A heuristic-based fuzzy co-clustering algorithm for categorization of high-dimensional data

William-Chandra Tjhi*, Lihui Chen

Nanyang Technological University, School of Electrical and Electronic Engineering, Republic of Singapore

Received 10 October 2006; received in revised form 28 June 2007; accepted 1 October 2007
Available online 12 October 2007

Abstract

Fuzzy co-clustering is a technique that performs simultaneous fuzzy clustering of objects and features. It is known to be suitable for categorizing high-dimensional data, due to its dynamic dimensionality reduction mechanism achieved through simultaneous feature clustering. We introduce a new fuzzy co-clustering algorithm called Heuristic Fuzzy Co-clustering with the Ruspini’s condition (HFCR), which addresses several issues in some prominent existing fuzzy co-clustering algorithms. Among these issues are the performance on data sets with overlapping feature clusters and the unnatural representation of feature clusters. The key idea behind HFCR is the formulation of the dual-partitioning approach for fuzzy co-clustering, replacing the existing partitioning-ranking approach. HFCR adopts an efficient and practical heuristic method that can be shown to be more robust than our earlier effort for the dual-partitioning approach. We explain the proposed algorithm in details and provide an analytical study on its advantages. Experimental results on 10 large benchmark document data sets confirm the effectiveness of the new algorithm.

© 2007 Elsevier B.V. All rights reserved.

Keywords: Co-clustering; Fuzzy clustering; Information retrieval

1. Introduction

The growing number of on-line high-dimensional data repositories, such as collections of web pages in the Web, has induced the need for a sophisticated yet practical categorization technique. To answer to this challenge, various research communities have made many significant efforts, particularly in the area of data clustering. It has been a common practice in this field to model the data to be clustered by a co-occurrence matrix with rows representing objects (or data) and columns representing features (or attributes) [3,9]. Based on this model, a typical way of performing clustering is by categorizing the rows based on the distribution of the columns. An enhancement to this traditional approach would be to perform a simultaneous categorization of rows and columns. The latter approach is usually called co-clustering. It has been argued that performing an extra step of categorizing the columns enables a co-clustering algorithm to capture the inherent structure of data more accurately [9]. Furthermore, this column categorization scheme can simultaneously be beneficial when dealing with high-dimensional data as it implies a local dynamic dimensionality reduction process [3,4]. Another advantage of co-clustering is the generation of feature clusters, which can be utilized to describe the

* Corresponding author.
E-mail addresses: william_chandra@pmail.ntu.edu.sg (W.-C. Tjhi), elhchen@ntu.edu.sg (L. Chen).

0165-0114/$ - see front matter © 2007 Elsevier B.V. All rights reserved.
doi:10.1016/j.fss.2007.10.003
meanings of the resulting object clusters [8]. Hence, co-clustering generally can give more interpretable results than
the ones generated by its conventional clustering counterpart.

Fuzzy co-clustering is a variant of co-clustering that generates fuzzy co-clusters, i.e. fuzzy object and feature clusters.
Compared to its crisp counterpart, fuzzy co-clustering can produce more realistic results by allowing co-clusters to
overlap one another. In the literatures, there are two important fuzzy co-clustering algorithms that are relevant to
our work. They are Fuzzy Clustering for Categorical Multivariate Data (FCCM) [11] and Fuzzy Co-Clustering of
Documents and Keywords (Fuzzy CoDoK) [7]. Both algorithms share the same principle where co-clustering is seen
as a combination of object partitioning and feature ranking (or weighting). We refer to such an approach as the
partitioning-ranking approach. The works in [11, 7] have demonstrated the effectiveness of this existing approach.

Despite this, a couple of issues related to the formulations of FCCM and Fuzzy CoDoK remain unsolved. First, the
feature-ranking formulation adopted may adversely affect the algorithms’ performances in co-clustering data sets with
highly overlapping feature clusters. As we shall see, in such an overlapping environment, the resulting feature ranks can
contradict the natural grouping structure of the features. This in turn can affect the overall categorization accuracies of the
algorithms. The second issue is on the representation of feature clusters. In the partitioning-ranking approach, objects
and features are processed differently. Hence, care must be taken when interpreting their respective memberships.
Specifically in the case of features, the memberships’ constraint imposed often causes the feature membership values
to be unrealistically small when we have a large number of features in a data set. Hence, there is still a need for a
more natural method for the feature-clustering aspect of fuzzy co-clustering. On the more practical side, there are
at least two problems that limit the performances of the existing algorithms. First, due to its inherent computational
overflow problem, the FCCM algorithm only works in data sets that have small numbers of objects [7]. Fuzzy CoDoK
was proposed to address this problem. However, the solution offered leads to another practical problem, in which the
membership values can take on any real value without any upper or lower bound whatsoever. This can prevent the
algorithm from converging. In its current version, some extra external steps are required for Fuzzy CoDoK to function
properly.

To address the existing problems above, we propose a new heuristic-based fuzzy co-clustering algorithm called
Heuristic Fuzzy Co-clustering with the Ruspini’s condition (HFCR). Different from FCCM and Fuzzy CoDoK, which
are based on the partitioning-ranking approach, HFCR is based on the dual-partitioning approach. In the latter ap-
proach, the algorithm performs a simultaneous partitioning of objects and features. This can be achieved by imposing
the partitioning constraints on both object and feature memberships. The partitioning constraint is also known in the
literatures as the Ruspini’s condition [13] and it is well-known for being used as the membership constraint in the
Fuzzy C-means algorithm [1]. We will show that, when categorizing data sets with many overlapping feature clusters,
by partitioning, instead of ranking the features, we can get a more natural feature cluster membership. This, in turn,
can enhance the categorization ability of the algorithm. At the same time, by uniforming the constraints of both object
and feature memberships, we can resolve the existing issues on the representation of feature clusters. Furthermore, our
formulation is also able to simultaneously tackle the computational overflow and the membership-range problems that
occur in FCCM and Fuzzy CoDoK, respectively.

As an additional note, HFCR is an enhanced version of one existing dual-partitioning-based algorithm, called Fuzzy
Co-clustering with the Ruspini’s condition (FCR), that we proposed earlier [16]. While the two algorithms are related,
the heuristic formulation of HFCR allows it to be more robust to noises than FCR. Consequently HFCR can be more
applicable for real-world applications, in which noises are commonly present.

The remaining of this paper is organized as follows. In Section 2 the two related algorithms, FCCM and Fuzzy
CoDoK, are summarized. It is then followed by some discussions on the existing problems in these two algorithms.
We introduce HFCR in Section 3. This section also includes analytical discussions on the relationship between FCR
and HFCR, and on how the latter can be more robust than the former. Empirical studies including experiments on 10
large document data sets and two toy problems are provided in Section 4. Section 5 concludes the paper.

2. Background

2.1. Related works

There are a number of existing fuzzy co-clustering algorithms in the literatures, including FCCM [11], Fuzzy
CoDoK [7], Fuzzy Simultaneous KeyWord Identification and Clustering of text documents (FSKWIC) [5], and Fuzzy
Table 1
List of mathematical notations

<table>
<thead>
<tr>
<th>Notation</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>C, N, K</td>
<td>Numbers of (co-)clusters, objects, and features</td>
</tr>
<tr>
<td>u_{ci}, v_{cj}</td>
<td>Object and feature memberships</td>
</tr>
<tr>
<td>d_{ij}</td>
<td>Relatedness measure between an object and a feature, in our case d_{ij} \geq 0</td>
</tr>
<tr>
<td>T_u, T_v</td>
<td>Co-clustering user-defined membership parameters</td>
</tr>
<tr>
<td>m</td>
<td>Fuzzy C-means fuzzifier parameter</td>
</tr>
<tr>
<td>r</td>
<td>Number of iterations</td>
</tr>
<tr>
<td>\tau_{max}</td>
<td>Maximum number of iterations</td>
</tr>
<tr>
<td>\varepsilon</td>
<td>Convergence indicator parameter</td>
</tr>
<tr>
<td>\lambda_j, \gamma_c</td>
<td>Lagrange multipliers</td>
</tr>
</tbody>
</table>

Co-clustering with the Ruspini’s condition (FCR) [16]. As mentioned in Section 1, FCCM and Fuzzy CoDoK are two prominent fuzzy co-clustering algorithms based on the partitioning-ranking approach. FSKWIC can be regarded as a distance-based variant of FCCM and Fuzzy CoDoK [7]. FCR is a dual-partitioning-based fuzzy co-clustering algorithm that we developed earlier. It can be considered as the predecessor of the proposed HFCR. In this section, we focus our discussions on FCCM and Fuzzy CoDoK, because some of their underlying principles are essential in understanding the design of HFCR. We exclude the discussion on FSKWIC as its formulation is quite distant from that of HFCR. Section 3.3 discusses FCR and its insufficiency to handle real-world data sets, which simultaneously provides a justification on the need to formulate HFCR.

FCCM and Fuzzy CoDoK achieve fuzzy co-clustering through an iterative optimization process. In the case of FCCM, the objective function is as defined in Eq. (1). The maximization of this function is subject to the membership constraints given in Eqs. (2) and (3). The explanations on the mathematical notations used can be found in Table 1. The update membership rules in Eqs. (4) and (5) can then be derived by applying the Lagrange multiplier method. The FCCM algorithm proceeds by iteratively updating these two equations in an alternating fashion.

\[
J_{\text{FCCM}} = \sum_{c=1}^{C} \sum_{i=1}^{N} \sum_{j=1}^{K} u_{ci} v_{cj} d_{ij} - T_u \sum_{c=1}^{C} \sum_{i=1}^{N} u_{ci} \ln u_{ci} - T_v \sum_{c=1}^{C} \sum_{j=1}^{K} v_{cj} \ln v_{cj},
\]

\[
\sum_{c=1}^{C} u_{ci} = 1 \quad \text{for } i = 1, \ldots, N,
\]

\[
\sum_{j=1}^{K} v_{cj} = 1 \quad \text{for } c = 1, \ldots, C,
\]

\[
u_{ci} = \frac{\exp \left\{ \sum_{j=1}^{K} v_{cj} d_{ij} \right\}}{\sum_{j=1}^{C} \exp \left\{ \sum_{j=1}^{K} v_{fj} d_{ij} \right\}}, \tag{4}
\]

\[
v_{cj} = \frac{\exp \left\{ \sum_{i=1}^{N} u_{ci} d_{ij} \right\}}{\sum_{q=1}^{K} \exp \left\{ \sum_{i=1}^{N} u_{eq} d_{iq} \right\}}. \tag{5}
\]

The first term in Eq. (1) is called the degree of aggregation [11]. The maximization of this term is intended to make highly related objects and features (as indicated by high d_{ij} values) to be co-clustered together (i.e. assigned to
the same co-cluster). The motivation is the belief that a high quality co-cluster should be the one with a strong coherence bonding among its members (i.e. objects and features). The second and the third terms are the fuzzifier terms. Their purpose is similar to the parameter \( m \) in the Fuzzy C-means algorithm, i.e. to fuzzify the resulting co-clusters. The parameters \( T_u \) and \( T_v \) can be used to adjust the levels of fuzziness of the object and feature memberships, respectively.

The constraint in Eq. (2) conforms to the Ruspini’s condition [13]. Due to this constraint, the object memberships computed by FCCM reflect how the objects are partitioned across different co-clusters (i.e. similar to the memberships in Fuzzy C-means [1]). The constraint in Eq. (3), on the other hand, has a different orientation. By conforming to this constraint, the feature memberships computed by FCCM reflects the features’ ranks in the co-clusters. In this case, in every co-cluster, the membership of a feature is decided based on where its similarity to the co-cluster ranks in comparisons to all the other features. Features with higher similarities to the co-cluster will be assigned higher memberships (i.e higher ranks). Thus, based on its constraints, FCCM is essentially a fuzzy co-clustering algorithm that combines object partitioning with feature ranking (i.e. a partitioning-ranking algorithm).

Another related fuzzy co-clustering algorithm is Fuzzy CoDoK. The objective function of this algorithm can be found in Eq. (6). Similar to FCCM, the maximization of Eq. (6) in Fuzzy CoDoK is subject to the constraints in Eqs. (2) and (3). Because of this, Fuzzy CoDoK adopts the same partitioning-ranking approach discussed above. Eqs. (7) and (8) show the derived update membership rules of Fuzzy CoDoK:

\[
J_{FCODOK} = \sum_{c=1}^{C} \sum_{i=1}^{N} \sum_{j=1}^{K} u_{ci} v_{cj} d_{ij} - T_u \sum_{c=1}^{C} \sum_{i=1}^{N} u_{ci}^2 - T_v \sum_{c=1}^{C} \sum_{j=1}^{K} v_{cj}^2,
\]

\[
u_{ci} = \frac{1}{C} + \frac{1}{T_u} \left( \sum_{j=1}^{K} v_{cj} d_{ij} - \frac{1}{C} \sum_{f=1}^{C} \sum_{j=1}^{K} v_{fj} d_{ij} \right),
\]

\[
u_{cj} = \frac{1}{K} + \frac{1}{T_v} \left( \sum_{i=1}^{N} u_{ci} d_{ij} - \frac{1}{K} \sum_{q=1}^{K} \sum_{i=1}^{N} u_{qi} d_{iq} \right).
\]

2.2. The existing problems

In this section, we detail some existing problems found in FCCM and Fuzzy CoDoK.

2.2.1. The problem with overlapping feature clusters

Consider the problem of co-clustering the matrix \( D_1 \) depicted in Fig. 1. In \( D_1 \), rows represent objects and columns represent features. The three rectangles illustrate the ideal co-clusters of \( D_1 \), i.e. \( CC_1 \), \( CC_2 \), and \( CC_3 \). It can be seen that \( CC_1 \) overlaps with \( CC_2 \) at feature 3 (the indices of rows and columns start from 1) and that feature 3 is equally relevant to both co-clusters. For this reason, it should be natural for feature 3 to have an equal membership value to \( CC_1 \) and \( CC_2 \). It can also be seen that \( CC_2 \) and \( CC_3 \) overlap at feature 5. This time, however, we have feature 5 slightly more relevant to \( CC_3 \) than to \( CC_2 \). Therefore, the ideal would be for feature 5 to have a higher membership value to \( CC_3 \) than to \( CC_2 \). The following are the object (\( U \)) and feature (\( V \)) membership matrices generated by FCCM.
and Fuzzy CoDoK when applied on $D_1$, with rows correspond to co-clusters and columns correspond to objects and feature, respectively:

$$U_{FCCM} = U_{FCODOK} = \begin{bmatrix} 1 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 1 \end{bmatrix},$$

$$V_{FCCM} = \begin{bmatrix} 0.59 & 0.22 & 0.11 & 0.02 & 0.02 & 0.02 \\ 0.03 & 0.03 & 0.15 & 0.58 & 0.15 & 0.03 \\ 0.02 & 0.02 & 0.12 & 0.27 & 0.53 \end{bmatrix},$$

$$V_{FCODOK} = \begin{bmatrix} 0.52 & 0.31 & 0.17 & 0 & 0 & 0 \\ 0 & 0.23 & 0.54 & 0.23 & 0 & 0 \\ 0 & 0 & 0 & 0.18 & 0.34 & 0.48 \end{bmatrix}.$$
to represent the ranks of the feature in co-clusters. This illustrates one problem that may affect the representation of the feature clusters in FCCM and Fuzzy CoDoK.

Another problem with regard to the feature clusters representation in FCCM and Fuzzy CoDoK is that, as the number of features increases, the feature membership values become smaller. This happens due to the feature membership constraint in Eq. (3). It should be easy to picture (by observing Eq. (3)) that if the co-cluster $CC_1$ in the data set $D_2$ were to contain 1000 features instead of 3, the membership value of each of this feature would be equal to 0.001 instead of 0.33. While scaling up the resulting feature membership values may artificially solve this problem, a more natural method to perform feature clustering should provide a better solution.

2.2.3. The computational overflow problem

This practical problem occurs when applying FCCM on a data set with a large number of objects. The reason can be observed from the FCCM’s update membership equation in Eq. (5) given in Section 2. In this equation, both the numerator and the denominator are in the form of $\sum_{i=1}^{N} u_{ci} d_{ij}$. When dealing with large data sets, i.e. with a large value of $N$, this exponential function may be equal to a very large number. Such a very large number can cause a computational overflow once its value passes the maximum limit allowable by a computer [7]. Because of this problem, it is prohibitive to apply FCCM for data sets containing more than a few hundreds objects.

2.2.4. The membership-range problem

This problem occurs in Fuzzy CoDoK because its update membership equations in Eqs. (7) and (8) allows $u_{ci}$ and $v_{cj}$ to take on any real values between $-\infty$ and $\infty$ (i.e. these memberships are not confined in the more conventional range $0 \leq u_{ci}, v_{cj} \leq 1$). If left as it is, this problem can prevent the optimization in Fuzzy CoDoK from converging. For this reason, in [7] if any membership becomes negative, its value is clipped to 0 and the values of the remaining memberships are renormalized according to the constraint in Eq. (3). One problem with this workaround is that, the clipped memberships lose their meanings. Thus, by performing such a clipping and renormalization of memberships, Fuzzy CoDoK may fail to capture the actual grouping structure of the data set. This, in turn, could reduce the accuracies of the resulting co-clusters.

3. The proposed algorithm

In this section, we discuss our contribution. We first explain the dual-partitioning approach to fuzzy co-clustering. This approach provides the basis for the proposed HFCR. We then go through the detailed formulation of HFCR. Lastly, we explain why there is a need for HFCR, despite our earlier effort in [16] to devise FCR, a fuzzy co-clustering algorithm based on the dual-partitioning approach that can be regarded as the predecessor of HFCR.

3.1. The dual-partitioning approach

We now discuss the dual-partitioning approach, which is an alternative to the partitioning-ranking approach adopted by FCCM and Fuzzy CoDoK. In the dual-partitioning approach, both objects and features are partitioned. In order for this to happen, the feature membership constraint given in Eq. (9) should be incorporated. Similar to the object membership constraint in Eq. (2), the new feature membership constraint conforms to the Ruspini’s condition. Therefore, by imposing the constraints in Eqs. (2) and (9), we can have a dual partitioning of objects and features:

$$\sum_{c=1}^{C} v_{cj} = 1 \quad \text{for} \quad j = 1, \ldots, K. \quad (9)$$

By partitioning the features, the resulting feature memberships reflect how features are partitioned across different co-clusters. Consequently, the computation of the feature-partitioning memberships must take into account the features’ contributions in all the existing co-clusters. This is as opposed to the feature ranking scheme where a feature membership value, while dependent on the states of the other features, is totally independent on the states of the other co-clusters. Because overlap occurs between co-clusters, the feature-partitioning scheme, which takes into consideration the positions of features in different co-clusters, should be able to capture overlapping feature clusters better than its feature-ranking counterpart. To illustrate the effectiveness of the feature-partitioning scheme when dealing
with overlaps, we conduct a simple experiment by applying FCCM and Fuzzy CoDoK on $D_T^1$ in Fig. 3, which is the transposed version of the matrix $D_1$ in Fig. 1. This time the overlaps are at object (row) 3 and object (row) 5 instead of feature (column) 3 and feature (column) 5. By using this example, we can get a glimpse into how the algorithms would handle the matrix $D_1$ in Fig. 1, if they were to partition, instead of to rank the features.

The following shows the resulting object membership matrices with rows correspond to co-clusters and columns correspond to objects for $U$ (or features for $V$):

$$U_{\text{FCCM}} = \begin{bmatrix} 1 & 1 & 0.5 & 0 & 0 & 0 \\ 0 & 0 & 0.5 & 1 & 0.38 & 0 \\ 0 & 0 & 0 & 0.62 & 1 & 1 \end{bmatrix},$$

$$U_{\text{FCODOK}} = \begin{bmatrix} 1 & 1 & 0.5 & 0 & 0 & 0 \\ 0 & 0 & 0.5 & 1 & 0.43 & 0 \\ 0 & 0 & 0 & 0.57 & 1 & 1 \end{bmatrix}.$$

This illustration shows that by performing object partitioning, the two algorithms manage to represent all the object-to-co-cluster assignments correctly. This includes the assignments of objects 3 and 5 (or features 3 and 5 in the case of $D_1$), which are positioned at the overlapping regions. These results support our proposition that if both objects and features are partitioned, a fuzzy co-clustering algorithm should be more effective in dealing with data sets with overlapping feature clusters.

Another important point is that by partitioning the features, the representations of both object and feature clusters are uniformized because both object and feature memberships observe similar constraints (i.e. Eqs. (2) and (9)). Thus, instead of getting object clusters and feature ranks as in the case of the partitioning-ranking approach, the dual-partitioning approach enables us to get object clusters and feature clusters. For this reason, we argue that the dual-partitioning approach can provide a more natural way of representing feature clusters than the existing partitioning-ranking counterpart.

3.2. The HFCR algorithm

We now introduce HFCR, the proposed heuristic fuzzy co-clustering algorithm based on the dual-partitioning approach. We start from the most direct way to formulate the concept of dual partitioning, that is, by simply replacing the feature membership constraint from Eqs. (3) to (9) in the existing FCCM formulation. The aim of the algorithm is therefore to maximize the function $J_R$ in Eq. (10), subject to the constraints given in Eqs. (2) and (9). Note that $J_R$ is actually equal to $J_{\text{FCCM}}$ in Eq. (1). We distinguish the two notations to avoid any confusion between the discussions on HFCR and the discussion on FCCM. By applying the Lagrange multiplier method, we can come up with the update membership rules for $J_R$, i.e. the $u_{ci}$ update in Eq. (4) given in Section 2.1 and the $v_{cj}$ update in Eq. (11) given below:

$$J_R = \sum_{c=1}^{C} \sum_{i=1}^{N} \sum_{j=1}^{K} u_{ci} v_{cj} d_{ij} - T_u \sum_{c=1}^{C} \sum_{i=1}^{N} u_{ci} \ln u_{ci} - T_v \sum_{c=1}^{C} \sum_{j=1}^{K} v_{cj} \ln v_{cj},$$

(10)
We propose two auxiliary functions in Eqs. (12) and (13) with the same constraints as the directions of the membership updates in such a way that they will direct the optimization properly, free from the existing bias. We let

\[
v_{cij} = \frac{\exp \left\{ \sum_{i=1}^{N} u_{ci} d_{ij} \right\}}{\sum_{f=1}^{C} \exp \left\{ \sum_{i=1}^{N} u_{fi} d_{ij} \right\}}.
\]

Eqs. (4) and (11) obviously reflect the dual-partitioning nature of the memberships (i.e. both memberships conform to the Ruspini’s condition). Despite this, such a direct transformation from the partitioning-ranking to the dual-partitioning schemes causes a fundamental flaw that prevents an effective fuzzy co-clustering from being accomplished. Let us briefly elaborate this further. Let denote the component-wise inner product of two matrices. Also let denote the component-wise inner product of two matrices, i.e.

\[
\sum_{i=1}^{N} \sum_{f=1}^{C} u_{ci} v_{cj} = \begin{bmatrix} m_{11} & \cdots & m_{1K} \\ \vdots & \ddots & \vdots \\ m_{N1} & \cdots & m_{NK} \end{bmatrix} \begin{bmatrix} d_{11} & \cdots & d_{1K} \\ \vdots & \ddots & \vdots \\ d_{N1} & \cdots & d_{NK} \end{bmatrix} = \begin{bmatrix} \sum_{i=1}^{N} u_{ci} d_{ij} \cdot v_{cj} \\ \vdots \end{bmatrix}.
\]

From Eqs. (4) and (11), we have \(0 \leq \sum_{i=1}^{N} \sum_{j=1}^{K} m_{ij} \leq N K\). Based on the degree of aggregation’s expression above, this variation implies that the maximization of the degree of aggregation in this case is biased towards the construction of co-clusters with larger \(\sum_{i=1}^{N} \sum_{j=1}^{K} m_{ij}\) values (\(d_{ij} \geq 0\) in our case). In reality, however, it is not necessary for co-clusters to have large \(\sum_{i=1}^{N} \sum_{j=1}^{K} m_{ij}\) values in order to accurately capture the inherent grouping structure of a given data set. Therefore, this bias becomes a fundamental flaw in the formulation. It can be proven that the update memberships in Eqs. (4) and (11) provide directions that always point to the maximization \(J_R\) (see Appendix A), and therefore they should point to the intended co-clusters (i.e. maximizing the degree of aggregation leads to the formation of highly coherent co-clusters as discussed in Section 2.1). However, due to the bias, such directions may not be able to lead to the desired outcome [16,11]. At this point we note, by observing the membership constraints in Eqs. (2) and (3), that we have \(\sum_{i=1}^{N} \sum_{j=1}^{K} m_{ij}\) always equal to a constant (i.e. \(N\)) in FCCM and Fuzzy CoDoK. For this reason they are spared from the bias problem.

Let us now discuss the proposed heuristic scheme incorporated to achieve the dual-partitioning approach to fuzzy co-clustering, while simultaneously avoiding the pitfall discussed above. The key idea behind this scheme is to alter the directions of the membership updates in such a way that they will direct the optimization properly, free from the existing bias. We propose two auxiliary functions in Eqs. (12) and (13) with the same constraints as \(J_R\) given in Eqs. (2) and (9) to guide the directions:

\[
J_1 = \sum_{c=1}^{C} \sum_{i=1}^{N} \sum_{j=1}^{K} u_{ci} v_{cj} \ln u_{ci} - T_{u} \sum_{c=1}^{C} \sum_{i=1}^{N} \ln u_{ci} - T_{v} \sum_{c=1}^{C} \sum_{j=1}^{K} v_{cj} \ln v_{cj},
\]

\[
J_2 = \sum_{c=1}^{C} \sum_{i=1}^{N} \sum_{j=1}^{K} u_{ci} v_{cj} \ln u_{ci} - T_{u} \sum_{c=1}^{C} \sum_{i=1}^{N} \ln u_{ci} - T_{v} \sum_{c=1}^{C} \sum_{j=1}^{K} v_{cj} \ln v_{cj}.
\]

It should be quite obvious, from their structures, that both \(J_1\) and \(J_2\) share similar principles with \(J_R\), i.e. the maximization of these three functions are intended to make highly related objects and features (i.e. those with high \(d_{ij}\) values) in a data set to be co-clustered together, resulting in highly coherent co-clusters. The three, however, differ in how the normalization applied on their degree of aggregation terms. In \(J_R\), no normalization takes place in its degree of aggregation term. In \(J_1\), however, the degree of aggregation of every co-cluster \(c\) is normalized by \(\sum_{q=1}^{K} v_{cq}\), while in \(J_2\), it is normalized by \(\sum_{p=1}^{K} u_{cp}\). This normalization is essential in our formulation in order to avoid the bias in the maximization of the degree of aggregation. Again, let: denote the component-wise inner product of two matrices. Also let \((m_1)_{ij} = \sum_{c=1}^{C} u_{ci} \frac{v_{cj}}{\sum_{q=1}^{K} v_{cq}}\) and \((m_2)_{ij} = \sum_{c=1}^{C} u_{ci} \frac{v_{cj}}{\sum_{p=1}^{K} u_{cp}}\). Then the degree of aggregations of \(J_1\) and \(J_2\) can be expressed, respectively, as follows:

\[
\sum_{c=1}^{C} \sum_{i=1}^{N} \sum_{j=1}^{K} u_{ci} v_{cj} \frac{v_{cj}}{\sum_{q=1}^{K} v_{cq}}d_{ij} = \begin{bmatrix} (m_1)_{11} & \cdots & (m_1)_{1K} \\ \vdots & \ddots & \vdots \\ (m_1)_{N1} & \cdots & (m_1)_{NK} \end{bmatrix} \begin{bmatrix} d_{11} & \cdots & d_{1K} \\ \vdots & \ddots & \vdots \\ d_{N1} & \cdots & d_{NK} \end{bmatrix},
\]
Table 2
HFCR pseudo-code

<table>
<thead>
<tr>
<th>Procedure:</th>
</tr>
</thead>
<tbody>
<tr>
<td>1. Set parameters $T_u, T_v, C, \tau_{\text{max}},$ and $\epsilon$;</td>
</tr>
<tr>
<td>2. Set $\tau = 0$;</td>
</tr>
<tr>
<td>3. Randomly initialize $u_{ci}, 0 \leq u_{ci} \leq 1$;</td>
</tr>
<tr>
<td>4. REPEAT</td>
</tr>
<tr>
<td>5. Update $v_{cj}$ by Eq. (15);</td>
</tr>
<tr>
<td>6. Update $u_{ci}$ by Eq. (14);</td>
</tr>
<tr>
<td>7. Update $\tau = \tau + 1$;</td>
</tr>
<tr>
<td>8. UNTIL $\max_{c,i}</td>
</tr>
</tbody>
</table>

$$
\begin{align*}
\sum_{c=1}^{C} \sum_{i=1}^{N} \sum_{j=1}^{K} u_{ci} v_{cj} d_{ij} &= \begin{bmatrix} (m_2)_{11} & \cdots & (m_2)_{1K} \\ \vdots & \ddots & \vdots \\ (m_2)_{N1} & \cdots & (m_2)_{NK} \end{bmatrix} \begin{bmatrix} d_{11} & \cdots & d_{1K} \\ \vdots & \ddots & \vdots \\ d_{N1} & \cdots & d_{NK} \end{bmatrix}.
\end{align*}
$$

By observing the constraints in Eqs. (2) and (9), it should be obvious that \( \sum_{i=1}^{N} \sum_{j=1}^{K} (m_1)_{ij} = N \) and \( \sum_{i=1}^{N} \sum_{j=1}^{K} (m_2)_{ij} = K \). By eliminating the variations in the values of \( \sum_{i=1}^{N} \sum_{j=1}^{K} (m_1)_{ij} \) and \( \sum_{i=1}^{N} \sum_{j=1}^{K} (m_2)_{ij} \), just like in the case of FCCM and Fuzzy CoDoK, we remove the biases in $J_1$ and $J_2$, respectively.

$$
\begin{align*}
\frac{u_{ci}}{\sum_{j=1}^{K} \frac{u_{cj} d_{ij}}{T_v \sum_{j=1}^{K} v_{cj}}} &= \exp \left\{ \frac{\sum_{j=1}^{K} v_{cj} d_{ij}}{T_v \sum_{j=1}^{K} v_{cj}} \right\}, \\
\frac{v_{cj}}{\sum_{i=1}^{N} \frac{u_{ci} d_{ij}}{T_u \sum_{i=1}^{N} u_{ci}}} &= \exp \left\{ \frac{\sum_{i=1}^{N} u_{ci} d_{ij}}{T_u \sum_{i=1}^{N} u_{ci}} \right\}.
\end{align*}
$$

Table 2 shows the proposed heuristic procedure for fuzzy co-clustering based on the dual-partitioning approach. As exhibited in the table, the HFCR algorithm is essentially an iterative process of $u_{ci}$ and $v_{cj}$ updates. The update equations in Eqs. (14) and (15) are, respectively, derived from $\frac{\partial L_1}{\partial u_{ci}} = 0$ and $\frac{\partial L_2}{\partial v_{cj}} = 0$, where $L_1$ and $L_2$ are the Lagrangian functions of $J_1$ and $J_2$, respectively, taking into account the constraints in Eqs. (2) and (9). We would like to recapture several important points that motivate the adoption of this heuristic procedure for HFCR below.

Firstly, it can be proven that the Hessian matrix $\nabla^2 J_1(u)$ is negative definite if $u_{ci}$ is updated from Eq. (14) (see Appendix A). This indicates that for a given $v = [v_{11}, v_{12}, \ldots, v_{CN}]^T$, a $u_{ci}$ update in HFCR always points to a local maximum of $J_1$ in the region that satisfies the constraints in Eqs. (2) and (9), and thus the update always seeks to increase the $J_1$ function. As we mentioned earlier, from the function’s definition, an increase in $J_1$, similar to an increase in $J_2$, promotes the formations of highly coherent co-clusters by assigning highly related objects and features into the same co-clusters. In a similar fashion we can prove that for a given $u = [u_{11}, u_{12}, \ldots, u_{CN}]^T$, a $v_{cj}$ update in HFCR always points to a local maximum of $J_2$ in the region that satisfies the constraints in Eqs. (2) and (9), and thus the update always seeks to increase the $J_2$ function. Again from the definition of $J_2$, an increase in the function promotes the formations of highly coherent co-clusters just like in the case of $J_1$. Based on this understanding, we can state that the heuristic-natured updates in HFCR directs the algorithm to a proper aim of generating highly coherent co-clusters.

The second and perhaps the most important point is that, unlike the combination of Eqs. (4) and (11) in our earlier attempt above, the combination of Eqs. (14) and (15) in HFCR provide unbiased update directions, leading to the generation of the desired fuzzy co-clusters. This comes from the fact that Eqs. (14) and (15) are, respectively, derived
from $J_1$ and $J_2$, which we have proven earlier to be free from the unwanted bias in the maximization of the degree of aggregation discussed above. In fact by contrasting with Eqs. (4) and (11), the bias-free property of Eqs. (14) and (15) can be clearly observed from the update equations themselves. Consider Eq. (4). The fundamental component of this equation lies in the term \( \exp \left\{ \frac{\sum_{j=1}^{K} v_{c,j} d_{i,j}}{\sum_{j=1}^{P} v_{c,j}} \right\} \), which measures the similarity between an object $i$ and a co-cluster $c$. It can be seen, however, that, since in the dual-partitioning scheme we have \( \sum_{j=1}^{K} v_{c,j} \neq \text{constant} \), the similarity measure favors co-clusters with larger \( \sum_{j=1}^{K} v_{c,j} \) values (i.e. such co-clusters tend to be assigned larger membership values). In other words, it favors co-clusters that contain more features. Contrast this with the term \( \exp \left\{ \frac{\sum_{j=1}^{K} u_{c,j} d_{i,j}}{\sum_{j=1}^{P} u_{c,j}} \right\} \) in Eq. (14) of HFCR. Here we have the argument of the exponential term normalized by \( \sum_{j=1}^{K} v_{c,j} \), removing the bias like the one occurs in Eq. (4). A similar phenomenon can be observed in the $v_{c,j}$ update equations as well.

Thirdly, based on our discussions on the bias in the maximization of the degree of aggregation, an ideal way to achieve fuzzy co-clustering based on the dual-partitioning approach is by using the bias-free $J_1$ or $J_2$ (i.e. either one) as the objective function to derive both $u_{c,i}$ and $v_{c,j}$ update rules. However, such an approach does not directly yield in the closed-form expression of one of the updates, e.g. from the first necessary condition $\frac{\partial L_1}{\partial v_{c,j}} = 0$ one cannot come up with the closed-form expression of $v_{c,j}$. It is possible to solve the equation numerically. This, however, can potentially compromise the efficiency of the algorithm. For this reason and based on the understanding that both $J_1$ and $J_2$ seek similar objectives of constructing highly coherent co-clusters, we derive a more practical alternative heuristic approach that uses both functions simultaneously to derive the $u_{c,i}$ and $v_{c,j}$ update rules.

As shown in Table 2, we adopt two stopping criteria for HFCR. The first criterion is when changes in the memberships have become insignificantly small, indicating the state of convergence of the iterations. The second one is when the number of iterations has exceeded $\tau_{\text{max}} = 200$, which we find, based on our experiments, to be a practical stopping criterion in the case of prolonged convergence.

At this point, we can observe two practical advantages of HFCR compared to FCCM and Fuzzy CoDoK. First, we can see from Eqs. (14) and (15) that $0 \leq u_{c,i}, v_{c,j} \leq 1$, and hence, there is no need to clip and renormalize the memberships like in the case of Fuzzy CoDoK. The second thing is that HFCR does not suffer from the same computational overflow problem as FCCM. The reason is because all the arguments of the exponential functions in Eq. (15) are normalized according to the number objects in the co-clusters (i.e. normalized by $\sum_{p=1}^{P} u_{c,p}$). This ensures that the values of these arguments will never be too large regardless of the size of the data sets. In Section 4, we demonstrate how HFCR addresses the two remaining problems detailed in Section 2.2.

The time complexity of HFCR is $O(CNK\tau)$ (with $\tau$ denoting the number of iterations). We can see from Eq. (15) that the computations of the numerator and the denominator can be performed independently. For a given $c$ and $j$, the computation of the numerator of Eq. (15) requires $O(N\tau)$. There are $CK$ of such numerators. Therefore, the computations of all different numerators of Eq. (15) require $O(CNK)$. As for the denominator of Eq. (15), there are $K$ different denominators, and to compute each one of them requires $O(CN)$. Therefore, the computations of all different denominators of Eq. (15) require $O(CNK)$. So the time complexity of updating all feature memberships $v_{c,j}$ per iteration is $O(CNK)$. In the same manner, we can find that the time complexity of updating all object memberships per iteration is $O(CNK)$. Therefore, taking into account the number of iterations, the time complexity of HFCR becomes $O(CNK\tau)$.

### 3.3. FCR and its limitation

As mentioned in the beginning of this section, we have earlier introduced a fuzzy co-clustering algorithm, called FCR, based on the dual-partitioning approach [16]. Here we explain why, despite FCR, we need to formulate a new dual-partitioning-based fuzzy co-clustering algorithm in HFCR. We first provide a brief review of FCR, followed by an analytical discussion on the limitation of FCR in real-world applications, which is caused by the algorithm’s lack of robustness to noises.

In FCR, we try to maximize the objective function in Eq. (16), subject to the constraints in Eqs. (2) and (9). Note that $J_{\text{FCR}}$ here is not exactly of the same version as the one we presented in [16]. Fundamentally the two are the same, and one can verify that both versions result in the same update membership equations (derived using the Lagrange multiplier method). The reason we show this version of $J_{\text{FCR}}$ is that from its expression, we can observe the non-robustness of...
FCR to noises relatively more easily. As explained in [16], the purpose of the second term of $J_{\text{FCR}}$ is to prevent the disruptive bias towards larger $\sum_{i=1}^{N} \sum_{j=1}^{K} m_{ij}$ values (where $m_{ij} = \sum_{c=1}^{C} u_{ci} v_{cj}$) when maximizing the degree of aggregation. By applying the Lagrange multiplier method, we can derive the update membership equations of FCR in Eqs. (17) and (18). The algorithm proceeds by updating these two equations until $J_{\text{FCR}}$ converges:

$$J_{\text{FCR}} = \sum_{c=1}^{C} \sum_{i=1}^{N} \sum_{j=1}^{K} u_{ci} v_{cj} d_{ij} - \frac{\sum_{c=1}^{C} \sum_{i=1}^{N} \sum_{j=1}^{K} u_{ci} v_{cj} \sum_{i=1}^{N} \sum_{j=1}^{K} d_{ij}}{NK}$$

$$- Tu \sum_{c=1}^{C} \sum_{i=1}^{N} u_{ci} \ln u_{ci} - Tv \sum_{c=1}^{C} \sum_{j=1}^{K} v_{cj} \ln v_{cj},$$

$$u_{ci} = \frac{\sum_{j=1}^{K} v_{cj} d_{ij} - \sum_{p=1}^{N} \sum_{j=1}^{K} d_{pj} \sum_{j=1}^{K} v_{cj}}{Tu}$$

$$\sum_{f=1}^{C} \exp \left( \frac{\sum_{j=1}^{K} v_{fj} d_{ij} - \sum_{p=1}^{N} \sum_{j=1}^{K} d_{pj} \sum_{j=1}^{K} v_{fj}}{Tu} \right).$$

$$v_{cj} = \frac{\sum_{i=1}^{N} u_{ci} d_{ij} - \sum_{q=1}^{N} \sum_{i=1}^{K} d_{iq} \sum_{i=1}^{K} u_{ci}}{Tv}$$

$$\sum_{f=1}^{C} \exp \left( \frac{\sum_{i=1}^{N} u_{fi} d_{ij} - \sum_{q=1}^{N} \sum_{i=1}^{K} d_{iq} \sum_{i=1}^{K} u_{fi}}{Tv} \right).$$

We showed in [16] how FCR can perform co-clustering effectively on some data sets. However, as we will explain shortly, FCR cannot handle most real-world data sets well due to the algorithm’s formulation that is not robust to noises. To see this, let us represent the first two terms of $J_{\text{FCR}}$ as follows:

$$\sum_{c=1}^{C} \sum_{i=1}^{N} \sum_{j=1}^{K} u_{ci} v_{cj} d_{ij} - \frac{\sum_{c=1}^{C} \sum_{i=1}^{N} \sum_{j=1}^{K} u_{ci} v_{cj} \sum_{i=1}^{N} \sum_{j=1}^{K} d_{ij}}{NK}$$

$$= \left[ \begin{array}{ccc} m_{11} & \cdots & m_{1K} \\ \vdots & \ddots & \vdots \\ m_{NK} & \cdots & m_{NK} \end{array} \right] \cdot \left[ \begin{array}{ccc} d_{11} & \cdots & d_{1K} \\ \vdots & \ddots & \vdots \\ d_{NK} & \cdots & d_{NK} \end{array} \right] - \left[ \begin{array}{ccc} B & \cdots & B \\ \vdots & \ddots & \vdots \\ B & \cdots & B \end{array} \right]$$

$$= \left[ \begin{array}{ccc} m_{11} & \cdots & m_{1K} \\ \vdots & \ddots & \vdots \\ m_{NK} & \cdots & m_{NK} \end{array} \right] \cdot \left[ \begin{array}{ccc} (d_{11} - B) & \cdots & (d_{1K} - B) \\ \vdots & \ddots & \vdots \\ (d_{NK} - B) & \cdots & (d_{NK} - B) \end{array} \right],$$

where $:$ denotes the component-wise inner product of two matrices (i.e. $X : Y = \sum_{i} \sum_{j} X_{ij} Y_{ij}$), $m_{ij} = \sum_{c=1}^{C} u_{ci} v_{cj}$, and $B = \frac{\sum_{p=1}^{N} \sum_{q=1}^{K} d_{pq}}{NK}$, i.e. the mean of the object-feature matrix. In this perspective, we can say that the purpose of the second term of $J_{\text{FCR}}$ in Eq. (16) is to classify whether or not an object is considered related to a feature. An object $x$ is considered related to a feature $y$ if $d_{xy} > B$. In this case, a positive value in the corresponding $m_{xy}$ will
increase the objective function (see the expression above), bringing it closer to its local maximum. On the other hand, the same object and feature are considered not related if \( d_{xy} < B \). Since we have \( 0 \leq m_{ij} \leq 1 \) for \( i = 1, \ldots, N \) and \( j = 1, \ldots, K \), the algorithm will then try to assign a zero value to the corresponding \( m_{xy} \) (i.e. \( m_{xy} = 0 \)) to prevent any decrease in the objective function. In FCR, the value of \( m_{ij} \), from its definition \( m_{ij} = \sum_{c=1}^{C} u_{ci} v_{cj} \), determines the membership distributions and consequently, the co-clusters that are generated by the algorithm. We can see from this perspective, however, that the process of forming co-clusters in FCR depends very much on the value of \( B \). While in an ideal case \( B \) alone can be a good option to act as a central point separating related object-feature pairs (high \( d_{xy} \)) from the non-related ones (low \( d_{xy} \)), in reality, due to the presence of noises in a data set, high-dependency on \( B \) like what we have in FCR may become prohibitive. It is a well-known fact that a mean value is not robust in the presence of noises \[12\]. Because of this, FCR may not perform effectively when applied on real-world data sets, which usually contain significant numbers of noises. This point is demonstrated later in our experiments in Section 4.2.2.

It is because of this limitation in FCR that a new fuzzy co-clustering algorithm based on the dual-partitioning approach needs to be formulated. Unlike FCR, HFCR, as can be observed from our discussion in Section 3.2, does not rely on the mean of the object-feature matrix \( B \) to form co-clusters. Thus, HFCR is more robust to noises than its predecessor. Our experimental results in Table 8 confirm this claim.

4. Experimental results

4.1. Toy problems

In this section, we discuss the performances of HFCR when being applied on two toy problems \( D_3 \) and \( D_4 \) (rows represent objects and columns represent features) in Table 3. The table also shows the expected intuitive outcomes for the given toy problems in the forms of the ideal membership matrices (rows represent co-clusters and columns represent objects/features). Table 4 shows the membership matrices generated by HFCR, FCCM, and Fuzzy CoDoK. In this table, some membership matrices are transposed to save some space. By comparing the HFCR’s membership matrices in Table 4 and the ideal ones in Table 3, we can see that HFCR manages to generate the expected object and feature memberships for both data sets. In the case of FCCM and Fuzzy CoDoK, however, the resulting feature membership matrices \( V \) do not accurately represent the ideal structure of the feature clusters. Particularly for \( D_3 \), which contains overlapping feature clusters (at features 2, 3, 5, and 6), these misrepresentations would lead to the generation of totally different feature clusters from the ideal ones (e.g. if we defuzzify the \( V \) matrices based on the maximum membership values, we will get different clusters’ assignments from the ideal case). We can also observe especially in the toy problem \( D_4 \) that FCCM and Fuzzy CoDoK produced smaller feature membership values than the ideal values.

As discussed in Section 2.2.2, this is caused by the constraint in Eq. (3) imposed on the feature memberships. If the number of features is very large (such as in the case our experiments reported in Section 4.2), the membership values would become unnaturally smaller. From these observations, we argue that HFCR, due to its dual-partitioning nature, should be able to give a more natural representation of the feature clusters than the existing partitioning-ranking-based algorithms.

<table>
<thead>
<tr>
<th>Data set</th>
<th>Ideal result</th>
</tr>
</thead>
</table>
| \( D_3 \) | \[
\begin{bmatrix}
1 & 0.7 & 0.5 & 0 & 0 & 0 & 0 \\
1 & 0.7 & 0.5 & 0 & 0 & 0 & 0 \\
0 & 0.3 & 0.5 & 0.8 & 0.5 & 0.3 & 0 \\
0 & 0.3 & 0.5 & 0.8 & 0.5 & 0.3 & 0 \\
0 & 0 & 0 & 0 & 0.55 & 1 & 1.2 \\
0 & 0 & 0 & 0 & 0.55 & 1 & 1.2 \\
\end{bmatrix}
\] |
| \( D_4 \) | \[
\begin{bmatrix}
1 & 1 & 1 & 1 & 0 & 0 & 0 \\
1 & 1 & 1 & 1 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 1 & 1 & 1 \\
0 & 0 & 0 & 0 & 1 & 1 & 1 \\
\end{bmatrix}
\] |

Table 3
Toy problems

<table>
<thead>
<tr>
<th>Data set</th>
<th>Ideal result</th>
</tr>
</thead>
</table>
| \( D_3 \) | \[
U = \begin{bmatrix}
1 & 1 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 1 & 1 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 1 & 1 \\
\end{bmatrix}
\] |
| \( D_4 \) | \[
U = \begin{bmatrix}
1 & 1 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 1 & 1 & 1 \\
\end{bmatrix}
\] |

| \( B \) | \[
V = \begin{bmatrix}
1 & a_1 & 0.5 & 0 & 0 & 0 & 0 \\
0 & a_2 & 0.5 & 1 & b_2 & e_2 & 1 \\
0 & 0 & 0 & 0 & b_3 & e_3 & 1 \\
0.5 \leq a_1, b_2, e_3 \leq 1 \\
e_2 < 0.5 \\
\end{bmatrix}
\] |
Table 4
Toy problems’ results

<table>
<thead>
<tr>
<th>Data set</th>
<th>HFCR result</th>
<th>FCCM result</th>
<th>FCoDoK result</th>
</tr>
</thead>
<tbody>
<tr>
<td>$D_3$</td>
<td>$U = \begin{bmatrix} 1 &amp; 1 &amp; 0 &amp; 0 &amp; 0 &amp; 0 \ 0 &amp; 0 &amp; 1 &amp; 1 &amp; 0 &amp; 0 \ 0 &amp; 0 &amp; 0 &amp; 0 &amp; 1 &amp; 1 \end{bmatrix}$</td>
<td>$U = \begin{bmatrix} 1 &amp; 1 &amp; 0 &amp; 0 &amp; 0 &amp; 0 \ 0 &amp; 0 &amp; 1 &amp; 1 &amp; 0 &amp; 0 \ 0 &amp; 0 &amp; 0 &amp; 0 &amp; 1 &amp; 1 \end{bmatrix}$</td>
<td>$U = \begin{bmatrix} 1 &amp; 1 &amp; 0 &amp; 0 &amp; 0 &amp; 0 \ 0 &amp; 0 &amp; 1 &amp; 1 &amp; 0 &amp; 0 \ 0 &amp; 0 &amp; 0 &amp; 0 &amp; 1 &amp; 1 \end{bmatrix}$</td>
</tr>
<tr>
<td>$V^T$</td>
<td>$\begin{bmatrix} 0.75 \ 0.17 \ 0.06 \end{bmatrix}$</td>
<td>$\begin{bmatrix} 0.06 \ 0.14 \end{bmatrix}$</td>
<td>$\begin{bmatrix} 0.45 \ 0.32 \end{bmatrix}$</td>
</tr>
</tbody>
</table>

| $D_4$    | $U = \begin{bmatrix} 1 & 1 & 0 & 0 \\ 0 & 0 & 1 & 1 \end{bmatrix}$ | $U = \begin{bmatrix} 1 & 1 & 0 & 0 \\ 0 & 0 & 1 & 1 \end{bmatrix}$ | $U = \begin{bmatrix} 1 & 1 & 0 & 0 \\ 0 & 0 & 1 & 1 \end{bmatrix}$ |
| $V^T$    | $\begin{bmatrix} 0.25 \\ 0.25 \\ 0.25 \end{bmatrix}$ | $\begin{bmatrix} 0.5 \\ 0.5 \end{bmatrix}$ | $\begin{bmatrix} 0.25 \end{bmatrix}$ |

Table 5
List of document data sets

<table>
<thead>
<tr>
<th>Data set</th>
<th>No. of docs.</th>
<th>No. of words</th>
<th>Clusters (No. of docs.)</th>
</tr>
</thead>
<tbody>
<tr>
<td>$Classic3$</td>
<td>3891</td>
<td>2176</td>
<td>Medical (1033), Aerospace (1398), Inform. Retrieval (1460)</td>
</tr>
<tr>
<td>$Multi5$</td>
<td>500</td>
<td>2889</td>
<td>Computer Graphics (100), Motorcycle (100), Baseball (100), Space (100), Middle-east (100)</td>
</tr>
<tr>
<td>$Binary$</td>
<td>500</td>
<td>3377</td>
<td>Politics (250), Middle-east (250)</td>
</tr>
<tr>
<td>$SM$</td>
<td>2000</td>
<td>5450</td>
<td>Soccer (1000), Motorsport (1000)</td>
</tr>
<tr>
<td>$SS$</td>
<td>2000</td>
<td>6337</td>
<td>Soccer (1000), Sport (1000)</td>
</tr>
<tr>
<td>$ABJ$</td>
<td>3000</td>
<td>7688</td>
<td>Astronomy (1000), Biology (1000), Java (1000)</td>
</tr>
<tr>
<td>$Yahoo_K1$</td>
<td>2340</td>
<td>3640</td>
<td>Health (494), Entertainment (1389), Sports (141), Politics (114), Technology (60), Business (142)</td>
</tr>
<tr>
<td>$CJB$</td>
<td>2250</td>
<td>6781</td>
<td>Commercial Bank (1000), Java (750), Biology (500)</td>
</tr>
<tr>
<td>$CAB$</td>
<td>2250</td>
<td>6980</td>
<td>Commercial Bank (1000), Astronomy (750), Biology (500)</td>
</tr>
</tbody>
</table>

4.2. Large document datasets

To demonstrate the usefulness of HFCR in categorizing real-world data, we performed experiments on 10 benchmark document data sets. In these experiments, we compare the performances of several algorithms on various techniques: the dual-partitioning with the partitioning-ranking approaches, fuzzy co-clustering with fuzzy clustering, and HFCR with FCR. For the dual-partitioning vs. the partitioning-ranking, we compared the performances of HFCR with Fuzzy CoDoK. For fuzzy co-clustering vs. fuzzy clustering, we compared the performances of HFCR with HFCM, which is a variant of Fuzzy C-means for high-dimensional data.

4.2.1. Data sets and implementation details

Since the experiments were conducted on document data sets, it is important to highlight that in the case of document-word co-clustering, documents are treated as objects and words are treated as features. The 10 benchmark document data sets are summarized in Table 5. The data sets $Classic3$ and $Yahoo_K1$ can be downloaded from ftp://ftp.cs.cornell.edu/pub/smart and ftp://ftp.cs.umn.edu/dept/users/boley/pddpdata/doc-K respectively. The $Binary$ and $Multi5$ data sets are the subsets of the 20News group data set available in http://people.csail.mit.edu/jrennie/20Newsgroups. The remaining data sets are the subsets of the large web document collections found in [15]. By
<table>
<thead>
<tr>
<th>Data set</th>
<th>HFCR Precision</th>
<th>HFCR Recall</th>
<th>HFCR Purity</th>
<th>Fuzzy CoDoK Precision</th>
<th>Fuzzy CoDoK Recall</th>
<th>Fuzzy CoDoK Purity</th>
</tr>
</thead>
<tbody>
<tr>
<td>BP</td>
<td>96.5 (1.4)</td>
<td>96.4 (1.5)</td>
<td>96.4 (1.5)</td>
<td>95.5 (1.6)</td>
<td>95.4 (1.6)</td>
<td>95.4 (1.6)</td>
</tr>
<tr>
<td>Classic3</td>
<td>98.6 (0.1)</td>
<td>98.5 (0)</td>
<td>98.6 (0)</td>
<td>98.6 (0)</td>
<td>98.3 (0)</td>
<td>98.5 (0)</td>
</tr>
<tr>
<td>Multi5</td>
<td>96.8 (0.2)</td>
<td>96.2 (0.2)</td>
<td>96.2 (0.2)</td>
<td>94.8 (7.6)</td>
<td>93.6 (9.7)</td>
<td>93.8 (9.3)</td>
</tr>
<tr>
<td>Binary</td>
<td>81.2 (16.1)</td>
<td>81.2 (16.1)</td>
<td>81.2 (16.1)</td>
<td>73.9 (14.1)</td>
<td>72.8 (14.3)</td>
<td>72.8 (14.3)</td>
</tr>
<tr>
<td>SM</td>
<td>81.6 (7)</td>
<td>81.2 (6.8)</td>
<td>81.2 (6.8)</td>
<td>68.4 (9.4)</td>
<td>67.3 (8.9)</td>
<td>67.3 (8.9)</td>
</tr>
<tr>
<td>SS</td>
<td>88.9 (0.1)</td>
<td>88.8 (0.1)</td>
<td>88.8 (0.1)</td>
<td>74.8 (11)</td>
<td>74.2 (10.8)</td>
<td>74.2 (10.8)</td>
</tr>
<tr>
<td>ABJ</td>
<td>85.2 (0)</td>
<td>84.8 (0)</td>
<td>84.8 (0)</td>
<td>81.4 (2.7)</td>
<td>76.5 (2.6)</td>
<td>76.5 (2.6)</td>
</tr>
<tr>
<td>YahooK1</td>
<td>54.6 (1.4)</td>
<td>69.6 (1.9)</td>
<td>83.3 (1.5)</td>
<td>52.1 (5.2)</td>
<td>64.8 (7.8)</td>
<td>83.6 (2.7)</td>
</tr>
<tr>
<td>CJB</td>
<td>88 (6.8)</td>
<td>88.7 (6.6)</td>
<td>88.9 (6.1)</td>
<td>86.1 (4)</td>
<td>85.7 (2.6)</td>
<td>86.4 (3.3)</td>
</tr>
<tr>
<td>CAB</td>
<td>78.2 (10.2)</td>
<td>78.2 (10.6)</td>
<td>81.8 (6.3)</td>
<td>73.4 (8.9)</td>
<td>70.8 (10.6)</td>
<td>75.9 (5.8)</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Data set</th>
<th>HFCR Precision</th>
<th>HFCR Recall</th>
<th>HFCR Purity</th>
<th>HFCM Precision</th>
<th>HFCM Recall</th>
<th>HFCM Purity</th>
</tr>
</thead>
<tbody>
<tr>
<td>BP</td>
<td>96.5 (1.4)</td>
<td>96.4 (1.5)</td>
<td>96.4 (1.5)</td>
<td>97.6 (0)</td>
<td>97.5 (0)</td>
<td>97.5 (0)</td>
</tr>
<tr>
<td>Classic3</td>
<td>98.6 (0.1)</td>
<td>98.5 (0)</td>
<td>98.6 (0)</td>
<td>99 (0)</td>
<td>98.8 (0)</td>
<td>98.9 (0)</td>
</tr>
<tr>
<td>Multi5</td>
<td>96.8 (0.2)</td>
<td>96.2 (0.2)</td>
<td>96.2 (0.2)</td>
<td>86.4 (12)</td>
<td>85.7 (11.6)</td>
<td>88 (8.4)</td>
</tr>
<tr>
<td>Binary</td>
<td>81.2 (16.1)</td>
<td>81.2 (16.1)</td>
<td>81.2 (16.1)</td>
<td>77.6 (5)</td>
<td>70 (7.5)</td>
<td>70 (7.5)</td>
</tr>
<tr>
<td>SM</td>
<td>81.6 (7)</td>
<td>81.2 (6.8)</td>
<td>81.2 (6.8)</td>
<td>66.2 (10.5)</td>
<td>66.1 (10.5)</td>
<td>66.1 (10.5)</td>
</tr>
<tr>
<td>SS</td>
<td>88.9 (0.1)</td>
<td>88.8 (0.1)</td>
<td>88.8 (0.1)</td>
<td>75.8 (11.2)</td>
<td>75.4 (11.4)</td>
<td>75.4 (11.4)</td>
</tr>
<tr>
<td>ABJ</td>
<td>85.2 (0)</td>
<td>84.8 (0)</td>
<td>84.8 (0)</td>
<td>59.3 (1.5)</td>
<td>57.7 (1.6)</td>
<td>60.8 (0.7)</td>
</tr>
<tr>
<td>YahooK1</td>
<td>54.6 (1.4)</td>
<td>69.6 (1.9)</td>
<td>83.3 (1.5)</td>
<td>53.4 (7)</td>
<td>65.1 (11.2)</td>
<td>82.6 (3.2)</td>
</tr>
<tr>
<td>CJB</td>
<td>88 (6.8)</td>
<td>88.7 (6.6)</td>
<td>88.9 (6.1)</td>
<td>74.2 (13.8)</td>
<td>75.1 (14.9)</td>
<td>80 (8.9)</td>
</tr>
<tr>
<td>CAB</td>
<td>78.2 (10.2)</td>
<td>78.2 (10.6)</td>
<td>81.8 (6.3)</td>
<td>62.2 (6.8)</td>
<td>62.8 (6.2)</td>
<td>73.8 (2)</td>
</tr>
</tbody>
</table>

Observing the data sets’ categories in Table 5, we can see variations of the data sets’ complexity natures due to, among other things, more overlapping categories and more unbalanced size categories. Through these variations, we can observe how the algorithms perform in different data sets’ conditions.

The documents were pre-processed using the Matrix Creation (MC) tool [2]. For all data sets, words occurring in less than 0.5% and more than 99.5% of the number of documents were removed. Stopwords (e.g., “a”, “to”, “of”) were removed and the experiments were performed without any stemming. In all the experiments, we used the normalized tf-idf [14] as the measure of document-word relatedness $d_{ij}$.

As mentioned above, in the experiments, we tested four algorithms: HFCR, Fuzzy CoDoK, HFCM, and FCR. The four algorithms were randomly initialized throughout the experiments. Since different initializations may result in different outcomes in all these four algorithms, for each data set, every algorithm was tested in 30 trials (or runs). The results reported in Tables 6–8 are averages of the 30-trial simulations.

The parameter $\gamma_{\max}$ was set to 200, limiting the number of iterations in both algorithms to 200. The number of clusters $C$ was set according to the number of the ground-truth categories of documents given in each data set, as indicated in Table 5. The convergence indicator parameter $\varepsilon$ was set to $10^{-5}$ throughout the experiments. The degree of fuzziness parameters $T_u$ and $T_v$ were set based on the trial and error approach. For HFCR, we used $T_u = T_v = 0.001$ on all data sets except Binary and Multi5, where we used $T_u = 0.001$ and $T_v = 0.005$. For Fuzzy CoDoK, we set $T_u = 0.00001$ and $T_v = 1.5$ on data sets with less overlap, while on data sets with more overlapping categories, we used $T_u = 0.001$. For HFCM, we set $m = 1.02$ in all the experiments. For FCR, we set $T_u = T_v = 1.0$ on all the data sets it was tested.
Table 8
Performance comparisons HFCR and FCR

<table>
<thead>
<tr>
<th>Data set</th>
<th>HFCR Precision</th>
<th>HFCR Recall</th>
<th>HFCR Purity</th>
<th>FCR Precision</th>
<th>FCR Recall</th>
<th>FCR Purity</th>
</tr>
</thead>
<tbody>
<tr>
<td>BP</td>
<td>96.5 (1.4)</td>
<td>96.4 (1.5)</td>
<td>96.4 (1.5)</td>
<td>82.4 (9.3)</td>
<td>67.4 (21.4)</td>
<td>67.4 (21.4)</td>
</tr>
<tr>
<td>Multi5</td>
<td>96.8 (0.2)</td>
<td>96.2 (0.2)</td>
<td>96.2 (0.2)</td>
<td>50.8 (6.6)</td>
<td>48.9 (6.9)</td>
<td>51 (6.7)</td>
</tr>
<tr>
<td>Binary</td>
<td>81.2 (16.1)</td>
<td>81 (16.1)</td>
<td>81 (16.1)</td>
<td>61.4 (7.8)</td>
<td>55.2 (6)</td>
<td>55.2 (6)</td>
</tr>
<tr>
<td>SM</td>
<td>81.6 (7)</td>
<td>81.2 (6.8)</td>
<td>81.2 (6.8)</td>
<td>71.1 (0)</td>
<td>52.6 (0)</td>
<td>52.6 (0)</td>
</tr>
<tr>
<td>SS</td>
<td>88.9 (0.1)</td>
<td>88.8 (0.1)</td>
<td>88.8 (0.1)</td>
<td>54.8 (3.9)</td>
<td>50.9 (0.6)</td>
<td>50.9 (0.6)</td>
</tr>
</tbody>
</table>

Three evaluation measures were used, namely precision, recall, and purity. The precision and recall of are defined as follows [3]:

\[
\text{precision} = \frac{\sum_{c=1}^{C} \left( \frac{|R_c|}{|R_c| + |S_c|} \right)}{C},
\]

\[
\text{recall} = \frac{\sum_{c=1}^{C} \left( \frac{|R_c|}{|R_c| + |T_c|} \right)}{C},
\]

where \(R_c\) denotes the set of all documents correctly assigned to \(c\), \(S_c\) denotes the set of all documents incorrectly assigned to \(c\), and \(T_c\) denotes the set of all documents incorrectly not assigned to \(c\). When computing precision and recall, we use the most optimum document-to-cluster assignment (i.e. the one that resulted in the best precision and recall) to guide us in deciding which of the ground-truth categories a cluster represented. The purity is defined as follows [10]:

\[
\text{purity} = \sum_{c=1}^{C} \frac{\max_{x \in X} |G_{cx}|}{|R_c| + |S_c|},
\]

where \(X\) denotes the set of all ground-truth document categories, \(G_{cx}\) denotes the set of all documents in the ground-truth category \(x\) and assigned to cluster \(c\), \(R_c\) and \(S_c\) are as defined above.

4.2.2. Results and discussions

Tables 6 and 7 show the performance comparisons between HFCR and Fuzzy CoDoK, and between HFCR and HFCM, respectively. In each of these tables, in addition to the 30-trial average precision, recall, and purity, we also provide information about the standard deviation, as shown in the bracket. Several observations can be made based on the results in Table 6. When data sets are relatively simple such as in the case of Classic3, BP, and Multi5, both algorithms perform equally well. In certain data sets with overlapping categories (i.e. several categories have similar contents to one another), such as: Binary, SS, SM, and ABJ; HFCR performs better than Fuzzy CoDoK. In a data set that contains highly unbalanced size categories such as Yahoo_K1, both HFCR and Fuzzy CoDoK do not perform as well as on the other data sets. When the unbalanced levels are not as extreme as Yahoo_K1, as in the case of CJB and CAB, HFCR and Fuzzy CoDoK can still result in satisfactory outcomes. We note that, generally, in the case of data sets containing overlapping categories such as: Binary, SS, SM, and ABJ, and CAB; HFCR categorizes documents better than Fuzzy CoDoK. Since overlapping categories indicate overlapping features or words, these results are consistent with our analysis that, when there are many overlapping feature clusters in a data set, the partitioning-ranking approach, due to some possible misrepresentations in its feature memberships representation (as discussed in Section 2.2.1), may be less effective in categorizing the data set’s objects or documents as compared to its dual-partitioning counterpart. From Table 7, we can also see that HFCR outperforms HFCM in majority of the data sets. This reaffirms the benefits of the co-clustering compared to its standard clustering counterpart. All the performance differences shown in Tables 6 and 7 have been tested for statistical significance. We include the details of this test in Appendix B.
We have discussed why FCR is more vulnerable to noises than HFCR. We now demonstrate how this vulnerability can affect the performance of FCR when applied on five of the document data sets listed in Table 5 (we only show five to save some space). Note that each document in these five data sets contains a document header, which is not removed during the preprocessing. These headers contain some information not related to the actual contents of the documents such as: e-mail addresses, date, path, or hyperlinks. For this reason, these headers can be considered as noises. Table 8 shows the performance results of HFCR and FCR on the five data sets. The results in the table strongly support our claim that HFCR is more robust to noises than FCR. The test for statistical significance for the performance comparisons between HFCR and FCR is provided in Appendix B.

In our experiments, we found that all the four algorithms always converged within 200 iterations. Depending on the nature of the data set tested, these algorithms including HFCR has no guarantee to converge to the optimum outcomes. For relatively simpler data sets such as Classic3, we found that all the different trials yield accurate clustering, as reflected by the high average precision, recall, and purity values. For complex data sets such as Yahoo_K1, our experiments show a significant number of trials result in not so accurate clustering. This is why we get low average precision and recall values on this particular data set.

5. Conclusion and future work

We have presented HFCR, a new heuristic-based fuzzy co-clustering algorithm based on the dual-partitioning approach. We have discussed how HFCR offers an alternative way of performing fuzzy co-clustering that enables us to tackle several issues in the existing fuzzy co-clustering formulations concerning the representation of feature clusters and also two other existing practical limitations. We have also shown, analytically and empirically, that HFCR is more robust to noise than its predecessor, FCR. We have demonstrated, through experiments, the new algorithm’s effectiveness in clustering 10 large document data sets. Since the algorithm can be implemented in a relatively efficient manner, we believe HFCR can become a good choice to tackle real-world categorization problems.

There are some potential and practical future directions. First, we may consider some other forms of distance-based objective function suitable for high-dimensional data, such as suggested by [6]. Second, we intend to conduct further investigations on how to better utilize the resulting fuzzy feature clusters in order to generate more meaningful summaries of object clusters.

Appendix A

Lemma 1. The \( u_{ci} \) and \( v_{cj} \) updates from Eqs. (4) and (11) never decrease \( J_R \) given in Eq. (10).

Proof. Eq. (4) can be derived from \( \frac{\partial L_R}{\partial u_{ci}} = 0 \) and the constraint in Eq. (2), where \( L_R \) is the Lagrangian of \( J_R \) taking into accounts the constraints in Eqs. (2) and (9) as follows:

\[
L_R = \sum_{c=1}^{C} \sum_{i=1}^{N} \sum_{j=1}^{K} u_{ci} v_{cj} d_{ij} - T_u \sum_{c=1}^{C} \sum_{i=1}^{N} u_{ci} \ln u_{ci} - T_v \sum_{c=1}^{C} \sum_{j=1}^{K} v_{cj} \ln v_{cj} \\
+ \sum_{i=1}^{N} \lambda_i \left( \sum_{c=1}^{C} u_{ci} - 1 \right) + \sum_{j=1}^{K} \gamma_j \left( \sum_{c=1}^{C} v_{cj} - 1 \right).
\]

Thus, the membership vector \( u^* = [u_{11}, u_{12}, \ldots, u_{CN}] \), with every element updated from Eq. (4), is a stationary point of \( L_R \) for a given \( v_{11}, v_{12}, \ldots, v_{CK} \). In addition, the Hessian matrix \( \nabla^2 J_R(u^*) \) can be found to be as follows:

\[
\nabla^2 J_R(u^*) = \begin{bmatrix}
\frac{\partial^2 J_R(u^*)}{\partial u_{11}^2} & \ldots & \frac{\partial^2 J_R(u^*)}{\partial u_{11} u_{CN}} \\
\vdots & \ddots & \vdots \\
\frac{\partial^2 J_R(u^*)}{\partial u_{CN} u_{11}} & \ldots & \frac{\partial^2 J_R(u^*)}{\partial u_{CN} u_{CN}}
\end{bmatrix} = \begin{bmatrix}
-T_u & 0 \\
0 & -T_u
\end{bmatrix}.
\]

Since at \( u^* \) we have \( 0 \leq u_{11}, u_{12}, \ldots, u_{CN} \leq 1 \) (from Eq. (4)), and we always set \( T_u \) to be positive, the Hessian matrix \( \nabla^2 J_R(u^*) \) is negative definite. From this we can conclude that \( u^* \) is a local maximum of \( L_R \) for a given
Lemma 2. For a given \( v_{11}, v_{12}, \ldots, v_{CK} \). Thus the update from Eq. (4) never decreases \( L_R \). From this we know that the update from Eq. (4) never decreases \( J_R \) as well because we have \( u_{ci} \) and \( v_{cj} \) always satisfy Eqs. (2) and (9), respectively, and therefore the constraint terms in \( L_R \) are always equal to 0. In a similar fashion we can show that update from Eq. (11) never decreases \( J_R \). □

\[ \begin{align*}
&\text{Lemma 2. For a given } v_{11}, v_{12}, \ldots, v_{CK}, \text{ the } u_{ci} \text{ update from Eq. (14) never decreases } J_1 \text{ given in Eq. (12).} \\
&\text{Proof. Eq. (14) is derived from } \frac{\partial L_1}{\partial u_{ci}} = 0 \text{ and the constraint in Eq. (2), where } L_1 \text{ is the Lagrangian function of } J_1 \text{ as follows:} \\
&L_1 = \sum_{c=1}^{C} \sum_{i=1}^{N} \sum_{j=1}^{K} u_{ci} v_{cj} d_{ij} - T_u \sum_{c=1}^{C} \sum_{i=1}^{N} u_{ci} \ln u_{ci} - T_v \sum_{c=1}^{C} \sum_{j=1}^{K} v_{cj} \ln v_{cj} \\
&+ \sum_{i=1}^{N} \lambda_i \left( \sum_{c=1}^{C} u_{ci} - 1 \right) + \sum_{j=1}^{K} \gamma_j \left( \sum_{c=1}^{C} v_{cj} - 1 \right).
\end{align*} \]

Thus, the membership vector \( u^* = [u_{11}, u_{12}, \ldots, u_{CN}] \), with every element updated from Eq. (14), is a stationary point of \( L_1 \) for a given \( v_{11}, v_{12}, \ldots, v_{CK} \). We can show that the Hessian matrix \( \nabla^2 J_1(u^*) \) can be found to be as follows:

\[ \nabla^2 J_1(u^*) = \begin{bmatrix}
\frac{\partial J_1(u^*)}{\partial u_{11} u_{11}} & \cdots & \frac{\partial J_1(u^*)}{\partial u_{11} u_{CN}} \\
\vdots & \ddots & \vdots \\
\frac{\partial J_1(u^*)}{\partial u_{CN} u_{11}} & \cdots & \frac{\partial J_1(u^*)}{\partial u_{CN} u_{CN}}
\end{bmatrix} = \begin{bmatrix}
-T_u & 0 \\
0 & \ddots \\
0 & 0 & -T_u
\end{bmatrix}.
\]

Since at \( u^* \) we have \( 0 \leq u_{11}, u_{12}, \ldots, u_{CN} \leq 1 \) (from Eq. (14)), and we always set \( T_u \) to be positive, the Hessian matrix \( \nabla^2 J_1(u^*) \) is negative definite. From this we can conclude that \( u^* \) is a local maximum of \( L_1 \) for a given \( v_{11}, v_{12}, \ldots, v_{CK} \). Thus the update from Eq. (14) never decreases \( L_1 \) or equivalently \( J_1 \). □

Lemma 3. For a given \( u_{11}, u_{12}, \ldots, u_{CN} \), the \( v_{cj} \) update from Eq. (15) never decreases \( J_2 \) given in Eq. (13).

Proof. Lemma 3 can be proven in a similar fashion as Lemma 2. □

Appendix B

We apply the one-sided two-sample \( t \)-test for equal means to evaluate the statistical significance of the performance differences reported in Tables 6–8. We state the following null and alternative hypotheses:

\[ \begin{align*}
&H_0: \text{The average of precision/recall/purity of HFCR is equal to the average of precision/recall/purity of Fuzzy CoDoK/HFCM/FCR} \\
&\text{Fuzzy CoDoK/HFCM/FCR:} \\
&\begin{cases}
&\text{The average of precision/recall/purity of HFCR is greater than the average of precision/recall/purity of Fuzzy CoDoK/HFCM/FCR.} \\
&\text{Fuzzy CoDoK/HFCM/FCR is worse than the performance of Fuzzy CoDoK/HFCM/FCR.}
&\end{cases}
\end{align*} \]

\[ \begin{align*}
&H_1: \text{The average of precision/recall/purity of HFCR is less than the average of precision/recall/purity of Fuzzy CoDoK/HFCM/FCR.} \\
&\text{Fuzzy CoDoK/HFCM/FCR:} \\
&\begin{cases}
&\text{The average of precision/recall/purity of HFCR is better than the performance of Fuzzy CoDoK/HFCM/FCR.} \\
&\text{Fuzzy CoDoK/HFCM/FCR is better than the performance of Fuzzy CoDoK/HFCM/FCR.}
&\end{cases}
\end{align*} \]
Table 9
Test statistic for the performance comparisons between HFCR and Fuzzy CoDoK

<table>
<thead>
<tr>
<th>Data set</th>
<th>Precision</th>
<th>Recall</th>
<th>Purity</th>
</tr>
</thead>
<tbody>
<tr>
<td>BP</td>
<td>2.58</td>
<td>2.5</td>
<td>2.5</td>
</tr>
<tr>
<td>Classic3</td>
<td>0</td>
<td>NA</td>
<td>NA</td>
</tr>
<tr>
<td>Multi5</td>
<td>1.44</td>
<td>1.47</td>
<td>1.41</td>
</tr>
<tr>
<td>Binary</td>
<td>1.87</td>
<td>2.09</td>
<td>2.09</td>
</tr>
<tr>
<td>SM</td>
<td>6.17</td>
<td>6.8</td>
<td>6.8</td>
</tr>
<tr>
<td>SS</td>
<td>7.02</td>
<td>7.4</td>
<td>7.4</td>
</tr>
<tr>
<td>ABJ</td>
<td>7.71</td>
<td>17.48</td>
<td>17.48</td>
</tr>
<tr>
<td>YahooK1</td>
<td>2.54</td>
<td>3.27</td>
<td>0.53</td>
</tr>
<tr>
<td>CJB</td>
<td>1.32</td>
<td>2.32</td>
<td>1.97</td>
</tr>
<tr>
<td>CAB</td>
<td>1.94</td>
<td>2.7</td>
<td>3.77</td>
</tr>
</tbody>
</table>

Table 10
Test statistic for the performance comparisons between HFCR and HFCM

<table>
<thead>
<tr>
<th>Data set</th>
<th>Precision</th>
<th>Recall</th>
<th>Purity</th>
</tr>
</thead>
<tbody>
<tr>
<td>BP</td>
<td>4.3</td>
<td>4.02</td>
<td>4.02</td>
</tr>
<tr>
<td>Classic3</td>
<td>21.91</td>
<td>NA</td>
<td>NA</td>
</tr>
<tr>
<td>Multi5</td>
<td>4.75</td>
<td>4.96</td>
<td>5.35</td>
</tr>
<tr>
<td>Binary</td>
<td>1.17</td>
<td>3.39</td>
<td>3.39</td>
</tr>
<tr>
<td>SM</td>
<td>6.68</td>
<td>6.61</td>
<td>6.61</td>
</tr>
<tr>
<td>SS</td>
<td>6.41</td>
<td>6.44</td>
<td>6.44</td>
</tr>
<tr>
<td>ABJ</td>
<td>94.57</td>
<td>92.77</td>
<td>187.79</td>
</tr>
<tr>
<td>YahooK1</td>
<td>0.92</td>
<td>2.17</td>
<td>1.7</td>
</tr>
<tr>
<td>CJB</td>
<td>4.91</td>
<td>4.57</td>
<td>4.52</td>
</tr>
<tr>
<td>CAB</td>
<td>7.15</td>
<td>6.87</td>
<td>6.63</td>
</tr>
</tbody>
</table>

We use the following test statistic:

\[ T = \frac{\bar{X}_1 - \bar{X}_2}{\sqrt{\frac{s_1^2}{N_1} + \frac{s_2^2}{N_2}}} \]

where \( \bar{X} \) denotes the average value, \( s \) denotes the standard deviation, and \( N_1 = N_2 = 30 \) indicating the number of trials in our case. We reject the null hypothesis \( H_0 \) if \( T > t_{(2, v)} \), where \( t_{(2, v)} \) denotes the critical value of the \( t \)-distribution, \( \alpha = 0.05 \), and the degree of freedom \( v \) is defined as follows:

\[ v = \frac{\left( \frac{s_1^2}{N_1} + \frac{s_2^2}{N_2} \right)^2}{\left( \frac{s_1^2}{N_1} \right)^2 + \left( \frac{s_2^2}{N_2} \right)^2} \cdot \frac{(N_1 - 1)}{(N_2 - 1)}. \]

From the \( t \)-distribution table, we found that for all the results, \( t_{(2, v)} \) is approximately equal to 1.7. The following Tables 9–11 show the test statistic values \( T \) for the performance differences in Tables 6–8, respectively:

In Tables 9–11, NA indicates that the results of the two algorithms being compared have 0 standard deviation. A bolded entry indicates that the null hypothesis \( H_0 \) is rejected in favor of the alternative hypothesis \( H_1 \); in other words, the performance difference between the two algorithms is significant. In Section 4.2.2, we state that HFCR outperforms Fuzzy CoDoK on Binary, SM, SS, ABJ, and CAB. From Table 9, we confirm that the performance differences on these data sets are statistically significant. We also mention in Section 4.2.2, that HFCR generally achieves better
performances than HFCM. From Table 10, we can see that the performance differences between HFCR and HFCM are generally statistically significant. Finally, Table 11 shows all the performance differences between HFCR and FCR are statistically significant. Therefore, as discussed in Section 4.2.2, HFCR consistently outperforms FCR significantly on all the data sets tested.

References