Bilateral $k$-Means Algorithm for Fast Co-Clustering

Junwei Han,† Kun Song,‡ Feiping Nie,† Xuelong Li§

†School of Automation, Northwestern Polytechnical University, Xi’an, 710072, Shaanxi, P. R. China
‡School of Computer Science and Center for OPTIMAL, Northwestern Polytechnical University, Xi’an, 710072, P. R. China
§Center for OPTIMAL, State Key Laboratory of Transient Optics and Photonics, Xi’an Institute of Optics and Precision Mechanics, Chinese Academy of Sciences, Xi’an 710119, Shaanxi, P. R. China.

{junweihan2010, songkun123000, feipingnie}@gmail.com, xuelong_li@ieee.org

Abstract

With the development of the information technology, the amount of data, e.g., text, image and video, has been increased rapidly. Efficiently clustering those large scale data sets is a challenge. To address this problem, this paper proposes a novel co-clustering method named bilateral $k$-means algorithm (BKM) for fast co-clustering. Different from traditional $k$-means algorithms, the proposed method has two indicator matrices $P$ and $Q$ and a diagonal matrix $S$ to be solved, which represent the cluster memberships of samples and features, and the co-cluster centres, respectively. Therefore, it could implement different clustering tasks on the samples and features simultaneously. We also introduce an effective approach to solve the proposed method, which involves less multiplication. The computational complexity is analyzed. Extensive experiments on various types of data sets are conducted. Compared with the state-of-the-art clustering methods, the proposed BKM not only has faster computational speed, but also achieves promising clustering results.

Introduction

Co-clustering, also referred as biclustering, seeks to cluster the set of samples and the set of features in a data matrix simultaneously. By doing permutations of rows and columns, the co-clustering algorithms aim to reorganize the initial data matrix into homogeneous blocks. These blocks also called co-clusters can therefore be defined as subsets of the data matrix characterized by a set of observations and a set of features whose elements are similar. The surveys of types of co-cluster and the related co-clustering algorithms are referred in literatures (Madeira and Oliveira 2004; Tanay, Sharan, and Shamir 2005; Charrad and Ahmed 2011). Since co-clustering algorithms utilize the relations between samples clusters and feature clusters, they make the data sets more predictable and the co-clustering performance more excellent compared with traditional one-side clustering.

Recently, several research topics involving co-clustering become hot issues due to the development of internet technology. Examples include text data mining, image retrieving, image segmentation, and video analysis (Zhang et al. 2004; Lu, Yuan, and Yan 2013; Lu, Wu, and Yuan 2014; Ailem, Role, and Nadif 2015; Liu and Lin 2015; Han et al. 2015; Cheng et al. 2015). In those tasks, the data sets are often represented by sparse matrices, and most of the related co-clustering methods (Ding, Li, and Jordan 2010; Ding et al. 2006) are based on checkerboard co-cluster model (as illustrate in Figure 1). In fact, the checkerboard co-cluster model is not quit suitable for those applications. The checkerboard co-cluster supposes every element in the data matrix should belong to one co-cluster, however, much of the entries of sparse matrix equals to zero which contribute none to the co-clustering. In real applications, the sparse data matrices are often mixed with noises which may change the values of elements equaling zero. It would reduce the co-clustering performance greatly. Therefore, according to the duality of the sample clustering and features clustering, using the diagonal co-cluster structure (as illustrate in Figure 1) instead of the checkerboard co-cluster model in those applications is a reasonable choice.

Among the diagonal co-cluster based algorithms, bipartite spectral graph partition (BSGP) approach (Inderjit 2001) is the most famous one due to its notable performance. Its variants have been applied in many fields (Zhang et al. 2004; Li, Wu, and Chang 2012; Trivedi et al. 2012). However, BSGP is computationally prohibitive for large data collections since it involves the singular value decomposition in the solution process, which severely limits the range of BSGP in the real world applications. In addition, BSGP is based on bipartitioning $N$cuts of a spectral graph. When dealing with multipartitioning problem, it needs to relax the origin discrete problem to a continuous problem firstly, then use $k$-means algorithm to output discrete result. In this discrete-continuous-discrete transformation procedure, the final resulting optimization problem is much deviate from the original objective problem of BSGP, and the performance of BSGP would be damaged.

In this paper, we propose a novel co-clustering method named bilateral $k$-means algorithm (BKM), by discovering the shortages of BSGP. Our proposed method is interesting from following perspectives:

1. Different from BSGP transforming the optimization problem of minimizing normalized cuts into a continuous relaxation, BKM relaxes the minimum normalized cuts problem to a special non-negative matrix decomposition (NMF) which involves constraints of indicator matrices. Such indi-
Bipartite spectral graph partitioning (BSGP) based co-clustering approach, constructs a bipartite graph $G = \{V, A\}$ between samples and features of data matrix $X$. Here, $V$ is the set of vertices corresponding to samples and features, and $A$ is the adjacency matrix constructed by data matrix $X$ as Eq. (1)

\[
A = \begin{bmatrix}
0 & X \\
X^T & 0
\end{bmatrix}
\] (1)

Using the bipartite graph model, the feature cluster $\Omega_i$ and sample cluster $\Phi_i$ are determined as follows.

\[
\Omega_i = \{x_i: \sum_{j\in\Theta_i} X_{ij} \geq \sum_{j\in\Phi_i} X_{ij}, \forall k = 1, \ldots, c\} \quad (2)
\]

\[
\Theta_i = \{x_j: \sum_{i\in\Phi_i} X_{ij} \geq \sum_{i\in\Omega_i} X_{ij}, \forall k = 1, \ldots, c\} \quad (3)
\]

It is easy to see that there is a recursive relationship between $\Omega_i$ and $\Theta_i$, and the relations described in Eq.(2) and Eq.(3) determine BSGP is based on diagonal co-cluster structure.

Then, BSGP aims to finding the minimal NCuts of $G$ by solving the optimization problem in Eq. (4)

\[
\min_{y} \frac{\frac{1}{2}y^T Ly}{y^T Dy}, \text{s.t.} y \in \{-1, 1\}^{(d+n)\times 1} \quad (4)
\]

where $D$ is the diagonal “degree” matrix with $D_{ii} = \sum_k A_{ik}, L = D - A, y = [p^T, q^T]^T, p \in \{1, -1\}^{d\times 1}$ stores the membership of features, and $q \in \{-1, 1\}^{n\times 1}$ stores the membership of samples. When $p_i = 1$, the $i$-th feature belongs to the first feature cluster, otherwise, it belongs to the second feature cluster. Similarly, when $q_i = 1$, the $i$-th sample belongs to the first sample cluster, otherwise, it belongs to the second sample cluster.

Since the problem in Eq. (4) is a NP-complete problem, it is relaxed into Eq. (5)

\[
\min_{q \neq 0} \frac{q^T L q}{q^T D q}, \text{s.t.} q^T D e = 0 \quad (5)
\]

where $e$ is a $(d+n) \times 1$ matrix with all elements equal to 1. Considering the structure of data matrix $A$, the problems in Eq. (5) can be solved by calculating singular vector corresponding to the second smallest singular value of matrix $D^{1/2}XD_2^{1/2}$, where $D_1 \in R^{d \times d}$ and $D_2 \in R^{n \times n}$ satisfy

\[
D = \begin{bmatrix}
D_1 & 0 \\
0 & D_2
\end{bmatrix} \quad (6)
\]

The procedure mentioned above is about bipartition clustering. When dealing with multipartitioning problem, suppose there are $c$ clusters in the data set, it needs to calculate $l = \log_2(c)$ left vectors $u_2, \cdots, u_{l+1}$ and $l$ right singular vectors $v_2, \cdots, v_{l+1}$, which is deemed to contain $c$-modal information about the data set. Thus we can form a $l$-dimensional data set

\[
Z = \begin{bmatrix}
D_1^{-1/2} U \\
D_2^{-1/2} V
\end{bmatrix} \quad (7)
\]

where $U = [u_2, \cdots, u_{l+1}], V = [v_2, \cdots, v_{l+1}]$. Lastly, the $k$-means algorithm is required to cluster the new data set $Z$, and output the clustering results, in which the first $d$ results of $Z$ correspond to the feature clustering results of $X$, and the last $n$ results correspond to sample clustering results.
Bilateral $K$-means algorithm

In this section, we replace the bipartitioning normalized cuts in BSGP by multipartitioning normalized cuts, and derive the formulation of the proposed method. Then, an efficient algorithm for solving the proposed optimization problem is introduced.

Formulation of bilateral $k$-means (BKM) algorithm

We now apply the multipartitioning Ncuts into BSGP. The optimization problem in Eq. (4) is transformed to

$$\min_P \sum_{k=1}^{c} \frac{y_k^T L y_k}{y_k^T D y_k}, \text{s.t. } P \in \Theta_{(d+n) \times c} \tag{8}$$

Since there is only one non-zero element in each row of $Y$, and $D$ is a diagonal matrix, the matrix $Y^T D Y$ is a diagonal matrix with $(k,k)$-element equal to $y_k^T D y_k$. Thus, optimization problem in Eq. (8) can be transformed to the matrix form as follows.

$$\min_P \text{Tr}(Y^T L Y (Y^T D Y)^{-1}), \text{s.t. } P \in \Theta_{(m+n) \times c} \tag{9}$$

where $\text{Tr}()$ is the matrix trace.

Substituting $L = D - A$ into the objective function of problem in Eq.(9), there is an equation as follows

$$\text{Tr}(Y^T L Y (Y^T D Y)^{-1}) = \text{Tr}(Y^T (D - A) Y (Y^T D Y)^{-1}) = \text{Tr}(I - 2 P^T X Q (Y^T D Y)^{-1})$$

where $I$ is an identity matrix.

As bipartite graph $G$ is constructed between samples and features, the indicator matrix $Y$ can be rewritten as $Y^T = [P^T, Q^T]$. $P$ contains the membership of features and $Q$ contains membership of samples. Substituting $A$ defined in Eq.(3) into Eq. (10), There is $\text{Tr}(Y^T L Y (Y^T D Y)^{-1}) = \text{Tr}(I - 2 P^T X Q (Y^T D Y)^{-1})$. Thus, the optimization problem in Eq. (9) is transformed into

$$\min_{P,Q} \text{Tr}(Y^T L Y (Y^T D Y)^{-1}) \tag{11}$$

The objective in Eq. (11) is a NP-complete problem (Yù and Shi 2003), so it should be relaxed into a easy solved problem. Different from the traditional way relax the discrete constrained problem of finding minimal Ncuts into that with continuous constraints, we relax the optimization problem in Eq. (11) into a matrix decomposition problem. By adding two terms $\text{Tr}((Y^T D Y)^{-1} P^T P (Y^T D Y)^{-1} Q^T Q)$ and $\text{Tr}(X^T X)$ into the objective function in Eq. (11), the optimization problem in Eq. (11) is changed to

$$\min_{P,Q} ||X - P (Y^T D Y)^{-1} Q^T||^2_P \tag{12}$$

The optimization problem in Eq. (12) involves the matrix inverse, which would increase the difficulty of finding the solution. We known $(Y^T D Y)^{-1}$ is a diagonal matrix. To simplify the problem, a diagonal matrix $S$ is used to replace $(Y^T D Y)^{-1}$, and $S$ is considered as a parameter to be solved. Therefore, we have the optimization problem of bilateral $k$-means algorithm as follows.

$$\min_{P,Q,S} ||X - P S Q^T||^2_P \tag{13}$$

where diag represents the set of diagonal matrices. In Eq. (13), $P$ and $Q$ are the indicator matrices storing the clustering results of features and samples, respectively. $S$ plays a role of connecting $P$ and $Q$. Then, we can transform $PSQ^T$ to a diagonal block matrix by doing some permutations on both columns and rows.

An efficient optimization algorithm

Before giving the solution of BKM, two useful propositions are presented.

**Definition 1.** Suppose $B \in R^{k \times k}$, its $(i, i)$-element is $b_{ii}$, a function $f(B)$ is defined as $f(B) = [b_{11}, \cdots, b_{kk}]^T = b$.

**Proposition 1.** If $C \in R^{k \times k}$ is a diagonal matrix, for an arbitrary matrix $B \in R^{k \times k}$, there exists $\text{Tr}(B C) = f(B)^T f(C)$.

**Proof:** Since $C$ is a diagonal matrix, $\text{Tr}(B C) = \sum_i b_{ii} c_{ii} = [b_{11}, \cdots, b_{kk}] [c_{11}, \cdots, c_{kk}]^T = f(B)^T f(C) \hfill \square$

**Proposition 2.** If matrices $B, C, D \in R^{k \times k}$ are diagonal matrices, there are $\text{Tr}(B C D) = \text{Tr}(B D C) = f(B)^T D f(C)$.

Since the proof is simple, we do not provide them here. Similar to the standard $k$-means algorithm, the proposed method is solved in an alternative way.

Firstly, with $P$ and $Q$ fixed, we solve $S$. Let us denote the objective function in Eq. (13) as $J_1$, and rewrite it into the sum of several matrix traces as follows.

$$J_1 = ||X - PSQ^T||^2_P$$

$$= \text{Tr}(X^T X) - 2\text{Tr}(Q^T X^T PS)$$

$$+ \text{Tr}(SP^T PSQ^T Q) \tag{14}$$

Since $P$ and $Q$ are diagonal matrices, $P^T P$ and $Q^T Q$ are diagonal matrices. According to Proposition 1, $\text{Tr}(SP^T PSQ^T Q) = \text{Tr}(SP^T PQ^T QS) = f(S)^T (P^T PQ^T Q) f(S)$. According to Proposition 2, $\text{Tr}(Q^T X^T PS) = f(S)^T f(P^T XQ)$.

We use $H$ to denote $P^T PQ^T Q$, $s$ to denote $f(S)$, and $r$ to denote $f(P^T XQ)$. Therefore, $J_1$ can be rewritten into

$$J_1 = \text{Tr}(X^T X) - 2 r^T s + s^T H s \tag{15}$$

Then the problem for solving $S$ is transformed to solve $s$. Let us calculate the partial derivative of $J_1$ with respect to $s$, and make it equal to 0, there is

$$\frac{\partial J_1}{\partial s} = 2(Hs - r) = 0 \tag{16}$$

since $H$ is a diagonal matrix, $H^{-1}$ is easily to be solved. So,

$$s = H^{-1} r \tag{17}$$
and \( S \) is solved.

Secondly, we calculate \( Q \). With \( P \) and \( S \) fixed, the optimization problem to solve \( Q \) can be decomposed into \( n \) simple subproblems for each \( i (1 \leq i \leq n) \)

\[
\begin{align*}
\min_{\Phi} & \left\| x_i - Rq_i^T \right\|^2_F \\
\text{s.t.} & \quad \Phi \in \Phi_{n \times c}
\end{align*}
\]

where \( R = PS \).

Because there is only one element equal to 1 and the rest are zeros in vector \( q_i \), the solution of Eq.(18) is determined by

\[
q_{ij} = \begin{cases} 
1 & j = \arg\min_k \left\| x_i - r_k \right\|^2 \\
0 & \text{otherwise}
\end{cases}
\]

where \( r_k \) is the \( k \)-th column of \( R \).

Finally, we calculate \( P \). With the \( Q \) and \( S \) fixed, the optimization problem for solving \( P \) is decomposed into \( m \) simple subproblems as in Eq. (20) for each \( j (1 \leq j \leq m) \).

\[
\begin{align*}
\min_{\Phi} & \left\| x_j - p_i^T L \right\|^2_F \\
\text{s.t.} & \quad \Phi \in \Phi_{d \times c}
\end{align*}
\]

where \( L = SQ^T \).

\[
p_{ij} = \begin{cases} 
1 & j = \arg\min_k \left\| x_j - l_k \right\|^2 \\
0 & \text{otherwise}
\end{cases}
\]

where \( l_k \) is the \( k \)-th row of \( L \).

The procedures of solving the model of BKM described in Eq. (13) are summarized in Algorithm 1.

**Algorithm 1 Bilateral \( k \)-means algorithm**

**Input**: Data matrix \( X = [x_1, \ldots, x_n] \in R^{d \times n} \).

**Initialize** \( P \) and \( Q \) with arbitrary class indicator matrices.

**Repeat**

1. Calculate \( S \) by Eq. (17);
2. Calculate \( P \) by Eq.(19);
3. Calculate \( Q \) by Eq.(21);

**Until Convergence**

**Output**: Indicator matrix \( P \) for feature clustering and \( Q \) for sample clustering.

---

**The computational complexity**

In this section, we compare the computational complexity of BKM with that of traditional \( k \)-means and BGSP.

As seen from Algorithm 1, the computational complexity of BKM consists of three parts. They are the computational complexities of solving \( S, P, Q \). Since operating multiplication requires more time than operating addition, we only consider the multiplications of proposed methods.

For solving \( S \), it involves \( c \) times multiplications, and \( ndct \) times multiplications in solving \( Q \). Here \( n \) and \( d \) represent the numbers of columns and rows of the data matrix respectively, \( c \) represents cluster number, and \( t \) represents iteration number. In summary, there are \( 2ndct + c \) times of multiplications while that of \( k \)-means algorithm is \( dct + c \). Thus, the computational complexity of BKM is \( O(ndct) \), which is the same with traditional \( k \)-means. In many real applications, \( dct \) is much less than \( n \), so the complexity of BKM and \( k \)-means is often referred as \( O(n) \).

BGSP involves a singular value decomposition (SVD), so its complexity is more than \( O(n^2d) \). When dealing with large scale data, the number of samples \( n \) is much larger than \( ct \). So the complexity of BSGP is much larger than the proposed model BKM. In summary, BKM is more suitable than BGSP for dealing with large scale data.

---

**Experiment results**

In this section, several experiments are developed to explore the performance of our proposed method. There are two aspects of experiments. The first is to compare the computational consumption of the proposed BKM with other clustering or co-clustering methods. The second is the comparison of clustering results.

We compare BKM with several state-of-the-art co-clustering methods, such as BSGP (Inderjit 2001), and Orthogonal NMTF(ONMTF) (Ding et al. 2006), Fast NMTF (FNMTF) (Cheng 2015), and multi-linear decomposition with sparse latent factors algorithm (MDSLF) (Papalexakis 2013). Among those methods, FNMTF is a fast co-clustering framework, and MDSLF is designed for sparse data matrix co-clustering. We also compare with the one-way clustering methods, i.e. \( k \)-means (MacQueen 1967), NMF (Lee 2001).

**Data description and experimental setting**

By following previous works, we adopt two types of data sets in our experiments, i.e. real world data sets and synthetic data sets.

The real world data sets consist of WebKB4 (Ding et al. 2006), WebACE (Cai, Wu, and Han 2008), CSTR (Gu and Zhou 2009) and RCV1 (Lewis, Rose, and Li 2004), which are summarized in Table 1. The first three data sets are widely used as benchmarks in clustering literatures. The last data set has very large samples size and feature size. We use it to evaluate the performance of the proposed method for dealing with large data set. For feasibility of calculation on our computer, the keywords (features) appearing less than 100 times in the corpus are removed, which results in 2979 (out of 47236) keywords in our experiments.

Since synthetic data matrix has the exact co-cluster structure, we use it to demonstrate the performance of the proposed method. In our paper, three \( 200 \times 600 \) block diagonal synthetic data matrices are generalized, and they are under different noise levels of \( \{0.05, 0.10, 0.15\} \) respectively. Each data matrix consists of \( c = 5 \) clusters, \( noise = N/(d \times n) \), where \( N \) is the number of non-zero elements in non-diagonal block area. Examples of synthetic data set are shown in Figure 2.

In the experiments, the number of column clusters is set equal to that of row clusters for all the co-clustering methods. Two parameters of MDSLF are set to equal empirically, and they are determined by method in (Papalexakis 2013). As the order presented above, for the four real data sets, the
parameters are \{80, 92, 43, 94\}, respectively. For the synthetic data set, they are \{37, 42, 51\}. For \(k\)-means and NFM, except the number of clusters, there is no parameter needed to be tuned.

<table>
<thead>
<tr>
<th>Data sets</th>
<th>#Sample</th>
<th>#Feature</th>
<th>#Classes</th>
</tr>
</thead>
<tbody>
<tr>
<td>WebKB4</td>
<td>1140</td>
<td>1644</td>
<td>4</td>
</tr>
<tr>
<td>WebACE</td>
<td>4199</td>
<td>1000</td>
<td>20</td>
</tr>
<tr>
<td>CSTR</td>
<td>2340</td>
<td>1000</td>
<td>4</td>
</tr>
<tr>
<td>RCV1</td>
<td>193844</td>
<td>2979</td>
<td>100</td>
</tr>
</tbody>
</table>

### Computational complexity

In this subsection, we explore the computational complexity of the comparison clustering methods. All the experiments run on the computer with Intel (R) Xeon(R) CPU E3-1225 V2, 3.2GHZ CPU and 16.0G memory.

Since most of comparison methods are solved alternately, the iteration number is closely related to the running speed. Thus, before testing the computational times, we examine iteration numbers of the compared methods when they reach the convergence. Every method takes 50 runs. The average results are reported in Table 2. As shown in Table 2, the methods involving discrete solution space (i.e. \(k\)-means, FNMF, BKM) have less iteration number than the methods with continuous solution space. This is because the searching scale of discrete solution space is smaller than that of continuous solution space. Among those methods with discrete solution space, the one-way clustering method \(k\)-means need more iterations than co-clustering methods FNMF and BKM. It is because co-clustering methods utilize the inter-relationship between samples cluster and feature cluster. The reason for BKM converging faster than FNMF is that diagonal-block co-cluster structure is better than checker-board structure in our tasks.

At last, we report the average convergence times of the comparison methods on four real data sets. Every method takes 50 runs, and the average results are presented in Figure 3. As shown in Figure 3, we can see that the BKM costs the least time to converge. Especially, BKM runs faster than \(k\)-means method which is well known as linear computational complexity as \(O(n)\). That is because BKM requires less iteration numbers than \(k\)-means. Thus the results are consistent with the analysis in previous section.

### Clustering results

We evaluate the proposed method on two types of data sets, i.e. real world data sets and synthesis data sets. Three measures widely adopted to evaluate the results of clustering in literatures are used, i.e. Accuracy (Cai, Wu, and Han 2008), NMI (Cai, Wu, and Han 2008) and Purity (Ding et al. 2006). In the experiments, each method takes 50 runs, and the average results are computed. In Table 3 and Table 4, the results of real world data sets and synthesis data sets are reported respectively. As seen from the Table 3, we can find that the proposed method BKM achieves the better results comparing with other methods. Meanwhile we also could observe that the co-clustering methods, including BKM, MDLSF, BSGP, ONMTF and FMNTF, outperform the traditional cluster methods. It is consistent with the widely accepted hypothesis that utilizing the duality of feature clustering and sample clustering can help clustering of data samples. As seen from Table 4, we can find that BKM achieves better results of column clustering and row clustering than other co-clustering approaches. Meanwhile, the clustering results of BKM on synthesis data sets are shown in Figure 2. It shows the excellent performance of BKM for clustering the columns and rows of data matrix simultaneously.

### Conclusions

In this paper, we have proposed a novel co-clustering method named bilateral \(k\)-means algorithm. We adopted the main idea of BSGP, and used multipartitioning normalized cuts to construct a new optimization problem for finding the minimal cuts. Different from BSGP, we relaxed the new optimization problem into a special non-negative matrix decomposition and yielded the model of BKM. The BKM is with constraints of indicator matrix, which makes its solution involve much less multiplication than BSGP and other state-of-the-art co-clustering or clustering methods. It has indicated that our method is good at dealing with real world large-scale data set. We also conducted extensive experiments to evaluate the computational complexity and clustering performance of the proposed method. Promising results have shown that BKM not only runs faster than other clustering methods, but also achieves a better performance.

### Acknowledgments

This work was supported in part by the National Science Foundation of China under Grant 61473231 and Grant 61522207.

### References


Figure 2: The figures in the first row are original data matrices with different noise rates. The figures in the second row are clustering results, and the blue lines in the diagonal blocks represent error clustered samples or features.

Figure 3: The average running times (s) on different real data sets. (a) WebKB4; (b) WebACE; (c) CSTR; (d) RCV1

Table 3: Clustering results of different methods measured by Accuracy/NMI/Purity on real world data sets.

<table>
<thead>
<tr>
<th>Data</th>
<th>Metrics</th>
<th>k-means</th>
<th>NMF</th>
<th>ONMTF</th>
<th>BSGP</th>
<th>MDLSF</th>
<th>FNMTF</th>
<th>BKM</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Accuracy</td>
<td>0.598</td>
<td>0.568</td>
<td>0.645</td>
<td>0.782</td>
<td>0.763</td>
<td>0.642</td>
<td>0.786</td>
</tr>
<tr>
<td>WebKB4</td>
<td>NMI</td>
<td>0.467</td>
<td>0.427</td>
<td>0.442</td>
<td>0.447</td>
<td>0.442</td>
<td>0.431</td>
<td>0.513</td>
</tr>
<tr>
<td></td>
<td>Purity</td>
<td>0.601</td>
<td>0.595</td>
<td>0.571</td>
<td>0.632</td>
<td>0.630</td>
<td>0.611</td>
<td>0.687</td>
</tr>
<tr>
<td></td>
<td>Accuracy</td>
<td>0.526</td>
<td>0.514</td>
<td>0.725</td>
<td>0.763</td>
<td>0.753</td>
<td>0.537</td>
<td>0.782</td>
</tr>
<tr>
<td>WebACE</td>
<td>NMI</td>
<td>0.519</td>
<td>0.512</td>
<td>0.528</td>
<td>0.521</td>
<td>0.532</td>
<td>0.532</td>
<td>0.536</td>
</tr>
<tr>
<td></td>
<td>Purity</td>
<td>0.479</td>
<td>0.481</td>
<td>0.405</td>
<td>0.512</td>
<td>0.521</td>
<td>0.513</td>
<td>0.611</td>
</tr>
<tr>
<td></td>
<td>Accuracy</td>
<td>0.763</td>
<td>0.759</td>
<td>0.796</td>
<td>0.849</td>
<td>0.852</td>
<td>0.781</td>
<td>0.876</td>
</tr>
<tr>
<td>CSRT</td>
<td>NMI</td>
<td>0.624</td>
<td>0.618</td>
<td>0.641</td>
<td>0.714</td>
<td>0.721</td>
<td>0.627</td>
<td>0.747</td>
</tr>
<tr>
<td></td>
<td>Purity</td>
<td>0.612</td>
<td>0.587</td>
<td>0.617</td>
<td>0.692</td>
<td>0.703</td>
<td>0.614</td>
<td>0.711</td>
</tr>
<tr>
<td></td>
<td>Accuracy</td>
<td>0.147</td>
<td>0.153</td>
<td>0.163</td>
<td>0.183</td>
<td>0.181</td>
<td>0.169</td>
<td>0.249</td>
</tr>
<tr>
<td>RCV1</td>
<td>NMI</td>
<td>0.262</td>
<td>0.261</td>
<td>0.264</td>
<td>0.321</td>
<td>0.316</td>
<td>0.263</td>
<td>0.328</td>
</tr>
<tr>
<td></td>
<td>Purity</td>
<td>0.126</td>
<td>0.123</td>
<td>0.153</td>
<td>0.159</td>
<td>0.153</td>
<td>0.149</td>
<td>0.162</td>
</tr>
</tbody>
</table>


Table 4: Clustering results of different methods measured by Accuracy/NMI/Purity on synthetic data sets.

<table>
<thead>
<tr>
<th>Data</th>
<th>Metrics</th>
<th>ONMTF</th>
<th>BSGP</th>
<th>MDSL</th>
<th>FNMTF</th>
<th>BKM</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>sample</td>
<td>feature</td>
<td>sample</td>
<td>feature</td>
<td>sample</td>
<td>feature</td>
</tr>
<tr>
<td>Noise=0.05</td>
<td>Accuracy</td>
<td>0.853</td>
<td>0.853</td>
<td>0.871</td>
<td>0.874</td>
<td>0.869</td>
</tr>
<tr>
<td></td>
<td>NMI</td>
<td>0.854</td>
<td>0.855</td>
<td>0.871</td>
<td>0.872</td>
<td>0.870</td>
</tr>
<tr>
<td></td>
<td>Purity</td>
<td>0.858</td>
<td>0.858</td>
<td>0.872</td>
<td>0.871</td>
<td>0.865</td>
</tr>
<tr>
<td>Noise=0.1</td>
<td>Accuracy</td>
<td>0.853</td>
<td>0.853</td>
<td>0.871</td>
<td>0.874</td>
<td>0.869</td>
</tr>
<tr>
<td></td>
<td>NMI</td>
<td>0.854</td>
<td>0.855</td>
<td>0.871</td>
<td>0.872</td>
<td>0.863</td>
</tr>
<tr>
<td></td>
<td>Purity</td>
<td>0.858</td>
<td>0.858</td>
<td>0.872</td>
<td>0.871</td>
<td>0.861</td>
</tr>
<tr>
<td>Noise=0.15</td>
<td>Accuracy</td>
<td>0.853</td>
<td>0.853</td>
<td>0.841</td>
<td>0.838</td>
<td>0.830</td>
</tr>
<tr>
<td></td>
<td>NMI</td>
<td>0.854</td>
<td>0.855</td>
<td>0.852</td>
<td>0.843</td>
<td>0.833</td>
</tr>
<tr>
<td></td>
<td>Purity</td>
<td>0.858</td>
<td>0.858</td>
<td>0.831</td>
<td>0.833</td>
<td>0.827</td>
</tr>
</tbody>
</table>

ACM.


