Pattern Recognition 44 (2011) 222–235

Contents lists available at ScienceDirect

Pattern Recognition

journal homepage: www.elsevier.com/locate/pr

Approximate pairwise clustering for large data sets via sampling plus extension

Liang Wang [a,*], Christopher Leckie [b], Ramamohanarao Kotagiri [b], James Bezdek [b]

[a] National Lab of Pattern Recognition, Institute of Automation, Chinese Academy of Sciences, Beijing 100190, China
[b] Department of Computer Science and Software Engineering, The University of Melbourne, Parkville, Victoria 3010, Australia

ARTICLE INFO

Article history:
Received 31 August 2009
Received in revised form
19 April 2010
Accepted 7 August 2010

Keywords:
Pairwise data
Selective sampling
Spectral clustering
Graph embedding
Out-of-sample extension

ABSTRACT

Pairwise clustering methods have shown great promise for many real-world applications. However, the computational demands of these methods make them impractical for use with large data sets. The contribution of this paper is a simple but efficient method, called eSPEC, that makes clustering feasible for problems involving large data sets. Our solution adopts a "*sampling, clustering plus extension*" strategy. The methodology starts by selecting a small number of representative samples from the relational pairwise data using a *selective sampling* scheme; then the chosen samples are grouped using a *pairwise clustering* algorithm combined with *local scaling*; and finally, the label assignments of the remaining instances in the data are extended as a classification problem in a low-dimensional space, which is explicitly learned from the labeled samples using a *cluster-preserving* graph embedding technique. Extensive experimental results on several synthetic and real-world data sets demonstrate both the *feasibility* of approximately clustering large data sets and *acceleration* of clustering in loadable data sets of our method.

1. Introduction

As an exploratory data analysis tool, clustering aims to group objects of a similar kind into their respective categories. Various clustering algorithms have been developed and used on data of varying types and sizes (see [4] for a comprehensive survey). In general, the set of objects in the data may be described by either *object data* or *relational data*. Let $\mathcal{O}=\{o_1,\ldots,o_N\}$ denote a set of N objects (e.g., flowers, beers, etc.). Object data generally have the form $\mathcal{F}=\{\boldsymbol{f}_1\boldsymbol{f}_2,\ldots\boldsymbol{f}_N\}$, $\boldsymbol{f}_i\in\mathcal{R}^v$, where each object o_i is represented by a v-dimensional feature vector $\boldsymbol{f}_i$, while relational data are usually represented by an $N\times N$ dissimilarity (or similarity) matrix $\mathbf{D}_N$, in which each element d_{ij} describes some relation between objects o_i and o_j. It is always possible to convert $\mathcal{F}$ into $\mathbf{D}_N$ by computing pairwise distances $d_{ij}=\|\boldsymbol{f}_i-\boldsymbol{f}_j\|$ in any vector norm on $\mathcal{R}^v$. Generally, $\mathbf{D}_N$ satisfies $0\le d_{ij}\le 1; d_{ij}=d_{ji}; d_{ii}=0$, for $1\le i,j\le N$.

Compared with traditional *central clustering* methods such as c-means and Gaussian Mixture Model (GMM) fitting [1], *relational clustering* methods are more general in the sense that they are applicable to situations in which the objects to be clustered may be not representable in terms of feature vectors. Naturally, a pairwise relational data representation also allows the use of more flexible metrics for measuring proximities in sets of objects, e.g., the χ^2 distance for comparing histogram-based feature vectors [28] and the Hausdorff distance for computing the dissimilarity of any two action sequences of different durations [24]. In addition, pairwise relational clustering keeps the algorithm generic and independent of specific data representations [10]. In particular, pairwise data clustering is more practical when groups of similar objects cannot be represented effectively by a single prototype (e.g., centroid), and has been demonstrated to be advantageous in applications that involve highly complex clusters.

Pairwise data clustering has its roots in psychology and bioinformatics. The earliest method of this type was the graph-theoretic method that Cattell called single linkage [34]. Many pairwise clustering methods have been proposed in the recent literature [5,7,8,10,9,23]. However, partitioning pairwise relational data is generally considered a much harder problem than clustering in object vectorial data since the inherent structure of the data is hidden in N^2 pairwise relations. In addition, pairwise clustering algorithms cannot generally handle large data sets efficiently because of their huge *computation* and *storage* requirements. For example, spectral clustering methods are computationally expensive (or infeasible) for very large data sets since they rely on the eigendecomposition of an $N\times N$ similarity matrix (note that comparing all possible pairs of distances between N objects in a large data set is computationally expensive in itself). There are always data sets that are too large to be efficiently analyzed using traditional clustering techniques with readily available computing resources. As data sets become larger and more varied, additional strategies to extend the existing algorithms to adapt to the growing data sizes are thus required to maintain both cluster quality and speed.

One way to attack this problem is "extensibility". A *literal clustering* scheme directly applies the clustering algorithm

* Corresponding author.
E-mail address: wangliangnlpr@gmail.com (L. Wang).

0031-3203/$ - see front matter © 2010 Elsevier Ltd. All rights reserved.
doi:10.1016/j.patcog.2010.08.005

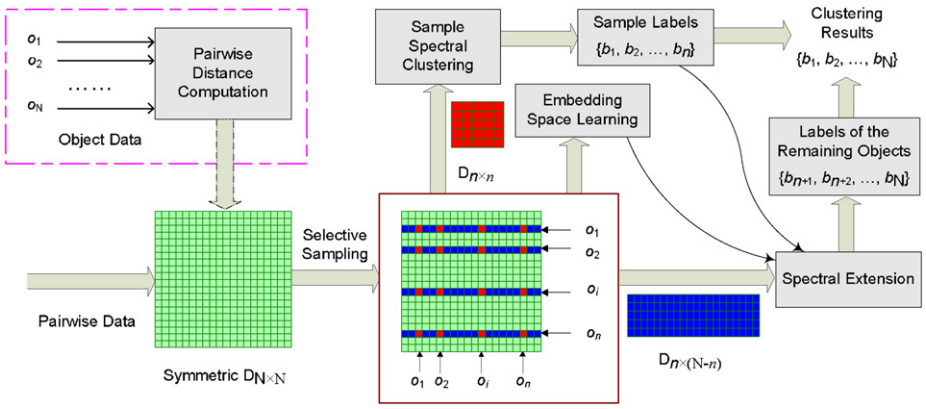

Fig. 1. Architecture of our approximate relational clustering method. Note that if the object data is given, it is possible to transform them to pairwise relational data by comparing all possible pairs of distances, as highlighted by the top-left box (dashed line).

without any modifications to the full data set. In contrast, an *extended clustering* scheme applies a clustering algorithm to a representative and manageably sized sample set of the full data, and then non-iteratively extends the results from this sample set to obtain (approximate) clusters for the remaining data [3]. The main objective of this paper is to develop such an *extensible pairwise clustering* method for approximate clustering of large data sets in an accurate and efficient manner. In other words, given the pairwise relational data matrix $\mathbf{D}_N$ corresponding to $\mathcal{O}$ (i.e., where a vectorial representation of the original objects is not necessarily available), we wish to partition the data set into c groups, i.e., $C_1,\ldots,C_c$, so that $C_i \cap C_j = \emptyset$ if $i \neq j$ and $C_1 \cup C_2 \cup \cdots \cup C_c = \mathcal{O}$. Our proposed method is performed in a "sampling, clustering plus extension" manner, as shown in Fig. 1. First, a selective sampling scheme selects a small number of representative samples from the full data set. Then the selected samples are clustered using a spectral clustering algorithm coupled with a local scaling scheme. Finally, the labeled samples are used to learn a compact embedding space using a cluster-preserving graph embedding technique, in which the labels of the remaining examples in the data set can be effectively predicted as a classification problem, i.e., by assigning out-of-sample examples to the previously determined clusters. An empirical evaluation on both synthetic and real-world data sets demonstrates that our method is not only feasible for large data partitioning problems where literal clustering fails, but also drastically reduces the computational burden associated with the processing of large data sets.

The remainder of the paper is organized as follows. Section 2 briefly reviews related work. Sections 3 describes our approximate relational clustering method including three steps, namely selective sampling, sample clustering and out-of-sample spectral extension. Experimental results are given and analyzed in Section 4, prior to a discussion and conclusion in Section 5.

2. Related work

With the increase in data sizes, it is becoming increasingly important to develop large-scale clustering techniques for clustering large data sets. Key points of *sampling-based* algorithms [17,18,20] are how to choose an appropriate number of samples to maintain the important geometrical properties of clusters, and how to extend the sample results to the remainder of the data. *Incremental* algorithms [16,19] load a subset of the data that fits into main memory at one time for clustering, and keep sufficient statistics or past knowledge of clusters from a previous run for use in improving the model incrementally for the remaining data. *Distributed* clustering algorithms [36,35] usually take into consideration that the data may be inherently distributed to different loosely coupled sites connected through a network. However, most of the existing large-scale algorithms are restricted to object (vectorial) data. In this paper, our concern is to extend a pairwise data clustering algorithm to large relational data sets. Clustering of large relational data sets has received little attention. The following reviews related work.

There have been a number of pairwise data clustering methods in the recent literature [30,7,8,23]. Hofmann and Buhmann [7] proposed to perform pairwise data clustering by using deterministic annealing. Roth et al. [8] reformulated the pairwise data representation in terms of a vectorial data representation by using a constant shift embedding. In particular, a family of spectral clustering algorithms [30] has been widely studied and used, e.g., for image segmentation [6] and trajectory pattern learning [29]. However, an important problem associated with pairwise data grouping algorithms is their scaling behavior with the number of data items N, in terms of their memory limitations and computational overheads, which greatly hinders their applicability to very large data sets. Several recent attempts, e.g., dominant sets [10], the methods based on the Nyström approximation [11,14] and eNERF [3,22], have been proposed to deal with this problem.

Dominant sets [10] generalize the notion of a maximal clique to edge-weighted graphs and have non-trivial connections to continuous quadratic optimization and spectral grouping. However, it would require a significant amount of time to use a complete graph to find dominant sets. To deal with large data sets, out-of-sample extensions of dominant-set clusters are proposed in [10]. First a small number of samples are selected *randomly* to find the dominant sets; then a label prediction for new examples is found by computing an approximation of the degree of cluster membership. However, the prediction (or classification) scheme for new examples inherits the potential drawbacks of the computationally complex dominant set formulation. Also, it may happen that for some instances there is no cluster that satisfies the prediction rule (and thus no assignment).

A spectral grouping approach using the Nyström method is proposed in [11]. The approach is based on first solving a small-scale eigenvalue problem with *randomly* chosen sample data, followed by computing approximated eigenvectors by extrapolation, and finally using them to perform classical c-means clustering. Strictly speaking, the approach does not fall into the category of approaches that are based on "sampling, clustering plus extension", as used in this paper and other previous works

[3,22,10]. Another similar work based on the Nyström method, for out-of-sample extensions but not for clustering itself, is described in Bengio et al. [14], in which a new example is mapped as a linear weighted combination of the corresponding eigenfunctions for the training samples. When extending the mapping, it is necessary to simultaneously extend the kernel function. However, the extension of the kernel is not trivial, especially when the kernel itself is an unknown function defined in the feature space.

The eNERF algorithm described in [3] performs approximate clustering for large pairwise relational data. Instead of the use of simple random sampling (as used in [11,10]), this method adopts a *progressive sampling* (PS) procedure based on a statistical test to extract the representative examples from the pairwise data. Then the method performs sample clustering with non-Euclidean relational fuzzy *c*-means (NERF), followed by indirectly extending the clusters to the remainder of the data with an iterative procedure. However, progressive sampling often results in significant over-sampling [22]. Sample clustering in the eNERF algorithm is based on a centering clustering algorithm, i.e., the relational fuzzy *c*-means algorithm, and hence, will probably fail for clusters with non-hyperellipsoidal shapes.

It should be mentioned that an incremental spectral clustering algorithm is proposed to handle *dynamic* changing data in [21]. This method targets a class of applications (e.g., monitoring of evolving communities such as the websphere and blogsphere), which need to handle not only insertion or deletion of data points but also similarity changes between existing items over time. The method extends standard spectral clustering to adapt to evolving data by introducing the incidence vector/matrix to represent two kinds of dynamics in the same framework and by incrementally updating the eigenvalue system. Computational efficiency is improved at the expense of lower cluster quality.

In this paper, our aim is to develop an extended pairwise clustering method to partition large "*static*" data sets using a "*sampling plus extension*" strategy. We choose spectral clustering as the basis of our method due to its continuing success in many recent applications. Accordingly, we call this method eSPEC (extensible Spectral Clustering). Motivations for such a study can be summarized as follows: When the data set is large or very large and unloadable given the available computing resources, sampling plus extension can offer an approximate clustering solution i.e., makes clustering feasible, whereas it is impossible to use the literal approach alone. If the data set is small, medium-sized, or merely large but still loadable, then an extended scheme may offer an approximate solution comparable to the literal solution at a significantly reduced computational cost. That is, it accelerates the literal scheme. The benefits of an extended clustering scheme in these two cases can be summarized as *feasibility* for very large data and *acceleration* for large data, as depicted in Table 1. A fundamental difference between these two cases involves the calculation of an *approximation error*. For case I, we can assess the approximation error by measuring the difference between the clustering results obtained using the corresponding extended and literal schemes. For case II, the only solution available is that obtained by the extended scheme, in which case the approximation error cannot be measured. As mentioned before, key points of *sampling-based* algorithms are how to choose an appropriate number of representative samples to maintain the important geometrical properties of the underlying clusters as much as possible, and how to extend the clustering results from the sample set to the remainder of the data. In this work, we specifically propose to use an elegant combination of selective sampling with graph-embedding-based out-of-sample extension, which will be described in the following sections.

Table 1
Characteristics of the extended clustering scheme for large (R_L) and very large (R_{VL}) data sets.

Case	Property	Objective	Approximate error
I (R_L)	Loadable	Acceleration	Measurable
II (R_{VL})	Unloadable	Feasibility	Immeasurable

3. Our method

3.1. Selective sampling

Using a sample set from the full data set can speed up the clustering process, but this is only acceptable if it does not reduce the quality of the discovered knowledge. Most data reduction techniques are based on statistical sampling, such as uniform random sampling, stratified sampling or non-uniform probabilistic sampling [2]. There have also been some methods that incorporate random sampling with adaptive procedures involving a specific data mining tool such as decision trees [32,33]. However, there have been few studies on direct sampling of pairwise relational data. Simple *random sampling* (RS) may work well when the sample size is sufficiently large. But a probably unnecessarily large sample size will naturally increase the computational workload. We wish to take few representative samples while still achieving satisfactory performance.

For this purpose, we use a modification of the *selective sampling* (SS) scheme developed in [22]. This heuristic sampling scheme was shown to be superior to the progressive sampling scheme in [3] (which has a tendency to over-sample) and to random sampling (when the sample size is not sufficient). In addition, the SS algorithm is computationally efficient, thus allowing the processing of truly large data sets. In brief, our modified selective sampling scheme first selects h distinguished objects (using a max–min farthest point strategy to ensure that they are mutually far away from each other) from the dissimilarity matrix $\mathbf{D}_N$, which are used as cluster seeds to guide the sampling process. Next, each object in $\{o_1,o_2,\ldots,o_N\}$ is associated with its nearest distinguished object. The final step of SS randomly selects a small number of samples from each group R_i ($i=1,2,\ldots,h$), where R_i denotes the set of objects grouped with the i-th distinguished object. The sampling algorithm is summarized as follows.

Selective sampling
Input: $\mathbf{D}_N$, an $N\times N$ pairwise dissimilarity matrix of N objects; h, the number of distinguished objects; and n, the number of the samples to be chosen.
Output: $\mathbf{D}_n$, an $n\times n$ matrix which is a submatrix of $\mathbf{D}_N$ corresponding to the row/column indices in the sample set $\mathcal{S}$.

1. Select the indices $p_1,\ldots,p_h$ of the h distinguished objects from the rows of $\mathbf{D}_N$.
   - Randomly select the first index from the index set $\{1,2,\ldots,N\}$, e.g., $p_1=1$, without loss of generality.
   - Initialize the search array by $\boldsymbol{s}=(s_1,\ldots,s_N)=(d_{1,1},\ldots,d_{1,N})$.
   - Successively update $\boldsymbol{s}$ to $(\min\{s_1,d_{p_{i-1},1}\},\ldots,\min\{s_N,d_{p_{i-1},N}\})$ for $i=2,\ldots,h$, and select $p_i=\mathrm{argmax}_j\{s_j\}$.
2. Associate each object in $\{o_1,\ldots,o_N\}$ with its nearest cluster seed according to the dissimilarities (note that this is only a coarse pre-clustering process based on d_{ij}).
   - Initialize the index set of each of the respective distinguished objects $R_1=R_2=\cdots=R_h=\emptyset$.
   - For $i=1$ to N, select $q=\mathrm{argmin}_{1\le j\le h}\{d_{p_j,i}\}$ and accordingly update $R_q=R_q\cup\{i\}$.

Select the sample data from each group $R_i (i = 1, \ldots, h)$.

3. • For $i = 1, \ldots, h$, compute a sub-sample size of the i-th group $n_i = \lfloor n \cdot |R_i|/N \rfloor$, and randomly select n_i indices from R_i without replacement.
   • Let the sample set $\mathcal{S}$ denote the union of all the randomly selected indices and define $n = |\mathcal{S}|$.

We note the following points: (1) $\mathbf{D}_N$ does not need to be fully loaded into memory but just a small portion $h \times N$ needs to be loaded. Moreover, if the input data begin as object data, only a $h \times N$ distance matrix is computed for the sampling process. For subsequent processing, we only need to load $\mathbf{D}_{n \times n}$ for sample clustering, and $\mathbf{D}_{n \times (N-n)}$ for out-of-sample extension (fully loaded or incrementally loaded depending on the computational platform). (2) If N is very large, we may select a subset sampled uniformly and randomly from $\mathbf{D}_N$ to act as the input to the SS algorithm. This is necessary for handling truly very large data sets, and it is plausible because, in general, the number of clusters $c \ll N$ (i.e., unnecessarily many examples basically provide redundant information to characterize the structure of the whole data set). (3) If a set of objects $\mathcal{O}$ can be partitioned into $c \geq 1$ *compact* and *separated* (CS) clusters, and if $h \geq c$, then this sampling algorithm will select at least one distinguished object from each cluster, which provides a guarantee that the selected samples do not miss a potential cluster (see [3] for the proof). In addition, the proportion of objects in the sample set from each cluster approximately equals the proportion of objects in the population from the same cluster. (4) To obtain more representative samples, we may set h to an overestimate of the true but unknown number of clusters (i.e., $h \geq c$). It might be appropriate that h be set to a larger value in order to sufficiently capture irregular-shaped data structures. Thus a group (possibly with a complex shape) in the data might be approximated by more subgroups. As h increases, the likelihood that the sampling algorithm would include enough representative examples improves.

Note that the step of identifying the distinguished objects requires $O(hN)$ time. The step of classifying each object to its nearest distinguished object requires $O(N)$ time. Thus the time complexity of this algorithm is $O(hN)$. If the available data is just object data in $\mathcal{R}^v$, then the acquisition of the required elements in the first step and the computation of $\mathbf{D}_n$ from original object data have additional time requirements of $O(vhN)$ and $O(vn^2)$. In this case, the runtime complexity is $O(vhN+vn^2)$ or $O(\max(vhN, vn^2))$. In summary, this sampling scheme is scalable since the runtime complexity is linear in N.

3.2. Sample clustering

Two commonly used clustering methods for object data, c-means and fitting a GMM via the EM algorithm, are inapplicable when we have no original vectorial representations of objects (e.g., where only $\mathbf{D}_n$ is available). Spectral clustering is a powerful method for finding complex structure in data using spectral properties of a pairwise similarity matrix [30]. Moreover, manifold learning [12,15] and spectral clustering are intimately related because the clusters that spectral clustering manages to capture can be arbitrarily curved manifolds, which provides a rational basis for our out-of-sample extension strategy (see Section 3.3).

Let us regard $\mathbf{D}_n$ as a complete weighted graph $\mathcal{G}(\mathbf{V}, \mathbf{A})$ having a set of nodes $\mathbf{V}$ corresponding to n sample objects. The edge weights are defined by a $n \times n$ symmetric affinity matrix $\mathbf{A}$, whose element A_{ij} represents the relation of the edge connecting nodes i and j. Generally, the distance matrix $\mathbf{D}_n$ may be transformed into $\mathbf{A}$ by $A_{ij} = \exp(-d_{ij}^2/\sigma^2)$, where σ is a scale parameter that controls how rapidly the affinity A_{ij} falls off with the distance d_{ij} between objects o_i and o_j. With this representation, the spectral clustering problem can be reformulated as a graph cut problem, such as normalized cut [6] and min–max-cut [31]. For clustering the sample data $\mathbf{D}_n$, we use a clustering algorithm described in [5]. How to choose an optimal scale parameter σ to construct a "good" affinity matrix $\mathbf{A}$ from $\mathbf{D}_n$ is critical. If the data set has different local statistics between clusters or a cluttered background, it has been shown in [9] that a local scaling method can outperform the use of a global scale parameter. For this reason, we use a local scale parameter in our approach. The sample clustering algorithm is summarized as follows.

Sample clustering

Input: $\mathbf{D}_n$, an $n \times n$ pairwise dissimilarity matrix (corresponding to the set of n sample objects in $\mathcal{S}$).
Output: A set of (crisp) labels for the n sample objects in $\mathcal{S}$, i.e., $\{b_1, b_2, \ldots, b_n\}$, where $b_i \in \{1, 2, \ldots, c\}$.

1. Compute a local scale σ_i for each object o_i in $\mathcal{S}$ using $\sigma_i = d(o_i, o_r) = d_{ir}$ where o_r is the r-th nearest neighbor of o_i.
2. Form the affinity matrix $\mathbf{A} \in \mathcal{R}^{n \times n}$ as $A_{ij} = \exp(-d_{ij}d_{ji}/\sigma_i\sigma_j)$ for $i \neq j$, and $A_{ii} = 0$.
3. Define $\mathbf{H}$ to be a diagonal matrix with $H_{ii} = \sum_{j=1}^{n} A_{ij}$, and construct the normalized Laplacian matrix $\mathcal{L} = \mathbf{H}^{-1/2}\mathbf{A}\mathbf{H}^{-1/2}$.
4. Find $\boldsymbol{e}_1, \boldsymbol{e}_2, \ldots, \boldsymbol{e}_c$, the c largest eigenvectors of $\mathcal{L}$ and form the matrix $\mathbf{E} = [\boldsymbol{e}_1, \ldots, \boldsymbol{e}_c] \in \mathcal{R}^{n \times c}$ by stacking the eigenvectors in columns. Then re-normalize the rows of $\mathbf{E}$ to unit length to generate $\mathbf{T} \in \mathcal{R}^{n \times c}$.
5. For $i = 1, 2, \ldots, n$, let $\boldsymbol{t}_i \in \mathcal{R}^c$ be the vector corresponding to the i-th row of $\mathbf{T}$, and cluster these $\boldsymbol{t}_i$ into c groups $B_1, \ldots, B_c$ via the c-means method.
6. Assign each object o_i to cluster j if and only if the corresponding row i of $\mathbf{T}$ was assigned to cluster j, thus obtaining final clusters $C_1, \ldots, C_c$ with $C_j = \{i | \boldsymbol{t}_i \in B_j\}$.

The scale parameter σ_i, or r here, is important for the results of spectral clustering, but hard to determine optimally. In real applications, r should not be generally set to a very large value in order to preserve good locality. For this sample clustering algorithm, the runtime complexity depends on three major steps, i.e., computing the local scale σ_i, the eigendecomposition of the normalized Laplacian matrix and performing c-means. The corresponding runtime complexities for these three steps are, respectively, $O(rn^2)$, $O(n^3)$ and $O(i_{max}c^2n)$, where i_{max} is the maximum iteration number, and in the spectral embedding space, "new" instances corresponding to original objects have c dimensions, in the worst case. Thus the total time complexity is $O(n^3 + rn^2 + i_{max}c^2n)$.

3.3. Out-of-sample extension

Next we need to address the problem of grouping out-of-sample objects. We are supplied with an $n \times (N-n)$ matrix $\mathbf{D}_{N-n}$ consisting of the pairwise dissimilarities between the n samples and the remaining $N-n$ objects in the data. We need to assign each of the remaining $N-n$ objects to one of the c previously determined clusters. In [10,8], the authors considered the assignment of labels of unseen data as a classification problem. The extension stage of our method follows this principle. We consider the whole $n \times N$ matrix of pairwise dissimilarities between N objects $[\mathbf{D}_n \ \mathbf{D}_{N-n}]$ as a new semantically meaningful vectorial representation, i.e., each object has n attributes, each of which corresponds to a dissimilarity relation between the object and one of the n previously sampled objects. It is plausible to assume that

such feature vectors from the same class share the same label more often than not. This intuitive observation is our rationale for the use of the classification method to estimate the labels of unlabeled data points in the cluster-preserving embedding space.

For this purpose, we wish to learn an explicit mapping using the sample data $\mathbf{D}_n$ so that we can obtain the embedding representations of unseen data in a new feature space, and accordingly provide a straightforward extension for out-of-sample examples. A non-linear manifold discovery method that is very similar to the mapping procedure used in spectral clustering algorithms is *Laplacian Eigenmaps* (LE) [12]. We adopt its computationally efficient linear version, namely *Locality Preserving Projections* (LPP) [13], for explicitly learning the cluster-preserving embedding space from $\mathbf{D}_n$. More crucially, LPP are defined everywhere in the feature space rather than just at the training sample points, so it is clear how to evaluate the map for new non-sample points. The locality preserving property of LPP also makes it possible that a nearest neighbor search in the low-dimensional embedding space yields similar results to that in the high-dimensional input space. The out-of-sample spectral extension is summarized as follows.

Out-of-Sample Spectral Extension
Input: $\mathbf{D}_n$, the dissimilarity matrix corresponding to the set of labeled sample objects (which we re-denote $\mathbf{D}_n=[\boldsymbol{x}_1,\boldsymbol{x}_2,\ldots,\boldsymbol{x}_n],\boldsymbol{x}_i\in\mathcal{R}^n$ for convenience of description), and the dissimilarity matrix $\mathbf{D}_{N-n}=[\boldsymbol{x}_1^e,\boldsymbol{x}_2^e,\ldots,\boldsymbol{x}_{N-n}^e],\boldsymbol{x}_i^e\in\mathcal{R}^n$ consisting of the pairwise dissimilarity values between the n sample objects and the remaining $N-n$ objects.
Output: A complete set of (crisp) labels of N objects in the data $\mathcal{O}$, i.e., $\{b_1,b_2,\ldots,b_n,b_{n+1},\ldots,b_N\}$, where $b_i\in\{1,2,\ldots,c\}$.

1. Constructing the adjacency graph: Let $\mathcal{G}$ denote an undirected graph with n nodes corresponding to the n labeled samples. An edge occurs between nodes i and j if $\boldsymbol{x}_i$ and $\boldsymbol{x}_j$ are "close", according to K-nearest neighbors ($K\in\mathcal{N}$) (i.e., $\boldsymbol{x}_i$ is among the K-nearest neighbors of $\boldsymbol{x}_j$ or if $\boldsymbol{x}_j$ is among the K-nearest neighbors of $\boldsymbol{x}_i$).
2. Weighting the edges: Let $\mathbf{W}$ be a symmetric $n\times n$ matrix. Its element W_{ij} is the weight of the edge joining the nodes i and j, and is 0 if there is no such edge. $\mathbf{W}$ is sparse as most weights are 0. We use the cosine similarity to compute the edge weights, i.e., $W_{ij}=(\boldsymbol{x}_i\cdot\boldsymbol{x}_j)/(|\boldsymbol{x}_i|\cdot|\boldsymbol{x}_j|)$.
3. Eigenmaps: Solve the generalized eigenvector problem $\mathbf{XLX}^T\mathbf{u}=\lambda\mathbf{XHX}^T\mathbf{u}$, where the i-th column of the matrix $\mathbf{X}$ is $\boldsymbol{x}_i$, $\mathbf{L}=\mathbf{H}-\mathbf{W}$ is the graph Laplacian and $\mathbf{H}$ is a diagonal matrix whose entries are column (or row) sums of symmetric $\mathbf{W}$. Let the column vectors $\mathbf{u}_1,\ldots,\mathbf{u}_c$ be the eigenvectors, ordered according to their eigenvalues, $\lambda_1<\cdots<\lambda_c$. Thus, the embedding in the c-dimensional spectral space is represented as $\boldsymbol{x}_i\rightarrow\boldsymbol{y}_i=\mathbf{U}^T\boldsymbol{x}_i,\mathbf{U}=[\mathbf{u}_1,\mathbf{u}_1,\ldots,\mathbf{u}_c]$.
4. Extension: For $j=n+1,n+2,\ldots,N$ in $\mathbf{D}_{N-n}$,[1] project each object $\boldsymbol{x}_j^e$ that needs to be extended in the learned c-dimensional embedding space by $\boldsymbol{y}_j^e=\mathbf{U}^T\boldsymbol{x}_j^e$. Together with the embedding $\boldsymbol{y}_i$ of n labeled samples, we may assign this new object o_j to the class label with the maximum votes from its k nearest neighbors as measured in the spectral domain.

This algorithm includes two main steps: (1) LPP-based embedding space learning and (2) out-of-sample extension using a k-nearest neighbor (k NN) classifier in the embedding space. The complexity of LPP is basically dominated by two parts: K nearest neighbor search and generalized eigenvector computation. For the former, the time complexity is $O((n+K)n^2)$, where nn^2 stands for the time complexity of computing the distances between any two n-dimensional data points, and Kn^2 stands for the time complexity of finding the K nearest neighbors for all n data points. For the latter, to solve a generalized eigenvector problem $A\mathbf{u}=\lambda B\mathbf{u}$, where A and B are $n\times n$ symmetric and positive semi-definite matrices in this case, we first need to compute the SVD of B, requiring a time of $O(n^3)$. Then, we need to compute the first c smallest eigenvectors of an $n\times n$ matrix, with a time of $O(cn^2)$. Thus the total runtime complexity of the generalized eigenvector problem is $O((n+c)n^2)$. That is, the time of the LPP method is $O((n+K)n^2+(n+c)n^2)$. The time for the extension of the $N-n$ remaining objects using k NN is $O((N-n)kcn)$. This suggests that the total time complexity for the algorithm is $O((2n+K+c)n^2+(N-n)kcn)$. Assuming $N\gg n$ and $n\gg K+c$, the dominant component of the total time complexity for the algorithm is $O(n^3+Nkcn)$. Since the sample number $n\ll N$, the complete extension algorithm may be treated as linearly scalable in terms of the number of objects N.

[1] $\mathbf{D}_{N-n}$ can be fully loaded or incrementally loaded depending on the computational platform. Moreover, if the input data begin as object data, for each object to be extended, only the distances between it and the n samples are computed for use in extension.

4. Experimental evaluation

We carried out a large number of experiments on several synthetic and real-world data sets to evaluate the eSPEC method. How to automatically determine the number of clusters c in unlabeled data sets is a hard problem, and is beyond the scope of this paper. In our experiments, we generally set it to the number of real physical classes in the data sets. The other four important parameter choices made in our implementation were motivated by simplicity and these parameter settings can be refined if necessary. Unless mentioned specifically, we set $h=3c$, $r=7$, $K=7$ and $k=5$ as default values in the following experiments. All experiments were implemented in a Matlab 7.2 environment on a PC with Intel CPU 2.4 GHz and 2 GB memory. Our programs have not been optimized for run-time efficiency.

Following [27,26,21], we use an accuracy (AC) metric that relies on permuting the cluster labels to evaluate the clustering performance. Suppose that l_i^c is the clustering label result of a given example o_i and l_i^g is the corresponding ground truth label, AC is defined by $\mathrm{AC}=\max_{map}\sum_{i=1}^{\mathrm{N}}\delta(l_i^g,map(l_i^c))/\mathrm{N}$ where $\delta(l_1,l_2)$ is the delta function that equals 1 if and only if $l_1=l_2$ and 0 otherwise, and *map* is the mapping function that permutes clustering labels to match equivalent labels given by the ground truth. The Kuhn–Munkres algorithm is usually used to obtain the best permutation [25]. For each experiment, we performed our eSPEC algorithm multiple times, and reported results in terms of the average resubstitution error rate (i.e., $1-\mathrm{AC}$), as well as the average computation time.

To make the experimental procedure clear, we carried out our experiments in the following order. First we evaluate our method on five small-sized data sets, i.e., three synthetic data sets (with sizes of 3000, 2000 and 2000 objects) and two real-world data sets (with sizes of 3000 and 5850 objects). Such choices of the data sizes allow us to implement both the approximate clustering method and the literal clustering using the whole data, so as to examine their difference in clustering accuracy and computational efficiency. Then, we carried out a set of experiments on one synthetic data set of these five data sets to evaluate the effects of several parameters on the clustering accuracy, and further, we compared several approximate clustering methods applied to the remaining two synthetic and two real-world data sets. In addition, we also performed a comparative experiment with respect to a realistic "pure relational" data set to examine our method's

performance. Finally, we evaluated our method using several larger data sets, including three high-resolution images with sizes of $N=154{,}401$ pixels, one synthetic data set with a size of $N=3{,}000{,}000$ objects and clustering of one real-world MRI data set with a size of $N=1{,}132{,}545$.

4.1. Results on small synthetic data sets

We begin with several synthetic data sets of different types and sizes to measure the clustering difference between the output of the literal clustering method and the output of our approximate clustering method. These synthetic data sets include (almost) linearly and non-linearly separable clusters, and combinations thereof (see their scatter plots in the left column of Fig. 2). (1) Data set I, 5-Gaussian, was generated from a mixture of five bi-variate normal distributions, i.e., spherical shapes. The size of this data set ($c=5$) was $N=3000$, and the number of objects in each group was, respectively, 600, 600, 900, 600, and 300. (2) Data set II, 2-HalfMoon, was composed of two half-moon-like patterns, i.e., manifold shapes. The size of this data set ($c=2$) was $N=2000$, with 1000 points in each cluster. (3) Data set III, 2-Gaussian + 1-HalfMoon, was generated from a combination of a mixture of two bi-variate normal distributions and one half-moon-like pattern. The size of this data set ($c=3$) was $N=2000$, including 1000 points for the half-moon pattern and 500 points for each of the two Gaussian shapes.

We computed pairwise dissimilarity matrices $\mathbf{D}_N$ with the Euclidean distance as inputs to our algorithm. For each data set, we tried multiple sampling rates (i.e., the ratio between the sample size and the total data size, n/N). It should be noted that the use of the sampling rate is only for description consistency and convenience. It might be better to simply use the sample size n, not n/N, in order to avoid the possible misunderstanding that the sampling rate represents the scalability. Actually, the ideal sample size n depends only on the structure of the data including the number of clusters c and their distributions, rather than the data size N. That is, the necessary sample size is basically fixed to

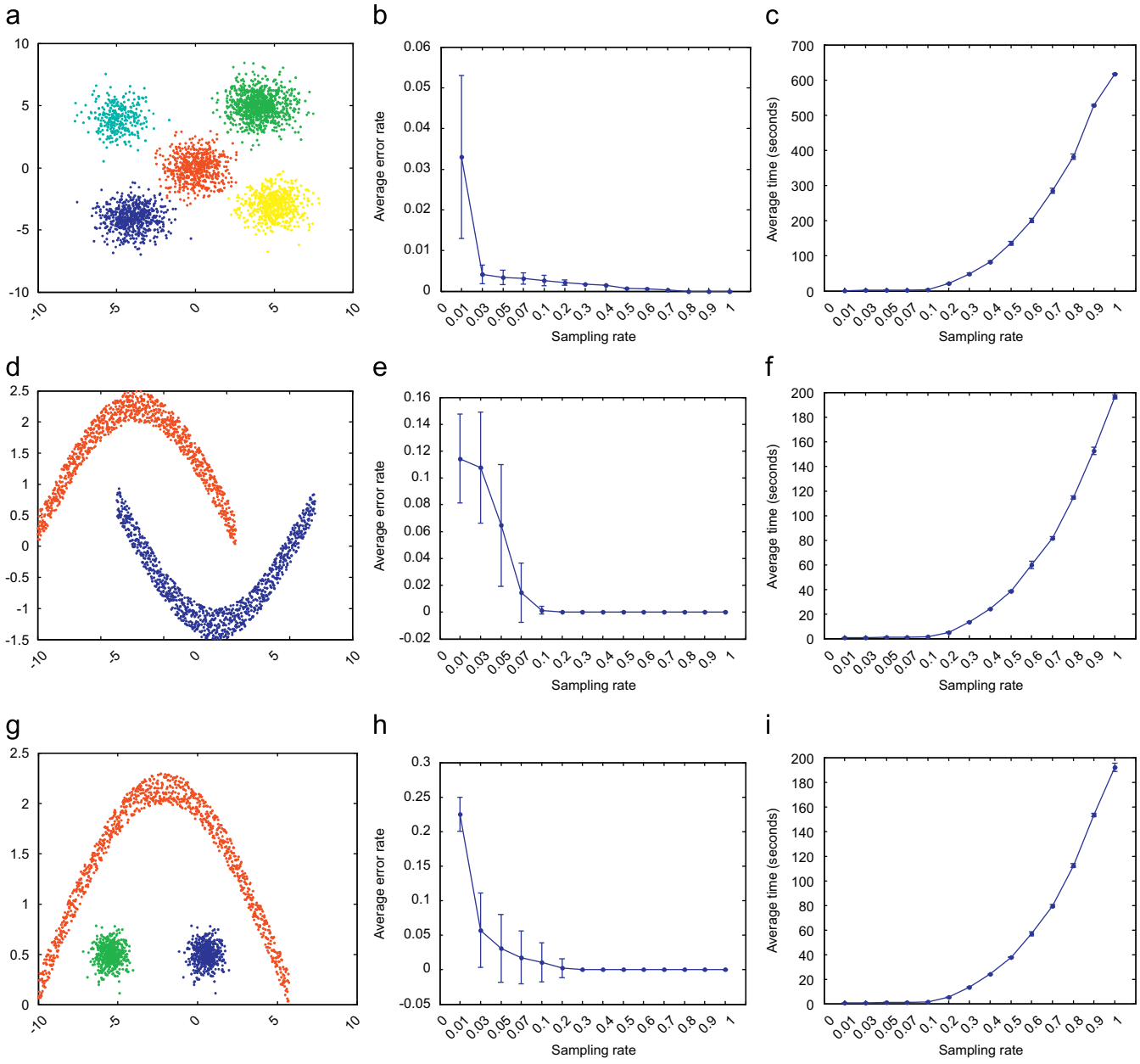

Fig. 2. Results on synthetic data sets: data sets (left), average error rates vs. sampling rates (middle), and average computation times vs. sampling rates (right).

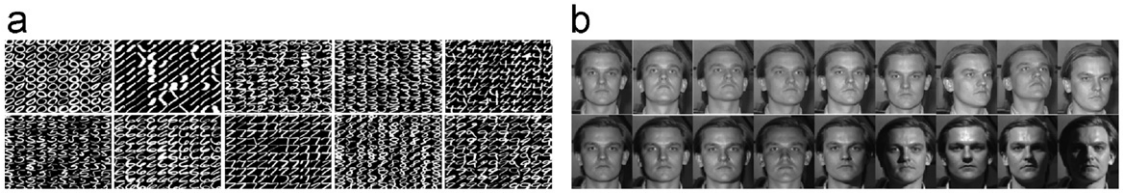

Fig. 3. Example images: (a) the USPS digits data set and (b) the Yale-B face data set with pose and illumination variations.

sufficiently encode the structure of the underlying data, and so it can remain unchanged regardless of the real data size. For each sampling rate, 100 trials were made. For each experiment, we computed the average error rates (AER) of clustering based on the known cluster memberships, as well as the average computation time (ACT) consumed by the whole clustering procedure, as shown in Fig. 2. It shows that:

- Our method can achieve a good approximation, sometimes with the same accuracy as the literal solution with the sampling rate of 1. Moreover, these estimates are obtained using only a small fraction of the data (see (b), (e) and (h)), and in much less time (up to several hundred times faster, as shown in (c), (f) and (i)).
- For more complex-shaped clusters, more samples (or larger sampling rates) are generally required to obtain stable results (see Fig. 2 (middle)). For Data Set III, the required sampling rate is 0.2; for Data Set II, the required sampling rate is 0.1; and for Data Set I, the required sampling rate is 0.07 (note that above 0.07, the average error rates are very close to that of the dense problem).
- When the sampling rate (or say the number of samples) is sufficient, the accuracy curve remains flat as the sample size further increases, but the computation time increases drastically (as shown by Fig. 2 (middle) and (right)).

These numerical experiments suggest that the strategy of first grouping a small number of data items and then classifying the out-of-sample instances in the spectral domain can obtain essentially the same results as the literal approach in much less time. This demonstrates that our method achieves acceleration, with little loss of accuracy when compared to the literal clustering approach.

4.2. Results on small real-world data sets

Next we consider the USPS Digits data and Yale-B Face data sets, both of which have been widely used for testing classification methods in the pattern recognition community. (1) The USPS digits data set consists of 16×16 gray-scale images of hand-written digits scanned from envelopes by the U.S. Postal Service.[2] Some example images are shown in Fig. 3(a). We wish to distinguish between the digits "0"–"9" using the USPS data set. In our experiment, we used all digits, with 300 examples of each digit ($N=3000$ images in total). Each image was converted into a 256-dimensional (i.e., 16×16) vector representation in row-major order. Linear principal component analysis (PCA) was then used as a pre-processing step to reduce the input dimensionality from 256 to 148 (accounting for 98% of the variance). (2) The Yale-B face data set[3] contains single light source images of 10 different subjects, each seen under 576 viewing conditions (9 poses $\times$ 64 illumination conditions). For every subject in a particular pose, an image with ambient (background) illumination was also captured. Hence, the total number of images is $N = 576 \times 10 + 9 \times 10 = 5850$. Some sample images are shown in Fig. 3(b). We wish to group human faces by person using the Yale-B data set. In our experiments, we used images of 10 individuals, and down-sampled each original image to 30×40 pixels, producing a 1200-dimensional (i.e., 30×40) vector representation in row-major order. PCA was then applied to reduce the input dimensionality of the images from 1200 to 294 (accounting for 98% of the variance).

For each of these two image data sets, we computed a pairwise dissimilarity matrix using Euclidean distance to act as the input to our algorithm. We set the number of clusters to $c=10$ corresponding to 10 different subjects in the Yale-B data set (i.e., individuals 1–10) and $c=10$ corresponding to 10 different digits in the USPS data set (i.e., digits "0"–"9"). For each case, we applied our algorithm 25 times for each of various sampling rates. The AERs and ACTs on these two image data sets are shown in Fig. 4.

From Fig. 4, it can be seen that, when the samples are sufficient, the results using clustering on a small number of samples plus out-of-sample extension are generally close or comparable to those using the full set of examples in the data set, but in much less time, which is basically consistent with the results for synthetic data sets. More interestingly, some results using a small number of samples outperform those of the literal method using full samples for the Yale-B face database. This is probably because a small portion of representative samples may be enough to effectively reveal the complete structure of the complete data set, while the introduction of more (or possibly noisy) data will have a slightly negative influence on the use of the algorithm. In addition, irrelevant dimensions in the high-dimensional image data (still 294 dimensions after PCA) can affect the selective sampling results from the pairwise relational matrix (constructed in the original high-dimensional input space), and confuse clustering algorithms by hiding clusters in noisy data.

4.3. Parameter evaluation

As stated before, we use the default parameter settings for the above experiments. How to set such parameters in an optimal manner remains challenging. To examine their effects on accuracy, we performed a group of parameter evaluation experiments. Here we used the third synthetic data set (i.e., 2-Gaussian + 1-HalfMoon), and chose three sampling rates of 0.01, 0.05 and 0.1 (i.e., $n=20$, 100 and 200) for these experiments. Each time, we changed one of the parameters according to its potential range, while keeping the remaining ones unchanged. For $n=20$, we select h to change over 1–15, while r, k and K varying over 1–7. For $n=100$ we varied h, r, k and K over 1–25, and for $n=200$, we varied h, r, k and K over 1–50. The determination of such parameter ranges is obtained by the assumption that the approximate numbers of the sampled objects in each of the three clusters are expected to be $n/2$, $n/4$ and $n/4$ (according to their real cluster sizes). For each group of parameter settings, we performed our algorithm 50 times, and computed the average of the resulting clustering accuracies. The results of parameter evaluation are shown in Fig. 5, from which we can see:

- The sample size has the greatest influence on the clustering accuracy compared to other parameters (see those curves with different colors, i.e., green curves generally have the lowest

[2] http://www.cs.toronto.edu/~ roweis/data.html
[3] http://markus-breitenbach.com/machine_learning_data.php

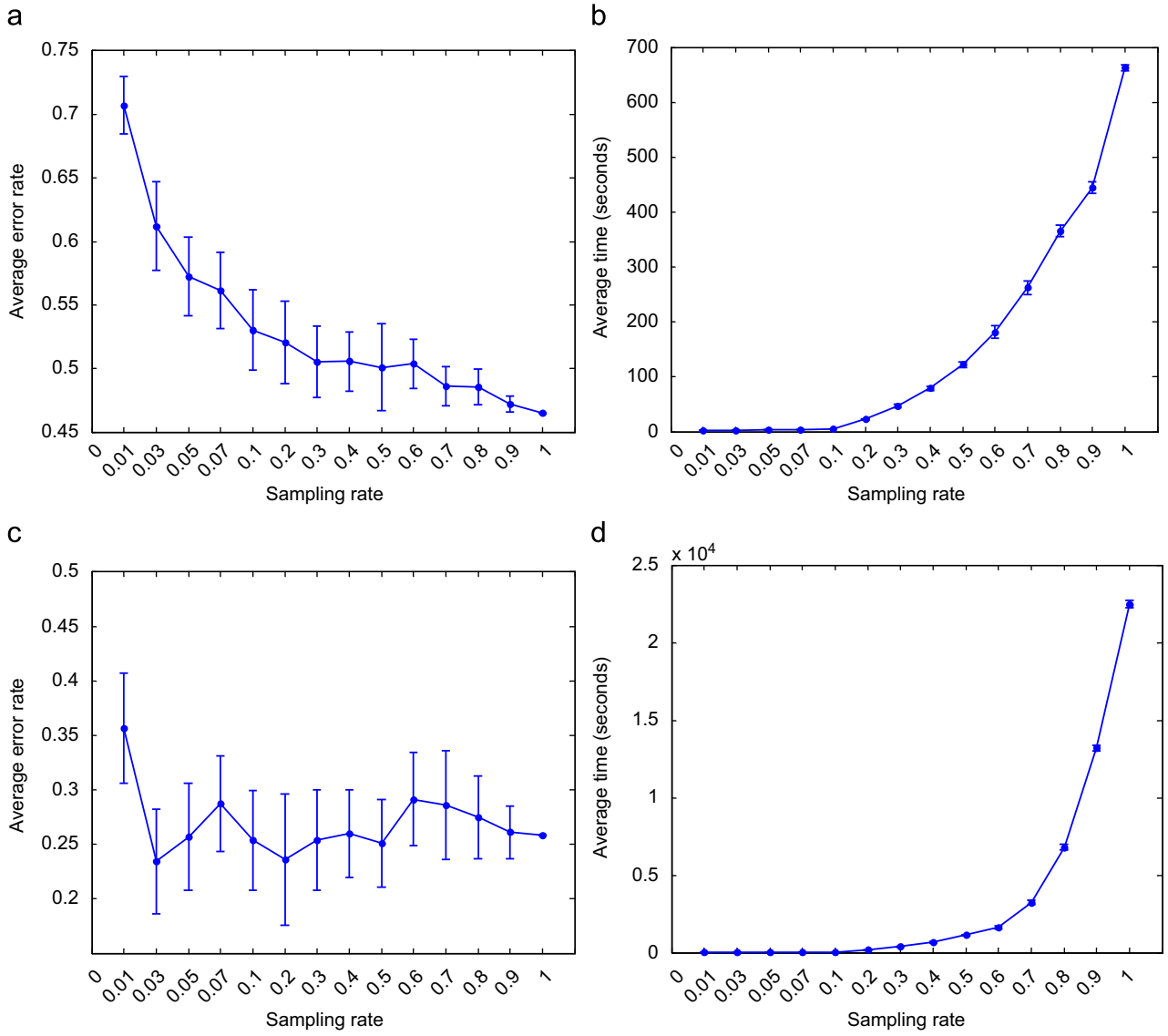

Fig. 4. Results in terms of AERs and ACTs on real-world image data sets: the USPS digit data set (top) and Yale-B face data set (bottom).

clustering errors with quite small variances). In addition, for very small (or insufficient) sample sizes (e.g., $n=20$), these four parameters have very different effects on the results, especially r and k (which are critical for sample clustering and thus out-of-sample extension).

- For h, when the sample size is insufficient (e.g., $n=20$), an increase in h will naturally decrease the clustering error. When the sample size is closer to being sufficient, the results basically remain steady even as h increases further. For K, basically, when it is larger than 3, the results tend to be stable.
- For k, the results become much steadier as k increases from the nearest-neighbor classifier (i.e., $k=1$) to the larger values of k.
- The results are somewhat sensitive to r. This is not surprising because r determines the quality of the sample clustering directly which has a continuous influence on the out-of-sample extension (and thus the overall clustering accuracy). The experiments on this data set show that it may be set in a narrow range of smaller values (e.g., $3-8$), which is consistent with the common concern that it is better to choose a value that is not too large for better preserving the locality of each data point.

In summary, as expected, the sample size is a dominant factor in determining the algorithm performance. The parameters h, k and K are easily chosen over a wider range of values. However, we should be more careful in choosing a suitable r because it controls the degree of locality of each example. Empirical experiments show that $r=7$ performs well on all the used data sets in [9]. Though our experiment also shows that r has its best effect for a relatively small range, e.g., $3-8$, it is plausible that this range might vary because of different geometric structures and properties of the data sets to be analyzed.

4.4. Comparison of approximate clustering methods

To further examine the performance of eSPEC, we compared it to the three existing approximate clustering approaches, namely eNERF with progressive sampling (PS, [3]), eNERF with selective sampling (SS, [22]) and the spectral grouping method based on the Nyström approximation [11]. We re-implemented and tested the three methods on the remaining two synthetic data sets I and II (i.e., 5-Gaussian and 2-halfMoon), and two real image data sets (i.e., USPS and Yale-B), for this comparative experiment. Note that in [11], Fowlkes et al. used random sampling and a global scale parameter to construct the affinity matrix. For the method based on the Nyström approximation in [11], we specially used the

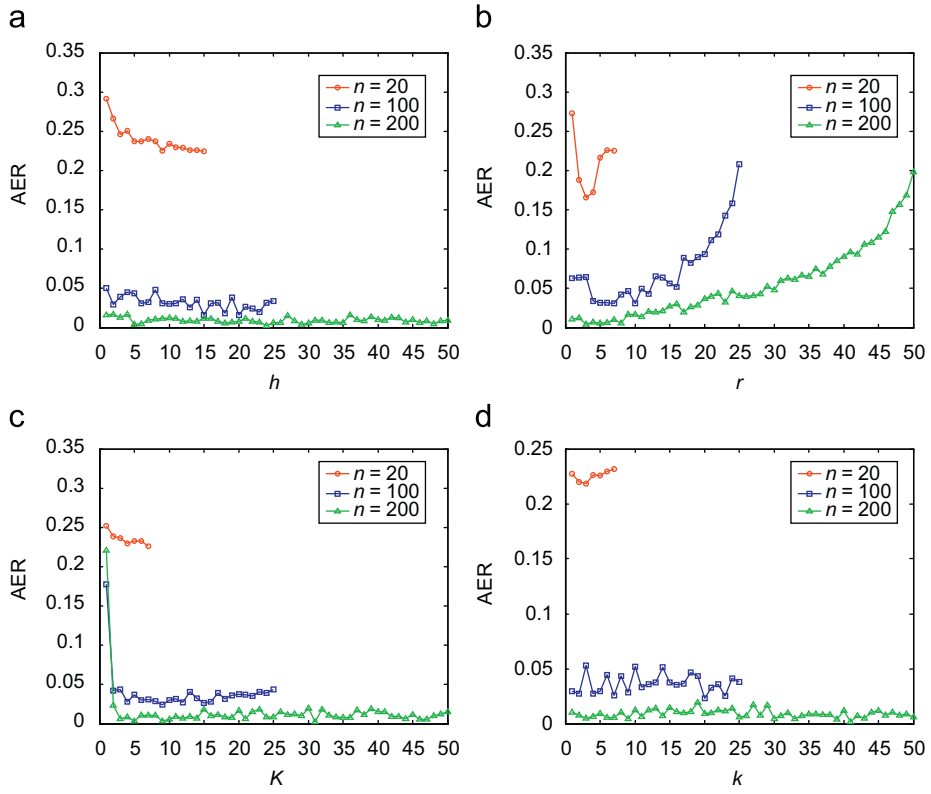

Fig. 5. Evaluation of key algorithmic parameters. (a) $r=7$, $K=7$, $k=5$. (b) $h=9$, $K=7$, $k=5$. (c) $h=9$, $r=7$, $k=5$. (d) $h=9$, $r=7$, $K=7$.

same selective sampling scheme and local scaling scheme (as used in eSPEC) for obtaining as fair a comparison as possible. The number of clusters was set to the number of real physical classes for all algorithms considered. For eNERF (either PS or SS), we used the following parameter values (see [3] for detailed definitions): number of DF candidates $H=N$ for PS and $NN=N$ for SS; number of DFs $h=3c$; the fuzzy weighting constant $m=2$, and the stopping tolerances were for LNERF $\varepsilon_L=0.00001$ and for xNERF, $\varepsilon_x=0.001$. For PS, we set the following additional parameters: divergence acceptance threshold $\varepsilon_{PS}=0.80$; number of histogram bins $b=10$; initial sample percentage $p=10\%$; and incremental sample percentage $p=1\%$. According to the above parameter evaluation results, we use $h=3\cdot c, k=5$ and $K=7$ for eSPEC. For both eSPEC and the method based on the Nyström approximation, the results could depend on the selection of a local scale parameter r. Here we tested a suitable range of r for both of them, and reported the best results. For each method, we performed 25 trials at the sampling rates of 0.01, 0.03, 0.05, 0.07, 0.1 and 0.2, and the results in terms of AERs and ACTs are, respectively, shown in Fig. 6, from which we can conclude that:

- Both the eSPEC and Nyström methods apparently outperform the eNERF-based methods (except for the similar results on the simplest synthetic 5-Gaussian data). Although eNERF with either PS or SS requires the least time among these compared methods, both eNERF methods have significantly lower accuracy than either eSPEC or the method in [11].
- For synthetic data sets (see (a) and (b)), the Nyström method performs a little better than the eSPEC method in clustering accuracy, but requires longer (for 5-Gaussian) or very similar (for 2-HalfMoon) time. Note that the clustering error rates of both methods are themselves very small for these two small data sets. In particular, the advantage of the Nyström method over eSPEC in accuracy on the 5-Gaussian data and in computation time on the 2-HalfMoon data is almost negligible.
- eSPEC performs better than the other three methods in terms of accuracy when the required sampling rate is above 0.03, especially for the two real image data sets (see (c) and (d)). Overall, the Nyström method is worse than eSPEC in both the accuracy and computational efficiency. Again, the eNERF methods have poor clustering accuracy, especially for non-Gaussian data sets, though they are computationally more efficient.

In summary, the eSPEC algorithm is far superior to both eNERF methods. Although the Nyström approximation is a well-studied method based on a theory how the Gram matrix can be approximated by a few eigenvectors of a sub-matrix, empirical results still show that our method is competitive compared to the method based on the Nyström approximation in terms of both computation time and accuracy (see also the comparison results in terms of high-definition image segmentation in the following subsection). Theoretically, any methods based on sampling plus out-of-sample extension (such as our eSPEC) can work for arbitrarily large data sets, as long as the sampling and sample clustering can be performed, which is quite likely to be feasible in practice. However, for the Nyström method, it still involves multiplication of a large matrix depending on N and the c-means clustering, which naturally creates the problem of whether it can be performed on a truly very large data set. In this way, the eSPEC algorithm seems to be superior to the method based on the Nyström approximation in terms of scalability.

4.5. Results on a "pure relational" data set

In the previous series of experiments, the object vectors for the data sets used are available, so that we can construct the pairwise distance matrices as the input of our algorithm. In this section, we

a

Average error rates							
Sampling rate	0.01	0.03	0.05	0.07	0.1	0.2	0.49
SS + eNERF	0.0136	0.0021	0.0019	0.0018	0.0017	0.0016	-
PS + eNERF	-	-	-	-	-	-	0.0016
SS + Nystrom	**0.0118**	**0.0031**	**0.0015**	**0.0015**	**0.0016**	**0.0016**	-
eSPEC	0.0170	0.0032	0.0027	0.0024	0.0021	**0.0016**	-
Average computation time (s)							
Sampling rate	0.01	0.03	0.05	0.07	0.1	0.2	0.49
SS + eNERF	0.40	0.42	0.44	0.45	0.46	0.54	-
PS + eNERF	-	-	-	-	-	-	1.66
SS + Nystrom	2.05	2.77	3.20	4.04	5.38	**15.66**	-
eSPEC	**1.58**	**1.95**	**2.46**	**3.12**	**4.71**	20.11	-

b

Average error rates							
Sampling rate	0.01	0.03	0.05	0.07	0.1	0.2	0.39
SS + eNERF	0.115	0.110	0.104	0.103	0.103	0.103	-
PS + eNERF	-	-	-	-	-	-	0.103
SS + Nystrom	**0.031**	**0.001**	**0.000**	**0.000**	**0.000**	**0.000**	-
eSPEC	0.094	0.042	0.019	0.005	0.001	**0.000**	-
Average computation time (s)							
Sampling rate	0.01	0.03	0.05	0.07	0.1	0.2	0.39
SS + eNERF	0.18	0.18	0.19	0.19	0.21	0.21	-
PS + eNERF	-	-	-	-	-	-	0.26
SS + Nystrom	**0.61**	**0.66**	**0.82**	**1.00**	**1.44**	**4.80**	-
eSPEC	0.62	0.73	0.95	1.07	1.48	4.98	-

c

Average error rates							
Sampling rate	0.01	0.03	0.05	0.07	0.1	0.2	0.61
SS + eNERF	0.791	0.799	0.802	0.797	0.809	0.809	-
PS + eNERF	-	-	-	-	-	-	0.811
SS + Nystrom	**0.588**	**0.570**	0.553	0.553	0.545	0.545	-
eSPEC	0.701	0.593	**0.541**	**0.533**	**0.514**	**0.521**	-
Average computation time (s)							
Sampling rate	0.01	0.03	0.05	0.07	0.1	0.2	0.61
SS + eNERF	0.33	0.36	0.37	0.39	0.44	0.57	-
PS + eNERF	-	-	-	-	-	-	1.40
SS + Nystrom	5.43	6.04	6.59	7.04	9.77	23.85	-
eSPEC	**2.75**	**3.15**	**3.67**	**4.41**	**6.21**	**22.50**	-

d

Average error rates							
Sampling rate	0.01	0.03	0.05	0.07	0.1	0.2	0.66
SS + eNERF	0.841	0.812	0.799	0.798	0.798	0.798	-
PS + eNERF	-	-	-	-	-	-	0.798
SS + Nystrom	**0.355**	0.233	0.220	0.217	0.208	**0.202**	-
eSPEC	0.357	**0.194**	**0.176**	**0.174**	**0.174**	0.220	-
Average computation time (s)							
Sampling rate	0.01	0.03	0.05	0.07	0.1	0.2	0.66
SS + eNERF	0.92	1.05	1.14	1.27	1.41	2.01	-
PS + eNERF	-	-	-	-	-	-	2.50
SS + Nystrom	11.50	13.35	19.01	22.89	39.14	144.40	-
eSPEC	**5.11**	**7.30**	**11.70**	**19.27**	**39.07**	**143.86**	-

Fig. 6. Comparison of several approximate clustering algorithms. (a) Synthetic Data Set I: 5-Gaussian. (b) Synthetic Data Set II: 2-HalfMoon. (c) USPS Digit Data Set. (d) Yale-B Face Data Set.

selected a "pure relational" data set that has been used elsewhere in the literature. Chicken Pieces Silhouettes Database in [38] was used in this experiment. This data set consists of 446 images of chicken pieces, each of which belongs to one of five categories, representing wing (117), back (76), drumstick (96), thigh and back (61), and breast (96). A number of string edit distance matrices with respect to different configurations are made available for this data set. We selected the *chickenpieces-norm*20.0-*AngleCostFunction*60.0 matrix for our experiment. We applied our method and the spectral grouping method based on the Nyström approximation [11] to this distance matrix. For these two methods we used the same selective sampling for a fair comparison. For each method, we performed 50 trials at various sampling rates, and the results in terms of AER are shown in Fig. 7. From Fig. 7, it can be seen that the performance of our method tends to consistently improve as the sampling rate increases; whereas the performance of the Nyström method is not steady across different sampling rates. For lower sampling rates, the Nyström method performs better. In contrast, when the sampling rate is increased to include a sufficient number of samples, our method performs better. This behavior of the Nyström method seems to be the result of over-fitting when

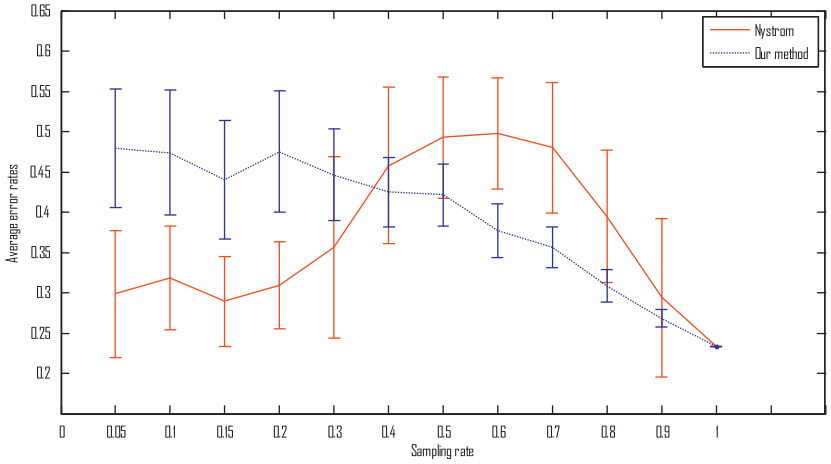

Fig. 7. Average error rates on the Chicken Piece Silhouettes Database.

approximating the eigenvectors using more samples. We show later that the same behavior occurs in the image segmentation results. In contrast, our method does not result from this drawback.

4.6. Results on high-definition image segmentation

We further applied eSPEC to relatively larger real-world data sets, i.e., the problem of high-resolution image segmentation, where it is generally infeasible to directly use the literal spectral clustering algorithm. In [11], high-resolution image segmentation is also used to evaluate the approximate spectral grouping technique based on the Nyström approximation. Different visual features have been used to describe the affinities between image pixels, e.g., brightness, color, texture, proximity, or their fusion. Although the use of multiple cues may obtain better segmentation results, we just tried the brightness feature since our major concern is to demonstrate the performance of our approximate clustering algorithm in the application of image segmentation, and not purely image segmentation. Fig. 8 shows segmentation results on three 481×321 images taken from the Berkeley database.[4] We set $c=2$ or 3 for these three images according to the number of visually meaningful components, viz., $c=3$ for the house image, and $c=2$ for the airplane and elephant images.

Running a spectral clustering algorithm on the whole image which contains $N=481 \times 321=154,401$ pixels would be simply impractical using Matlab. For these images, the number of sampled pixels was empirically chosen to be 150 (less than 0.1% of the number of total pixels), considering that there are far fewer coherent groups (i.e., $c \ll N$) in a scene than pixels. We cannot measure the clustering error in this case because literal spectral clustering cannot be performed and the correct partition is not known. So, the best we can do for evaluation here is to resort to visual inspection of the segmentation results. In these three cases, our algorithm partitioned the images into meaningful components (see Fig. 8, in which pixels of each color in the segmented images represent one component). We also implemented the normalized cut algorithm with the Nyström approximation [11] on these images using the same number of samples. The results on the elephant image are similar to those of our method, but are worse than those of our method on the other two images. For example, in Fig. 8(b), the sky region is divided into two components and one of them is merged with the region of the balcony, which is not meaningful in terms of visual perception; and in Fig. 8(e), the region corresponding to the airplane is connected with a part of the background region.

[4] http://www.eecs.berkeley.edu/Research/Projects/CS/vision/grouping/segbench/

4.7. Results on very large data sets

Next, we consider the application of our method to two very large data sets. One is a synthetic data set used in [22], which is a set of $N=3,000,000$ 2D points drawn from a mixture of normal distributions (MND, $c=5$), and the other is a real-world data set used in [37], which is a set of $N=1,132,545$ 3D objects derived from magnetic resonance image (MRI) data. Note that five clusters in the NMD data set are visually apparent, but there is a high level of mixing between outliers from components in the mixture. The MRI data set was created by concatenating 45 slices of MR images of the human brain of size 256×256 from modalities T1, PD and T2. After air was removed, there are slightly more than 1 million examples (1,132,545 examples, three features).

Computing squared Euclidean distances between pairs of vectors yields a matrix $\mathbf{D}_N$ with $N \times (N-1)/2 = O(10^{12})$ dissimilarities for these two data sets. We are not able to calculate, load and process a full distance data matrix of this size. Here we use an approximate "lookup table" mode by just storing the object data set, accessing only the vectors needed to make a particular distance computation, and releasing the memory used by these vectors immediately after to avoid "out of memory" exceptions. If we had only the relational data, this would represent a very large (VL) clustering problem for the computing environment available. To avoid exhausting memory in Matlab, the processing was partitioned by calling the extension routine multiple times (depending on the chunk size), and each chunk was used to extend the partition for loaded objects. We set $c=5$ for the NMD data set like [22] and $c=9$ for the MRI data set like [37], and tried several different sample sizes, i.e. $n=300$, 450, 600, and 2500 (or sampling rates 0.00010, 0.00015, 0.00020, and 0.00083 for the NMD data set, and sampling rates 0.00026, 0.00040, 0.00053, and 0.0022 for the MRI data set). The whole eSPEC process took relatively longer for each case with different values of n, thus we only list the results with respect to one run per case in Table 2.

For the NMD data set, we report the clustering accuracy by comparing the clustering labels obtained by the eSPEC method with the known labels, as well as the computation time. For the MRI data set, since we have no real labels, we cannot compute the clustering accuracy. Instead, we computed two indices for

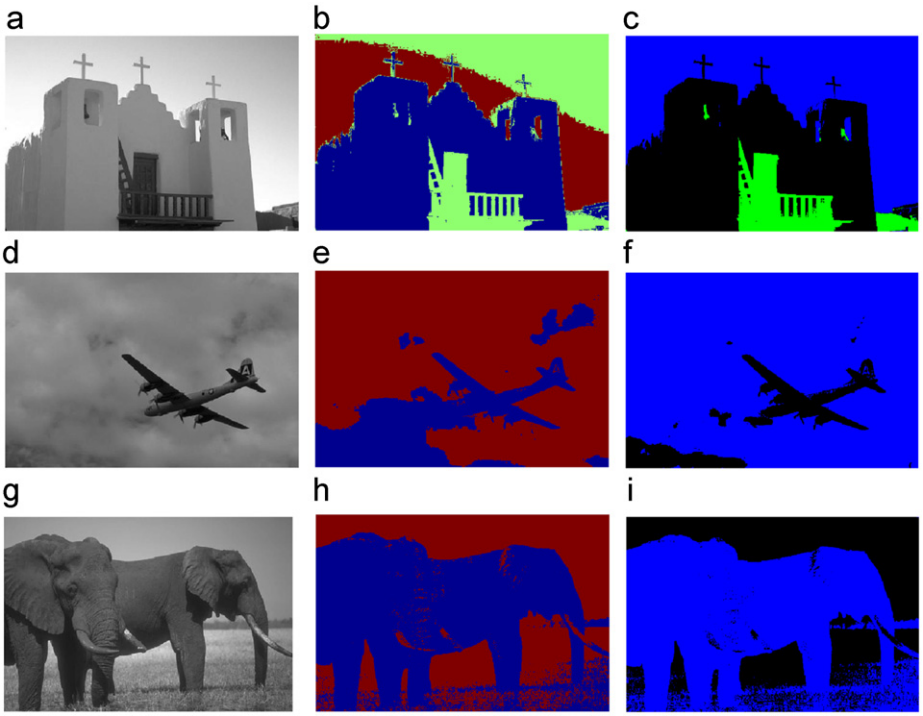

Fig. 8. Image segmentation: original images (left), segmentation results of the Nyström method (middle) and segmentation results of our method (right).

Table 2
Results of eSPEC on two very large data sets (one run per case).

Sample sizes	MND (N=3,000,000)		MRI (N=1,132,545)		
	Error rate	Time (s)	ASW	CHI (10^6)	Time (s)
300	0.0103	13,002	0.4770	1.522	4546.3
450	0.0073	14,892	0.4713	1.707	6936.0
600	0.0054	27,331	0.5231	1.813	9232.9
2500	0.0062	363,860	0.5675	1.856	113,870

cluster validity, i.e., the average silhouette width (ASW) and the Calinski-Harabasz index (CHI) [4], which are two commonly used measures of correspondence between a cluster structure and the data from which it has been generated, when the ground-truth labels are unavailable. Generally, the greater such measures, the better the clustering. From Table 2, it can be seen that all measures of error for the NMD data set are very small. This was certainly not an easy clustering problem, in terms of how well separated the clusters actually are. Though not exactly accuracy measures, the values of the two indices obtained on the MRI data set, to some extent, implicitly show the results to be satisfactory. In summary, the point here was to demonstrate the feasibility property of eSPEC on truly VL data, and these two examples demonstrated this.

5. Discussion and conclusion

Our contribution in this paper is a feasible and effective solution to the pairwise relational clustering of large data sets, in the "sampling, clustering plus extension" manner. The proposed solution is composed of three successive steps, namely selective sampling, sample clustering and out-of-sample spectral extension. A major innovation is an LPP-based out-of-sample extension strategy, which is critical to the algorithm's accuracy and efficiency. We treat the pairwise relational matrix between objects as a semantically meaningful vectorial representation, and use the sample submatrix to learn a graph embedding space. This elegantly converts out-of-sample extensions to label predictions in the embedding space, which is very similar to the mapping process in spectral clustering. This key step enables us to effectively handle irregular data structures and perform fast extension compared to traditional complex iterative procedures. It could be argued that the methodology combines several known techniques (i.e., our previously proposed selective sampling for sample selection and sample clustering using spectral clustering) with a new strategy of out-of-sample spectral extension. However, we believe that elegant and non-obvious combinations of methods are a worthwhile contribution to this challenging problem. In particular, extensive experimental results on synthetic and real-world data sets have shown that eSPEC is not only feasible for very large data sets, but is also faster when applied to large data sets without reduction in accuracy.

Accuracy and efficiency are two important factors in data clustering. Note that the sizes of the data sets in our experiments have been larger or the same as those used in previous methods [11,3,22]. Comparative results have also shown that our method performs better (or is highly comparable) to the three comparison methods in terms of accuracy. The computation time of our method is higher than that of the eNERF methods, but comparable to the method based on the Nyström approximation. Fortunately, in most cases our method does not require a large sampling rate to obtain good results, thus leading to somewhat lower computational costs. Theoretically, the computation time of our method is dependent on the sample size n, but is linear to the data size N. To summarize, eSPEC provides the best tradeoff between accuracy and efficiency amongst these four methods.

How to automatically and effectively set several specific parameters of our algorithm is potential for future work. In SS, we need to specify the parameters h and n. As discussed before, h

should be generally greater than c so that we do not miss any clusters and accurately capture the unknown shape of the clusters which can potentially be quite complex. Increasing the sample size n would be beneficial to the learning of the spectral embedding space, and hence spectral extension. But how to select a suitable n is indeed a difficult problem. Although we cannot offer an optimal solution to this problem, an accepted rule of thumb is to set n to an appropriate size according to the value of c so as to make sure that the sample includes enough examples from each cluster. At the same time, the sample size n must be manageable to ensure that spectral clustering of the sample can be performed on the available computing platform. It is possible to perform the algorithm multiple times using different sampling rates so as to finally select a sampling rate that is able to achieve the best accuracy. For the r, k and K parameters in spectral clustering and extension, experiments show that both k and K can easily be set in a relatively wide range, while r needs to be set to a small value for preserving "locality". Compared with the sample size, there are far more examples that need to be extended (i.e., $N-n \gg n$). Thus, the smaller the k in the k-NN classifier, the more computationally efficient the extension algorithm.

References

[1] J.M. Buhmann, Data Clustering and Learning, in: M.A. Arbib (Ed.), The Handbook of Brain Theory and Neural Networks, 1995, pp. 278–281.

[2] P. Drineas, R. Kannan, M. Mahoney, Fast Monte Carlo algorithms for matrices II: computing a low-rank approximation to a matrix, SIAM Journal on Computing 36 (1) (2006) 158–183.

[3] J. Bezdek, R. Hathaway, J. Huband, C. Leckie, R. Kotagiri, Approximate clustering in very large relational data, International Journal of Intelligent Systems 21 (2006) 817–841.

[4] R. Xu, D. Wunsch II, Survey of clustering algorithms, IEEE Transactions on Neural Networks 16 (3) (2005) 645–678.

[5] A. Ng, M. Jordan, Y. Weiss, On spectral clustering: analysis and an algorithm, in: Proceedings of the Advances in Neural Information Processing Systems, 2002.

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

[7] T. Hofmann, J. Buhmann, Pairwise data clustering by deterministic annealing, IEEE Transactions on Pattern Analysis and Machine Intelligence 19 (1997) 1–14.

[8] V. Roth, J. Laub, M. Kawanabe, J. Buhmann, Optimal cluster preserving embedding of nonmetric proximity data, IEEE Transactions on Pattern Analysis and Machine Intelligence 25 (12) (2003) 1540–1550.

[9] L. Zelnik-Manor, P. Perona, Self-tuning spectral clustering, in: Proceedings of the Advances in Neural Information Processing Systems, 2004.

[10] M. Pavan, M. Pelillo, Efficient out-of-sample extension of dominant-set clusters, in: Proceedings of the Advances in Neural Information Processing Systems, 2004.

[11] C. Fowlkes, S. Belongie, F. Chung, J. Malik, Spectral grouping using the Nyström method, IEEE Transactions on Pattern Recognition and Machine Intelligence 26 (2) (2004) 214–225.

[12] M. Belkin, P. Niyogi, Laplacian eigenmaps and spectral techniques for embedding and clustering, in: Proceedings of the Advances in Neural Information Processing Systems, 2002.

[13] X. He, P. Niyogi, Locality preserving projections, in: Proceedings of the Advances in Neural Information Processing Systems, 2003.

[14] Y. Bengio, J. Paiement, P. Vincent, O. Delallean, N. Roux, M.Ouimet, Out-of-sample extensions for LLE, Isomap, MDS, eigenmaps, and spectral clustering, in: Proceedings of the Advances in Neural Information Processing Systems, 2004.

[15] S. Roweis, L. Saul, Nonlinear dimensionality reduction by locally linear embedding, Science 290 (5500) (2000) 2323–2326.

[16] T. Zhang, R. Ramakrishnan, M. Livny, BIRCH: an efficient data clustering method for very large databases, in: Proceedings of the ACM International Conference on Management of Data, 1996, pp. 103–114.

[17] S. Guha, R. Rastogi, K. Shim, CURE: an efficient clustering algorithm for large databases, in: Proceedings of the ACM International Conference on Management of Data, 1998, pp. 73–84.

[18] R. Hathaway, J. Bezdek, Extending fuzzy and probabilistic clustering to very large data sets, Computational Statistics and Data Analysis 51 (2006) 215–234.

[19] C. Gupta, R. Grossman, GenIc: a single pass generalized incremental algorithm for clustering, in: Proceedings of the SIAM International Conference on Data Mining, 2004, pp. 22–24.

[20] P. Domingos, G. Hulten, A general method for scaling up machine learning algorithms and its application to clustering, in: Proceedings of the International Conference on Machine Learning, 2001, pp. 106–113.

[21] H.Z. Ning, W. Xu, Y. Chi, Y. Gong, T.S. Huang, Incremental spectral clustering with application to monitoring of evolving blog communities, in: Proceedings of the SIAM International Conference on Data Mining, 2007.

[22] L. Wang, J. Bezdek, C. Leckie, R. Kotagiri, Selective sampling for approximate clustering of very large data sets, International Journal of Intelligent Systems 23 (3) (2008) 313–331.

[23] H. Frigui, C. Hwang, Adaptive concept learning through clustering and aggregation of relational data, in: Proceedings of the SIAM International Conference on Data Mining, 2007.

[24] L. Wang, D. Suter, Learning and matching of dynamic shape manifolds for human action recognition, IEEE Transactions on Image Processing 16 (6) (2007) 1646–1661.

[25] L. Lovasz, M. Plummer, Matching Theory, Akadémiai Kiadó, North-Holland, Budapest, 1986.

[26] W. Xu, X. Liu, Y. Gong, Document clustering based on non-negative matrix factorization, in: Proceedings of the ACM SIGIR Conference on Information Retrieval, 2003.

[27] D. Cai, X. He, J. Han, Document clustering using locality preserving indexing, IEEE Transactions on Knowledge and Data Engineering 17 (2) (2005) 1624–1637.

[28] L. Zelnik-Manor, M. Irani, Event-based analysis of video, in: Proceedings of the International Conference on Computer Vision and Pattern Recognition, vol. 2, 2001, pp. 123–130.

[29] F. Porikli, Learning object trajectory patterns by spectral clustering, in: Proceedings of the International Conference on Multimedia and Expo, vol. 2, 2004, pp. 1171–1174.

[30] M. Filippone, F. Camastra, F. masulli, S. Rovetta, A survey of kernel and spectral methods for clustering, Pattern Recognition 41 (2008) 176–190.

[31] C. Ding, X. He, H. Zha, M. Gu, H.D. Simon, A min–max cut algorithm for graph partitioning and data clustering, in: Proceedings of the IEEE International Conference on Data Mining, 2001, pp. 107–114.

[32] G. John, P. Langley, Static versus dynamic sampling for data mining, in: Proceedings of the International Conference on Knowledge Discovery and Data Mining, 1996.

[33] F. Provost, D. Jensen, T. Oates, Efficient progressive sampling, in: Proceedings of the International Conference on Knowledge Discovery and Data Mining, 1999, pp. 23–32.

[34] R.B. Cattell, A note on correlation clusters and cluster search methods, Psychometrika 9 (3) (1944) 169–184.

[35] H.P. Kriegel, P. Kröger, A. Pryakhin, M. Schubert, Effective and efficient distributed model-based clustering, in: Proceedings of the International Conference on Data Mining, 2005.

[36] M. Klusch, S. Lodi, G.L. Moro, Distributed clustering based on sampling local density estimates, in: Proceedings of the International Conference on AI, 2003, pp. 485–490.

[37] P. Hore, L.O. Hall, D.B. Goldgof, Single pass fuzzy C means, in: Proceedings of the International Conference on Fuzzy Systems, 2007.

[38] B. Spillmann, Description of the distance matrices, Institute of Computer Science and Applied Mathematics, University of Bern, 2004.

Liang Wang obtained the Ph.D. degree from the National Laboratory of Pattern Recognition, Institute of Automation, Chinese Academy of Sciences, in 2004. He was then with the Imperial College London, UK, and Monash University, Australia. Since 2007, he has been with the Department of Computer Science and Software Engineering, the University of Melbourne, Australia. He serves as a program committee member of many international conferences, as well as Co-chairing several international workshops. Currently he is an Associate Editor of IEEE Transactions on System, Man and Cybernetics—Part B (IEEE T-SMCB), International Journal of Image and Graphics (IJIG), Signal Processing, and Neurocomputing, and is a Guest Editor for three special issues in IEEE TSMC-B, Pattern Recognition Letters (PRL) and the International Journal of Pattern Recognition and Artificial Intelligence (IJPRAI). He has widely published in major international journals and conferences such as TPAMI, TIP, TKDE, TCSVT, CVIU, PR, ICDM, ICCV and CVPR. His main research interest includes pattern recognition, computer vision, image and video processing, machine learning, and data mining.

Christopher Leckie received the B.Sc. degree in 1985, the B.E. degree in Electrical and Computer Systems Engineering (with first class honours) in 1987, and the Ph.D. degree in Computer Science in 1992, all from Monash University, Australia. He joined Telstra Research Laboratories in 1988, where he conducted research and development into artificial intelligence techniques for various telecommunication applications. In 2000, he joined the University of Melbourne, Australia, where he is currently an Associate Professor with the Department of Computer Science and Software Engineering. His research interests include using artificial intelligence for network management and intrusion detection, and data mining techniques such as clustering.

Ramamohanarao Kotagiri received his degrees B.E. at Andhra University, ME at the Indian Institute of Science, Bangalore and Ph.D. at Monash University. He was awarded the Alexander von Humboldt Fellowship in 1983. He has been at the University Melbourne since 1980 and was appointed a Professor in Computer Science in 1989. Rao held several senior positions including Head of Computer Science and Software Engineering, Head of the School of Electrical Engineering and Computer Science at the University of Melbourne, Deputy Director of Centre for Ultra Broadband Information Networks, Co-Director of the Key Centre for Knowledge-Based Systems, and Research Director for the Cooperative Research Centre for Intelligent Decision Systems. He served as a member of the Australian Research Council Information Technology Panel. He served on the Prime Ministers Science, Engineering and Innovation Council working party on Data for Scientists. At present he is on the Editorial Boards for Universal Computer Science, KAIS, IEEE TKDE, Journal of Statistical Analysis and Data Mining and VLDB Journal. He served as a program committee member of several International conferences including SIGMOD, IEEE ICDM, VLDB, ICLP and ICDE. He was the program Co-Chair for VLDB, PAKDD, DASFAA and DOOD conferences. He is a steering committee member of IEEE ICDM, PAKDD and DASFAA. Rao is a fellow of the Institute of Engineers Australia, Australian Academy Technological Sciences and Engineering and Australian Academy of Science. He has research interests in the areas of database systems, logic based systems, agent oriented systems, information retrieval, data mining, intrusion detection and machine learning.

James Bezdek received the Ph.D. in Applied Mathematics from Cornell University in 1973. He is past president of three professional societies: NAFIPS (North American Fuzzy Information Processing Society); IFSA (International Fuzzy Systems Association); and the IEEE Computational Intelligence Society. He is the founding editor of two journals: the International Journal of Approximate Reasoning; and the IEEE Transactions on Fuzzy Systems. He is a fellow of the IEEE and IFSA, and recipient of the IEEE 3rd Millennium, IEEE CIS Fuzzy Systems Pioneer, and IEEE CIS Rosenblatt medals. His interests include woodworking, optimization, motorcycles, pattern recognition, cigars, fishing, image processing, blues music, and cluster analysis.