corel	Segment	DNA	Satimage	Pendigit	USPS	Letter	Mnist-Test	Mnist	RCV1	Covtype
# Instances	2,074	2,310	3,186	4,435	10,992	11,000	15,000	10,000	60,000	193,844	581,012
Dimensionality	144	19	180	36	16	256	16	784	784	47,236	54
Class	18	7	3	6	10	10	26	10	10	103	7

graph. The weight of each edge in the affinity graph is computed by: $a_{ij} = \exp\{-\frac{\|x_i - x_j\|^2}{2\gamma^2}\}$, where x_i and x_j are two data samples, and γ is the bandwidth of Gaussian kernel. In our experiments, γ is automatically set in a self-tuning fashion [72]. Normalized graph Laplacian [2] can be immediately derived from the affinity graph by $L = D^{-\frac{1}{2}}AD^{-\frac{1}{2}}$,[8] where D is a diagonal matrix, with $D_{ii} = \sum_{j=1}^{m} a_{ij}$.

In the experiments, we use all the data in each dataset for graph construction, and clustering. We use the following methods and their abbreviations in our experiments: Original Graph [50], Intra-Graph [36], Inter-Graph, Clustered Low Rank (C-LR) [36], Clustered-SVD (C-SVD) [36], Nyström (NYS) [49], Clustered Nyström (C-NYS) [51], LSC [70], MEKA [40], and Ours. Specifically, we compare with Intra-Graph, Inter-Graph, C-LR, and C-SVD on graph approximation experiments, and compare with NYS, C-NYS, LSC, and MEKA on spectral clustering experiments since they are kernel approximation methods suitable for large-scale spectral clustering.

"Original Graph" follows the conventional way to build the sparse affinity graph and normalized graph Laplacian, as stated in [2]. "Intra-Graph" means we directly take the approximations of diagonal blocks in A_α as the approximate sparse graph. Compared to "Intra-Graph", C-LR [36] takes an extra step to extrapolate the approximations of diagonal blocks to off-diagonal blocks, which somehow enhances the performance, especially when the graph is not highly sparse. However, it will dramatically increase the computational burden. "C-SVD" [36] is very close to "C-LR", except that it uses conventional SVD decomposition rather than randomized low-rank approximation for each block in the intra-graph. Inter-Graph is proposed in this paper, and detailed in Section 3.3. "NYS" [49] has been extensively adopted for large-scale spectral clustering, due to its time and spatial efficiency. "MEKA" [40] is very similar to C-LR, but use Nyström instead of randomized low-rank approximation for intra-graph. Besides, it accelerates the computation of inter-graph by extra constraints. "C-NYS" [51] is a special version of "NYS", which takes the cluster clusters as the sampled data for Nyström method. Finally, "LSC" seeks t nearest neighbors of each data sample, and use the similarities between the data and their neighbors as the new feature. Therefore, the affinity graph can be naturally decomposed into two low-rank matrices, which saves huge amount of time for the following eigen-decomposition.

8. We use $D^{-\frac{1}{2}}AD^{-\frac{1}{2}}$ instead of $I - D^{-\frac{1}{2}}AD^{-\frac{1}{2}}$ as the normalized graph Laplacian for the convenience of our proof. This will not change the eigenvectors, but change the corresponding eigenvalues from the first several smallest ones to the leading ones.

In this paper, all the experiments are ran on an Intel i7-3770K PC with 32 GB memory. The software environment is Windows platform + MATLAB 2014b.

5.3 Measurements and Factors

We use following criteria for evaluations of each algorithm:

Approximation error reflects the difference between ground truth and approximate graph by Frobenius norm: $\|A - \hat{A}\|_F$. We further normalized it by the norm of ground truth graph: $\|A - \hat{A}\|_F / \|A\|_F$. Intuitively, higher approximation error means lower accuracy of the approximation.

Clustering accuracy [73] is the average performance of label matching results between resulted labels and ground truth labels for spectral clustering, which can be formulated as

$$\text{Clustering Accuracy} = \max_f \frac{\sum_i \psi(y_i, f(z_i))}{m}, \tag{16}$$

where z_i/y_i is the cluster/class label of x_i, $f(\cdot)$ is a permutation function that maps each cluster label to a class label. $\psi(y_i, z_i)$ is a function equal to 1 if $y_i = z_i$, 0 otherwise.

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

$$\text{NMI} = \frac{\sum_i \sum_j n_{i,j} \log\left(\frac{m \cdot n_{i,j}}{n_i \cdot n_j}\right)}{\sqrt{\left(\sum_i n_i \log \frac{n_i}{m}\right)\left(\sum_j n_j \log \frac{n_j}{m}\right)}}, \tag{17}$$

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.

Space and time consumptions are key indicators in our evaluations. First, we need to check if the space use of our method is smaller than the original graph or competitors. In addition, the running time of ours should be comparable with the original one; otherwise, it is nothing but a trade-off between time and space. To empirically measure the space consumption, we store the matrices for graph reconstruction into binary files and measure their memory consumption on Windows platform. For example, our method needs to store k triplets $\{U_i, \Sigma_i, V_i\}$ and $k(k-1)$ pairs $\{a_{ij}, \Lambda_{ij}\}$, while for Nyström methods, we need to store intermediate matrices W and A.[9] For the actual time complexity, we record the end-to-end running time of each algorithm. Note that theoretical time/space complexity of our algorithm can be fond in Sections 4.3 and 4.4.

9. The matrices have different meanings in [50].

Number of Clusters/Neighbors and Rank. First, the "number of clusters" will affect the integrity of each cluster. If more clusters are assumed, then more inter-cluster relationships are kept in the off-diagonal blocks of the graph matrix. On the other hand, if there are fewer clusters, most inter-cluster relationships are embedded in diagonal blocks of the graph matrix where intra-graph matrix becomes the dominance. Second, the "number of neighbor" is also a critical factor identified by spectral clustering and other graph based dimensionality reduction methods. In general, there is no fixed pattern to follow [74]. However, in most cases, a small number of neighbors yields good results (This saves space as well). In experiments, we vary the number of neighbors and set it relatively small. Finally, "rank" is an essential measurement of the matrix and a higher rank leads to more linearly independent columns/rows. If the target rank is close to the real matrix rank, it may provide a lower approximation error.

5.4 Results and Analysis

5.4.1 Approximation Error and Model Parameters

First, we will illustrate the *approximation error* and *memory consumption* results on four datasets with varied number of clusters/neighbors and ranks in Fig. 4, as each of them will affect our model performance. In addition, as both intra- and inter-graphs are important components of our method, they are illustrated as the comparisons.

On the 1st row of Fig. 4, we vary the number of clusters for small-scale datasets from 10 to 100, and then for mid-scale datasets from 20 to 200. We set the number of neighbors at 30, and rank at 2 (for each block) for both small- and mid-scale datasets. The performance of Inter-Graph and Ours increases gradually with the change of numbers of cluster. More clusters mean the inter-graph is dominant, and more neighbors come from the outside of the cluster, as we discussed in *Theorem 2*. Since the improvement of Inter-Graph is greater than the decrease caused by Intra-Graph, ours still achieves improvements. In addition, the memory consumption of Inter-Graph and Ours keeps going up as more off-diagonal entries will be non-zeros with a larger k.

On the 2nd row of Fig. 4, we keep the number of cluster at 50, and 100 respectively for small- and mid-scale datasets and rank = 2 while vary the number of neighbors for these datasets. We can observe that the number of neighbors do not impact the approximation error so much on intra-graph based methods, but do on Inter-Graph and Ours, as we analyzed in *Theorem 2*. When more neighbors are allowed in the affinity graph, more values can be used for the approximation of inter-graph, therefore, leading to a better performance. This however will consume more memory to approximate the off-diagonal blocks of graph.

On the 3rd row of Fig. 4, we vary the rank of intra-graph, and set cluster = 50 and neighbor = 30 for small-scale datasets, and cluster = 100, neighbor = 30 for mid-scale datasets. For most of the cases, when the target rank for each block is increasing, the Intra-Graph and Ours perform better. However, there are two exceptions from "Corel" and "Segment" datasets, where both Intra-Graph and Ours have decreased performance when rank is approaching 20. The reason is the intra-graph approximation error's upper bound is not a monotonic function of rank r, as we explained in Section 4.1 under *Theorem 1*. Increasing r will enlarge the first factor of this upper bound. When this increase cannot be compensated by the second factor, the approximation error will inevitably increase rather than decrease. In addition, we can see a larger target rank increases the memory consumption, as we discussed in Section 4.4.

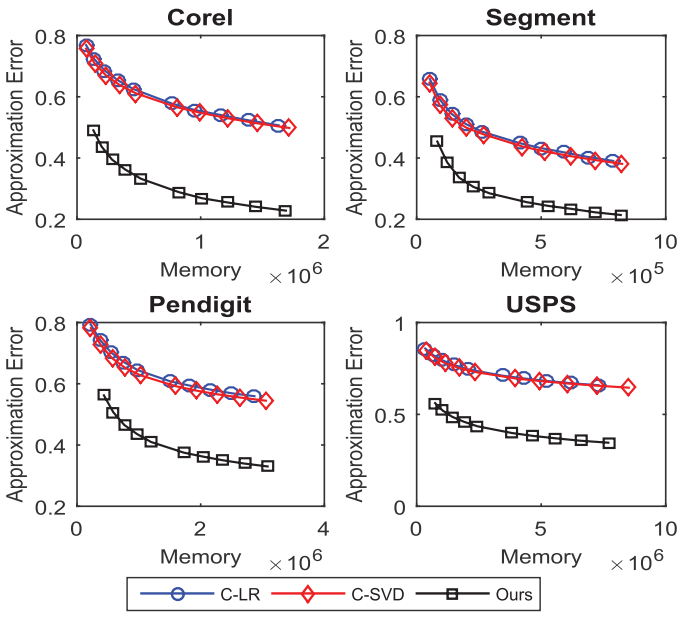

Fig. 5. Experimental results of graph approximation error (↓ better) on eight datasets. The *X*-axis represents the size of memory (Byte), and the *Y*-axis represents the approximation error. We compare three different methods: Clustered-Low-Rank (C-LR), Clustered-SVD (C-SVD), and our method.

Second, in Fig. 5, we fix the number of clusters/neighbors, and compare with the state-of-the-art sparse graph approximation methods: C-LR, C-SVD,[10] by changing the memory use of approximation methods. We tune the target rank of each method, and manage to keep the memory use of all methods falling into similar ranges. Clearly, our method performs better than others. In addition, we can see that C-LR works comparably to C-SVD, demonstrating the practical value of randomized low-rank approximation.

5.4.2 Spectral Clustering Results

Next, we focus on the performance of spectral clustering by the approximate normalized graph Laplacian. Recall that we introduce two popular measurements for evaluations: clustering accuracy and NMI, which reflect different aspects of the algorithm in Figs. 6 and 7. Here we first summarize the end-to-end procedure of using our approximate graph for spectral clustering in Algorithm 2.

Sightly different from the approximation error experiments that only adopt sparse graph as the experiment subject, in spectral clustering experiments, we use both sparse graph (Ours), and dense graphs (NYS, C-NYS, LSC, MEKA) for evaluations, since both of them are compatible with spectral clustering. Although the graph itself has different forms among these methods, we use storage space as a factor to measure the performance of these approximation methods. Note that

10. Truncated, or top-k SVD by quick solver, e.g., ARPACK [34] is also appropriate for the sparse graph approximation. However, it has been proved in [36] that C-LR and C-SVD consistently perform better than it in terms of memory usage.

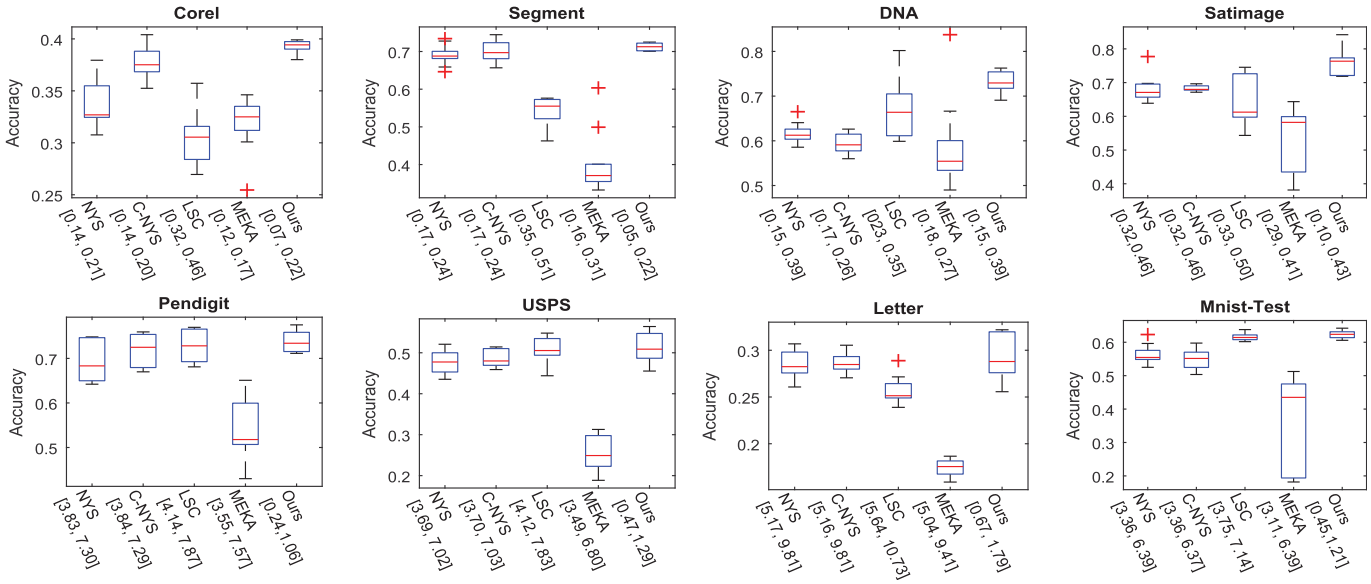

Fig. 6. Experimental results of clustering accuracy (↑ better) with different memory use (Mb) on eight datasets. *X*-axis represents different approximation methods, and their ranges of memory use, and *Y*-axis represents the clustering accuracy featured by box-plot.

we tune the bandwidth of Gaussian kernel for each method independently to ensure their best performance.

Algorithm 2. Spectral Clustering with Approximate Graph

Input: Dataset $X \in \mathbb{R}^{d\times m}$, sparse affinity graph $A \in \mathbb{R}^{m\times m}$, predefined number of partition k, rank r, and number of neighbors t.

Output: Cluster label set $\{z_i\}, i \in [1, m]$.

1: Partition the X by vector/graph clustering approaches.
2: Rearrange the sub-graphs A_{ij} in A according to the partitions of X, i.e., entries in A_{ij} only reflect the similarities of samples from X_i and X_j.
3: Compute the graph Laplacian for A.
4: Approximate A_α and A_β by Sections 3.2 and 3.3.
5: Compute the eigenvectors of $A \approx \hat{A}_\alpha + \hat{A}_\beta$.
6: Use the row-space of the eigenvectors as the new representation for k-means clustering, and obtain new cluster label set $\{z_i\}, i \in [1, m]$.

The way we change the memory use for each method is different: (1) for Ours and MEKA, we gradually increase the target rank of the approximation algorithms; (2) for NYS, and C-NYS, we increase their sampling amounts; (3) for LSC, we gradually increase the number of neighbors used for computing similarities. In fact, we can hardly keep the memory usage of all the methods in the same range due to different strides. Therefore, we only promise to use approximately the same memory over different methods, and keep the unions of these ranges as large as possible. For each method, 10 results with different memory use are computed, and their average performance and standard deviation are shown in the box-plots from Figs. 6 and 7. Note the numbers in the bracket below each box-plot indicate the range of actual memory usage. In most cases, Ours performs comparably with the state-of-the-art methods, in terms of accuracy and NMI, but using less memory, which is clearly shown by the mid-scale datasets' results in Fig. 7.

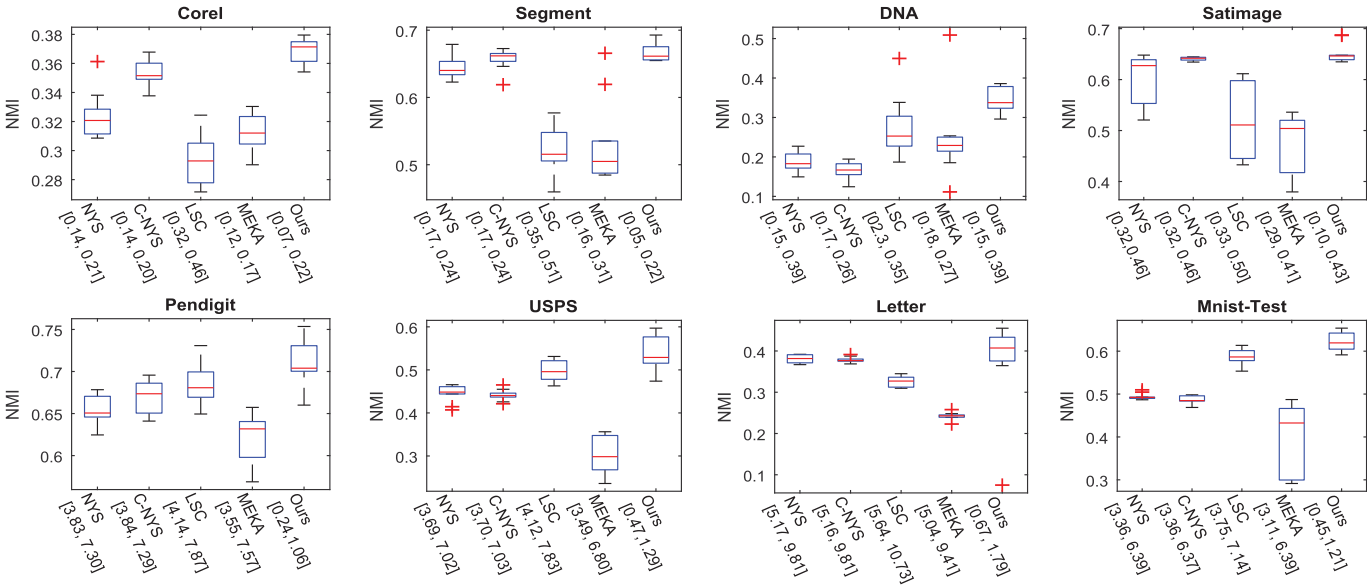

Fig. 7. Experimental results of NMI (↑ better) with different memory use (Mb) on eight datasets. *X*-axis represents different approximation methods, and their ranges of memory use, and *Y*-axis represents the NMI featured by box-plot.

TABLE 4
Large-Scale Experimental Results, Set-1

Methods/DB	Mnist		RCV1		Covtype	
Criteria	Apx	Time	Apx	Time	Apx	Time
Intra-Graph	94.9	**2.3**	97.1	**17.7**	98.3	**252.3**
Cluster-LR	94.7	91.4	96.9	1,948.2	97.9	15,376.4
Cluster-SVD	94.2	117.7	96.7	2,543.5	97.4	18,263.1
Inter-Graph	73.8	3.2	78.8	33.7	82.3	393.5
Ours	**66.7**	5.6	**75.0**	51.5	**79.5**	645.8

For "Approximation Error (Apx, %)" and "Time (s)", lower means better.

TABLE 5
Large-Scale Experimental Results, Set-2

Methods/DB	Mnist		RCV1		Covtype	
Criteria	Acc	Space	Acc	Space	Acc	Space
Original Graph	57.1	24.9	12.6	81.5	**41.3**	180.7
Nyström	49.3	5.03	12.9	58.5	23.5	103.4
C-Nyström	58.6	5.0	15.7	58.3	24.8	102.8
LSC	50.5	449.2	16.6	1,393.1	25.6	4,336.5
MEKA	41.4	4.7	14.5	30.4	22.8	78.2
Ours	**66.8**	**3.2**	**18.3**	**9.1**	27.4	**39.7**

For "Accuracy (Acc, %)" higher means better, while for "Space (MB)" lower means better.

Remarks. From Fig. 4 we can see that the number of clusters k, neighbor t, and rank r will affect the graph approximation error. In addition, we can learn that given limited memory, our method can perform fairly well compared to the existing methods on clustering task, measured by accuracy and NMI. In experiments of different scales, we found that 10-100 clusters work well for small-scale dataset, while 20-200 clusters are good for large-scale dataset, e.g., 580 K data samples. The parameter choosing within these ranges are usually trade-off between memory use and performance. We also suggest a small $r \in [1, 20]$ for the target rank of intra-graph. It is also interesting to find from Figs. 5, 6, and 7 that a moderate approximation error will yield acceptable clustering performance, meaning we may save more memory if using the graph for clustering after compression. Finally, oversampling factor p is empirically set to $\max\{1, 0.5r\}$, providing good results on all the datasets.

5.4.3 Large-Scale Data

We run experiments on three large-scale datasets: Mnist(60 K), RCV1(200 K), Covtype(580 K) and compare with the same methods used in previous experiments for evaluation. Note our method uses the same parameters: clusters = 100, neighbor = 30, rank = 2. For both NYS and C-NYS, the number of sampling is set to 100. For MEKA[11] the target rank is set to 100, and the number of clusters is set to 10. Finally, the number of neighbors in LSC is set to 100. Experimental results are shown in Tables 4 and 5, Fig. 8.

In Table 4, our method achieves better results on approximation error while using comparable time (compared to Intra- and Inter-Graph). This time complexity advantage becomes significant compared to both C-LR and C-SVD, which is caused by the time-consuming outer product of vectors in the extrapolation of C-LR and C-SVD. In addition, since the inter-graph is very sparse, the extrapolation to the inter-graph does not work well, and the improvement from Intra-Graph to either C-LR, or C-SVD is very limited.

In Table 5, our method obtains better results on clustering accuracy (Acc). An interesting phenomenon is that even the original graph does not perform well compared to other graph approximation methods on Mnist and RCV1 datasets. We believe that the huge size of sparse graph degenerates the performance of SVD operation in MATLAB implementation. We should point out our method uses even less memory than those column sampling methods, e.g., NYS, C-NYS because we compress the original sparse graph according to its underlying structure. In addition, we found that although MEKA has shown superior performance in dense graph approximation, its performance on spectral clustering is not good enough. We believe the reason is that it does not well preserve the graph structure. Finally, the most competitive method is LSC in terms of Acc in this experiment; however, it uses more than 100 times of memory (9.1 →1,393.1 Mb) compared to Ours. These facts demonstrate that the proposed method is memory-efficient especially with large-scale data.

In addition, we also conduct experiments to evaluate approximation error by varying the memory use in Fig. 8, where we compare C-LR, C-SVD, and our method. Similar to the results from Fig. 5, using more memory, approximation error is gradually decreased by three methods. When using approximately same amount of memory, our method performs better than the other two with large margins.

5.4.4 Impacts of Graph Partitions

As the first step of the proposed graph approximation method, graph partition will affect the quality of the approximate graph. This is mainly due to the random factors introduced by the clustering methods, e.g., k-means. In the following experiments, we will evaluate such impact factors by running our methods with k-means 50 times for graph approximation on Corel and Letter datasets in Fig. 9

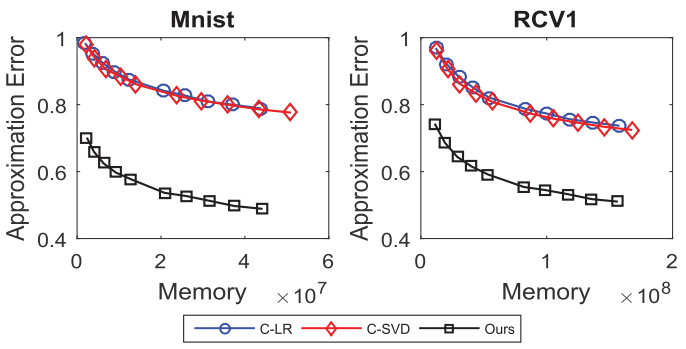

Fig. 8. Experimental results of approximation error (↓ better) with different memory consumption (Byte) on large-scale datasets. *X*-axis represents different memory use, and *Y*-axis represents approximation error.

11. MEKA is very sensitive to the number of clusters, and a large one usually leads to exceptions in eigen-decomposition.

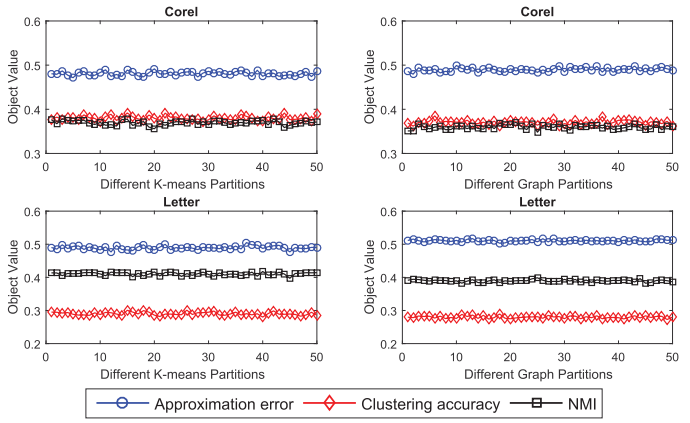

Fig. 9. Experimental results of stableness of graph approximation. Specifically, we run our methods 50 times with different vector/graph clustering results, and report the approximation error (↓ better), clustering accuracy (↑ better), and NMI(↑ better). *X*-axis indexes different trials.

(left). In addition, we also evaluate the difference between the vector clustering, e.g., k-means and graph clustering, e.g., Constrained Laplacian Rank method [63] in Fig. 9 (right). From these results, we can see that vector clustering performs slightly better in terms of average performance on both Corel and Letter datasets while graph clustering performs stable on Letter dataset. This gives us some hints on the performance and stableness of different clustering methods for our graph approximation algorithm.

6 Conclusions

In this paper, we proposed a nearest neighbor sparse graph approximation algorithm by exploiting the underlying graph structure. Through graph partition in the first step, we decomposed the whole graph into intra- and inter-graphs. Then we approximated both intra- and inter-graphs according to their underlying structures, which significantly reduces the computational burden. To theoretically demonstrate the correctness of the proposed method, both intra- and inter-graphs' error bounds and their time/space costs were provided. Finally, we conducted extensive experiments on eleven datasets in different scales, and demonstrated that when using comparable resources (time/space), our method could achieve better performance.

Acknowledgments

This work is supported in part by the US National Science Foundation IIS award 1651902, US National Science Foundation CNS award 1314484, ONR award N00014-12-1-1028, ONR Young Investigator Award N00014-14-1-0484, and US Army Research Office Young Investigator Award W911NF-14-1-0218.

References

[1] J. Shi and J. Malik, "Normalized cuts and image segmentation," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 22, no. 8, pp. 888–905, Aug. 2000.
[2] A. Ng, M. Jordan, and Y. Weiss, "On spectral clustering: Analysis and an algorithm," in *Proc. Advances Neural Inf. Process. Syst.*, 2002, vol. 2, pp. 849–856.
[3] F. Nie, Z. Zeng, I. W. Tsang, D. Xu, and C. Zhang, "Spectral embedded clustering: A framework for in-sample and out-of-sample spectral clustering," *IEEE Trans. Neural Netw.*, vol. 22, no. 11, pp. 1796–1808, Nov. 2011.
[4] F. Nie, D. Xu, and X. Li, "Initialization independent clustering with actively self-training method," *IEEE Trans. Syst. Man Cybern. Part B (Cybern.)*, vol. 42, no. 1, pp. 17–27, Feb. 2012.
[5] F. Nie, X. Wang, and H. Huang, "Clustering and projected clustering with adaptive neighbors," in *Proc. ACM SIGKDD Int. Conf. Knowl. Discovery Data Mining*, 2014, pp. 977–986.
[6] P. Li, J. Bu, L. Zhang, and C. Chen, "Graph-based local concept coordinate factorization," *Knowl. Inf. Syst.*, vol. 43, no. 1, pp. 103–126, 2015.
[7] I. Jolliffe, *Principal Component Analysis*. Hoboken, NJ, USA: Wiley, 2005.
[8] K. Fukunaga, *Introduction to Statistical Pattern Recognition*. Cambridge, MA, USA: Academic, 1990.
[9] X. He and P. Niyogi, "Locality preserving projections," in *Proc. Advances Neural Inf. Process. Syst.*, 2004, vol. 16, pp. 153–160.
[10] X. He, D. Cai, S. Yan, and H. Zhang, "Neighborhood preserving embedding," in *Proc. IEEE Int. Conf. Comput. Vis.*, 2005, vol. 2, pp. 1208–1213.
[11] M. Belkin and P. Niyogi, "Laplacian Eigenmaps for dimensionality reduction and data representation," *Neural Comput.*, vol. 15, no. 6, pp. 1373–1396, 2003.
[12] L. Saul and S. Roweis, "Think globally, fit locally: Unsupervised learning of low dimensional manifolds," *J. Mach. Learn. Res.*, vol. 4, pp. 119–155, 2003.
[13] J. Tenenbaum, V. De Silva, and J. Langford, "A global geometric framework for nonlinear dimensionality reduction," *Science*, vol. 290, no. 5500, pp. 2319–2323, 2000.
[14] S. Yan, D. Xu, B. Zhang, H. Zhang, Q. Yang, and S. Lin, "Graph embedding and extensions: A general framework for dimensionality reduction," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 29, no. 1, pp. 40–51, Jan. 2007.
[15] F. Nie, D. Xu, I. W.-H. Tsang, and C. Zhang, "Flexible manifold embedding: A framework for semi-supervised and unsupervised dimension reduction," *IEEE Trans. Image Process.*, vol. 19, no. 7, pp. 1921–1932, Jul. 2010.
[16] V. Vapnik, *The Nature of Statistical Learning Theory*. Berlin, Germany: Springer, 2000.
[17] B. Schölkopf, A. Smola, and K. Müller, "Nonlinear component analysis as a kernel eigenvalue problem," *Neural Comput.*, vol. 10, no. 5, pp. 1299–1319, 1998.
[18] R. I. Kondor and J. Lafferty, "Diffusion kernels on graphs and other discrete structures," in *Proc. 19th Int. Conf. Mach. Learn.*, 2002, pp. 315–322.
[19] C. Zhang, F. Nie, and S. Xiang, "A general kernelization framework for learning algorithms based on kernel PCA," *Neurocomputing*, vol. 73, no. 4, pp. 959–967, 2010.
[20] F. Nie, S. Xiang, Y. Song, and C. Zhang, "Orthogonal locality minimizing globality maximizing projections for feature extraction," *Optical Eng.*, vol. 48, no. 1, 2009, Art. no. 017202.
[21] B. Cheng, J. Yang, S. Yan, Y. Fu, and T. S. Huang, "Learning with l1-graph for image analysis," *IEEE Trans. Image Process.*, vol. 19, no. 4, pp. 858–866, Apr. 2010.
[22] J. Casas-Roma, J. Herrera-Joancomartí, and V. Torra, "Anonymizing graphs: Measuring quality for clustering," *Knowl. Inf. Syst.*, vol. 44, no. 3, pp. 507–528, 2015.
[23] F. D. Malliaros, V. Megalooikonomou, and C. Faloutsos, "Estimating robustness in large social graphs," *Knowl. Inf. Syst.*, vol. 45, no. 3, pp. 645–678, 2015.
[24] C. Wang, J. Liu, N. Desai, M. Danilevsky, and J. Han, "Constructing topical hierarchies in heterogeneous information networks," *Knowl. Inf. Syst.*, vol. 44, no. 3, pp. 529–558, 2015.
[25] F. R. Chung, *Spectral Graph Theory*. Providence, RI, USA: Amer. Math. Soc., 1997, vol. 92.
[26] W. Dong, C. Moses, and K. Li, "Efficient k-nearest neighbor graph construction for generic similarity measures," in *Proc. 20th Int. Conf. World Wide Web*, 2011, pp. 577–586.
[27] K. Hajebi, Y. Abbasi-Yadkori, H. Shahbazi, and H. Zhang, "Fast approximate nearest-neighbor search with k-nearest neighbor graph," in *Proc. 22nd Int. Joint Conf. Artif. Intell.*, 2011, pp. 1312–1317.
[28] D. Wang, L. Shi, and J. Cao, "Fast algorithm for approximate k-nearest neighbor graph construction," in *Proc. IEEE 13th Int. Conf. Data Mining Workshops*, 2013, pp. 349–356.
[29] S. Fine and K. Scheinberg, "Efficient SVM training using low-rank kernel representations," *J. Mach. Learn. Res.*, vol. 2, pp. 243–264, 2002.

[30] C. Rasmussen and C. Williams, *Gaussian Processes for Machine Learning*. Cambridge, MA, USA: MIT Press, 2006.
[31] S. Kumar, M. Mohri, and A. Talwalkar, "On sampling-based approximate spectral decomposition," in *Proc. 26th Annu. Int. Conf. Mach. Learn.*, 2009, pp. 553–560.
[32] M. W. Mahoney, "Algorithmic and statistical perspectives on large-scale data analysis," *arXiv preprint arXiv:1010.1609*, 2010.
[33] N. Halko, P. Martinsson, and J. Tropp, "Finding structure with randomness: Probabilistic algorithms for constructing approximate matrix decompositions," *SIAM Rev.*, vol. 53, no. 2, pp. 217–288, 2011.
[34] V. Hernandez, J. Roman, A. Tomas, and V. Vidal, "A survey of software for sparse eigenvalue problems," Universitat Politècnica de València, Valencia, Spain, Tech. Rep. STR-6, 2009.
[35] M. Li and J. T.-Y. Kwok, "Making large-scale Nystrom approximation possible," in *Proc. 27th Int. Conf. Mach. Learn.*, 2010, pp. 631–638.
[36] B. Savas and I. Dhillon, "Clustered low rank approximation of graphs in information science applications," in *Proc. SIAM Int. Conf. Data Mining*, 2011, pp. 164–175.
[37] T. Cormen, C. Leiserson, R. Rivest, and C. Stein, *Introduction to Algorithms*. Cambridge, MA, USA: MIT Press, 2001.
[38] T. Zhang, D. Tao, X. Li, and J. Yang, "Patch alignment for dimensionality reduction," *IEEE Trans. Knowl. Data Eng.*, vol. 21, no. 9, pp. 1299–1313, Sep. 2009.
[39] N. Guan, D. Tao, Z. Luo, and B. Yuan, "Manifold regularized discriminative nonnegative matrix factorization with fast gradient descent," *IEEE Trans. Image Process.*, vol. 20, no. 7, pp. 2030–2048, Jul. 2011.
[40] S. Si, C.-J. Hsieh, and I. Dhillon, "Memory efficient kernel approximation," in *Proc. 31st Int. Conf. Mach. Learn.*, 2014, pp. 701–709.
[41] S. Si, C.-J. Hsieh, and I. Dhillon, "A divide-and-conquer solver for kernel support vector machines," in *Proc. 31st Int. Conf. Mach. Learn.*, 2014, pp. 566–574.
[42] A. Gittens and J. Tropp, "Error bounds for random matrix approximation schemes," 2009. [Online]. Available: http://arxiv.org/abs/0911.4108
[43] D. Spielman and N. Srivastava, "Graph sparsification by effective resistances," *SIAM J. Comput.*, vol. 40, no. 6, pp. 1913–1926, 2011.
[44] A. Ruston, "Auerbach's theorem," *Math. Proc. Cambridge Philos. Soc.*, vol. 56, pp. 476–480, 1964.
[45] P. Drineas, A. Frieze, R. Kannan, S. Vempala, and V. Vinay, "Clustering large graphs via the singular value decomposition," *Mach. Learn.*, vol. 56, no. 1, pp. 9–33, 2004.
[46] M. Ouimet and Y. Bengio, "Greedy spectral embedding," in *Proc. Int. Workshop Artif. Intell. Statistics*, 2005, pp. 253–260.
[47] A. K. Farahat, A. Elgohary, A. Ghodsi, and M. S. Kamel, "Greedy column subset selection for large-scale data sets," *Knowl. Inf. Syst.*, vol. 45, no. 1, pp. 1–34, 2015.
[48] F. Bach and M. Jordan, "Predictive low-rank decomposition for kernel methods," in *Proc. 22nd Int. Conf. Mach. Learn.*, 2005, pp. 33–40.
[49] C. Fowlkes, S. Belongie, F. Chung, and J. Malik, "Spectral grouping using the Nystrom method," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 26, no. 2, pp. 214–225, Feb. 2004.
[50] W.-Y. Chen, Y. Song, H. Bai, C.-J. Lin, and E. Y. Chang, "Parallel spectral clustering in distributed systems," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 33, no. 3, pp. 568–586, Mar. 2011.
[51] K. Zhang and J. Kwok, "Clustered Nyström method for large scale manifold learning and dimension reduction," *IEEE Trans. Neural Netw.*, vol. 21, no. 10, pp. 1576–1587, Oct. 2010.
[52] G. Stewart, "Four algorithms for the the efficient computation of truncated pivoted QR approximations to a sparse matrix," *Numerische Mathematik*, vol. 83, no. 2, pp. 313–323, 1999.
[53] H. Cheng, Z. Gimbutas, P. Martinsson, and V. Rokhlin, "On the compression of low rank matrices," *SIAM J. Sci. Comput.*, vol. 26, no. 4, pp. 1389–1404, 2005.
[54] P. Drineas, M. Mahoney, and S. Muthukrishnan, "Relative-error cur matrix decompositions," *SIAM J. Matrix Anal. Appl.*, vol. 30, no. 2, pp. 844–881, 2008.
[55] P. Drineas, R. Kannan, and M. Mahoney, "Fast Monte Carlo algorithms for matrices III: Computing a compressed approximate matrix decomposition," *SIAM J. Comput.*, vol. 36, no. 1, pp. 184–206, 2006.
[56] T. Hofmann, "Probabilistic latent semantic indexing," in *Proc. 22nd Annu. Int. ACM SIGIR Conf. Res. Develop. Inf. Retrieval*, 1999, pp. 50–57.
[57] P.-G. Martinsson, V. Rokhlin, and M. Tygert, "A randomized algorithm for the decomposition of matrices," *Appl. Comput. Harmonic Anal.*, vol. 30, pp. 47–68, 2011.
[58] D. Yan, L. Huang, and M. Jordan, "Fast approximate spectral clustering," in *Proc. 15th ACM SIGKDD Int. Conf. Knowl. Discovery Data Mining*, 2009, pp. 907–916.
[59] R. Duda, P. Hart, and D. Stork, *Pattern Classification*, vol. 2. New York, NY, USA: Wiley, 2001.
[60] S. Dasgupta and Y. Freund, "Random projection trees and low dimensional manifolds," in *Proc. 40th Annu. ACM Symp. Theory Comput.*, 2008, pp. 537–546.
[61] W. Chen, M. Shao, and Y. Fu, "Clustering based fast low-rank approximation for large-scale graph," in *Proc. IEEE Int. Conf. Data Mining Workshops*, 2011, pp. 787–792.
[62] V. D. Blondel, J.-L. Guillaume, R. Lambiotte, and E. Lefebvre, "Fast unfolding of communities in large networks," *J. Statistical Mech.: Theory Experiment*, vol. 2008, no. 10, pp. 1–12, 2008.
[63] F. Nie, X. Wang, M. I. Jordan, and H. Huang, "The constrained Laplacian rank algorithm for graph-based clustering," in *Proc. 13th AAAI Conf. Artif. Intell.*, 2016, pp. 1969–1976.
[64] F. Nie, J. Yuan, and H. Huang, "Optimal mean robust principal component analysis," in *Proc. 31st Int. Conf. Mach. Learn.*, 2014, pp. 1062–1070.
[65] G. Stewart and J. Sun, *Matrix Perturbation Theory*. Cambridge, MA, USA: Academic, 1990.
[66] R. Gray, "Vector quantization," *IEEE ASSP Mag.*, vol. 1, no. 2, pp. 4–29, Apr. 1984.
[67] S. Lloyd, "Least squares quantization in PCM," *IEEE Trans. Inf. Theory*, vol. IT-28, no. 2, pp. 129–137, Mar. 1982.
[68] A. M. Davie and A. J. Stothers, "Improved bound for complexity of matrix multiplication," *Proc. Royal Soc. Edinburgh: Section A Math.*, vol. 143, no. 2, pp. 351–369, 2013.
[69] F. Woolfe, E. Liberty, V. Rokhlin, and M. Tygert, "A fast randomized algorithm for the approximation of matrices," *Appl. Comput. Harmonic Anal.*, vol. 25, no. 3, pp. 335–366, 2008.
[70] X. Chen and D. Cai, "Large scale spectral clustering with landmark-based representation," in *Proc. 25th AAAI Conf. Artif. Intell.*, 2011, pp. 313–318.
[71] D. D. Lewis, Y. Yang, T. G. Rose, and F. Li, "RCV1: A new benchmark collection for text categorization research," *J. Mach. Learn. Res.*, vol. 5, pp. 361–397, 2004.
[72] L. Zelnik-Manor and P. Perona, "Self-tuning spectral clustering," in *Proc. Advances Neural Inf. Process. Syst.*, 2004, pp. 1601–1608.
[73] M. Wu and B. Schölkopf, "A local learning approach for clustering," in *Proc. Advances Neural Inf. Process. Syst.*, 2006, pp. 1529–1536.
[74] U. Von Luxburg, "A tutorial on spectral clustering," *Statistics Comput.*, vol. 17, no. 4, pp. 395–416, 2007.

Ming Shao received the BE degree in computer science, the BS degree in applied mathematics, and the ME degree in computer science from Beihang University, Beijing, China, in 2006, 2007, and 2010, respectively. He received the PhD degree in computer engineering from Northeastern University, Boston, Massachusetts, in 2016. He is a tenure-track assistant professor affiliated with the College of Engineering, University of Massachusetts Dartmouth, since 2016 Fall. His current research interests include low-rank matrix analysis, deep learning, and social media analytics. He received the Presidential Fellowship of State University of New York at Buffalo from 2010 to 2012, and the best paper award winner of IEEE ICDM 2011 Workshop on Large Scale Visual Analytics. He is a member of the IEEE.

Xindong Wu received the BS and MS degrees in computer science from the Hefei University of Technology, China, and the PhD degree in artificial intelligence from the University of Edinburgh, Britain. He is a Alfred and Helen Lamson Endowed professor of computer science with the University of Louisiana at Lafayette. His research interests include data mining, big data analytics, knowledge-based systems, and web information exploration. He is currently the steering committee chair of the IEEE International Conference on Data Mining (ICDM), the editor-in-chief of the *Knowledge and Information Systems* (KAIS, by Springer), and a series editor-in-chief of the Springer Book Series on *Advanced Information and Knowledge Processing* (AI&KP). He was the editor-in-chief of the *IEEE Transactions on Knowledge and Data Engineering* between 2005 and 2008. He served as program committee chair/co-chair of the 2003 IEEE International Conference on Data Mining, the 13th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, and the 19th ACM Conference on Information and Knowledge Management. He is a fellow of the IEEE and the AAAS.

Yun Fu (S'07-M'08-SM'11) received the BEng degree in information engineering and the MEng degree in pattern recognition and intelligence systems from Xi'an Jiaotong University, China, respectively. He received the MS degree in statistics and the PhD degree in electrical and computer engineering from the University of Illinois at Urbana-Champaign. He is an interdisciplinary faculty member affiliated with the College of Engineering and the College of Computer and Information Science, Northeastern University, since 2012. His research interests include machine learning, computational intelligence, big data mining, computer vision, pattern recognition, and cyber-physical systems. He has extensive publications in leading journals, books/book chapters and international conferences/workshops. He serves as and associate editor, chairs, PC member, and reviewer of many top journals and international conferences/workshops. He received seven Prestigious Young Investigator Awards from NAE, ONR, ARO, IEEE, INNS, UIUC, Grainger Foundation; seven Best Paper Awards from IEEE, IAPR, SPIE, SIAM; three major Industrial Research Awards from Google, Samsung, and Adobe, etc. He is currently an associate editor of the *IEEE Transactions on Neural Networks and Leaning Systems*. He is a fellow of the IAPR, a lifetime senior member of the ACM and the SPIE, lifetime member of the AAAI, the OSA, and the Institute of Mathematical Statistics, member of the Global Young Academy (GYA), the INNS, and Beckman Graduate fellow during 2007-2008. He is a senior member of the IEEE.