Efficient Kernel Clustering Using Random Fourier Features

Radha Chitta, Rong Jin and Anil K. Jain
Department of Computer Science and Engineering,
Michigan State University,
East Lansing, MI 48824, USA
chittara@msu.edu, rongjin@cse.msu.edu, jain@cse.msu.edu

***Abstract*—Kernel clustering algorithms have the ability to capture the non-linear structure inherent in many real world data sets and thereby, achieve better clustering performance than Euclidean distance based clustering algorithms. However, their quadratic computational complexity renders them non-scalable to large data sets. In this paper, we employ random Fourier maps, originally proposed for large scale classification, to accelerate kernel clustering. The key idea behind the use of random Fourier maps for clustering is to project the data into a low-dimensional space where the inner product of the transformed data points approximates the kernel similarity between them. An efficient linear clustering algorithm can then be applied to the points in the transformed space. We also propose an improved scheme which uses the top singular vectors of the transformed data matrix to perform clustering, and yields a better approximation of kernel clustering under appropriate conditions. Our empirical studies demonstrate that the proposed schemes can be efficiently applied to large data sets containing millions of data points, while achieving accuracy similar to that achieved by state-of-the-art kernel clustering algorithms.**

***Keywords*-Kernel clustering, Kernel *k*-means, Random Fourier features, Scalability**

I. Introduction

Clustering, a primary data mining task, has found use in a variety of applications such as web search, social network analysis, image retrieval, medical imaging, gene expression analysis, recommendation systems and market analysis [1].

Kernel-based clustering techniques, which employ a non-linear distance function, have the ability to find clusters of varying densities and distributions, characteristics inherent in many real data sets. They embed the data points into a high-dimensional non-linear manifold where the clusters tend to be more separable. However, these algorithms require the computation of an $n \times n$ kernel matrix, where n is the size of the data set. This renders them non-scalable to large data sets that contain more than a few tens of thousands of points. On the other hand, linear clustering algorithms based on the Euclidean distance measure are more efficient in terms of computational and memory complexity. An elegant technique for deriving the best of both worlds was proposed by Rahimi et al. in [2].

The data is mapped into a low-dimensional randomized feature space spanned by m basis vectors (called the Fourier components) drawn from the Fourier transform of the kernel function. The inner product of the data points in this feature space approximates the kernel similarity between them. A linear learning algorithm is then applied to this data based on random Fourier features, while obtaining performance similar to that of the kernel-based learning algorithm. This technique has been successfully applied to several learning tasks including classification, regression and retrieval [3]–[5].

In this article, we propose to take advantage of this scheme to speed-up kernel clustering algorithms. We focus on the kernel *k*-means algorithm [6] due to its simplicity and equivalence with most other kernel based clustering algorithms [7]. Given the vector representations of the data points based on the random Fourier features, we propose to apply the k-means algorithm [1] to find the clusters in the data. We show that, when compared to the kernel k-means algorithm, the proposed clustering algorithm achieves an approximation error of $O(1/\sqrt{m})$, where m is the number of Fourier components. To the best of our knowledge, this is the first work that explores random Fourier features for kernel clustering and provides theoretical guarantees. We also propose a more effective scheme, which executes *k*-means on the top singular vectors of the transformed data matrix to identify the clusters in the data set. Our analysis shows that this method achieves an approximation error of $O(1/m)$, provided certain reasonable assumptions are satisfied. We demonstrate the accuracy and efficiency of the proposed algorithms empirically on several large data sets.

The rest of the paper is organized as follows: Section II outlines some of the related literature. We discuss the preliminaries about kernel *k*-means and random Fourier feature maps in Section III, and describe the proposed methods for large scale kernel clustering in Section IV. We present our empirical results in Section V and conclude in Section VI.

II. Related Work

Large scale clustering has been the focus of the data mining research community for the past 20 years. A major challenge that is faced in clustering large data sets containing millions of data points is the high computational and memory complexity.

Several methods have been proposed to cluster large data sets efficiently. Some of the early methods included incremental [8], [9] and divide-and-conquer [10] based clustering algorithms. Algorithms like BIRCH, CURE, CLARA and CLARANS [11]–[13] employed data sampling and summarization techniques to handle large data sets. Of late, distributed techniques, which split the clustering task among a large number of processors are gaining prominence [14]–[17]. Parallel implementations of conventional clustering algorithms like the *k*-means [1] and Canopy clustering [18] algorithms on the MapReduce framework [19] are readily available [20]. However, these approaches are limited to the Euclidean distance based similarity and cannot identify arbitrarily shaped clusters.

Kernel clustering techniques address this limitation by embedding the data points into a high-dimensional non-linear manifold. A number of kernel-based clustering methods such as kernel *k*-means [6], spectral clustering [21], and kernel Self-Organizing Maps (SOM) [22] have been developed. While they usually achieve better performance than the Euclidean distance based clustering algorithms, they are not scalable to large data sets due to their high runtime and memory complexity. Several approximation techniques have been used to address these challenges. Efficiency of spectral clustering was improved using the Nystrom approximation and random projection [23]–[25]. In [26], parallel eigensolvers were employed to obtain the eigenvectors of a sparse approximation of the kernel matrix and perform spectral clustering. The approximate kernel *k*-means algorithm [27] employed the Nystrom approximation to reduce the complexity of kernel *k*-means clustering. In our study, we employ random Fourier maps to improve the efficiency of kernel clustering.

Random Fourier Maps project the data set into a low dimensional space spanned by vectors drawn from the Fourier transform of a shift-invariant kernel function. This technique was deployed in classification [2], [3], kernel regression [4] and data compression [5]. They allow the use of efficient linear learning algorithms to approximate kernel-based learning algorithms, with only a small reduction in accuracy.

III. Background

Kernel *k*-means: Let $\mathcal{D} = \{\mathbf{x}_1, \ldots, \mathbf{x}_n\}$, where $x_i \in \Re^d$ represent the input data set of n data points, to be clustered into C clusters. Let $\kappa(\cdot,\cdot)$ be the kernel function used to compute the similarity between any two data points. In our study, we assume: (i) $\kappa(\cdot,\cdot)$ is shift-invariant[1], i.e. $\kappa(\mathbf{x},\mathbf{y}) = \kappa(\mathbf{x}-\mathbf{y})$, and (ii) $\kappa(\mathbf{x},\mathbf{x}) = \kappa(0) = 1$. Let $\mathcal{H}_\kappa$ be the Reproducing Kernel Hilbert space (RKHS) endowed by the kernel function, and $|\cdot|_{\mathcal{H}_\kappa}$ represent the functional norm for $\mathcal{H}_\kappa$.

The objective of the kernel *k*-means problem is to minimize the *clustering error*, defined as the sum of the squared distance from each point to its cluster center. This can be posed as the following optimization problem:

$$\min_{U\in\{0,1\}^{n\times C}} \max_{\{c_k(\cdot)\in\mathcal{H}\}_{k=1}^{C}} \sum_{k=1}^{C}\sum_{i=1}^{n} \frac{U_{i,k}}{n} |c_k(\cdot) - \kappa(x_i,\cdot)|^2_{\mathcal{H}_\kappa}, \quad (1)$$

where U is the cluster membership matrix satisfying $U_{i,k} = 1$ if data point x_i belongs to cluster k and 0 otherwise, $\{c_k(\cdot)\}_{k=1}^{C}$ are the cluster centers, and $\mathcal{H} = \text{span}(\kappa(\mathbf{x}_1,\cdot),\ldots,\kappa(\mathbf{x}_n,\cdot))$. The factor $1/n$ is introduced only for the purpose of normalization.

Given the cluster membership U, the cluster centers $\{c_k(\cdot)\}_{k=1}^{C}$ can be expressed as

$$c_k(\cdot) = \frac{1}{n_k}\sum_{i=1}^{n} U_{i,k}\kappa(x_i,\cdot), \quad (2)$$

where n_k is the number of data points assigned to cluster k. By substituting (2) in (1), we can express the kernel *k*-means problem as the following trace maximization problem:

$$\max_{U\in\{0,1\}^{n\times C}} \text{tr}(\widetilde{U}^\top K \widetilde{U}), \quad (3)$$

where the normalized kernel matrix $K = \frac{1}{n}[\kappa(x_i,x_j)]_{n\times n}$, $\widetilde{U} = UD^{-1/2}$ and $D = \text{diag}(U^\top \mathbf{1})$.

To solve the problem in (3), we need to compute the $n \times n$ matrix K, which is infeasible for data sets containing millions of data points.

Random Fourier Features: The authors of [2] propose a novel technique for finding a low dimensional mapping of any given data set, such that the dot product of the mapped data points approximates the kernel similarity between them.

Let $p(\mathbf{w})$ denote the Fourier transform of the kernel function $\kappa(\mathbf{x}-\mathbf{y})$, i.e.

$$\kappa(\mathbf{x}-\mathbf{y}) = \int p(\mathbf{w})\exp(j\mathbf{w}^\top(\mathbf{x}-\mathbf{y}))d\mathbf{w}.$$

According to Bochner's theorem [29], $p(\mathbf{w})$ is a valid probability density function, provided the kernel function is continuous and positive-definite.

Let $\mathbf{w}$ be a d-dimensional vector sampled from $p(\mathbf{w})$. The kernel function can be approximated as

$$\kappa(x,y) = \text{E}_{\mathbf{w}}\left[f(\mathbf{w},\mathbf{x})^\top f(\mathbf{w},\mathbf{y})\right], \quad (4)$$

where

$$f(\mathbf{w},\mathbf{x}) = (\cos(\mathbf{w}^\top\mathbf{x}), \sin(\mathbf{w}^\top\mathbf{x}))^\top.$$

We can approximate the expectation in (4) with the empirical mean over m Fourier components $\mathbf{w}_1,\ldots,\mathbf{w}_m$, sampled from the distribution $p(\mathbf{w})$, to obtain a low dimensional

[1]Although this assumption restricts the kernel functions that may be employed, shift-invariant kernels like the RBF kernel are known to perform well on most data sets. Also, many studies including [28] and references therein extend the use of random feature maps to kernels which do not satisfy these criteria.

representation of the point $\mathbf{x}$:

$$\mathbf{z}(\mathbf{x}) = \frac{1}{\sqrt{m}}(\cos(\mathbf{w}_1^\top \mathbf{x}), \ldots, \cos(\mathbf{w}_m^\top \mathbf{x}), \sin(\mathbf{w}_1^\top \mathbf{x}), \ldots, \sin(\mathbf{w}_m^\top \mathbf{x})) \quad (5)$$

Using this mapping, the kernel matrix K can be approximated as

$$\widehat{K} = H^\top H, \quad (6)$$

where

$$H = \left[\mathbf{z}(\mathbf{x}_1)^\top, \ldots, \mathbf{z}(\mathbf{x}_n)^\top\right]. \quad (7)$$

The following theorem shows that the Frobenius norm (denoted by $|\cdot|_F$) of the difference between K and $\widehat{K}$ decreases at the rate of $O(1/\sqrt{m})$[2]. For the sake of simplicity, we present only our main results here and detail the proofs in the Appendix.

Theorem 1: For any $\delta \in (0,1)$, with probability $1-\delta$, we have

$$|\widehat{K} - K|_F \leq \frac{2\ln(2/\delta)}{m} + \sqrt{\frac{2\ln(2/\delta)}{m}} = O\left(\frac{1}{\sqrt{m}}\right).$$

$\widehat{K}$ is a good approximation of K provided that the number of Fourier components m is sufficiently large.

IV. Kernel Clustering using Random Fourier Features

A simple and straightforward method of using random Fourier features for kernel clustering is to replace the kernel matrix K in (3) with the approximate kernel matrix $\widehat{K}$ in (6), leading to the following optimization problem:

$$\max_{U \in \{0,1\}^{n \times C}} \text{tr}(\widetilde{U}^\top H^\top H \widetilde{U}). \quad (8)$$

It is clear that the problem in (8) can be solved by executing k-means [1] on the matrix H. Algorithm 1, which we call the **RFF clustering algorithm**, shows the procedure for clustering using the random Fourier features obtained from the RBF kernel.

In the following theorem, we bound the difference between the solutions of (3) and (8):

Theorem 2: Let U^* and U_m^* be the optimal solutions of (3) and (8), respectively. Let $\widetilde{U}^* = U^*[D^*]^{-1/2}$ and $\widetilde{U}_m^* = U_m^*[D_m^*]^{-1/2}$ denote the normalized versions of U^* and U_m^*, where $D^* = \text{diag}([U^*]^\top \mathbf{1})$ and $D_m^* = \text{diag}([U_m^*]^\top \mathbf{1})$.

For any $\delta \in (0,1)$, with probability $1-\delta$, we have

$$\text{tr}\left([\widetilde{U}^* - \widetilde{U}_m^*]^\top K [\widetilde{U}^* - \widetilde{U}_m^*]\right) \leq \frac{4\ln(2/\delta)}{m} + \sqrt{\frac{8\ln(2/\delta)}{m}} = O\left(\frac{1}{\sqrt{m}}\right).$$

Theorem 2 indicates that the approximation error of the RFF clustering algorithm reduces as the number of random Fourier components m increases.

[2]This is a significant improvement over the $O(m^{-1/4})$ approximation error achieved by the Nystrom technique for kernel approximation [30].

Despite its simplicity, RFF clustering may suffer from high computational cost. A large number of random Fourier components may be required to achieve a low approximation error, as a result of which, we need to execute k-means over a high-dimensional space, leading to high runtime complexity. To address this problem, we propose using an idea similar to that in [27] and constrain the cluster centers to lie in the subspace spanned by the top eigenvectors of the kernel matrix. Let $\{(\lambda_i, \mathbf{v}_i)\}_{i=1}^n$ denote the eigenvalues and eigenvectors of the kernel matrix K, ranked in the descending order of the eigenvalues. It can be proved that when the last $n-C$ eigenvalues $\{\lambda_i\}_{i=C+1}^n$ of K are sufficiently small, the subspace $\mathcal{H}$ can be well approximated by the subspace spanned by the top C eigenvectors of K, i.e. $\mathcal{H} \sim \mathcal{H}_a = \text{span}(\mathbf{v}_1, \ldots, \mathbf{v}_C)$[3]. Hence, the kernel k-means problem in (1) can be approximated as

$$\min_{U \in \{0,1\}^{n \times C}} \max_{\{c_k(\cdot) \in \mathcal{H}_a\}_{k=1}^C} \sum_{k=1}^{C} \sum_{i=1}^{n} \frac{U_{i,k}}{n} |c_k(\cdot) - \kappa(x_i, \cdot)|_{\mathcal{H}_\kappa}^2, \quad (9)$$

which can be solved by executing k-means on the top eigenvectors of K, i.e. by solving the following optimization problem:

$$\arg\max_{Z \in \{0,1\}^{n \times C}} \text{tr}(Z^\top [V_C V_C^\top] Z), \quad (10)$$

where $V_C = (\mathbf{v}_1, \ldots, \mathbf{v}_C)$ and Z represents the cluster membership matrix. This method leads to a significant reduction in computational cost when compared to the RFF clustering algorithm, as each data point is represented by a C-dimensional vector and k-means needs to be executed over a lower dimensional space.

However, computing the eigenvectors of K requires the computation of the $n \times n$ kernel matrix, which is infeasible when n is large. We address this challenge by approximating the eigenvectors of K using the singular vectors of the random Fourier features, and thereby avoid computing the full kernel matrix. More specifically, we compute the top C singular values and the corresponding left singular vectors of H, denoted by $\{(\widehat{\lambda}_i, \widehat{\mathbf{v}}_i)\}_{i=1}^C$, where C is the number of clusters[4], and represent the data points in $\mathcal{D}$ by the matrix $\widehat{V}_C = (\widehat{\mathbf{v}}_1, \ldots, \widehat{\mathbf{v}}_C)$. We then solve the approximate optimization problem

$$\arg\max_{Z \in \{0,1\}^{n \times C}} \text{tr}(Z^\top [\widehat{V}_C \widehat{V}_C^\top] Z), \quad (11)$$

by executing k-means on the matrix $\widehat{V}_C$ to obtain the C clusters. This procedure, named as the **SV clustering algorithm**, is outlined in Algorithm 2. It has the same input and output as the RFF clustering algorithm, but differs in the final two steps.

As the dimensionality of the input to the k-means clustering step in the SV clustering algorithm is significantly

[3]Refer Theorem 4 in Appendix.

[4]We follow the framework of spectral clustering, and set the number of eigenvectors equal to the number of clusters.

Algorithm 1 RFF clustering

Input:

- $\mathcal{D} = (x_1, \ldots, x_n)$: set of n data points to be clustered
- λ: RBF kernel width parameter
- m: number of Fourier components
- C: number of clusters

Output: Cluster membership matrix $U \in \{0,1\}^{n \times C}$

1: Draw m independent samples $\mathbf{w}_1, \ldots, \mathbf{w}_m$ from the Gaussian distribution $\mathcal{N}(0, \frac{1}{\lambda} I)$.
2: Compute the matrix H, as defined in (7).
3: Run the *k*-means algorithm on H with the number of clusters set to C and obtain the membership matrix U.

Algorithm 2 SV clustering

Input:

- $\mathcal{D} = (x_1, \ldots, x_n)$: set of n data points to be clustered
- λ: RBF kernel width parameter
- m: number of Fourier components
- C: number of clusters

Output: Cluster membership matrix $U \in \{0,1\}^{n \times C}$

1: Draw m independent samples $\mathbf{w}_1, \ldots, \mathbf{w}_m$ from the Gaussian distribution $\mathcal{N}(0, \frac{1}{\lambda} I)$.
2: Compute the matrix H, as defined in (7).
3: Compute the left singular vectors of H corresponding to its top C singular values to obtain the matrix $\widehat{V}_C = (\widehat{\mathbf{v}}_1, \ldots, \widehat{\mathbf{v}}_C)$.
4: Run the *k*-means algorithm on $\widehat{V}_C$ with the number of clusters set to C and obtain the membership matrix U.

smaller than that in the RFF clustering algorithm, SV clustering is more efficient than RFF clustering, despite the overhead of computing the singular vectors. In the following theorem, we show that the SV clustering algorithm yields a better approximation of kernel clustering than the RFF clustering algorithm, provided there is a sufficiently large gap in the eigenspectrum.

Theorem 3: Let Z^* and Z^*_m be the optimal solutions of (10) and (11), and let $\widetilde{Z}^*$ and $\widetilde{Z}^*_m$ represent their normalized versions (as in Theorem 2), respectively.

Given $\delta \in (0,1)$, assume $(\lambda_C - \lambda_{C+1}) \geq 3\Delta$, where

$$\Delta = \frac{2\ln(2/\delta)}{m} + \sqrt{\frac{2\ln(2/\delta)}{m}}. \tag{12}$$

With probability $1 - \delta$, we have

$$\text{tr}\left([\widetilde{Z}^* - \widetilde{Z}^*_m]^\top [\widetilde{Z}^* - \widetilde{Z}^*_m]\right) \leq \frac{18\Delta^2}{(\lambda_C - \lambda_{C+1})^2} = O\left(\frac{1}{m}\right).$$

Theorem 3 shows that, like the RFF clustering algorithm, the SV clustering algorithm's approximation error reduces as the number of Fourier components increases. We note that the assumption of the existence of a large eigengap, adopted by many earlier kernel-based algorithms which rely on the spectral embedding of the data [21], essentially implies that most attributes of the data can be well approximated by vectors in a low-dimensional space.

Computational Complexity Sampling from the Fourier transform of the kernel function is a relatively inexpensive operation for most shift-invariant kernels. For instance, several efficient techniques have been proposed for sampling from a Gaussian distribution in the literature [31]. The crux of the proposed algorithms thus lies in computing the low-dimensional random Fourier features H and performing Singular Value Decomposition (SVD) of H. Given m d-dimensional Fourier components, the mapping to the matrix H can be performed in $O(ndm)$ time. Executing *k*-means on H takes $O(nmCl)$ time, where l is the number of iterations required for convergence.

In the SV clustering algorithm, if the top singular vectors of H are found using conventional methods, the runtime complexity of the SVD step would be $O(nm^2)$. We reduce this complexity in our implementation by using the approximate SVD technique proposed in [32]. We sample s rows from H to form an $s \times m$ matrix S. The top eigenvectors of $S^\top S$, denoted by $\widetilde{V} = (\widetilde{\mathbf{v}}_1, \ldots, \widetilde{\mathbf{v}}_C)$, are close to the top eigenvectors of $H^\top H$ and the singular vectors of H can be recovered from the eigenvectors of $S^\top S$ as $H\widetilde{V}$. Using this approximation, the runtime complexity of SVD is reduced to $O(s^2 m)$. The time taken to execute *k*-means on the singular vectors is $O(nC^2 l)$.

When $\max(m, s, l, C) \ll n$, the proposed algorithms have linear time complexity. We note that the time taken by the *k*-means step in the SV clustering algorithm is $O(nC^2 l)$, as opposed to $O(nmCl)$, the time taken by the *k*-means step in the RFF clustering algorithm. As seen in Theorem 2, the value of m needs to be large for the RFF clustering algorithm to achieve a low approximation error. As a consequence, $m \gg C$ and therefore, the SV clustering algorithm is more efficient than the RFF clustering algorithm.

The values chosen for m and s introduce a trade-off between the clustering quality and efficiency. Higher values result in better clustering quality but lesser speedup. In our implementation, we found that a reasonably good accuracy can be achieved by setting the value of m to range between 1% and 2% of n, and setting s to around 2% of n. Lower m/n ratio values work well as n increases.

V. Empirical Analysis

In this section, we evaluate the proposed clustering algorithms in terms of their clustering accuracy and scalability. We compare their performance with state-of-the-art kernel clustering algorithms on four public domain data sets: MNIST, Forest Cover Type, Poker and Network Intrusion.

A. Data sets

We use four benchmark data sets to evaluate the large scale clustering performance of the proposed schemes:

- **MNIST:** The MNIST data set [33] is a subset of the database of handwritten digits available from NIST. It contains $60,000$ training images and $10,000$ test images from 10 classes. Each image is represented as a 784-dimensional feature vector. For the purpose of clustering, we combine the training and test images to form a data set containing $70,000$ images.
- **Forest Cover Type:** The Forest Cover Type data set [34] is composed of $581,012$ data points from the US Geological Survey (USGS) and the US Forest Service (USFS). Each data point is represented by a vector of 54 dimensions and assigned to one of 7 classes, each class representing a Forest Cover type.
- **Poker:** This data set [35] contains $1,025,010$ data points, each represented by a 30-dimensional vector and assigned to one of 10 classes. Each class represents a poker hand.
- **Network Intrusion:** The Network Intrusion data set [36] contains $4,898,431$ 50-dimensional data points representing the TCP dump data from seven weeks of a local-area network traffic. The data is classified into 23 classes representing different types of network traffic. We filtered out the data from classes which contain fewer than 500 data points, to form a data set with $4,897,988$ data points from 10 classes.

B. Baseline algorithms

We compare the proposed algorithms with the kernel *k*-means algorithm to demonstrate that they achieve similar clustering accuracy. We also compare their performance with the approximate kernel *k*-means algorithm [27], which employs a randomly sampled subset of the data to speedup kernel *k*-means, and the Nystrom approximation based spectral clustering algorithm [24], which clusters the top C eigenvectors of a low rank approximate kernel matrix obtained from a randomly sampled subset of the data, using the Nystrom approximation technique [23]. We also gauge the performance of our algorithms against that of the *k*-means algorithm to show that they achieve better clustering accuracy.

C. Parameters

We use the RBF kernel for all the kernel-based algorithms on all the data sets. We set λ equal to $\rho\mathfrak{d}$, where $\mathfrak{d}$ is the average pairwise Euclidean distance between the data points and parameter ρ is tuned in the range $[0, 1]$ to obtain optimal performance[5]. We vary the number of Fourier components m from 100 to 2000 for the first three data sets. On the Network Intrusion data set, the value of m is varied only upto 1000, in order to limit the memory requirement to 40 GB. For the approximate kernel *k*-means and spectral clustering algorithms, m represents the size of the sample drawn from the data set. The value of s, the number of rows sampled from H to compute the approximate singular vectors, is set to 2% of the total number of data points. The number of clusters C is set equal to the true number of classes in the data set.

All algorithms were implemented in MATLAB[6] and run on a 2.8 GHz processor using 40 GB RAM. All results are averaged over 10 runs of the algorithms. In each run of the proposed algorithms, we use a different set of randomly sampled Fourier components. For the baseline algorithms which use a subset of the data, we use different randomly sampled subsets in each run.

D. Experimental results

Clustering accuracy

We measure the clustering accuracy of the proposed algorithms in terms of the *Normalized Mutual Information* (NMI) [39] between the cluster labels and the true class labels. Let $U_c = (\mathbf{u}_1^c, \cdots, \mathbf{u}_C^c)$ and $U_t = (\mathbf{u}_1^t, \cdots, \mathbf{u}_C^t)$ be the membership matrices corresponding to the cluster labels assigned by the clustering algorithm, and the true class labels respectively[7]. The NMI achieved by the clustering algorithm, is defined by

$$NMI(U_c, U_t) = \frac{\sum_{i=1}^{C}\sum_{j=1}^{C} n_{i,j}^{c,t} \log\left(\frac{n.n_{i,j}^{c,t}}{n_i^c n_j^t}\right)}{\sqrt{\left(\sum_{i=1}^{C} n_i^c \log\frac{n_i^c}{n}\right)\left(\sum_{j=1}^{C} n_j^t \log\frac{n_j^t}{n}\right)}},$$

where $n_i^c = \mathbf{u}_i^{c\top}\mathbf{1}$ represents the number of data points that have been assigned the cluster label i, $n_j^t = \mathbf{u}_j^{t\top}\mathbf{1}$ represents the number of data points that belong to class j, and $n_{i,j}^{c,t} = \mathbf{u}_i^{c\top}\mathbf{u}_j^t$ represents the number of data points that have been assigned label i but belong to class j. An NMI value of 1 indicates a perfect match between the cluster labels and the true class labels, whereas 0 indicates perfect mismatch.

The NMI values achieved by the algorithms on the four data sets are shown in Fig. 1. We first observe that the accuracy of all the kernel based algorithms, including the proposed algorithms, is better than that of the *k*-means algorithm, demonstrating the fact that incorporating a non-linear distance function improves the clustering performance.

We also note that, although the SV clustering algorithm performs worse than the RFF clustering algorithm in terms of NMI when m is small, it yields similar performance as

[5]Note that the kernel width parameter can be set using other heuristics [21], [37], [38]. Parameter selection is not part of the clustering algorithms. The performance of our algorithms relative to the baseline algorithms, is not dependent on the heuristics employed to choose the kernel parameters, provided the parameters are tuned to achieve optimal performance on the given data set.

[6]We used the *k*-means implementation in the MATLAB Statistics Toolbox and the Nystrom approximation based spectral clustering implementation [26] available at http://alumni.cs.ucsb.edu/~wychen/sc.html. The remaining algorithms were implemented in-house.

[7]Note that we set the number of clusters equal to the true number of classes in the data set.

the RFF clustering algorithm when $m \geq 500$ for all the data sets. On the MNIST data set, as the value of m increases from 100 to 2000, the average NMI achieved by the RFF clustering algorithm increases by about 15% whereas the SV clustering algorithm achieves an increase of 20%. Similar rates of increase are observed on other data sets also. This verifies our claim that the SV clustering algorithm has a faster convergence rate than the RFF clustering algorithm for kernel clustering.

On the MNIST data set, we observe that the performance of both our algorithms is similar to that of kernel *k*-means when $m \geq 1000$. The difference in NMI of the proposed algorithms and kernel *k*-means reduces as the number of Fourier components increases. Comparison with kernel *k*-means is not feasible on the remaining data sets due to their large size.

The proposed algorithms' performance is significantly better than that of the Nystrom approximation based spectral clustering algorithm on all data sets. On the MNIST data set, they perform slightly worse than the approximate kernel *k*-means algorithm, when m is small but the difference in NMI becomes negligible as m increases. On the remaining data sets, almost identical performance is achieved by our algorithms and the approximate kernel *k*-means algorithm, even for small values of m.

Scalability

The running time of the baseline algorithms and the proposed algorithms are recorded in Table I.

We first observe that the RFF clustering algorithm takes longer time than the SV clustering algorithm on all the data sets. Though both algorithms require the computation of the data matrix H, the time taken to perform this computation is insignificant when compared to the *k*-means clustering time. RFF clustering involves running *k*-means on a $2m$-dimensional matrix, which takes longer than running *k*-means on a C-dimensional matrix. Although the SV clustering algorithm includes computing the singular vectors of H, the overhead of performing SVD is small, rendering it more efficient than the RFF clustering algorithm. On the MNIST data set, the SV clustering algorithm is about 20 times faster than the RFF clustering algorithm, even when m is as small as 100. Similar speedup values are observed on the remaining data sets also. Given the fact that the SV clustering algorithm achieves similar clustering accuracy as the RFF clustering algorithm when the number of Fourier components is sufficiently large, we conclude that **the SV clustering algorithm is more suitable for large scale kernel clustering than the RFF clustering algorithm**. In the table, we emphasize the values of m for which the SV clustering algorithm is as accurate as the RFF clustering algorithm but more efficient.

Although both the algorithms are more efficient when compared to kernel *k*-means with the full kernel matrix on the MNIST data set, the SV clustering algorithm achieves a higher speedup than the RFF clustering algorithm. On an average, the SV clustering algorithm is about 30 times faster than kernel *k*-means.

The Nystrom approximation based spectral clustering algorithm finds the clusters by executing *k*-means on the top eigenvectors of a low rank approximate kernel matrix derived from a randomly sampled subset of the data set. It first obtains the eigenvectors of an $m \times m$ matrix and then extrapolates them to the top eigenvectors of the $n \times n$ kernel matrix. As the SV clustering algorithm only finds the top singular vectors of an $s \times m$ matrix, it is more efficient than the Nystrom approximation based spectral clustering algorithm in most cases. We note that the speedup on the Network Intrusion data set becomes significant only when $m \geq 500$. However, as seen earlier, the SV clustering algorithm achieves a better clustering accuracy even when m is small, thereby rendering it more efficient.

The SV clustering algorithm is also faster than approximate kernel *k*-means on all the data sets, primarily due to the fact that unlike the approximate kernel *k*-means algorithm, the dimensionality of the input to the *k*-means step (which dominates the running time) remains constant despite the increase in m. The input dimensionality of *k*-means in the approximate kernel *k*-means algorithm increases linearly with m.

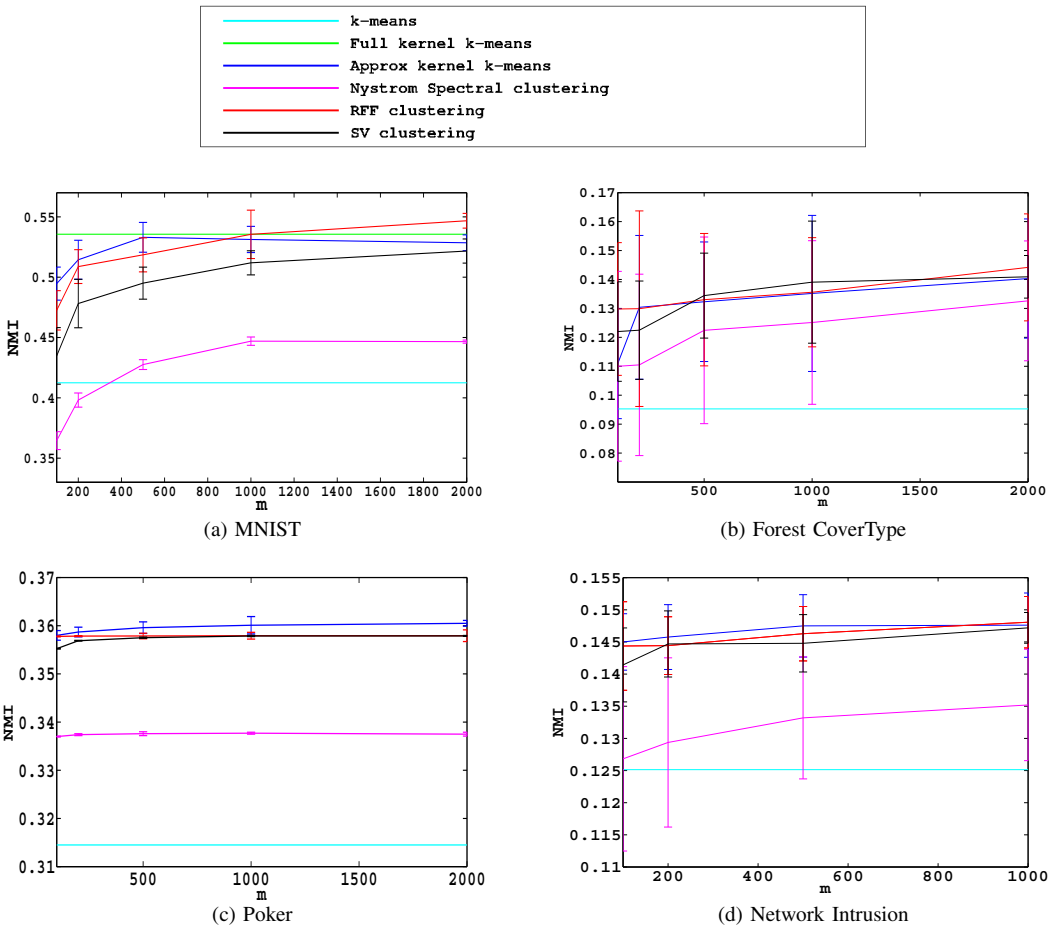
Again, as the dimensionality of each data set is greater than the number of clusters in it, the SV clustering algorithm run faster than the *k*-means algorithm, when m is sufficiently small.

All these results indicate that the SV clustering algorithm which uses the singular vectors of the random Fourier features to identify the clusters in the data set is an accurate and scalable approximation of kernel *k*-means.

VI. Conclusions

We have proposed two algorithms, the RFF clustering algorithm and the SV clustering algorithm, which use random Fourier features to obtain a good approximation of kernel clustering using an efficient linear clustering algorithm. We have analytically bound the approximation error of both these methods. We have shown that, when there is a large gap in the eigenspectrum of the kernel matrix, the SV clustering algorithm which clusters the singular vectors of the random Fourier features is a more effective and scalable approximation of kernel clustering, allowing large data sets with millions of data points to be clustered using kernel-based clustering. We have demonstrated the accuracy and scalability of our algorithms on several large data sets.

The proposed algorithms are currently applicable only to kernel functions with analytical inverse Fourier transforms. In the future, we plan to extend our work to other random feature maps and design more efficient clustering algorithms



(a) MNIST (b) Forest CoverType

(c) Poker (d) Network Intrusion

Figure 1: NMI values with respect to true labels. m represents the number of Fourier components in the context of the proposed methods, and it represents the size of the sample drawn from the data set in the context of approximate kernel k-means and Nystrom approximation based spectral clustering. (Figures best viewed in color)

using a combination of the Nystrom approximation and random feature maps.

ACKNOWLEDGMENT

This research was supported by the Office of Naval Research (ONR Grant N00014-11-1-0100).

APPENDIX

EIGENSPACE APPROXIMATION

The following theorem shows that when the last $n - C$ eigenvalues of the kernel matrix K are sufficiently small, an approximate solution to the kernel k-means problem in (1) can be obtained by constraining the cluster centers to the subspace $\mathcal{H}_a$ spanned by the eigenvectors of K corresponding to its top C eigenvalues $\{\lambda_i\}_{i=1}^C$:

Theorem 4: Let E and E_a represent the optimal clustering errors in (1) and (9), respectively. We have

$$|E - E_a| \leq \sum_{i=C+1}^{n} \lambda_i.$$

Proof: Let $\{c_k^*(\cdot)\}_{k=1}^C$ and U^* be the optimal solutions to (1). Let $c_k^a(\cdot)$ represent the projection of c_k^* into the subspace $\mathcal{H}_a$. For any $\kappa(\mathbf{x}_i, \cdot)$, let $g_i(\cdot)$ and $h_i(\cdot)$ be the projections of $\kappa(\mathbf{x}_i, \cdot)$ into the subspace $\mathcal{H}_a$ and into span$(\mathbf{v}_{C+1}, \ldots, \mathbf{v}_n)$, respectively. We have

$$\begin{aligned}
E_a &= \min_U \max_{c_k(\cdot) \in \mathcal{H}_a} \sum_{k=1}^{C} \sum_{i=1}^{n} \frac{U_{i,k}}{n} |c_k(\cdot) - \kappa(x_i, \cdot)|_{\mathcal{H}_\kappa}^2 \\
&\leq \sum_{k=1}^{C} \sum_{i=1}^{n} \frac{U_{i,k}^*}{n} |c_k^a(\cdot) - \kappa(x_i, \cdot)|_{\mathcal{H}_\kappa}^2 \\
&= \sum_{k=1}^{C} \sum_{i=1}^{n} \frac{U_{i,k}^*}{n} \left(|c_k^a(\cdot) - g_i(\cdot)|_{\mathcal{H}_\kappa}^2 + |h_i(\cdot)|_{\mathcal{H}_\kappa}^2 \right) \\
&\leq E + \frac{1}{n} \sum_{k=1}^{C} \sum_{i=1}^{n} |h_i(\cdot)|_{\mathcal{H}_\kappa}^2 \\
&\leq E + \sum_{i=C+1}^{n} \lambda_i.
\end{aligned}$$

Table I: Comparison of running time in seconds

k-means clustering time: 448.69 ($\pm$177.24)				
Kernel k-means: 1245.45 ($\pm$242.59)				
m*	Approx. kernel k-means	Nystrom Spectral clustering	RFF clustering	SV clustering
100	26.57 ($\pm$3.12)	6.00 ($\pm$0.89)	85.36 ($\pm$25.64)	3.85 ($\pm$2.37)
200	17.98 ($\pm$7.99)	46.70 ($\pm$8.51)	122.31 ($\pm$48.31)	4.66 ($\pm$1.78)
500	24.72 ($\pm$8.46)	342.38 ($\pm$105.80)	272.57 ($\pm$111.25)	9.22 ($\pm$1.22)
1000	36.34 ($\pm$6.92)	914.18 ($\pm$215.77)	517.48 ($\pm$44.6)	17.46 ($\pm$1.43)
2000	86.43 ($\pm$12.71)	4163.76 ($\pm$383.37)	1089.26 ($\pm$483.63)	**39.94 ($\pm$5.64)**

(a) MNIST

k-means clustering time: 29.28 ($\pm$9.12)				
m*	Approx. kernel k-means	Nystrom Spectral clustering	RFF clustering	SV clustering
100	19.10 ($\pm$6.35)	11.75 ($\pm$1.73)	154.97 ($\pm$65.72)	9.62 ($\pm$2.57)
200	24.21 ($\pm$12.48)	13.65 ($\pm$1.59)	174.88 ($\pm$65.36)	10.77 ($\pm$1.67)
500	32.48 ($\pm$11.64)	41.92 ($\pm$7.89)	534.01 ($\pm$216.18)	**22.15 ($\pm$6.08)**
1000	66.15 ($\pm$19.25)	124.83 ($\pm$38.32)	1032.58 ($\pm$221.56)	35.46 ($\pm$5.20)
2000	157.48 ($\pm$27.37)	534.77 ($\pm$323.76)	2078.63 ($\pm$617.22)	76.99 ($\pm$17.04)

(b) Forest Cover Type

k-means clustering time: 40.88($\pm$6.4)				
m*	Approx. kernel k-means	Nystrom Spectral clustering	RFF clustering	SV clustering
100	55.57 ($\pm$11.22)	10.88 ($\pm$1.65)	144.22 ($\pm$11.88)	12.32 ($\pm$1.70)
200	89.14 ($\pm$32.12)	46.78 ($\pm$4.21)	411.32 ($\pm$34.34)	17.35 ($\pm$2.07)
500	117.57 ($\pm$20.17)	90.57 ($\pm$18.57)	654.98 ($\pm$132.70)	**22.82 ($\pm$2.48)**
1000	202.84 ($\pm$44.25)	261.14 ($\pm$20.51)	2287.53 ($\pm$159.06)	27.37 ($\pm$2.09)
2000	256.26 ($\pm$44.84)	479.73 ($\pm$47.46)	4530.44 ($\pm$276.37)	41.08 ($\pm$2.57)

(c) Poker

k-means clustering time: 953.41 ($\pm$169.38)				
m*	Approx. kernel k-means	Nystrom Spectral clustering	RFF clustering	SV clustering
100	736.59 ($\pm$238.31)	145.21 ($\pm$22.76)	2252.44 ($\pm$465.94)	147.93 ($\pm$62.03)
200	697.04 ($\pm$442.25)	169.27 ($\pm$38.15)	5371.85 ($\pm$1765.02)	**258.86 ($\pm$41.32)**
500	586.14 ($\pm$130.23)	366.42 ($\pm$175.57)	5296.87 ($\pm$3321.66)	245.37 ($\pm$158.57)
1000	763.75 ($\pm$88.55)	589.57 ($\pm$54.14)	24151.47 ($\pm$6351.34)	435.53 ($\pm$189.07)

(d) Network Intrusion

*m represents the number of Fourier components in the context of the proposed methods, and it represents the size of the sample drawn from the data set in the context of approximate kernel k-means and Nystrom approximation based spectral clustering.
The values of m for which the SV clustering algorithm is as accurate as the RFF clustering algorithm but more efficient are emphasized.

■

PROOFS OF THEOREMS

Proof of Theorem 1

To prove this theorem, we use the following lemma from [40]:

Lemma 1: Let $\mathcal{H}$ be a Hilbert space and ξ be a random variable on (Z, ρ) with values in $\mathcal{H}$. Assume $\|\xi\| \leq M < \infty$ almost surely. Denote $\sigma^2(\xi) = \mathrm{E}(\|\xi\|^2)$. Let $\{z_i\}_{i=1}^m$ be independent random drawers of ρ. For any $0 < \delta < 1$, with confidence $1 - \delta$,

$$\left\| \frac{1}{m} \sum_{i=1}^{m} (\xi_i - \mathrm{E}[\xi_i]) \right\| \leq \frac{2M \ln(2/\delta)}{m} + \sqrt{\frac{2\sigma^2(\xi) \ln(2/\delta)}{m}}. \tag{13}$$

Define

$$\mathbf{a}(\mathbf{w}) = \frac{1}{\sqrt{n}} (\cos(\mathbf{w}^\top \mathbf{x}_1), \ldots, \cos(\mathbf{w}^\top \mathbf{x}_n))^\top \text{ and}$$
$$\mathbf{b}(\mathbf{w}) = \frac{1}{\sqrt{n}} (\sin(\mathbf{w}^\top \mathbf{x}_1), \ldots, \sin(\mathbf{w}^\top \mathbf{x}_n))^\top.$$

Let $\xi_i = \mathbf{a}(\mathbf{w}_i)\mathbf{a}(\mathbf{w}_i)^\top + \mathbf{b}(\mathbf{w}_i)\mathbf{b}(\mathbf{w}_i)^\top$. We have $\mathrm{E}[\xi_i] = \mathrm{E}[\mathbf{a}(\mathbf{w}_i)\mathbf{a}(\mathbf{w}_i)^\top + \mathbf{b}(\mathbf{w}_i)\mathbf{b}(\mathbf{w}_i)^\top] = K$ and $|\xi_i|_F^2 = |\mathbf{a}(\mathbf{w}_i)|^2 + |\mathbf{b}(\mathbf{w}_i)|^2 = 1$, which implies $M = \sigma^2 = 1$. We obtain the required result by substituting these values in (13).

Proof of Theorem 2

We have

$$\begin{aligned} \mathrm{tr}([\widetilde{U}^*]^\top K \widetilde{U}^*) &\leq \mathrm{tr}([\widetilde{U}^*]^\top \widehat{K} \widetilde{U}^*) + |K - \widehat{K}|_F \\ &\leq \mathrm{tr}([\widetilde{U}_m^*]^\top \widehat{K} \widetilde{U}_m^*) + |K - \widehat{K}|_F \\ &\leq \mathrm{tr}([\widetilde{U}_m^*]^\top K \widetilde{U}_m^*) + 2|K - \widehat{K}|_F. \end{aligned}$$

Since $\mathrm{tr}([\widetilde{U}^*]^\top K \widetilde{U}^*) \geq \mathrm{tr}([\widetilde{U}_m^*]^\top K \widetilde{U}_m^*)$, we have

$$|\mathrm{tr}([\widetilde{U}^*]^\top K \widetilde{U}^*) - \mathrm{tr}([\widetilde{U}_m^*]^\top K \widetilde{U}_m^*)| \leq 2|K - \widehat{K}|_F.$$

We complete the proof by using the result from Theorem 1 and the strong convexity property of $\mathrm{tr}(U^\top K U)$.

Proof of Theorem 3

We prove Theorem 3 by using the following two lemmas and performing analysis similar to that in the proof of Theorem 2:

Lemma 2: Given $\delta \in (0, 1)$, we assume $(\lambda_C - \lambda_{C+1}) \geq 3\Delta$, where Δ is defined in (12). Then, with probability $1-\delta$, there exists a matrix $P \in \mathbb{R}^{(n-C)\times C}$ satisfying

$$|P|_F \leq \frac{2\Delta}{\lambda_C - \lambda_{C+1} - \Delta},$$

such that

$$\widehat{V}_C = (V_C + \bar{V}_C P)(I + P^\top P)^{-1/2},$$

where $\widehat{V}_C = (\widehat{\mathbf{v}}_1, \ldots, \widehat{\mathbf{v}}_C)$, $V_C = (\mathbf{v}_1, \ldots, \mathbf{v}_C)$, and $\bar{V}_C = (\mathbf{v}_{C+1}, \ldots, \mathbf{v}_n)$.

Proof: We use the following result from matrix perturbation theory [41]:

Lemma 3: Let $(\lambda_i, \mathbf{v}_i), i \in [n]$ be the eigenvalues and eigenvectors of a symmetric matrix $A \in \mathbb{R}^{n \times n}$ ranked in the descending order of eigenvalues. Set $X = (\mathbf{v}_1, \ldots, \mathbf{v}_r)$ and $Y = (\mathbf{v}_{r+1}, \ldots, \mathbf{v}_n)$. Given a symmetric perturbation matrix E, let

$$(X, Y)^\top E (X, Y) = \begin{pmatrix} E_{11} & E_{12} \\ E_{21} & E_{22} \end{pmatrix}.$$

Let $\|\cdot\|$ represent a consistent family of norms and let

$$\gamma = \|E_{21}\|, \delta = \lambda_r - \lambda_{r+1} - \|E_{11}\| - \|E_{22}\|.$$

If $\delta > 0$ and

$$\frac{\gamma}{\delta} < \frac{1}{2},$$

then there exists a unique matrix $P \in \mathbb{R}^{(n-r) \times r}$ satisfying

$$\|P\| < \frac{2\gamma}{\delta},$$

such that

$$\begin{aligned} X' &= (X + YP)(I + P^\top P)^{-1/2}, \text{ and} \\ Y' &= (Y - XP^\top)(I + PP^\top)^{-1/2} \end{aligned}$$

are the eigenvectors of $A + E$.

Let $E = \widehat{K} - K$. Using Theorem 1, we have

$$\gamma = \|V_C^\top E \bar{V}_C\| \leq \|E\| \leq \Delta,$$

and

$$\begin{aligned} \delta &= \lambda_C - \lambda_{C+1} - \|V_C^\top E V_C\| - \|\bar{V}_C^\top E \bar{V}_C\| \\ &\geq \lambda_C - \lambda_{C+1} - \Delta > 0. \end{aligned}$$

As $\lambda_C - \lambda_{C+1} \geq 3\Delta$, we also have

$$\frac{\gamma}{\delta} < \frac{1}{2},$$

allowing us to apply Lemma 3 and obtain the required result. ■

Lemma 4: Under the assumptions of Lemma 2, with a probability $1 - \delta$, we have

$$\sum_{i=1}^{C} |\widehat{\mathbf{v}}_i - \mathbf{v}_i|^2 \leq 2|P|_F^2 \leq \frac{18\Delta^2}{(\lambda_C - \lambda_{C+1})^2},$$

where Δ is defined in (12).

Proof: Define $A = P(I + P^\top P)^{-1/2}$ and $B = I - (I + P^\top P)^{-1/2}$. Let $\{\gamma_i\}_{i=1}^C$ be the eigenvalues of $P^\top P$. Using the result from Lemma 2, we have

$$\begin{aligned} &\sum_{i=1}^{C} |\widehat{\mathbf{v}}_i - \mathbf{v}_i|^2 = |\bar{V}_C A|_F^2 + |V_C B|_F^2 \\ \leq\ & |A|_F^2 + |B|_F^2 \leq |P|_F^2 + \sum_{i=1}^{C} \frac{\gamma_i}{(1 + \sqrt{\gamma_i})^2} \leq 2|P|_F^2. \end{aligned}$$

We complete the proof by using the fact that

$$|P|_F \leq \frac{2\Delta}{\lambda_C - \lambda_{C+1} - \Delta} \leq \frac{3\Delta}{\lambda_C - \lambda_{C+1}}.$$

■

REFERENCES

[1] A. Jain, "Data clustering: 50 years beyond k-means," *Pattern Recognition Letters*, vol. 31, no. 8, pp. 651–666, 2010.

[2] A. Rahimi and B. Recht, "Random features for large-scale kernel machines," *Advances in Neural Information Processing Systems*, vol. 20, pp. 1177–1184, 2007.

[3] L. Bo and C. Sminchisescu, "Efficient match kernel between sets of features for visual recognition," *Advances in Neural Information Processing Systems*, vol. 22, pp. 135–143, 2009.

[4] O. Maillard and R. Munos, "Compressed least-squares regression," *Advances in Neural Information Processing Systems*, vol. 22, pp. 1213–1221, 2009.

[5] M. Raginsky and S. Lazebnik, "Locality-sensitive binary codes from shift-invariant kernels," *Advances in Neural Information Processing Systems*, vol. 22, pp. 1509–1517, 2009.

[6] B. Scholkopf, A. Smola, and K. Muller, "Nonlinear component analysis as a kernel eigenvalue problem," *Neural Computation*, vol. 10, no. 5, pp. 1299–1314, 1996.

[7] I. Dhillon, Y. Guan, and B. Kulis, "A unified view of kernel k-means, spectral clustering and graph cuts," University of Texas at Austin, Tech. Rep., 2004, (Tech. rep. TR-04-25).

[8] F. Can, "Incremental clustering for dynamic information processing," *ACM Transactions on Information Systems*, vol. 11, no. 2, pp. 143–164, 1993.

[9] P. Bradley, U. Fayyad, and C. Reina, "Scaling em (expectation-maximization) clustering to large databases," Microsoft Research, Tech. Rep., 1998.

[10] S. Guha, A. Meyerson, N. Mishra, R. Motwani, and L. O'Callaghan, "Clustering data streams: Theory and practice," *IEEE Transactions on Knowledge and Data Engineering*, pp. 515–528, 2003.

[11] T. Zhang, R. Ramakrishnan, and M. Livny, "BIRCH: an efficient data clustering method for very large databases," *ACM SIGMOD Record*, vol. 25, no. 2, pp. 103–114, 1996.

[12] S. Guha, R. Rastogi, and K. Shim, "CURE: an efficient clustering algorithm for large databases," *Information Systems*, vol. 26, no. 1, pp. 35–58, 2001.

[13] R. Ng and J. Han, "CLARANS: A method for clustering objects for spatial data mining," *IEEE Transactions on Knowledge and Data Engineering*, pp. 1003–1016, 2002.

[14] B. Bahmani, B. Moseley, A. Vattani, R. Kumar, and S. Vassilvitskii, "Scalable k-means++," *Proceedings of the VLDB Endowment*, vol. 5, no. 7, pp. 622–633, 2012.

[15] A. Ene, S. Im, and B. Moseley, "Fast clustering using mapreduce," in *Proceedings of the International Conference on Knowledge discovery and Data mining*, 2011, pp. 681–689.

[16] T. Liu, C. Rosenberg, and H. Rowley, "Clustering billions of images with large scale nearest neighbor search," in *Proceedings of the IEEE Workshop on Applications of Computer Vision*, 2007, pp. 28–33.

[17] R. Ferreira Cordeiro, C. Traina Junior, A. Machado Traina, J. López, U. Kang, and C. Faloutsos, "Clustering very large multi-dimensional datasets with mapreduce," in *Proceedings of the International Conference on Knowledge Discovery and Data mining*, 2011, pp. 690–698.

[18] A. McCallum, K. Nigam, and L. Ungar, "Efficient clustering of high-dimensional data sets with application to reference matching," in *Proceedings of International Conference on Knowledge Discovery and Data mining*, 2000, pp. 169–178.

[19] C. Ranger, R. Raghuraman, A. Penmetsa, G. Bradski, and C. Kozyrakis, "Evaluating mapreduce for multi-core and multiprocessor systems," in *IEEE Symposium on High Performance Computer Architecture*, 2007, pp. 13–24.

[20] S. Owen, R. Anil, T. Dunning, and E. Friedman, *Mahout in Action*. Manning Publications Co., 2011.

[21] U. Von Luxburg, "A tutorial on spectral clustering," *Statistics and Computing*, vol. 17, no. 4, pp. 395–416, 2007.

[22] D. MacDonald and C. Fyfe, "The kernel self-organising map," in *Proceedings of the International Conference on Knowledge-Based Intelligent Engineering Systems and Allied Technologies*, vol. 1, 2002, pp. 317–320.

[23] C. Williams and M. Seeger, "Using the nystrom method to speed up kernel machines," *Advances in Neural Information Processing Systems*, vol. 13, pp. 682–688, 2001.

[24] C. Fowlkes, S. Belongie, F. Chung, and J. Malik, "Spectral grouping using the nystrom method," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 26, no. 2, pp. 214–225, 2004.

[25] T. Sakai and A. Imiya, "Fast spectral clustering with random projection and sampling," *Machine Learning and Data Mining in Pattern Recognition*, pp. 372–384, 2009.

[26] W. Chen, Y. Song, H. Bai, C. Lin, and E. Chang, "Parallel spectral clustering in distributed systems," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 33, no. 3, pp. 568–586, 2011.

[27] R. Chitta, R. Jin, T. Havens, and A. Jain, "Approximate kernel k-means: Solution to large scale kernel clustering," in *Proceedings of the International Conference on Knowledge Discovery and Data mining*, 2011, pp. 895–903.

[28] P. Kar and H. Karnick, "Random feature maps for dot product kernels," in *Proceedings of the International Conference on Artificial Intelligence and Statistics*, 2012, pp. 583–591.

[29] W. Rudin, *Fourier Analysis on Groups*. Wiley-Interscience, 1990, vol. 12.

[30] M. Belabbas and P. J. Wolfe, "Spectral methods in machine learning and new strategies for very large data sets," *Proceedings of the National Academy of Sciences of the USA*, vol. 106, pp. 369–374, 2009.

[31] J. Doornik, "An improved ziggurat method to generate normal random samples," *University of Oxford*, 2005.

[32] A. Frieze, R. Kannan, and S. Vempala, "Fast monte-carlo algorithms for finding low-rank approximations," *Journal of the ACM*, vol. 51, no. 6, pp. 1025–1041, 2004.

[33] Y. LeCun, L. Bottou, Y. Bengio, and P. Haffner, "Gradient-based learning applied to document recognition," *Proceedings of the IEEE*, vol. 86, no. 11, pp. 2278–2324, 1998.

[34] J. Blackard and D. Dean, "Comparative accuracies of artificial neural networks and discriminant analysis in predicting forest cover types from cartographic variables," *Computers and Electronics in Agriculture*, vol. 24, no. 3, pp. 131–152, 1999.

[35] R. Cattral, F. Oppacher, and D. Deugo, "Evolutionary data mining with automatic rule generalization," *Recent Advances in Computers, Computing and Communications*, pp. 296–300, 2002.

[36] S. Stolfo, W. Fan, W. Lee, A. Prodromidis, and P. Chan, "Cost-based modeling for fraud and intrusion detection: Results from the jam project," in *Proceedings of the DARPA Information Survivability Conference and Exposition*, vol. 2, 2000, pp. 130–144.

[37] C. Alzate and J. Suykens, "Multiway spectral clustering with out-of-sample extensions through weighted kernel pca," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 32, no. 2, pp. 335–347, 2010.

[38] P. Perona and L. Zelnik-Manor, "Self-tuning spectral clustering," *Advances in Neural Information Processing Systems*, vol. 17, pp. 1601–1608, 2004.

[39] T. Kvalseth, "Entropy and correlation: Some comments," *IEEE Transactions on Systems, Man and Cybernetics*, vol. 17, no. 3, pp. 517–519, 1987.

[40] S. Smale and D. Zhou, "Learning theory estimates via integral operators and their approximations," *Constructive approximation*, vol. 26, no. 2, pp. 153–172, 2007.

[41] G. W. Stewart and J. Guang Sun, *Matrix Perturbation Theory*. Academic Press, 1990.