Consensus clustering based on constrained self-organizing map and improved Cop-Kmeans ensemble in intelligent decision support systems

Yan Yang a,b,*, Wei Tan a,b, Tianrui Li a,b, Da Ruan c

a School of Information Science and Technology, Southwest Jiaotong University, Chengdu 610031, PR China
b Key Lab of Cloud Computing and Intelligent Technology, Sichuan Province, Chengdu 610031, PR China
c Belgian Nuclear Research Centre (SCK\textsuperscript{CEN}), Mol & Ghent University, Gent, Belgium

1. Introduction

A decision support system (DSS) is a computer-based information system that supports business or organizational decision-making activities [1]. The term of intelligent decision support systems (IDSS) describes DSS that make extensive use of artificial intelligence (AI) techniques. Along with knowledge-based decision analysis models and methods, IDSS incorporate well databases, model bases and intellectual resources of individuals or groups to support effective decision making [2,3]. Some research in AI focused on enabling systems to respond to novelty and uncertainty in more flexible ways has been successfully used in IDSS. Different algorithms are defined as a kind of unsupervised learning and perform through to identify relations and rules for IDSS. Data mining tool include techniques like case-based reasoning, clustering analysis, classification, association rule mining, and data visualization. Data mining increases the “intelligence” of DSS and becomes an important component in designing IDSS.

Data mining processes data from different perspectives into useful knowledge, and becomes an important component in designing intelligent decision support systems (IDSS). Clustering is an effective method to discover natural structures of data objects in data mining. Both clustering ensemble and semi-supervised clustering techniques have been emerged to improve the clustering performance of unsupervised clustering algorithms. Cop-Kmeans is a K-means variant that incorporates background knowledge in the form of pairwise constraints. However, there exists a constraint violation in Cop-Kmeans. This paper proposes an improved Cop-Kmeans (ICop-Kmeans) algorithm to solve the constraint violation of Cop-Kmeans. The certainty of objects is computed to obtain a better assignment order of objects by the weighted co-association. The paper proposes a new constrained self-organizing map (SOM) to combine multiple semi-supervised clustering solutions for further enhancing the performance of ICop-Kmeans. The proposed methods effectively improve the clustering results from the validated experiments and the quality of complex decisions in IDSS.

© 2011 Elsevier B.V. All rights reserved.
not easy to be used as a single algorithm to explore variety of structures of data objects. If the algorithm is not well suited for the dataset, it will result in a worse clustering. In recent years, semi-supervised clustering and clustering ensemble have been emerged as powerful tools to solve the above-mentioned problems.

Inspired by multiple classifiers ensemble, clustering ensemble [7–13] has been proved to improve the performance of traditional clustering algorithms. It integrates multiple clustering components generated by different algorithms, the same algorithm with different initialization parameters and so on. The final consensus clustering with its higher stability and robustness is obtained after such a combination. Establishing consensus functions is the key issue for clustering ensemble. Fred and Jain [8] explored an evidence accumulation clustering approach with the single-link and average-link hierarchical agglomerative algorithms. Their approach maps the clustering ensemble into a new similarity measure between patterns by accumulating pairwise pattern co-associations. Strehl and Ghosh [9] used three graph-based partitioning algorithms such as CSPA, HGPA and MCLA to generate the combined clustering. Zhou and Tang [10] employed four voting methods to combine the aligned clusters through the selective mutual information weights. Ayad and Kamel [11] sought cumulative vote weighting schemes and corresponding algorithms to compute an empirical probability distribution summarizing the ensemble. Yang et al. [12] improved an ant-based clustering algorithm to produce multiple clustering components as the input of an Adaptive Resonance Theory (ART) network and obtained the final partition. Wang et al. [13] proposed Bayesian cluster ensembles, a mixed-membership generative model to obtain a consensus clustering by combining multiple base clustering results.

Semi-supervised clustering [14–20] algorithms obtain better results using some prior knowledge, which is often represented by seeds or pairwise constraints. The seeds give directly the class labels of data objects. The pairwise constraints indicate whether a pair of objects is classified into the same group (must-link, ML) or different groups (cannot-link, CL). The recent semi-supervised clustering algorithms consist of two types: the constraint-based methods and distance-based methods. Seeded-Kmeans and Constrained-Kmeans proposed by Basu et al. [14] both utilize seeds information to guide the clustering process. For Seeded-Kmeans, the seeds information is only used to initialize cluster centers. For Constrained-Kmeans, the seeds labels are kept unchanged during the iteration step besides the initialization of cluster centers. Wagstaff et al. [15] proposed the Cop-Kmeans algorithm, where Must-Link and Cannot-Link constraints are incorporated into the assignment step and cannot be violated. Basu et al. [16] proposed the PCK-Kmeans to impose the violation penalty on the objective function of K-means [21], and two kinds of constraints were violated by adding the penalty. Xing et al. [17] employed metric learning techniques to get an adaptive distance measure based on the given pairwise constraints. Zhu et al. [18] extended balancing constraints to size constraints, i.e., based on the prior knowledge of the distribution of the data, and assigned the size of each cluster to find a partition that satisfies the size constraints. Zhang and Lu [19] proposed a kernel-based fuzzy algorithm to learn a cluster from both the labeled and unlabeled data. Abdala and Jiang [20] pre-
sented a semi-supervised clustering ensemble method to combine multiple semi-supervised clustering components for ensuring that the given constraints cannot be violated in the final clustering result.

The authors in [22,23] have recently investigated the combination of semi-supervised clustering and clustering ensemble. Compared with the random selection of constraints, they utilized the active learning technique or bounded constraints selection to obtain more informative and valuable constraints, which need querying from oracles or intervention with users. However, it is not realistic to ensure that the entire requirement for constraints is satisfied by the feedback from the real world.

This paper explores a combination of semi-supervised clustering and consensus clustering in IDSS at a different perspective. Thanks to the sensitivity of assignment order for many semi-supervised clustering algorithms [15,24], the clustering ensemble in the form of the co-association matrix gets a new assignment order for data objects. An improved version of Cop-Kmeans algorithm is then employed to generate multiple clustering components [25]. The paper proposes a novel constrained self-organizing map as the consensus function to obtain the final partition. The case study shows that the improved methods can support better decision making in IDSS.

The rest of the paper is organized as follows: Section 2 introduces the system framework, including data-driven DSS based on data mining and the consensus clustering. Section 3 outlines the K-means algorithm, Cop-Kmeans algorithm and an improved Cop-Kmeans (ICop-Kmeans) algorithm. Section 4 presents a new assignment order based on the weighted co-association matrix. Section 5 proposes a novel semi-supervised clustering ensemble. Section 6 illustrates a case study to validate that clustering is an important component in IDSS. Section 7 shows the experimental evaluation. Section 8 concludes the paper with further research topics.

2. The architecture of data-driven DSS based on data mining

2.1. A framework of data-driven DSS

Today’s real world decision issues have become more complicated, time consuming and difficult for managers, leading to modern approaches to the decision-making processes. Such a process involves more factors and aspects like data mining and OLAP than traditional decision making models. Fig. 1 shows a framework of data-driven DSS, in which data mining (e.g., clustering) as an important component in IDSS supports better decision making.

2.2. A framework of the consensus clustering

Fig. 2 shows a framework of consensus clustering based on the constrained SOM and improved Cop-Kmeans ensemble. The first phase consists of H clustering components using the improved Cop-Kmeans (ICop-Kmeans) algorithm with different initial

---

Fig. 2. A framework of the consensus clustering based on the constrained SOM and ICOP-Kmeans ensemble.
parameters, each of which generates a clustering. And in the second phase, a new feature weighted matrix $A$ is constructed by computing the certainty of objects and reassigning objects order. Finally, a constrained SOM is taken as the consensus function (Cop-SOM-E) to generate the final clustering result $\pi'$. Further details about these processes will be covered later.

3. An improved Cop-Kmeans algorithm

3.1. The K-means algorithm

The K-means algorithm [21] partitions a dataset into $k$ groups in which each object belongs to the cluster with the nearest mean. It proceeds by selecting $k$ initial cluster centers and then by iteratively refining them as follows: each object $x_i$ is assigned to its closest cluster and each cluster center $C_j$ is updated to be the mean of its constituent objects. The algorithm initializes the clusters using objects chosen at random from the dataset and converges when there is no further change in the assignment of objects to clusters.

3.2. The Cop-Kmeans algorithm

The Cop-Kmeans algorithm [15] enhances the traditional K-means algorithm by adding two types of constraints: must-link and cannot-link, which imply that objects should be assigned into the same cluster or not. It works as follows: first, the transitive closures are implemented to extend must-link and cannot-link constraints and $k$ initial cluster centers are selected. Second, an object $x_i$ is assigned to the nearest cluster center under the premise of none of the extended constraints being violated, otherwise it returns an empty partition and the algorithm fails if such a cluster cannot be found. Finally, every cluster center is updated and the algorithm iterates until its convergence.

From the description of the algorithm and the experiments, the Cop-Kmeans sometimes do not give a valid result, which can be illustrated by Fig. 3. Objects $(x_i, x_k)$ and $(x_j, x_k)$ both belong to the cannot-link constraints whereas $x_i$ and $x_j$ have been assigned into the clusters $C_a$ and $C_b$ before deciding the label of $x_k$. When it is the time to assign $x_k$, there is not any right cluster that satisfies all the cannot-link constraints and the algorithm terminates.

Wagstaff [24] presented two methods for overcoming the disadvantages. One is to just restart the algorithm with a group of random initial cluster centers; the other is to return a previous solution when the current iteration meets the inconsistency of assignment. These schemes however suffer from time-consuming and often result in an unconvergent solution with low accuracy even invalid clusters if the failure occurs in the first iteration of the algorithm.

3.3. The improved Cop-Kmeans algorithm

For the case of Fig. 3, the sensitivity of Cop-Kmeans is for the assignment order of objects. The number of the clusters $k$ is 2, while $(x_i, x_k) \in CL$ and $(x_j, x_k) \in CL$. It means that the right class labels of $x_i$ and $x_j$ must be the same. However, $x_i$ and $x_j$ are in the different clusters ($C_a$ and $C_b$) at the moment because of the assignment of $x_i$ and $x_j$ before $x_k$. Therefore, the improper assignment order makes $x_i$ and $x_j$ separated and indirectly leads to the violation of cannot-link constraints for $x_k$.

Note that only the cannot-link constraints induce the failure of looking for a satisfying solution because the must-link constraints
are transitive and not influenced by the assignment order. For example, \((x_i, x_k) \in ML\) and \((x_j, x_k) \in ML\) imply \((x_i, x_j) \in ML\) so that \(x_i\) and \(x_j\) must be in the same cluster before assigning \(x_k\). However, \((x_i, x_k) \in CL\) and \((x_j, x_k) \in CL\) do not reveal the relationship between \(x_i\) and \(x_j\) so that the two objects are possible to be placed into the different clusters before assigning \(x_k\). Therefore, a cannot-link constraint may be violated when \(x_k\) comes.

To solve this problem, we consider reassigning the \(x_i\), \(x_j\) and \(x_k\) \([25]\). Assume \(u_a\) and \(u_b\) are the centers of \(C_a\) and \(C_b\) respectively. Supposing \(\text{Dist}(x_k, u_b) < \text{Dist}(x_k, u_a)\), we check the ones that have been already placed into \(C_b\) among the objects that have cannot-link constraints with \(x_k\). When finding \(x_i\) is in \(C_b\), the class label of \(x_i\) is canceled temporarily and \(x_k\) is moved to the closest cluster \(C_b\). Then, we reassign \(x_j\) to the closest cluster except \(C_b\). At last, \(x_j\) is assigned to \(C_b\) and the assignment conflict is solved. Fig. 4 illustrates all the processes for the assignment of conflicts.

Following the above heuristic analysis, we propose an improved Cop-Kmeans algorithm called ICop-Kmeans. The key difference between the Cop-Kmeans and the improved version is that the latter reassigns the order of several objects when the constraint violation happens. During the iteration, both Cop-Kmeans and ICop-Kmeans perform the same way if the cannot-link constraints are never broken. In case of a conflict, the corresponding process is adopted to solve the conflict and guarantees the smooth implementation of the algorithm without any interruption. Fig. 5 lists the detailed description of the ICop-Kmeans.

| Input: dataset \(X\), number of clusters \(k\), must-link constraints \(ML\), cannot-link constraints \(CL\), maximal iteration time \(T\). |
| Output: clusters \(C_1, \ldots, C_k\) |
| (1) Initialize \(k\) cluster centers \(C_1, \ldots, C_k\). |
| (2) For every object \(x_i\) in \(X\), assign it to the nearest cluster \(C_j\) to ensure that Violate-constraints \((x_i, C_j, ML, CL)\) returns false. If no such a cluster is found, Process-violation \((x_i, CL)\) is implemented. |
| (3) Update every cluster center \(C_j\) by calculating the mean of the objects that have been assigned to the cluster. |
| (4) Repeat (2) and (3) until the algorithm converges or the iteration time comes to \(T\). |
| (5) Return clusters \(C_1, \ldots, C_k\). |

Violate-constraints (object \(x\), cluster \(C\), must-link constraints \(ML\), cannot-link constraints \(CL\))

| (1) If \(\exists x_m\) is in the cluster \(C\) and \((x, x_m) \in CL\), return true. |
| (2) If \(\exists x_j\) is not in the cluster \(C\) and \((x, x_j) \in ML\), return true. |
| (3) Otherwise, return false. |

Process-violation (object \(x\), cannot-link constraints \(CL\), cluster \(C_j\))

| (1) Find the cluster \(C\) whose center is the nearest to \(x\) if the last parameter is not given. Otherwise, find the cluster \(C\) whose center is the nearest to \(x\) except \(C_j\). |
| (2) For every object that has a relationship of \(CL\) with \(x\), check the ones that have been already placed into \(C\) and then put these objects in a set \(S\). |
| (3) Cancel the class labels of the objects in \(S\), and assign \(x\) to \(C\). |
| (4) For every object \(x_j\) in \(S\), reassign it to the nearest cluster except \(C\) and make sure that the cannot-link constraints cannot be violated. If no such a cluster is found, Process-violation \((x, CL, C)\) is implemented. |

Note that Process-violation is a recursive function. When Violate-constraints return true, Process-violation is called for the first time without the last parameter. It is implemented again only if the objects that need to be reassigned also suffer from the unsuccessful assignment. This situation exists, but it does not happen frequently when the number of pairwise constraints is less than 80 in later experiments. We will compare these two algorithms in Section 7.

### 4. A new assignment order based on the weighted co-association matrix

#### 4.1. The weighted co-association matrix

Given a set of data objects \(X = \{x_1, x_2, \ldots, x_N\}\) where \(N\) is the number of the dataset and \(x_i = \{x_{i1}, x_{i2}, \ldots, x_{id}\}\), \(d\) is the dimensionality of features. Assume that multiple diverse clustering components \(P = \{p_1, p_2, \ldots, p_H\}\) are generated from \(X\), where \(H\) is the number of the clustering components. \(x_i \rightarrow \{\pi_1(x_i), \pi_2(x_i), \ldots, \pi_H(x_i)\}\) is the corresponding class label vector of \(x_i\) in each \(\pi_j\), which is a label vector representing the \(j\)-th clustering component. To denote the consistency of a pair of clustering components, we employ the mutual information \([9]\) to measure the similarity between the two clustering components.

Suppose \(\pi_a\) and \(\pi_b\) are two clustering components, the normalized mutual information is defined as follows:

\[
\text{Mutual Information} = \frac{\sum_{x, y} p(x, y) \log \frac{p(x, y)}{p(x)p(y)}}{\sum_{x} p(x) \log \frac{p(x)}{p(x)}}
\]
\[
\text{NMI}(\pi_a, \pi_b) = \sqrt{\frac{\sum_{h=1}^{k^{(a)}} \log \left( \frac{n_h^{(a)}}{N^{(a)}} \right) \sum_{h=1}^{k^{(b)}} \log \left( \frac{n_h^{(b)}}{N^{(b)}} \right)}{\sum_{h=1}^{k^{(a)}} n_h^{(a)} \log \left( \frac{n_h^{(a)}}{N^{(a)}} \right) + \sum_{h=1}^{k^{(b)}} n_h^{(b)} \log \left( \frac{n_h^{(b)}}{N^{(b)}} \right)}}
\]

where \(k^{(a)}\) and \(k^{(b)}\) are the numbers of \(\pi_a\) and \(\pi_b\), respectively. \(n_h^{(a)}\) denotes the number of objects in the class \(C^{(a)}_h\) belonging to \(\pi_a\), and \(n_h^{(b)}\) denotes the number of objects in the class \(C^{(b)}_h\) belonging to \(\pi_b\). \(n_h\) denotes the number of objects in the class \(C^{(a)}_h\) belonging to \(\pi_a\) as well as in the class \(C^{(b)}_h\) belonging to \(\pi_b\).

When every pair of clustering components is calculated, the average mutual information [9] of a single clustering component is defined as follows:

\[
\text{NMI}(\pi_m, A) = \frac{1}{H} \sum_{h=1}^{H} \text{NMI}(\pi_m, \pi^{(h)})
\]

where \(m = 1, 2, \ldots, H\), \(A\) denotes the set of all the label vectors.

Once all the clustering components are generated, we construct an \(N \times N\) co-association matrix [8], where each \((x_i, x_j)\) cell represents the number of times the given object pair is co-occurred in a cluster. The entry in the co-association matrix also indicates a kind of similarity between two objects, and it is defined as follows:

\[
\text{Co-association}(x_i, x_j) = \frac{1}{H} \sum_{h=1}^{H} \delta(\pi_h(x_i), \pi_h(x_j))
\]

where \(\delta(x, y) = \begin{cases} 1 & \text{if } x = y \\ 0 & \text{otherwise} \end{cases}\)

The current study on the co-association matrix mostly lacks of considering the quality of each clustering component treated in the same weight. In this section, we use the average mutual information as the weight of the clustering component. The higher the average mutual information is, the more information the corresponding clustering component. Therefore, Eq. (3) is updated as follows:

\[
\text{Co-association}(x_i, x_j) = \frac{\sum_{h=1}^{N} \delta(\pi_h(x_i), \pi_h(x_j)) \cdot \text{NMI}(\pi_m, A)}{\sum_{h=1}^{H} \text{NMI}(\pi_h, A)}
\]

### 4.2. The certainty of objects

After the co-association matrix integrates with the mutual information in the form of weights, we construct a better co-association matrix. For two objects \(x_i\) and \(x_j\), \(\text{Co-association}(x_i, x_j)\) is a real value between 0 and 1 representing a similarity relation of each pair \((x_i, x_j)\) being assigned together. When \(\text{Co-association}(x_i, x_j)\) is equal or close to 1, it implies that \(x_i\) and \(x_j\) are highly likely to be classified into the same cluster. When \(\text{Co-association}(x_i, x_j)\) is equal or close to 0, it implies that \(x_i\) and \(x_j\) are unlikely to be classified into the different clusters. However, \(\text{Co-association}(x_i, x_j)\) with a value equal or close to 0.5 is rather unsure to confirm that the objects are placed into the same cluster or not. Therefore, it is crucial to clearly describe the certainty of a single object instead of two objects. Here, we establish a function mapping by the given co-association matrix into a value that represents the association with all the objects. The mapping function is defined as follows:

\[
\text{Certainty}(x_i) = \frac{1}{N-1} \sum_{j=1, j \neq i}^{N} F(\text{Co-association}(x_i, x_j))
\]

where \(N\) denotes the number of the objects, \(0 \leq \text{Certainty}(x_i) \leq 1\). Note that \(F(x)\) is a segmental function that is continuous and monotone in the intervals of \([0, 0.5]\) and \([0.5, 1]\). If the entry of the co-association matrix is regarded as the independent variable of \(F(x)\), \(F(\text{Co-association}(x_i, x_j))\) denotes the relative certainty between \(x_i\) and \(x_j\). When \(\text{Co-association}(x_i, x_j)\) is equal or close to 1 or 0, it implies that are very likely or unlikely to be classified into the same cluster and \(x_i\) is very certain relative to \(x_j\). In this case, \(F(\text{Co-association}(x_i, x_j))\) is equal or close to 1. When \(\text{Co-association}(x_i, x_j)\) is equal or close to 0.5, it implies that the cluster relationship is unsure between \(x_i\) and \(x_j\) and uncertain relative to \(x_j\). In this case, \(F(\text{Co-association}(x_i, x_j))\) is equal or close to 0. For \(x_i\), the whole certainty \(\text{Certainty}(x_i)\) is defined in Eq. (6). Hence, the higher the \(\text{Certainty}(x_i)\) is, the larger the certainty of object \(x_i\).

### 4.3. The generation of a new assignment order

Many semi-supervised clustering algorithms like Cop-K-means are sensitive to the assignment order of data objects. If an object with the low certainty is assigned before an object with the high certainty and an ML or CL constraint is connected between them, there is a greater probability to identify their class labels inaccurately. Therefore, we rearrange all the objects in a descending order by the certainty of each object. Finally, a new assignment order is produced.

### 5. A semi-supervised clustering ensemble

To further improve the accuracy of semi-supervised clustering algorithms, we focus on the method of unsupervised clustering ensemble. At present, many consensus functions of unsupervised clustering ensemble are proposed. However, Abdala and Jiang [20] indicated that if the consensus functions of unsupervised clustering ensemble are directly applied to combine multiple clustering components that satisfy the given pairwise constraints, it is possible that the constraints are violated in the final clustering result. In this section, we propose a constrained SOM algorithm based on pairwise constraints corresponding to last second block “Cop-SOM” in Fig. 2. The pairwise constraints are incorporated into the SOM and the constrained SOM algorithm as a kind of consensus function combines multiple semi-supervised clustering components.

#### 5.1. The SOM based on pairwise constraints

The SOM is a kind of unsupervised neural-network model proposed by Kohonen [26]. It is often used for clustering analysis and maps the feature space of data with high dimensions to a one- or two-dimensional space. The SOM is self-stable and keeps the topological structure of original input vectors. The SOM consists of two layers of neurons, an input layer and a competition layer. The topological structure of the competition layer is often represented by the grid of one dimension or two, and the neurons on the input layer are fully connected with the ones on the competition layer according to the connection weights. During the learning phase, the winning neuron is defined as follows:

\[
q(t) = \arg \min_{j} \| x_i(t) - w_j(t) \|
\]

where \(t\) denotes time, \(x_i(t)\) is the input vector and \(w_j(t)\) is the weight vector, \(q(t)\) is the winning neuron. After finding the winning neuron, the weight vectors between the two layers are updated so that the winning neuron and neighboring neurons are moved closer to the
input vector in the input space. The adjustment for the weight vectors is defined as follows:

$$w_j(t + 1) = \begin{cases} w_j(t) + \eta(t)(x_k(t) - w_j(t)) & j \in N_q(t) \\ w_j(t) & j \notin N_q(t) \end{cases}$$

where $\eta(t)$ is the learning rate and $N_q(t)$ denotes the neighbor area size. $\eta(t)$ and $N_q(t)$ often decrease over time. When the training ends, the weight vectors become very similar to the original input vectors.

For the given ML and CL constraints, if we cluster the data without considering the prior knowledge, the useful information is neglected and does not guide the clustering process. To incorporate pairwise constraints into the SOM, the learning phase of the SOM must meet the following requirements:

1. For two input vectors $x_i$ and $x_j$, if $(x_i, x_j) \in ML$, then $q_i(t) = q_j(t)$. It means that the winning neuron of $x_i$ and the winning neuron of $x_j$ must be the same on the competition layer.

2. For two input vectors $x_i$ and $x_j$, if $(x_i, x_j) \in CL$, then $q_i(t) \neq q_j(t)$. It means that the winning neuron of $x_i$ and the winning neuron of $x_j$ must be different on the competition layer.

When seeking the winning neuron of the input vector, make sure that the given pairwise constraints cannot be violated. If no such a neuron is found, a reasonable solution similar to the solution for the ICop-Kmeans is adopted for solving the problem of constraint violation. Here, we call the SOM based on pairwise constraints Cop-SOM, and the detailed description of Cop-SOM is listed in Fig. 6.

### 5.2. A constrained clustering ensemble based on Cop-SOM

For $X = \{x_1, x_2, \ldots, x_N\}$ each object satisfies $x_i = \{x_i1, x_i2, \ldots, x_id\} \in \mathbb{R}^d$. When $H$ clustering components $P = \{p_1, p_2, \ldots, p_H\}$ are generated from $X$ by the ICop-Kmeans, the original dataset $X$ with $d$-dimension features can be transformed into a new $N \times H$ feature space matrix $A$ shown in Table 1. Each entry in $A$ represents the class label $p_j(x_i)$ instead of the value in the original feature space, and $H$ clustering components are regarded as a new $H$-dimension feature.

When Cop-SOM serves as the consensus function, the feature-space matrix $A$ is taken as the network input. Fig. 7 lists the details.

### Table 1

<table>
<thead>
<tr>
<th>Class Label $p_1$</th>
<th>Class Label $p_2$</th>
<th>Class Label $p_3$</th>
<th>$\ldots$</th>
<th>Class Label $p_H$</th>
</tr>
</thead>
<tbody>
<tr>
<td>$x_1$</td>
<td>$p_1(x_1)$</td>
<td>$p_2(x_1)$</td>
<td>$p_3(x_1)$</td>
<td>$\ldots$</td>
</tr>
<tr>
<td>$x_2$</td>
<td>$p_1(x_2)$</td>
<td>$p_2(x_2)$</td>
<td>$p_3(x_2)$</td>
<td>$\ldots$</td>
</tr>
<tr>
<td>$x_3$</td>
<td>$p_1(x_3)$</td>
<td>$p_2(x_3)$</td>
<td>$p_3(x_3)$</td>
<td>$\ldots$</td>
</tr>
<tr>
<td>$\ldots$</td>
<td>$\ldots$</td>
<td>$\ldots$</td>
<td>$\ldots$</td>
<td>$\ldots$</td>
</tr>
<tr>
<td>$x_N$</td>
<td>$p_1(x_N)$</td>
<td>$p_2(x_N)$</td>
<td>$p_3(x_N)$</td>
<td>$\ldots$</td>
</tr>
</tbody>
</table>

**Fig. 6.** The detailed description of Cop-SOM.
tailed description of constrained clustering ensembles using the consensus function based on Cop-SOM (called Cop-SOM-E).

### 6. An illustrative example

In this section, we give a simple case-based example to show the performance of an IDSS (see Fig. 1) in which data clustering is as a component to support decision. In credit card management, predicting the customers’ behavior is a key to reduce the risk of credit card issuers. Credit card holders are important group customers to the banks. Clustering the holders into several clusters helps the managers make the right decision and pinpoint the right customers to markets.

Table 2 shows a simple example of credit clustering. For 10 customers, they may be clustered by their income, occupation and record of credit, which are the key factors for customer categories. The item of “income” is divided into five levels: the 5 denotes the highest income and 1 denotes the lowest income. The third column “occupation” indicates 5 with good positions such as professors, lawyers, and doctors and 1 with low positions. The entry of “record of credit” rank from 1 to 3, i.e., 1 shows the best credit record and 3 shows the worst.

The K-means, Cop-Kmeans, and ICop-Kmeans algorithms cluster the customers quickly to help managers make their decisions. For the 10 customers with three attributes in Table 2, the K-means, Cop-Kmeans, and ICop-Kmeans algorithms all cluster them into two classes (see Table 3). The last column “classes” is the results by the algorithms that categorize the 10 customers into A and B clusters finally. The customers with higher income, good occupations and better credit records are clustered into one partition A, while the other customers with low positions and worse credit records are grouped into another partition B.

In many application domains such as information retrieval, bioinformatics, a few labeled data or pairwise constraints can be obtained which may greatly help our decision-making. Semi-supervised clustering algorithms, e.g., Cop-Kmeans, ICop-Kmeans take advantages of this prior information about the dataset to aid the clustering process. For example, the customers 8 and 10 may be treated as in the same class by the experts of credit evaluation since they have the same values in entry “occupation” and “record of credit” with no much difference in “income”. That means the customers 8 and 10 should belong to ML constraints. Using this pairwise constraints, the Cop-Kmeans and ICop-Kmeans algorithms produce the clustering results: A = {1,2,3,7,9} and B = {4,5,6,8,10} (see Table 4).

However, in some case there may be more constraints from the experts. For example, there are two constraints: (1) The customers 8 and 9 cannot be in the same class; (2) The customers 9 and 10 cannot be in the same class. This will result in a constraint violation in the process of assigning the customer 9 using the Cop-Kmeans algorithm. In this case, the Cop-Kmeans algorithm cannot deal with this issue. The proposed ICop-Kmeans algorithm solves the constraint violation by reassigning the customer 10 to the class B and adding the customer 9 to the class A. The result obtained from the ICop-Kmeans algorithm is the same as that in Table 4.

This case-based example thus illustrates that clustering can help the managers make the right decision as its function shown in the data-driven DSS framework of Fig. 1.

### 7. Experimental evaluations

To test the performance of the proposed methods, we have performed several experiments. The dataset includes artificial and UCI types. Table 5 shows the description of datasets.

---

**Table 2**

An example of credit clustering.

<table>
<thead>
<tr>
<th>Customers</th>
<th>Income</th>
<th>Occupation</th>
<th>Record of credit</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>4</td>
<td>5</td>
<td>1</td>
</tr>
<tr>
<td>2</td>
<td>5</td>
<td>3</td>
<td>2</td>
</tr>
<tr>
<td>3</td>
<td>4</td>
<td>4</td>
<td>3</td>
</tr>
<tr>
<td>4</td>
<td>2</td>
<td>1</td>
<td>3</td>
</tr>
<tr>
<td>5</td>
<td>1</td>
<td>2</td>
<td>3</td>
</tr>
<tr>
<td>6</td>
<td>2</td>
<td>1</td>
<td>2</td>
</tr>
<tr>
<td>7</td>
<td>5</td>
<td>4</td>
<td>2</td>
</tr>
<tr>
<td>8</td>
<td>1</td>
<td>2</td>
<td>1</td>
</tr>
<tr>
<td>9</td>
<td>3</td>
<td>3</td>
<td>2</td>
</tr>
<tr>
<td>10</td>
<td>2</td>
<td>2</td>
<td>1</td>
</tr>
</tbody>
</table>

**Table 3**

An example of credit clustering with classes.

<table>
<thead>
<tr>
<th>Customers</th>
<th>Income</th>
<th>Occupation</th>
<th>Record of credit</th>
<th>Classes</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>4</td>
<td>5</td>
<td>1</td>
<td>A</td>
</tr>
<tr>
<td>2</td>
<td>5</td>
<td>3</td>
<td>2</td>
<td>A</td>
</tr>
<tr>
<td>3</td>
<td>4</td>
<td>4</td>
<td>3</td>
<td>A</td>
</tr>
<tr>
<td>4</td>
<td>2</td>
<td>1</td>
<td>3</td>
<td>B</td>
</tr>
<tr>
<td>5</td>
<td>1</td>
<td>2</td>
<td>3</td>
<td>B</td>
</tr>
<tr>
<td>6</td>
<td>2</td>
<td>1</td>
<td>2</td>
<td>B</td>
</tr>
<tr>
<td>7</td>
<td>5</td>
<td>4</td>
<td>2</td>
<td>A</td>
</tr>
<tr>
<td>8</td>
<td>1</td>
<td>2</td>
<td>1</td>
<td>B</td>
</tr>
<tr>
<td>9</td>
<td>3</td>
<td>3</td>
<td>2</td>
<td>A</td>
</tr>
<tr>
<td>10</td>
<td>2</td>
<td>2</td>
<td>1</td>
<td>A</td>
</tr>
</tbody>
</table>

**Table 4**

An example of credit clustering with one constraint.

<table>
<thead>
<tr>
<th>Customers</th>
<th>Income</th>
<th>Occupation</th>
<th>Record of credit</th>
<th>Classes</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>4</td>
<td>5</td>
<td>1</td>
<td>A</td>
</tr>
<tr>
<td>2</td>
<td>5</td>
<td>3</td>
<td>2</td>
<td>A</td>
</tr>
<tr>
<td>3</td>
<td>4</td>
<td>4</td>
<td>3</td>
<td>A</td>
</tr>
<tr>
<td>4</td>
<td>2</td>
<td>1</td>
<td>3</td>
<td>B</td>
</tr>
<tr>
<td>5</td>
<td>1</td>
<td>2</td>
<td>3</td>
<td>B</td>
</tr>
<tr>
<td>6</td>
<td>2</td>
<td>1</td>
<td>2</td>
<td>B</td>
</tr>
<tr>
<td>7</td>
<td>5</td>
<td>4</td>
<td>2</td>
<td>A</td>
</tr>
<tr>
<td>8</td>
<td>1</td>
<td>2</td>
<td>1</td>
<td>B</td>
</tr>
<tr>
<td>9</td>
<td>3</td>
<td>3</td>
<td>2</td>
<td>A</td>
</tr>
<tr>
<td>10</td>
<td>2</td>
<td>2</td>
<td>1</td>
<td>B</td>
</tr>
</tbody>
</table>
The ML and CL constraints are generated randomly in the experiments. We have add a pair of objects to the set of ML constraints if we chose two objects randomly from the dataset and the class labels of them are the same. Otherwise, we add a pair of objects to the set of CL constraints. The process is repeated until the number of pairwise constraints is satisfied.

To validate ICop-Kmeans for the first experiment, we have adopted the datasets whose cluster numbers are small. Like Cop-Kmeans, if the problem of constraint violation arises and an empty partition is returned, the algorithm fails. We compare the failure proportion of Cop-Kmeans and ICop-Kmeans and use F-measure [27] to evaluate the two algorithms. Both algorithms are implemented 500 times. For Cop-Kmeans and ICop-Kmeans, the ML and CL constraints, the initial parameters, the random assignment

<table>
<thead>
<tr>
<th>Datasets</th>
<th>Types</th>
<th>Samples</th>
<th>Classes</th>
<th>Dimensions</th>
</tr>
</thead>
<tbody>
<tr>
<td>3D3K</td>
<td>Artificial</td>
<td>300</td>
<td>3</td>
<td>3</td>
</tr>
<tr>
<td>8D5K</td>
<td>Artificial</td>
<td>1000</td>
<td>5</td>
<td>8</td>
</tr>
<tr>
<td>Iris</td>
<td>UCI</td>
<td>150</td>
<td>3</td>
<td>4</td>
</tr>
<tr>
<td>Wine</td>
<td>UCI</td>
<td>178</td>
<td>3</td>
<td>13</td>
</tr>
<tr>
<td>Glass</td>
<td>UCI</td>
<td>214</td>
<td>6</td>
<td>9</td>
</tr>
<tr>
<td>Sonar</td>
<td>UCI</td>
<td>208</td>
<td>2</td>
<td>60</td>
</tr>
</tbody>
</table>

The description of datasets.

Fig. 8. The average proportion of failure and F-measure of Cop-Kmeans and ICop-Kmeans.
order and the number of clusters are all the same each time. Fig. 8 shows the average proportion of failure and F-measure of Cop-Kmeans and ICop-Kmeans.

N denotes the number of the given pairwise constraints. The failure proportion of Cop-Kmeans increases when more constraints are given, as in Fig. 8(a). However, ICop-Kmeans never fail (in Fig. 8(b)) and its F-measure is almost the same or even a little better than Cop-Kmeans when the given number of pairwise constraints is equal (see Fig. 8(c)–(f)). The average F-measure of Cop-Kmeans only considers the successful cases. For the sonar dataset, when \( N = 80 \), the average F-measure of Cop-Kmeans is much higher than that of ICop-Kmeans because Cop-Kmeans is successfully implemented only 2% of 500 times whereas ICop-Kmeans is successfully implemented 500 times and it can even more reflect the average F-measure.

To further show the advantage of the time performance of ICop-Kmeans, we have compared the running time of Cop-Kmeans and ICop-Kmeans on each dataset. Both algorithms are implemented many times. For Cop-Kmeans and ICop-Kmeans, the ML and CL constraints, the initial parameters, the random assignment order and the cluster number are all the same each time. For Cop-Kmeans, if the problem of constraint violation arises, the algorithm restarts with a group of random initial cluster centers. We compute the total running time in which the two algorithms are implemented 500 times successfully on each dataset with its result shown in Fig. 9.

When the number of the given pairwise constraints is small, the total running time for the two algorithms being implemented 500 times successfully on each dataset is almost the same because the proportion of failure of Cop-Kmeans is low. When the number of the given pairwise constraints increases, the total running time for ICop-Kmeans being implemented 500 times successfully on each dataset is less than that of Cop-Kmeans. This is because the Cop-Kmeans needs to restart and select the random initial cluster centers frequently to ensure that the algorithm can find a reasonable solution. However, the ICop-Kmeans can be implemented 500 times successfully on each dataset in the experiment. Therefore, the time efficiency of ICop-Kmeans performs better than that of Cop-Kmeans.

After evaluating the validity of ICop-Kmeans, we have compared the influence of the random assignment order and produced the assignment order on the performance of ICop-Kmeans using the certainty of objects mentioned in Section 4. For Cop-Kmeans and ICop-Kmeans, the ML and CL constraints, the number of clusters and the initial cluster centers are the same each time, so the difference of the two algorithms depends on the different assignment order. For each time, the Kmeans algorithm with the variety of initial cluster centers is applied to generate 20 unsupervised clustering components to construct the weighted co-association matrix for producing a new assignment order of objects. The ICop-Kmeans with the produced assignment order generates 20 semi-supervised components, and the ICop-Kmeans with the random assignment order generates 20 semi-supervised components. The experiment is conducted 20 times. Figs. 10 and 11 show the maximum, minimum, average F-measure and the variance of F-measure.

Fig. 10 illuminates that the average F-measure of ICop-Kmeans with the produced order is higher than that of ICop-Kmeans with...
the random order as a whole. Generally, the maximum F-measure of ICop-Kmeans with the produced order is lower than that of ICop-Kmeans with the random order and the minimum F-measure of ICop-Kmeans with the produced order is higher than that of ICop-Kmeans with the random order. Fig. 11 also shows that the variance of ICop-Kmeans with the produced order is lower than that of ICop-Kmeans with the random order. Therefore, the ICop-Kmeans with the produced order performs better and is more stable than the ICop-Kmeans with the random order. For the dataset of Wine, the advantage of ICop-Kmeans with the produced order is rather obvious.

Next experiment aims to test whether Cop-SOM is effective to improve the performance of the unsupervised SOM or not. For Cop-SOM and SOM, we set the initial learning rate as 0.9, the total iteration times as 200. The initial field radius is the width of the competition layer. The ML and CL constraints are the same. Both of the algorithms are implemented 20 times and the average F-measure is shown in Fig. 12.

The average F-measure of Cop-SOM increases when more constraints are given except the Glass dataset. Compared with SOM, Cop-SOM effectively improves the performance of the unsupervised SOM with the pairwise constraints. Noted that SOM is a clus-
tering algorithm without the supervised information and two types of constraints do nothing to it, and Cop-SOM is considered as an unsupervised clustering algorithm equivalent to Kmeans when N=0. Therefore, Cop-SOM and SOM have the same F-measure in this case.

At last, we employ the clustering ensemble technique to further improve the performance of ICop-Kmeans. Cop-SOM is considered as the consensus function to combine multiple semi-supervised clustering components (Cop-SOM-E). In the experiment, we continue the previous experiment of ICop-Kmeans with a random order and produced order. According to 20 semi-supervised clustering components that are respectively generated by the ICop-Kmeans with the random order and produced order, the original dataset is transformed into a new feature-space matrix that is regarded as the input of the constrained consensus function. Cop-EAC-SL [20] is also a kind of semi-supervised consensus function presented by Abdala et al. Here, Cop-SOM and Cop-EAC-SL are both applied to combine the semi-supervised clustering components to obtain the final clustering result. The initial parameters of Cop-SOM-E consist with those of the last experiment. The experiment is conducted 20 times and the average F-measure evaluates the clustering results in Fig. 13.

Fig. 11. The variance of ICop-Kmeans with two kinds of order.
basically higher than that before the combination except on the dataset of Glass. Moreover, the result obtained by Cop-SOM-E and Cop-EAC-SL combining the semi-supervised clustering components generated by the ICop-Kmeans with the produced order is better than that obtained by Cop-SOM-E and Cop-EAC-SL combining the semi-supervised clustering components generated by the ICop-Kmeans with the random order on many datasets. For the Iris dataset, the latter is better than the former in some cases though the average F-measure of ICop-Kmeans with the random order that is lower than that of ICop-Kmeans with the produced order before the combination. Compared with Cop-EAC-SL, Cop-SOM-E performs better for combining the semi-supervised clustering components generated by the ICop-Kmeans with the random order as a whole whereas the performances of the two algorithms are almost the same for combining the semi-supervised clustering components generated by the ICop-Kmeans with the produced order.
8. Conclusions

In this paper, we have improved the Cop-Kmeans algorithm by the ICop-Kmeans for solving the problem of constraint violation in the Cop-Kmeans. Aiming at the sensitivity of assignment order, we constructed the weighted co-association to get the certainty of objects and obtained a new order according to the certainty. In the final step, we presented a constrained SOM algorithm Cop-SOM and considered as a kind of consensus function to combine multiple semi-supervised clustering components generated by the improved Cop-Kmeans with the random and produced order. From the validated experiments, the proposed methods could effectively overcome disadvantages of Cop-Kmeans, and ICop-Kmeans performs better using the produced order and its performance is further enhanced using the clustering ensemble technique. The case study has shown that these clustering methods are vital in IDSS to support better decision making. We will focus on the methods of dealing with large datasets as well as the improvement of other semi-supervised clustering algorithms as our future work.

Acknowledgements

This work is partially supported by the National Science Foundation of China (Nos. 60873108, 61170111 and 61003142) and the Fundamental Research Funds for the Central Universities (No. SWJTU11ZT08).
References