Data Min Knowl Disc (2013) 26:217–254
DOI 10.1007/s10618-012-0248-z

Parameter-less co-clustering for star-structured heterogeneous data

Dino Ienco · Céline Robardet · Ruggero G. Pensa · Rosa Meo

Received: 31 March 2011 / Accepted: 4 January 2012 / Published online: 15 January 2012

Abstract The availability of data represented with multiple features coming from heterogeneous domains is getting more and more common in real world applications. Such data represent objects of a certain type, connected to other types of data, the features, so that the overall data schema forms a star structure of inter-relationships. Co-clustering these data involves the specification of many parameters, such as the number of clusters for the object dimension and for all the features domains. In this paper we present a novel co-clustering algorithm for heterogeneous star-structured data that is parameter-less. This means that it does not require either the number of row clusters or the number of column clusters for the given feature spaces. Our approach optimizes the Goodman–Kruskal's τ, a measure for cross-association in contingency tables that evaluates the strength of the relationship between two categorical variables. We extend τ to evaluate co-clustering solutions and in particular we apply it in a higher

Responsible editor: Charu Aggarwal.

D. Ienco · R. G. Pensa · R. Meo
Department of Computer Science, University of Torino, 10139 Torino, Italy
e-mail: ienco@di.unito.it

R. G. Pensa
e-mail: pensa@di.unito.it

R. Meo
e-mail: meo@di.unito.it

C. Robardet
Université de Lyon, CNRS, INSA-Lyon, LIRIS UMR5205, 69621 Villeurbanne, France
e-mail: celine.robardet@insa-lyon.fr

D. Ienco (✉)
IRSTEA Montpellier, UMR TETIS, 34093 Montpellier, France
e-mail: dino.ienco@teledetection.fr

dimensional setting. We propose the algorithm CoStar which optimizes τ by a local search approach. We assess the performance of CoStar on publicly available datasets from the textual and image domains using objective external criteria. The results show that our approach outperforms state-of-the-art methods for the co-clustering of heterogeneous data, while it remains computationally efficient.

Keywords Co-clustering · Star-structured data · Multi-view data

1 Introduction

Clustering is by far one of the most popular techniques among researchers and practitioners interested in the analysis of data from different sources. It is often employed to obtain a first schematic view of what the data looks like, since it does not require much background knowledge, usually incorporated into the class labels, nor involves pre- or post-treatment of the data. Applications of clustering techniques range over a wide variety of domains, from biology to physics, from e-commerce to social network analysis. As such, there exists a wide literature which investigates new algorithms, measures, and application domains for data clustering.

Usually, data come in the form of a set of objects described by a set of attributes that can be heterogeneous for type and underlying properties. For instance, the most common ones are numeric, categorical, boolean, textual, sequential, or networked. In the most classic settings, objects are represented by features obtained by a unique source of information or view. For instance, census data are produced by a national statistical institute, textual data are extracted from a document repository, tagged items come from a social media website, and so on.

However, in some cases, computing clusters over a single view is not sufficient to capture all the intrinsic similarities or differences between objects. Consider, for instance, an e-commerce website specialized in high-tech products, where each product is described by a general description, a technical note, and reviews are posted by customers. It is quite common to look for certain products and be unable to decide because of some incomplete technical details, uninformative descriptions or misleading reviews. The typical customer behavior is then to merge all the different sources of information, by considering them for what they are: technical notes give details on components and features (usually taken from the producer's website); general descriptions provide synthetic views of the main characteristics and advantages of the products; customers' reviews possibly complete missing details and add further useful information.

As another example consider a scientific digital library where each paper is described by its textual content (e.g., title, abstract and/or keywords), by the list of co-authors, and by a list of cited papers. If the goal is, for instance, to find the scientific communities interested in the same research field, one can try to apply a clustering algorithm over one of the feature spaces. However, intersections between communities are very common: for instance, people interested in knowledge discovery and data mining may work and publish together with people coming from the information retrieval community. Abstracts or titles on their own are not sufficiently informative: they might focus on the applicative aspects of the work, or be too generic or specific. Citations can be misleading as well: in some cases papers refer to other works dealing with similar topics but in other fields of research; sometimes papers are cited for another purpose than reporting closely related works. Therefore, clustering papers using only one feature space could lead to imprecise results. Instead, taking into account all the feature space may help in identifying more interesting and relevant clusters.

A straightforward solution consists in merging all the feature spaces into a single one, and applying a clustering algorithm on the overall table. However this solution would not always work as expected, and this for several reasons: first the feature space may contain different types of attributes; second, even though the attribute type is homogeneous, each feature may fit a specific model, show properties or be affected by certain problems such as noise, which might affect in a different form the other feature space. Third, the form of data may vary significantly from one space to another: document-keyword data are typically sparse, while categorical data are usually dense. Networked data, such as social networks, contains highly dense portions of nodes, and sparse ones. Fourth, even though the spaces do not fall into the above-mentioned cases, their cardinality may vary drastically having the consequence that larger feature spaces have more influence on the clustering process than smaller ones.

To cope with the limitation of traditional clustering algorithms on multiple views data, a new clustering formulation was introduced in 2004, the so-called *multi-view clustering* approach (Bickel and Scheffer 2004). Instead of considering a standard clustering approach over each single-view, the authors proposed an algorithm based on EM that clusters each view on the basis of the partitions of the other views. A strongly related approach is the so-called *star-structured high-order heterogeneous co-clustering* (Gao et al. 2006). Though it has never been explicitly linked to *multi-view clustering*, this family of algorithms addresses the same issues of a co-clustering approach that simultaneously clusters objects and features from the different spaces. Co-clustering methods have two main advantages: they are more effective in dealing with the well-known problem of the curse of dimensionality, and they provide an intensional description of the object clusters by associating a cluster of features to each cluster of objects. When applied on multi-view data, the several views, or feature spaces, as well as the object set, are simultaneously partitioned. A common problem of existing co-clustering approaches [as well as of other dimensionality reduction techniques such as non-negative matrix factorization Lee and Seung 2001], is that the number of clusters/components is a parameter that has to be specified before the execution of the algorithm. Choosing a correct number of clusters is not easy, and mistakes can have undesirable consequences. As mentioned in Keogh et al. (2004), incorrect parameter settings may cause an algorithm to fail in finding the true patterns and may lead the algorithm to report spurious patterns that do not really exist. Existing multi-view co-clustering methods have as many parameters as there are views in the data, plus one for the partition of objects.

In this paper we propose a new co-clustering formulation for high-order star-structured data that automatically determines the number of clusters at running time. This is obtained thanks to the employment of a multiobjective local search approach over a set of Goodman–Kruskal's τ objective functions, that will be introduced in Sect. 3.1. The local search approach aims at maximizing these functions, comparing two solutions on the basis of a Pareto-dominance relation over the objective functions.

Due to the fact that those functions have a defined upper limit that is independent of the numbers of clusters, the τ functions can be used to compare co-clusterings of different sizes. On the contrary, measures that are usually employed in co-clustering approaches, like the loss of mutual information (Dhillon et al. 2003) or more generally the Bregman divergence (Banerjee et al. 2007), are optimized for a specific partition, usually the discrete one. Moreover, the stochastic local search algorithm we proposed enables the adaptation of the number of clusters during the optimization process. Last but not least, unlike most of the existing approaches, our algorithm is designed to fit any number of views: when a single feature space is considered, the algorithm is equivalent to the co-clustering approach presented in Robardet and Feschet (2001).

The remainder of this paper is organized as follows: Sect. 2 briefly explores the state of the art in star-structured co-clustering and multi-view clustering. The theoretic fundamental details are presented in Sect. 3 while the technical issues of our approach are provided in Sect. 4. In Sect. 5 we present the results of a comprehensive set of experiments on multi-view high-dimensional text and image data, as well as a qualitative evaluation on a specific result and a scalability analysis. Finally, Sect. 6 concludes.

2 Related work

Co-clustering has been studied in many different application contexts including text mining (Dhillon et al. 2003), gene expression analysis (Cho et al. 2004; Pensa and Boulicaut 2008) and graph mining (Chakrabarti et al. 2004) where these methods have yielded an impressive improvement in performance over traditional clustering techniques. The methods differ primarily by the criterion they optimize, such as minimum loss in mutual information (Dhillon et al. 2003), sum-squared distance (Cho et al. 2004), minimum description length (MDL) (Chakrabarti et al. 2004), Bregman divergence (Banerjee et al. 2007) and non-parametric association measures (Robardet and Feschet 2001; Ienco et al. 2009). Among these approaches, only those ones based on MDL and association measure are claimed to be parameter-free (Keogh et al. 2004). However, methods based on MDL are strongly restricted by the fact they can only handle binary matrices. Association measures, such as Goodman and Kruskal τ, are internal measures of the quality of a co-clustering based on statistical considerations. They have also another advantage: they can deal with both binary and counting/frequency data (Ienco et al. 2009; Robardet and Feschet 2001). From an algorithmic point of view, the co-clustering problem has been shown to be NP-hard (Anagnostopoulos et al. 2008) when the number of row and column clusters are fixed. Therefore, proposed methods so far are based on heuristic approaches.

Several clustering and co-clustering methods for heterogeneous star-structured data have been proposed recently. Long et al. (2006) use spectral clustering to iteratively embed each type of data objects into low dimensional spaces in a way that takes advantage of the interactions among the different feature spaces. A partitional clustering algorithm like k-means is then employed to obtain the final clustering computed on the transformed spaces. In Long et al. (2007), a parametric probabilistic approach to cluster relational data is proposed. A Monte Carlo simulation method is used to learn the parameters and to assign objects to clusters. The problem of clustering images described by segments and captions is considered in Bekkerman and Jeon (2007). The proposed algorithm is based on Markov random fields in which some of the nodes are random variables in the combinatorial problem.

The co-clustering problem on star-structured data was first considered in Gao et al. (2006) where Gao et al. propose to adapt the Information Theory co-clustering approach (Dhillon et al. 2003) to star-structured data. It consists in optimizing a weighted combination of mutual information evaluated over each feature space, where weights are chosen based on the supposed reliability/relevance of their correlation. Beyond the parameters inherited from the original algorithm, the weight involved in the linear combination also has to be fixed by the end-user. Another drawback of this approach is its complexity, that prevents its use on large-scale datasets. Greco et al. (2009) propose a similar approach based on the linear combination of mutual information evaluated on each feature space, where the parameter of the linear combination is automatically determined. Ramage et al. (2009) propose a generative clustering algorithm based on latent Dirichlet allocation to cluster documents using two different sources of information: document text and tags. Each source is modeled by a probability distribution and a weight value is used to weigh one vector space with respect to the other. During the learning step, the algorithm finds the distribution parameters, and models documents, words and tags. In addition to the weight parameter, the method has another drawback: it constrains the number of hidden topics in text and tag sources to be the same, which is a strong assumption on data that is not always true. Considering the multi-view clustering problem Cleuziou et al. (2009) propose to find a consensus between the clusters from different views. Their approach merges information from each view by performing a fusion process that identifies the agreement between the views and solves the conflicts. To this purpose they formulate the problem using a fuzzy clustering approach and extend fuzzy k-means algorithm by introducing a penalty term. This term aims at reducing the disagreement between any pair of partitions coming from the different views. This approach is sensitive to the parameter setting because, apart from the number of clusters, the algorithm requires two other parameters that may influence the result. Chen et al. (2009) can be considered as the first attempt that performs multi-view co-clustering. The method is an extension of the Non-Negative Matrix Factorization approach to deal with multi-view data. It computes new word–document and document–category matrices by incorporating user provided constraints through simultaneous distance metric learning and modality selection. This method is shown to be effective, but its formulation is not flexible. In fact, in order to use more than two feature spaces, one needs to reformulate the whole process. Another problem of this approach is that the number of clusters for each feature space is given as a parameter. Furthermore, the number of parameters grows with the number of feature spaces.

3 Measuring the quality of a co-clustering on star-structured data

In this section, we introduce the association measure optimized within our co-clustering approach. Originally designed for evaluating the dependence between two categorical variables, the Goodman and Kruskal τ measure can be used to evaluate the quality of a single-view co-clustering. We generalize its definition to a multi-view setting.

3.1 Evaluating the quality of a co-clustering

Association measures are statistical measures that evaluate the strength of the link between two or more categorical variables. Considering partitions of a co-clustering as categorical variables, these measures have been shown to be well adapted to determine co-clustering with high quality (Robardet and Feschet 2001). Goodman and Kruskal τ measure (Goodman and Kruskal 1954) is one of them that estimates the association between two categorical variables X and Y by the proportional reduction of the error in predicting X knowing or not the variable Y:

$$\tau_{X|Y} = \frac{e_X - E[e_{X|Y}]}{e_X}$$

Let say that variable X has m categories $X_1, \ldots, X_m$ with probabilities $p_1, \ldots, p_m$, variable Y has n categories $Y_1, \ldots, Y_n$ with probabilities $q_1, \ldots, q_n$ and their joint probabilities are denoted r_{ij}, for $i = 1 \ldots m$ and $j = 1 \ldots n$. The error in predicting X can be evaluated by $e_X = \sum_{i=1}^{m} p_i(1 - p_i) = 1 - \sum_{i=1}^{m} p_i^2$. This is the probability that two independent observations from the marginal distribution of X fall in different categories. The error is minimum (equals to 0) when every p_i's is 0 or 1. The error is maximum when $p_i = \frac{1}{m}$ for all i. $E[e_{X|Y}]$ is the expectation of the conditional error taken with respect to the distribution of Y:

$$E[e_{X|Y}] = \sum_{j}^{n} q_j\, e_{X|Y_j} = \sum_{j}^{n} q_j \sum_{i}^{m} \frac{r_{ij}}{q_j}\left(1 - \frac{r_{ij}}{q_j}\right) = 1 - \sum_{i}^{m}\sum_{j}^{n} \frac{r_{ij}^2}{q_j}$$

The Goodman–Kruskal's $\tau_{X|Y}$ association measure is then equal to:

$$\tau_{X|Y} = \frac{\sum_i \sum_j \frac{r_{ij}^2}{q_j} - \sum_i p_i^2}{1 - \sum_i p_i^2} \tag{1}$$

Similarly, the proportional reduction of the error in predicting Y while X is known is given by:

$$\tau_{Y|X} = \frac{e_Y - E[e_{Y|X}]}{e_Y} = \frac{\sum_i \sum_j \frac{r_{ij}^2}{p_i} - \sum_j q_j^2}{1 - \sum_j q_j^2} \tag{2}$$

Let's now describe how this association measure can be employed to quantify the goodness of a co-clustering. Let $\mathcal{O}$ be a set of objects and $\mathcal{F}$ a set of features. A dataset $\mathcal{D}$ associates with an object o_u and a feature f_v a value d_{uv} that represents the frequency of feature v in the description of object u (or it could also be a Boolean value that stands for presence/absence of the feature). Figure 1a is an example of such a dataset. A co-clustering is made of one partition of objects, $CO_1, \ldots, CO_m$, and one partition of features, $CF_1, \ldots, CF_n$. To evaluate the quality of the co-clustering, we compute a contingency table that empirically estimates the joint probabilities of the two partitions and the marginal probabilities of each partition. It associates with each co-cluster (CO_i, CF_j) the value t_{ij} that aggregates the frequency values of features from cluster CF_j on objects of CO_i:

$$t_{ij} = \sum_{o_u \in CO_i} \sum_{f_v \in CF_j} d_{uv} \tag{3}$$

Figure 1b is an example of such a contingency table with two clusters of objects and two clusters of features.

In order to review the meaning of τ for the evaluation of a co-clustering consider the table in Fig. 1b whose cell at the intersection of the row CO_i with the column CF_j contains the frequency of objects in cluster CO_i having the features in cluster CF_j. T_{O_i} is the total counting for cluster CO_i, T_{F_j} is the sum of counts for cluster CF_j, and T is the global total, i.e., $T_{O_i} = \sum_j t_{ij}$, $T_{F_j} = \sum_i t_{ij}$ and $T = \sum_i \sum_j t_{ij}$. Therefore, the probability that an object o_u is in cluster CO_i may be estimated by $p_i = \frac{T_{O_i}}{T}$, the relative frequency of the cluster. Similarly, the probability that a feature f_v is in cluster CF_j is estimated by $q_j = \frac{T_{F_j}}{T}$. Finally, the joint probability that an object o_u and a feature f_v are in co-cluster (CO_i, CF_j) is $r_{ij} = \frac{t_{ij}}{T}$. Such a statistical estimate is represented by a two-dimensional contingency table or a two-way frequency table. We denote by X the random variable associated with the partition of objects, and by Y the random variable associated with the partition of features. $\tau_{X|Y}$ is the proportional reduction in prediction error of partition $\{CO_1, \ldots, CO_m\}$ given partition $\{CF_1, \ldots, CF_n\}$.

Example 1 Consider the dataset in Fig. 2a. It could represent one of the possible views on a document collection: the view on the document terms or the view on the document tags that users adopted to annotate documents in a social network. Suppose the view is on the terms contained in the documents: each value in the matrix represents the number of occurrences of a specific word (indicated by the matrix column) in a specific document (indicated by the matrix row). We take into account two possible co-clusterings for this dataset:

	f_1	f_2	f_3
o_1	d_{11}	d_{12}	d_{13}
o_2	d_{21}	d_{22}	d_{23}
o_3	d_{31}	d_{32}	d_{33}
o_4	d_{41}	d_{42}	d_{43}

(a)

	CF_1	CF_2	
CO_1	t_{11}	t_{12}	T_{O_1}
CO_2	t_{21}	t_{22}	T_{O_2}
	T_{F_1}	T_{F_2}	T

(b)

Fig. 1 An example dataset (**a**) and the contingency table associated with a related co-clustering (**b**)

	f_1	f_2	f_3	f_4
o_1	3	4	1	1
o_2	5	3	0	2
o_3	6	4	1	0
o_4	0	1	7	7
o_5	1	0	6	8

(a)

	CF_1	CF_2	
CO_1	25	5	30
CO_2	2	28	30
	27	33	60

(b)

	CF_1	CF_2	
CO_1	15	4	19
CO_2	12	29	41
	27	33	60

(c)

Fig. 2 An example dataset (**a**) and the contingency tables that correspond to C_1 (**b**) and C_2 (**c**)

$$C_1 = \{\{o_1, o_2, o_3\}, \{o_4, o_5\}\}, \{\{f_1, f_2\}, \{f_3, f_4\}\} \quad \text{and}$$
$$C_2 = \{\{o_1, o_2\}, \{o_3, o_4, o_5\}\}, \{\{f_1, f_2\}, \{f_3, f_4\}\}$$

Tables in Fig. 2b, c represent the contingency tables for the first and second co-clustering respectively. The Goodman–Kruskal $\tau_{X|Y}$ and $\tau_{Y|X}$ for the first co-clustering are:

$$\tau_{X|Y} = \tau_{Y|X} = 0.5937$$

and for the second one are:

$$\tau_{X|Y} = 0.2158 \quad \text{and} \quad \tau_{Y|X} = 0.3375$$

We can observe that the first co-clustering better associates clusters of objects with clusters of features than the second one. In the first co-clustering, objects of CO_1 have mostly features of cluster CF_1, whereas objects of CO_2 are characterized by features of CF_2. In the second example, given an object having high frequency values on features of CF_1, it is difficult to predict if it belongs to CO_1 or CO_2. The τ measure corroborates these facts and in fact it has higher values for the first co-clustering.

Analyzing the properties of τ, we can observe that it satisfies many desirable properties of a co-clustering measure. First, it is invariant by rows and columns permutation. Second, it takes values between [0, 1]: (1) $\tau_{X|Y}$ is 0 iff knowledge of Y classification is of no help in predicting the X classification, i.e. there is independence between the object and feature clusters; (2) $\tau_{X|Y}$ is 1 iff knowledge of an individual's Y cluster completely specifies his X class, i.e. each row of the contingency table contains at most one non zero cell. Third, it has an operational meaning: given an object, it is the relative reduction in the prediction error of the object's cluster given the partition of features, in a way that preserves the category distribution (Goodman and Kruskal 1954). Finally, unlike many other association measures such as χ^2, τ has a defined upper limit that is independent of the numbers of classes m and n. Therefore, τ can be used to compare co-clusterings of different sizes.

3.2 The star schema

In the context of high-order star-structured co-clustering, the same set of objects is represented in different feature spaces. Such data represent objects of a certain type, connected to other types of data, the features, so that the overall data schema forms a star structure of inter-relationships. The co-clustering task consists in clustering simultaneously the set of objects and the set of values in the different feature spaces. In this way we obtain a partition of the objects influenced by each of the feature spaces and at the same time a partition of each feature space. Considering N feature spaces that have to be clustered, in this paper we study a way to extend the τ measure to perform co-clustering of the data when the number of the involved dimensions is high.

In Fig. 3 we show an example of a star-structured data schema. The set of objects is shown at the center of the star. For example it could be a set of documents. The set of objects is then described with multiple views by more features (*Feature1*, *Feature2*, *Feature3*). For instance, *Feature1* could be the set of the vocabulary words and the values d^1_{ij} represent the number of occurrences of the word f^1_j in the document Oi; *Feature2* could be the set of tags adopted by the users of a social community in their annotations and the values d^2_{ij} represent the number of users that used the tag f^2_j in the document Oi; *Feature3* could be the set of authors who cited documents in the given set and the feature values represent the number of citations. For the complete description of each object we have to make reference to the different feature spaces. The values of each feature have their own distributions in the document set and are described by the different contingency tables. At this point both the document set and the feature spaces might be partitioned into co-clusters where all the dimensions contribute to determine the partitions.

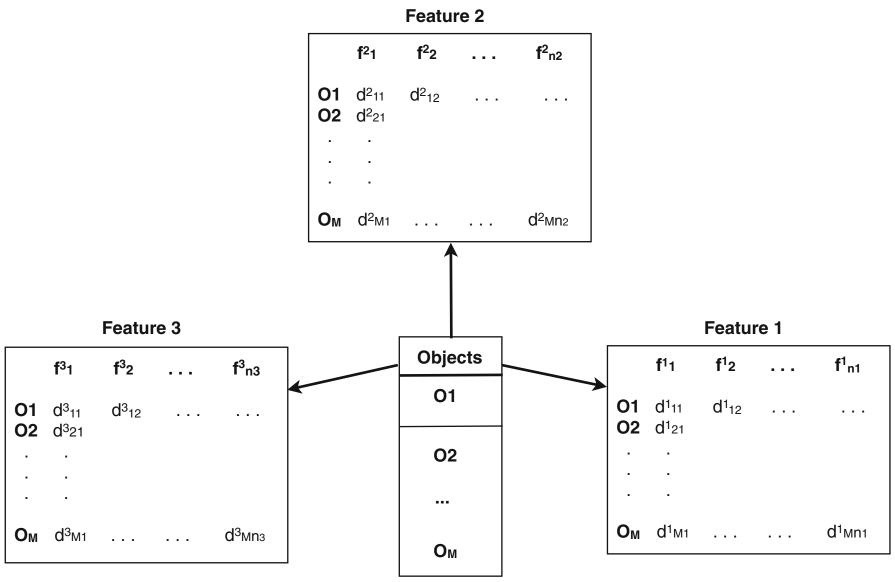


Fig. 3 The data star-schema

3.3 Evaluating the quality of a star-structured co-clustering

Evaluating the quality of the partition of objects, given the partitions of features, is formalized as follows. The partition of objects is associated with the dependent variable X, and the N partitions of the feature spaces are considered as many independent variables $\mathcal{Y} = \{Y^1, \ldots, Y^N\}$. Each variable $Y^k \in \mathcal{Y}$ has n_k categories $Y_1^k, \ldots, Y_{n_k}^k$ with probabilities $q_1^k, \ldots, q_{n_k}^k$ and X has m categories $X_1, \ldots, X_m$. However, for each variable Y^k, the m categories of X have different probabilities $p_1^k, \ldots, p_m^k$, $k = 1 \ldots N$. The joint probabilities between X and any $Y^k \in \mathcal{Y}$ are denoted by r_{ij}^k, for $i = 1 \ldots m$ and $j = 1 \ldots n_k$. The error in predicting X is the sum of the errors over the independent variables of $\mathcal{Y}$: $e_X = \sum_{k=1}^N \sum_{i=1}^m p_i^k(1 - p_i^k) = N - \sum_{k=1}^N \sum_{i=1}^m (p_i^k)^2$. $E[e_{X|\mathcal{Y}}]$ is the expectation of the conditional error taken with respect to the distributions of all $Y^k \in \mathcal{Y}$:

$$E[e_{X|\mathcal{Y}}] = \sum_k^N \sum_j^{n_k} q_j^k \, e_{X|Y_j^k} = \sum_k^N \sum_j^{n_k} q_j^k \sum_i^m \frac{r_{ij}^k}{q_j^k}\left(1 - \frac{r_{ij}^k}{q_j^k}\right)$$

$$= N - \sum_k^N \sum_i^m \sum_j^{n^k} \frac{\left(r^k{}_{ij}\right)^2}{q_j^k}$$

The generalized Goodman–Kruskal's $\tau_{X|\mathcal{Y}}$ association measure is then equal to:

$$\tau_{X|\mathcal{Y}} = \frac{\sum_k \sum_i \sum_j \frac{\left(r^k{}_{ij}\right)^2}{q_j^k} - \sum_k \sum_i \left(p_i^k\right)^2}{N - \sum_k \sum_i \left(p_i^k\right)^2} \tag{4}$$

If we consider Y^k as a dependent variable, and X as an independent variable, the corresponding $\tau_{Y^k|X}$ is computed as described in the previous section (see Eq. 2), i.e.:

$$\tau_{Y^k|X} = \frac{e_{Y^k} - E[e_{Y^k|X}]}{e_{Y^k}} = \frac{\sum_i \sum_j \frac{\left(r_{ij}^k\right)^2}{p_i^k} - \sum_j \left(q_j^k\right)^2}{1 - \sum_j \left(q_j^k\right)^2} \tag{5}$$

We now describe how the generalized association measure can be employed for computing the quality of a high-order star-structured co-clustering. Let $\mathcal{F} = \{F^1, \ldots, F^N\}$ be a set of N feature spaces. A dataset can be viewed under the different views given by the different feature spaces F^k. Therefore, the view D^k is associated with each feature space F^k. We call *star-structured dataset*, denoted $\mathcal{D} = \{D^1, \ldots, D^N\}$, the collection of matrices built over $\mathcal{O}$ and $\mathcal{F}$. The joint tables in Fig. 4b are the contingency tables representing a co-clustering of the multi-space dataset of Fig. 4a. Symbol CO_i represents ith cluster on objects, while CF_j^k represents jth cluster on the kth feature space (e.g. $(CO_1 = \{o_1, o_2, o_5\}, CO_2 = \{o_3, o_4\})$, $\left(CF_1^1 = \{f_1^1, f_2^1\},\ CF_2^1 = \{f_3^1, f_4^1\}\right)$ and $\left(CF_1^2 = \{f_1^2\}, CF_2^2 = \{f_2^2, f_3^2\}\right)$. The

Fig. 4 An example of a star-structured dataset (**a**) and the contingency table associated with a related star-structured co-clustering (**b**)

	f_1^1	f_2^1	f_3^1	f_4^1	f_1^2	f_2^2	f_3^2
o_1	d_{11}^1	d_{12}^1	d_{13}^1	d_{14}^1	d_{11}^2	d_{12}^2	d_{13}^2
o_2	d_{21}^1	d_{22}^1	d_{23}^1	d_{24}^1	d_{21}^2	d_{22}^2	d_{23}^2
o_3	d_{31}^1	d_{32}^1	d_{33}^1	d_{34}^1	d_{31}^2	d_{32}^2	d_{33}^2
o_4	d_{41}^1	d_{42}^1	d_{43}^1	d_{44}^1	d_{41}^2	d_{42}^2	d_{43}^2
o_5	d_{51}^1	d_{52}^1	d_{53}^1	d_{54}^1	d_{51}^2	d_{52}^2	d_{53}^2

(a)

	CF_1^1	CF_2^1		CF_1^2	CF_2^2	
CO_1	t_{11}^1	t_{12}^1	$T_{O_1}^1$	t_{11}^2	t_{12}^2	$T_{O_1}^2$
CO_2	t_{21}^1	t_{22}^1	$T_{O_2}^1$	t_{21}^2	t_{22}^2	$T_{O_2}^2$
	$T_{F_1^1}$	$T_{F_2^1}$	T^1	$T_{F_1^2}$	$T_{F_2^2}$	T^2

(b)

association of co-cluster (CO_i, CF_j^k) is represented by the value t_{ij}^k, at the intersection of the ith object cluster and the jth cluster of the kth feature space, and is equal to:

$$t_{ij}^k = \sum_{o_u \in CO_i} \sum_{f_v^k \in CF_j^k} d_{uv}^k \tag{6}$$

$T_{O_i}^k$ is the total counting for cluster CO_i within the kth feature space. $T_{F_j^k}$ is the total counting for the jth cluster of the kth feature space, and T^k is the global total for the kth feature space, i.e., $T_{O_i}^k = \sum_j t_{ij}^k$, $T_{F_j^k} = \sum_i t_{ij}^k$ and $T^k = \sum_i \sum_j t_{ij}^k$. Therefore, given a feature space k, the probability that an object o_u is in cluster CO_i is $p_i^k = \frac{T_{O_i}^k}{T^k}$. Similarly, the probability that a feature f_v^k is in cluster CF_j^k is $q_j^k = \frac{T_{F_j^k}}{T^k}$. Finally, the joint probability that an object o_u and a feature f_v^k are in co-cluster (CO_i, CF_j^k) is $r_{ij}^k = \frac{t_{ij}^k}{T^k}$.

Let X be the random variable associated with the partition on objects: its values range over all the possible partitions of $\mathcal{O}$. Let $\mathcal{Y}$ be the set of N random variables $Y^k, k = 1 \ldots N$, associated with the feature partitions. Each random variable Y^k has values that range over all the possible partitions of F^k. Figure 4b shows that a possible value for X is the partition $\{CO_1, CO_2\}$ and the values of the variables Y^1 and Y^2 are respectively the partitions $\{CF_1^1, CF_2^1\}$ and $\{CF_1^2, CF_2^2\}$.

$\tau_{X|\mathcal{Y}}$ is the proportional reduction in the prediction error of the object partition when the values of the feature partitions are known. The best co-clustering solution will be the one that reduces the most the prediction error of the object partition when the object values on the feature partitions are known. As we will see, also the opposite prediction problem will be considered in our approach, by evaluation of $\tau_{\mathcal{Y}|X}$: it considers the reduction in the prediction error of the feature partitions when the object partition is known.

Example 2 Example in Fig. 4a has been instantiated in the dataset of Fig. 5a: it could represent a collection of documents. Suppose the first feature space is the space of the terms occurring in the documents, while the second feature space is the space of the tags associated by the users of a social network. Each counting in the first feature space ($f_1^1 \dots f_4^1$) represents, for instance, the occurrences of a specific word in a specific document, while the counting in the second feature space ($f_1^2 \dots f_3^2$) are the number of users who associated a specific tag with a specific document. We take into account two possible co-clusterings for this dataset:

$$\begin{aligned} C_1 &= \{CO_1 = \{o_1, o_2, o_3\}, CO_2 = \{o_4, o_5\}\} \times \\ &\quad \{\{CF_1^1 = \{f_1^1, f_2^1\}, CF_2^1 = \{f_3^1, f_4^1\}\} \cup \{CF_1^2 = \{f_1^2\}, CF_2^2 = \{f_2^2, f_3^2\}\}\} \\ C_2 &= \{CO_3 = \{o_1, o_3, o_5\}, CO_4 = \{o_2, o_4\}\} \times \\ &\quad \{\{CF_3^1 = \{f_1^1, f_3^1\}, CF_4^1 = \{f_2^1, f_4^1\}\} \cup \{CF_1^2 = \{f_1^2\}, CF_2^2 = \{f_2^2, f_3^2\}\}\} \end{aligned}$$

Tables in Fig. 5b, c represent the contingency tables for C_1 and C_2 respectively. For the first co-clustering, the Goodman–Kruskal $\tau_{X|\mathcal{Y}}$ and $\tau_{Y^k|X}$ measures are:

$$\tau_{X|\mathcal{Y}} = 0.6390, \quad \tau_{Y^1|X} = 0.5937, \quad \tau_{Y^2|X} = 0.6890$$

τ values have been computed respectively on the contingency tables:

$$\begin{aligned} &\{CO_1, CO_2\} \times \{CF_1^1, CF_2^1, CF_1^2, CF_2^2\} \\ &\{CF_1^1, CF_2^1\} \times \{CO_1, CO_2\} \\ &\{CF_1^2, CF_2^2\} \times \{CO_1, CO_2\} \end{aligned}$$

Fig. 5 An example dataset (**a**) and two contingency tables associated with the star-structured co-clusterings C_1 (**b**) and C_2 (**c**)

	f_1^1	f_2^1	f_3^1	f_4^1	f_1^2	f_2^2	f_3^2
o_1	3	4	1	1	0	8	5
o_2	5	3	0	2	0	6	9
o_3	6	4	1	0	2	2	2
o_4	0	1	7	7	9	1	0
o_5	1	0	6	8	7	0	1

(a)

	CF_1^1	CF_2^1		CF_1^2	CF_2^2	
CO_1	25	5	30	2	32	34
CO_2	2	28	30	16	2	18
	27	33	60	18	34	52

(b)

	CF_1^1	CF_2^1		CF_1^2	CF_2^2	
CO_1	18	17	35	9	18	27
CO_2	12	13	25	9	16	25
	30	30	60	18	34	52

(c)

For the second co-clustering, they are:

$$\tau_{X|\mathcal{Y}} = 0.0119, \quad \tau_{Y^1|X} = 0.0234, \quad \tau_{Y^2|X} = 0.0008$$

Since all the three τ values for the first co-clustering are greater than the second ones, we can conclude that the first co-clustering better captures the interactions between objects and features of each space.

4 A stochastic local search approach for high-order star-structured co-clustering

We formulate our co-clustering approach for star-structured data as a multi-objective combinatorial optimization problem (Jaszkiewicz 2002) which aims at optimizing $N+1$ objective functions based on Goodman–Kruskal's τ measure. Given a star-structured dataset $\mathcal{D}$ over $\mathcal{O}$ and $\mathcal{F}$, the goal of the star-structured data co-clustering is to find a set of partitions $\mathcal{Y} = \{Y^1, \ldots, Y^k, \ldots, Y^N\}$ over the feature set $\mathcal{F} = \{F^1, \ldots, F^k, \ldots, F^N\}$, and a partition X of the object set $\mathcal{O}$ such that

$$\max_{\mathcal{Y}\cup X \in \mathcal{P}} \tau(\mathcal{Y} \cup X) = \left(\tau_{Y^1|X}, \ldots, \tau_{Y^k|X}, \ldots, \tau_{Y^N|X}, \tau_{X|\mathcal{Y}}\right) \tag{7}$$

where $\mathcal{P}$ is the discrete set of candidate partitions and τ is a function from $\mathcal{P}$ to $\mathbf{Z}$, where $\mathbf{Z} = [0, 1]^{N+1}$ is the set of vectors in the objective space, *i.e.*, $z_i = \tau_{Y^i|X}$ (see Eq. 5), $\forall i = 1, \ldots, N$ and $z_{N+1} = \tau_{X|\mathcal{Y}}$ (see Eq. 4).

To compare different candidate partitions of $\mathcal{P}$, one can either use a scalarization function, that maps $\mathbf{Z}$ into $\mathbb{R}$ (such as the weighted sum), or one can employ a dominance-based approach, that induces a partial order over $\mathcal{P}$. As for combinatorial problem there may exist many truly optimal partitions that are not optimal for any weighted sum scalarization function (Liefooghe et al. 2011), we consider in the following a Pareto-dominance approach. The goal is to identify optimal partitions $\mathcal{P}_{opt} \subset \mathcal{P}$, where optimality consists in the fact that no solution in $\mathcal{P} \setminus \mathcal{P}_{opt}$ is superior to any solution of $\mathcal{P}_{opt}$ on every objective function. This set of solutions is known as Pareto optimal set or non-dominated set. These concepts are formally defined below.

Definition 1 (*Pareto-dominance*) An objective vector $\mathbf{z} \in \mathbf{Z}$ is said to *dominate* an objective vector $\mathbf{z}' \in \mathbf{Z}$ iff $\forall i \in \{1, \ldots, N+1\}$, $z_i \geq z'_i$ and $\exists j \in \{1, \ldots, N+1\}$, such that $z_j > z'_j$. This relation is denoted $\mathbf{z} \succ \mathbf{z}'$ hereafter.

Definition 2 (*Non-dominated objective vector and Pareto optimal solution*) An objective vector $\mathbf{z} \in \mathbf{Z}$ is said to be *non-dominated* iff there does not exist another objective vector $\mathbf{z}' \in \mathbf{Z}$ such that $\mathbf{z}' \succ \mathbf{z}$.

A solution $p \in \mathcal{P}$ is said to be *Pareto optimal* iff its mapping in the objective space ($\tau(p)$) results in a non-dominated vector.

Thus, the star-structured data co-clustering problem is to seek for a Pareto optimal solution of Eq. 7. As this problem is more difficult than the co-clustering problem with fixed numbers of object and feature clusters, which is known to be NP-hard (Anagnostopoulos et al. 2008), we propose, in Sect. 4.1, a Stochastic Local Search algorithm CoStar to solve it. This algorithm searches for a near optimal solution given a local knowledge provided by the definition of a neighborhood (Paquete 2006). We demonstrate, in Sect. 4.2, that CoStar outputs a Pareto local optimum solution of the problem. In Sect. 4.3, we propose a way to evaluate a neighbor solution incrementally. Finally, Sect. 4.4 gives the overall complexity of CoStar and of its competitors.

4.1 Multiple feature space co-clustering algorithm

We build our algorithm on the basis of $\tau CoClust$ (Robardet and Feschet 2001), whose goal is to find a partition X of objects and a partition Y of features such that Goodman–Kruskal's $\tau_{X|Y}$ and $\tau_{Y|X}$ are maximized. The algorithm locally optimizes the two coefficients by updating step by step the partitions. The basic updating procedure consists in running through the neighborhood of each partition and choose the partition that increases the most the measures. This neighborhood is made of partitions where a peculiar element is moved from one cluster to another. Let us first consider the improvement of partition X. After randomly picking up a cluster and randomly picking up an object in that cluster, the procedure assigns this object to the cluster (possibly an empty one) that mostly increases $\tau_{X|Y}$. In a following phase, it considers the improvement of the features partition Y: it randomly picks up a feature from a randomly obtained cluster and assigns this feature to the cluster that mostly increases $\tau_{Y|X}$. Starting from the discrete partitions, the algorithm alternatively updates the partitions X and Y until a convergence criterion is satisfied. Unlike other co-clustering approaches, that require the number of co-clusters as parameter, $\tau CoClust$ is able to automatically determine the most appropriate numbers of clusters for X and Y. This is due to two characteristics: (1) the τ coefficient has a defined upper limit that is independent of the numbers of clusters (while other measures, such as the loss in mutual information Dhillon et al. (2003) have not) and thus enables comparison of co-clusterings of different sizes, and (2) the updating procedure can create or delete clusters at any time during the iterative process.

Let us now present algorithm CoStar as an extension of $\tau CoClust$ algorithm to multi-view co-clustering. CoStar is shown in Algorithm 1. It takes in input the dataset D and alternatively optimizes the partition of the objects, working on the random variable X, and each partition over each feature set F^k, working on the random variables Y^k, until a number of iterations N_{iter} is reached.

CoStar first initializes the partitions $Y^1, \ldots, Y^N$ and X with discrete partitions (that is, partitions with a single element per cluster) and then computes the contingency tables T^k between the partition X and alternatively each partition Y^k over the feature set F^k. The function ContingencyTable computes Eq. 6 (initialization step at lines 4–7). Then, from line 8 to 14, it performs a stochastic local search method by alternatively optimizing partition X (line 9) and each partition Y^k (line 11). The candidate partitions considered by the local search algorithm belong to the neighborhood of $\mathcal{Y} \cup X$ as defined in Eq. 8. Function OptimizeMultiObjectCluster substitutes the object partition X for the partition in the neighborhood of X that mostly increases $\tau_{X|\mathcal{Y}}$

Algorithm 1 CoStar(D, N_{iter})

```
1: Initialize $Y^1, \ldots, Y^N, X$ with discrete partitions
2: $i \leftarrow 0$
3: $T \leftarrow \emptyset$
4: for $k = 1$ to $N$ do
5:     $T^k \leftarrow$ ContingencyTable($X, Y^k, D^k$)
6:     $T \leftarrow T \bigcup T^k$
7: end for
8: while ($i \leq N_{iter}$) do
9:     $[X, T] \leftarrow$ OptimizeMultiObjectCluster($X, \mathcal{Y}, T$)
10:    for $k = 1$ to $N$ do
11:        $[Y^k, T^k] \leftarrow$ OptimizeFeatureCluster($X, Y^k, T^k$)
12:    end for
13:    $i \leftarrow i + 1$
14: end while
15: return $Y^1, \ldots, Y^N, X$
```

(see Algorithm 2). This measure takes into account all the partitions $Y^k \in \mathcal{Y}$ and their associated contingency tables $T = \bigcup_k T^k$. Function OptimizeFeatureCluster substitutes the feature partition Y^k for the partition in the neighborhood of Y^k that mostly increases $\tau_{Y^k|X}$ (see Algorithm 4). These operations are alternated and repeated until a stopping condition is satisfied. This condition can be based on a convergence criterion of τ, but, for simplicity, we bound the number of iterations by N_{iter} in the current version of the algorithm.

Algorithm 2 OptimizeMultiObjectCluster($X, \mathcal{Y}, T$)

```
1: Randomly choose a cluster $x_b \in X$
2: Randomly choose an object $o \in x_b$
3: $\mathcal{ND} \leftarrow \{x_b\}$
4: $max_{\tau_{X|\mathcal{Y}}} \leftarrow \tau_{X|\mathcal{Y}}(T)$
5: for all $x_e \in \{X \cup \emptyset\}$ $s.t.$ $x_b \neq x_e$ do
6:     $[X', T'] \leftarrow$ Update($X, T, o, x_b, x_e$)
7:     if ($\tau_{X'|\mathcal{Y}}(T') > max_{\tau_{X|\mathcal{Y}}}$) then
8:         $max_{\tau_{X|\mathcal{Y}}} \leftarrow \tau_{X'|\mathcal{Y}}(T')$
9:         $\mathcal{ND} \leftarrow \{x_e\}$
10:    else
11:        if ($\tau_{X'|\mathcal{Y}}(T') = max_{\tau_{X|\mathcal{Y}}}$) then
12:            $\mathcal{ND} \leftarrow \mathcal{ND} \cup \{x_e\}$
13:        end if
14:    end if
15: end for
16: $max_X, max_T \leftarrow$ FindParetoLocalOptimumObject($\mathcal{ND}, X, T, x_b, o$)
17: return $max_X, max_T$
```

The stochastic local optimization of the partition X works as follows (see Algorithm 2). The algorithm starts by randomly picking up a cluster x_b of X and an element o of x_b (lines 1 and 2). Then, all the partitions of the neighborhood of X, constituted by the partitions where o is moved to another cluster of X (possibly the empty one), are considered. To retrieve the partition $\max_X$ that increases at most $\tau_{X|\mathcal{Y}}$

Algorithm 3 FINDPARETOLOCALOPTIMUMOBJECT($\mathcal{ND}, X, T, x_b, o$)

```
1: while $Card(\mathcal{ND}) > 1$ do
2:   Take $x_i, x_j \in \mathcal{ND}\ s.t.\ x_i \neq x_j$
3:   $nb^+_{x_i} \leftarrow 0$
4:   $nb^+_{x_j} \leftarrow 0$
5:   $[X_i, T_i] \leftarrow$ UPDATE($X, T, o, x_b, x_i$)
6:   $[X_j, T_j] \leftarrow$ UPDATE($X, T, o, x_b, x_j$)
7:   for $k = 1$ to $N$ do
8:     if $(\tau_{Y^k|X_i}(T_i) > \tau_{Y^k|X_j}(T_j))$ then
9:       $nb^+_{x_i} \leftarrow nb^+_{x_i} + 1$
10:    else
11:      if $(\tau_{Y^k|X_i}(T_i) < \tau_{Y^k|X_j}(T_j))$ then
12:        $nb^+_{x_j} \leftarrow nb^+_{x_j} + 1$
13:      end if
14:    end if
15:  end for
16:  if $nb^+_{x_i} \geq nb^+_{x_j}$ then
17:    $\mathcal{ND} \leftarrow \mathcal{ND} \setminus \{x_j\}$
18:  else
19:    $\mathcal{ND} \leftarrow \mathcal{ND} \setminus \{x_i\}$
20:  end if
21: end while
22: return $[X_i, T_i] \leftarrow$ UPDATE($X, T, o, x_b, x_i$), with $x_i \in \mathcal{ND}$
```

among the partitions in the neighborhood of X and that is not dominated by any partition of this neighborhood, the algorithm uses a queue, $\mathcal{ND}$, that gathers the possibly non-dominated partitions of this neighborhood. Each partition of the neighborhood of X is identified by the cluster to which o belongs. Thus, the queue $\mathcal{ND}$ is initialized with x_b that represents the partition X (line 3). $\max_{\tau_{X|\mathcal{Y}}}$ is initialized with the current $\tau_{X|\mathcal{Y}}$ value. The loop from line 5 to 15 considers all the neighboring partitions of X (x_e being the cluster where o is moved to, $x_e \in X \cup \emptyset$). The function UPDATE (line 6) is called to modify the contingency tables $T = \bigcup_k T^k$ and the partition X with respect to the new cluster of o. To perform this step, it first moves the element o from the cluster x_b to cluster x_e and then performs one or more of the following three actions:

1. if cluster x_b is empty after removing o, it deletes cluster x_b and updates the contingency tables T' consequently;
2. if cluster x_e is the empty cluster, it adds a new cluster to the partition and updates the contingency tables T' consequently;
3. if the two above mentioned cases do not apply, it simply updates the content of T' by modifying x_b and x_e rows.

Each modification of T' consists actually in modifying the N contingency tables $T^k,\ k = 1, \ldots, N$.

From line 7 to 14, potentially non-dominated partitions of the neighborhood of X are stored in $\mathcal{ND}$. $\tau_{X'|\mathcal{Y}}$ is evaluated using the contingency tables T'. If it is strictly greater than $max_{\tau_{X|\mathcal{Y}}}$ then, given a partition of $\mathcal{ND}$, either X' dominates it or neither of the partitions dominates the other (see proof of Theorem 1). Therefore, $\mathcal{ND}$ and $max_{\tau_{X|\mathcal{Y}}}$ are reinitialized respectively with X' and $\tau_{X'|\mathcal{Y}}$. If $\tau_{X'|\mathcal{Y}}$ is equal to $max_{\tau_{X|\mathcal{Y}}}$

then x_e is added to the queue $\mathcal{ND}$. Considering X' and any partition X of $\mathcal{ND}$, all the Pareto dominance relation may exist between these two partitions: X' may dominate X, X' may be dominated by X or neither may dominate the other. Therefore, a partition that is not dominated by any partition of $\mathcal{ND}$ will be identified by the function FindParetoLocalOptimumObject.

When all the neighborhood has been explored, the function FindParetoLocal OptimumObject (see Algorithm 3) is called. It processes the queue $\mathcal{ND}$ in order to retrieve a partition non dominated by any other partition of $\mathcal{ND}$. To that end, the function compares two partitions X_i and X_j of $\mathcal{ND}$ by counting in $nb^+_{x_i}$ the number of times $\tau_{Y^k|X_i}$ is strictly greater than $\tau_{Y^k|X_j}$, and in $nb^+_{x_j}$ the number of times $\tau_{Y^k|X_j}$ is strictly greater than $\tau_{Y^k|X_i}$, for $k = 1, \ldots, N$. If $nb^+_{x_i} \geq nb^+_{x_j}$ then X_j is either dominated by X_i or neither X_i nor X_j dominate the other (see proof of Theorem 1). Thus, the function removes X_j from $\mathcal{ND}$ and continues until there is only one partition in $\mathcal{ND}$.

Algorithm 4 OptimizeFeatureCluster(X, Y^k, T^k)

```
1: Randomly choose a cluster $y_b \in Y^k$
2: Randomly choose a feature $f \in y_b$
3: $\mathcal{ND} \leftarrow \{y_b\}$
4: $max_{\tau_{Y^k|X}} \leftarrow \tau_{Y^k|X}(T^k)$
5: for all $y_e \in \{Y^k \cup \emptyset\}$ $s.t.$ $y_b \neq y_e$ do
6:     $[Y^{k\prime}, T^{k\prime}] \leftarrow$ Update($Y^k$, $T^k$, $f$, $y_b$, $y_e$)
7:     if ($\tau_{Y^{k\prime}|X}(T^{k\prime}) > max_{\tau_{Y^k|X}}$) then
8:         $max_{\tau_{Y^k|X}} \leftarrow \tau_{Y^{k\prime}|X}(T^{k\prime})$
9:         $\mathcal{ND} \leftarrow \{y_e\}$
10:    else
11:        if ($\tau_{Y^{k\prime}|X}(T^{k\prime}) = max_{\tau_{Y^k|X}}$) then
12:            $\mathcal{ND} \leftarrow \mathcal{ND} \cup \{y_e\}$
13:        end if
14:    end if
15: end for
16: $max_{Y^k}, max_{T^k} \leftarrow$ FindParetoLocalOptimumFeature($\mathcal{ND}$, $Y^k$, $T^k$, $y_b$, $f$)
17: return $max_{Y^k}, max_{T^k}$
```

Algorithms 4 and 5 optimize partition Y^k and work in a similar way than Algorithms 2 and 3. However, it is simpler because (1) modifying partition Y^k only impacts the contingency table $T^{k\prime}$, and (2), in Algorithm 5, the comparison of two partitions of the queue $\mathcal{ND}$ is only done on the basis of $\tau_{X|\mathcal{Y}}$ value, the others measures ($\tau_{Y^k|X}$, $k = 1, \ldots N$) being equal.

4.2 Local convergence of CoStar

Considering iterated local search algorithms for multi-objective optimization requires to extend the usual definitions of local optimum to that context (Paquete and Stützle 2006). Let $\mathcal{N} : \mathcal{P} \rightarrow 2^{\mathcal{P}}$ be a neighborhood structure that associates to every solution

Algorithm 5 FindParetoLocalOptimumFeature($\mathcal{ND}$, Y^k, T^k, y_b, f)

```
1: while $Card(\mathcal{ND}) > 1$ do
2:     Take $y_i, y_j \in \mathcal{ND}\ s.t.\ y_i \neq y_j$
3:     $nb^+_{y_i} \leftarrow 0$
4:     $nb^+_{y_j} \leftarrow 0$
5:     $[Y^k_i, T^k_i] \leftarrow$ Update($Y^k, T^k, f, y_b, y_i$)
6:     $[X^k_j, T^k_j] \leftarrow$ Update($Y^k, T^k, f, y_b, y_j$)
7:     $\mathcal{Y}_i \leftarrow \mathcal{Y} \setminus Y^k \cup Y^k_i$
8:     $\mathcal{Y}_j \leftarrow \mathcal{Y} \setminus Y^k \cup Y^k_j$
9:     if $(\tau_{X|\mathcal{Y}_i}(T^k_i) > \tau_{X|\mathcal{Y}_j}(T^k_j))$ then
10:        $nb^+_{y_i} \leftarrow nb^+_{y_i} + 1$
11:    else
12:        if $(\tau_{X|\mathcal{Y}_i}(T^k_i) < \tau_{X|\mathcal{Y}_j}(T^k_j))$ then
13:            $nb^+_{y_j} \leftarrow nb^+_{y_j} + 1$
14:        end if
15:    end if
16:    if $nb^+_{y_i} \geq nb^+_{y_j}$ then
17:        $\mathcal{ND} \leftarrow \mathcal{ND} \setminus \{y_j\}$
18:    else
19:        $\mathcal{ND} \leftarrow \mathcal{ND} \setminus \{y_i\}$
20:    end if
21: end while
22: return $[Y^k_i, T^k_i] \leftarrow$ Update($Y^k, T^k, f, y_b, y_i$), with $y_i \in \mathcal{ND}$
```

$p \in \mathcal{P}$ a set of neighboring solutions $\mathcal{N}(p) \subseteq \mathcal{P}$. We define a Pareto local optimum with respect to $\mathcal{N}$ as follows.

Definition 3 (*Pareto local optimum*) A solution $p \in \mathcal{P}$ is a Pareto local optimum with respect to the set of solutions $\mathcal{N}(p)$ that constitutes the neighborhood of p, if and only if there is no $r \in \mathcal{N}(p)$ such that $\tau(r) \succ \tau(p)$ (see Eq. 7).

In algorithm CoStar, the neighborhood function is defined as follows. Let $p = \mathcal{Y} \cup X$ be an element of $\mathcal{P}$. W^i denotes either the partition Y^i, if $i \in \{1, \ldots, N\}$, or the partition X, if $i = N + 1$. Let w_b be a cluster of W^i and let u be an element of w_b. The neighborhood $\mathcal{N}$ of p is determined by the movement of the element u from the cluster w_b to the cluster w_e in W^i. It is given by the substitution in p of W^i by $W^{i\prime}$:

$$\mathcal{N}_{(i,w_b,u)}(\mathcal{Y} \cup X) = \{\mathcal{Y} \cup X \setminus W^i \cup W^{i\prime}\} \tag{8}$$

with $W^{i\prime} = W^i \setminus w_b \setminus w_e \cup \{w_b \setminus \{u\}\} \cup \{w_e \cup \{u\}\}$ and $w_e \in \{W^i \cup \emptyset\}$.

Theorem 1 *The iterated local search algorithm* CoStar *terminates in a finite number of steps and outputs a Pareto local optimum with respect to* $\mathcal{N}_{(i,w_b,u)}$.

Proof We consider a partition $W^i \in \{\mathcal{Y} \cup X\}$. Algorithm CoStar (in functions OptimizeMultiObjectCluster and OptimizeFeatureCluster) randomly picks-up a cluster w_b from partition W^i and an element $u \in w_b$. Then, it systematically explores the neighborhood $\mathcal{N}_{(i,w_b,u)}(\mathcal{Y} \cup X)$ and retrieves the set of the corresponding neighboring solutions (co-clustering) $\mathcal{ND}$ defined by:

$$\mathcal{ND} = \{s \in \mathcal{N}_{(i,w_b,u)}(\mathcal{Y} \cup X) \mid \forall t \in \mathcal{N}_{(i,w_b,u)}(\mathcal{Y} \cup X), z_i(s) \geq z_i(t)\}$$

Considering a solution $s \in \mathcal{ND}$ and a solution $r \in \mathcal{N}_{(i,w_b,u)}(\mathcal{Y} \cup X)$, we have $z_i(s) \geq z_i(r)$ and the following situations can occur:

1. If $z_i(s) > z_i(r)$ and $\forall k \in \{1, \ldots, N+1\}, k \neq i, z_k(s) \geq z_k(r)$, then s dominates $r : s \succ r$.
2. If $z_i(s) > z_i(r)$ and $\exists k \in \{1, \ldots, N+1\}, k \neq i$, such that $z_k(s) < z_k(r)$, then, from Definition 1, $s \not\succ r$ and $r \not\succ s$.

   In both cases 1 and 2, r does not belong to $\mathcal{ND}$ since $z_i(s) > z_i(r)$ (see functions OptimizeMultiObjectCluster or OptimizeFeatureCluster from lines 7 to 9) and s is a Pareto local optimum with respect to r.

3. If $z_i(s) = z_i(r)$ and $\forall k \in \{1, \ldots, N+1\}, k \neq i, z_k(s) \geq z_k(r)$, then, either s is equivalent to r $(z(s) = z(r))$ or s dominates $r : s \succeq r$.

   Therefore, s is a Pareto local optimum with respect to r. From Algorithms OptimizeMultiObjectCluster or OptimizeFeatureCluster, both r and s belong to $\mathcal{ND}$. When partitions r and s are compared in Algorithms FindPa retoLocalOptimumObject or FindParetoLocalOptimumFeature, $nb_s^+ \geq nb_r^+$ and the algorithms delete r from $\mathcal{ND}$.

4. If $z_i(s) = z_i(r)$ and $\exists k \in \{1, \ldots, N+1\}, k \neq i$, such that $z_k(s) < z_k(r)$ and $\exists \ell \in \{1, \ldots, N+1\}, \ell \neq i \neq k$, such that $z_\ell(s) > z_\ell(r)$, then $s \not\succ r$ and $r \not\succ s$.

   Both partitions r and s are non-dominated from each other and are Pareto local optima. They both belong to $\mathcal{ND}$, and Algorithms FindParetoLocalOpti mumObject or FindParetoLocalOptimumFeature will remove the partition that is strictly greater than the other on the smallest number of objective functions z_k.

5. If $z_i(s) = z_i(r)$ and $\forall k \in \{1, \ldots, N+1\}, z_k(s) \leq z_k(r)$, and $\exists \ell \in \{1, \ldots, N+1\}, \ell \neq i$, such that $z_\ell(s) < z_\ell(r)$, then $r \succ s$.

   Therefore, r is a Pareto local optimum with respect to s. Both s and r belong to $\mathcal{ND}$ and Algorithms FindParetoLocalOptimumObject or FindPareto LocalOptimumFeature will identify that s is dominated by r and will remove s from $\mathcal{ND}$:
   (a) If $i = N + 1$, the set $\mathcal{ND}$ is processed by the function FindParetoLoca lOptimumObject, and s is removed from $\mathcal{ND}$ since

$$\sum_{k=1}^{N} \delta_{z_k(r) > z_k(s)} > \sum_{k=1}^{N} \delta_{z_k(s) > z_k(r)} \text{ with } \delta_C = \begin{cases} 1 & \text{if } C = \text{True} \\ 0 & \text{otherwise} \end{cases}$$

(b) If $i \neq N+1$, the set $\mathcal{ND}$ is processed by the function FINDPARETOLOCALOPTIMUMFEATURE, and s is removed from $\mathcal{ND}$ since

$$z_{N+1}(r) > z_{N+1}(s)$$

Indeed, in that case, $z_k(r) = z_k(s),\ \forall k = 1, \ldots, N$, since the modification of Y^i does not change $\tau_{Y^k|X}, \forall k \neq i$.

We can conclude that at the end of each call of OPTIMIZEMULTIOBJECTCLUSTER and OPTIMIZEFEATURECLUSTER, the output co-clustering is a Pareto local optimum with respect to the considered neighborhood. Finally, all the iterations are bounded and thus COSTAR terminates in a finite number of steps.

4.3 Optimized computation of $\tau_{Y^k|X}$ and $\tau_{X|\mathcal{Y}}$

Algorithms 2 and 4 modify, at each iteration, partitions X and Y^k, by evaluating functions $\tau_{X|\mathcal{Y}}$ and $\tau_{Y^k|X}$ respectively. Algorithms 3 and 5 find a non-dominated partition X and Y^k, among a set of candidates, based on the computation of $\tau_{Y^k|X}$ and $\tau_{X|\mathcal{Y}}$ respectively. The computational complexity of these functions is in $O(m \cdot \mathcal{N})$ and $O(m \cdot n_k)$ where $\mathcal{N} = \sum_k n_k,\ |Y^k| = n_k$ and $|X| = m$. In the worst case, these complexities are in $O(|O| \cdot \sum_k |F^k|)$ and $O(|O| \cdot |F^k|)$ where $|O|$ represents the cardinality of the objects dimension. Moreover, during each iteration of Algorithms 2 and 4, these operations are performed for each cluster (including the empty cluster). Thus the overall complexities are in $O(m \cdot m \cdot \mathcal{N})$ and $O(m \cdot n_k \cdot n_k)$.

When moving a single element from a cluster to another, a large part of the computation of these functions can be saved, since a large part in the τ formula remains the same. To take advantage of this point, we only compute the variations of $\tau_{Y^k|X}$ and $\tau_{X|\mathcal{Y}}$ from one step to another as explained in Sects. 4.3.1 and 4.3.2. We evaluate the variation of each measure when the partition X is modified as well as it is Y^k that is changed.

4.3.1 *Computing the variation of* $\tau_{Y^k|X}$

When partition Y^k is modified: Let's first consider the variation of $\tau_{Y^k|X}$ when one element f is moved from cluster y_b to cluster y_e of Y^k. This variation is the result of some changes on the co-occurrence table T^k. Moving f induces, for each cluster of objects x_i, the transfer of a quantity λ_i from t^k_{ib} to t^k_{ie} of T^k. The overall vector of transferred quantities is called $\boldsymbol{\lambda} = [\lambda_1, \ldots, \lambda_m]$. In the following, we call t^k_{ij} the elements of the co-occurrence table T^k before the update operation, and s^k_{ij} the same elements after the operation. We call T^k and S^k the two co-occurrence tables, and $\tau_{Y^k|X}(T^k)$, $\tau_{Y^k|X}(S^k)$ the values of $\tau_{Y^k|X}$ computed over T^k and S^k. The following equations hold for t^k_{ij} and s^k_{ij}:

$$\begin{aligned} s^k_{ij} &= t^k_{ij}, \text{ if } j \neq b, j \neq e \\ s^k_{ib} &= t^k_{ib} - \lambda_i \\ s^k_{ie} &= t^k_{ie} + \lambda_i \end{aligned}$$

Therefore, the variation of $\tau_{Y^k|X}$ is:

$$\begin{aligned} \Delta_{\tau_{Y^k|X}}(T^k, f, y_b, y_e) &= \tau_{Y^k|X}(T^k) - \tau_{Y^k|X}(S^k) \\ &= \frac{\sum_i \sum_j \frac{\left(t^k_{ij}\right)^2}{t^k_{i.}t^k_{..}} - \sum_j \frac{\left(t^k_{.j}\right)^2}{(t^k_{..})^2}}{1 - \sum_j \frac{\left(t^k_{.j}\right)^2}{(t^k_{..})^2}} - \frac{\sum_i \sum_j \frac{\left(s^k_{ij}\right)^2}{s^k_{i.}s^k_{..}} - \sum_j \frac{\left(s^k_{.j}\right)^2}{(s^k_{..})^2}}{1 - \sum_j \frac{\left(s^k_{.j}\right)^2}{(s^k_{..})^2}} \end{aligned}$$

where $t^k_{i.} = \sum_j t^k_{ij}$, $t^k_{.j} = \sum_i t^k_{ij}$ and $t^k_{..} = \sum_i \sum_j t^k_{ij}$. Setting $\lambda = \sum_i \lambda_i$, $\Omega = 1 - \sum_j \frac{(t^k_{.j})^2}{(t^k_{..})^2}$ and $\Gamma = 1 - \sum_i \sum_j \frac{(t^k_{ij})^2}{t^k_{i.}t^k_{..}}$, we obtain the following updating formula:

$$\Delta_{\tau_{Y^k|X}}(T^k, f, y_b, y_e) = \frac{\Omega \left(\sum_i \frac{2\lambda_i}{t^k_{i.}t^k_{..}} \left[t^k_{ib} - t^k_{ie} - \lambda_i\right] \right) + \Gamma \left(\frac{2\lambda}{(t^k_{..})^2} \left[t^k_{.e} - t^k_{.b} + \lambda\right] \right)}{\Omega^2 - \frac{2}{(t^k_{..})^2} \lambda \Omega \left(t^k_{.e} - t^k_{.b} + \lambda\right)}$$

Thanks to this approach, instead of computing $\tau_{Y^k|X}$ at lines 7 and 11 of Algorithm 4 with a complexity in $O(m \cdot n_k)$, we compute $\Delta_{\tau_{Y^k|X}}(T^k, f, y_b, y_e)$ once at line 7 with a complexity in $O(m)$ (that is in $O(|O|)$ in the worst case with the discrete partition X) and test if it is strictly positive in the first **if** condition or null in the second one. Computing Γ is in $O(m \cdot n_k)$ and Ω in $O(m)$ and is done only once at the beginning of Algorithm 4.

When partition X is modified Let's now consider the variation of $\tau_{Y^k|X}$ when one element o is moved from cluster x_b to cluster x_e of partition X. This operation induces, for each cluster of features y^k_j, the transfer of a quantity μ^k_j from t^k_{bj} to t^k_{ej} of T^k. The overall vector of transferred quantities is called $\boldsymbol{\mu^k} = [\mu^k_1, \ldots, \mu^k_{n_k}]$. We call t^k_{ij} the elements of the kth contingency table T^k before the update operation, and s^k_{ij} the same elements after the operation. $\tau_{Y^k|X}(T^k)$ and $\tau_{Y^k|X}(S^k)$ are the values of $\tau_{Y^k|X}$ computed over T^k and S^k. The following equations hold for t^k_{ij} and s^k_{ij}:

$$\begin{aligned} s^k_{ij} &= t^k_{ij}, \quad \text{if } i \neq b, i \neq e \\ s^k_{bj} &= t^k_{bj} - \mu^k_j \\ s^k_{ej} &= t^k_{ej} + \mu^k_j \end{aligned}$$

Therefore, the variation of $\tau_{Y^k|X}$ is equal to:

$$\Delta_{\tau_{Y^k|X}}(T^k, o, x_b, x_e) = \tau_{Y^k|X}(T^k) - \tau_{Y^k|X}(S^k)$$
$$= \frac{\sum_i \sum_j \frac{\left(t^k_{ij}\right)^2}{t^k_{i.} t^k_{..}} - \sum_j \frac{\left(t^k_{.j}\right)^2}{(t^k_{..})^2}}{1 - \sum_j \frac{\left(t^k_{.j}\right)^2}{(t^k_{..})^2}} - \frac{\sum_i \sum_j \frac{\left(s^k_{ij}\right)^2}{s^k_{i.} s^k_{..}} - \sum_j \frac{\left(s^k_{.j}\right)^2}{(s^k_{..})^2}}{1 - \sum_j \frac{\left(s^k_{.j}\right)^2}{(s^k_{..})^2}}$$

where $t^k_{i.} = \sum_j t^k_{ij}$, $t^k_{.j} = \sum_i t^k_{ij}$ and $t^k_{..} = \sum_i \sum_j t^k_{ij}$. Setting $\mu^k_{.} = \sum_j \mu^k_j$, $\Omega = 1 - \sum_j \frac{(t^k_{.j})^2}{(t^k_{..})^2}$, we obtain the following updating formula:

$$\Delta_{\tau_{Y^k|X}}(T^k, o, x_b, x_e) = \frac{1}{\Omega} \sum_j \left(\frac{\left(t^k_{ej}\right)^2}{t^k_{e.} t^k_{..}} - \frac{\left(t^k_{ej} + \mu^k_j\right)^2}{\left(t^k_{e.} + \mu^k_{.}\right) t^k_{..}} + \frac{\left(t^k_{bj}\right)^2}{t^k_{b.} t^k_{..}} - \frac{\left(t^k_{bj} - \mu^k_j\right)^2}{\left(t^k_{b.} - \mu^k_{.}\right) t^k_{..}} \right)$$

Thanks to this approach, instead of computing $\tau_{Y^k|X}$ at lines 8 and 11 of Algorithm 3 with a complexity in $O(m \cdot n_k)$, we compute $\Delta_{\tau_{Y^k|X}}(T^k, o, x_b, x_e)$ once at line 8 with a complexity in $O(n_k)$ (that is in $O(|F^k|)$ in the worst case with the discrete partition F^k) and test if it is strictly positive in the first **if** condition or strictly negative in the second one. Computing Ω is in $O(n_k)$ and is done only once at the beginning of Algorithm 3.

4.3.2 Computing the variation of $\tau_{X|\mathcal{Y}}$

When partition X is modified Let's now consider the variation of $\tau_{X|\mathcal{Y}}$ when one object o is moved from the cluster x_b to the cluster x_e of partition X. This operation induces, for each cluster of features y^k_j, the transfer of a quantity μ^k_j from t^k_{bj} to t^k_{ej} of T^k. The overall vector of transferred quantities is called $\boldsymbol{\mu} = [\mu^1_1, \ldots, \mu^1_{n_1}, \ldots, \mu^N_1, \ldots, \mu^N_{n_N}]$.

We call t^k_{ij} the elements of the kth contingency table T^k before the update operation, and s^k_{ij} the same elements after the operation. We call $\mathcal{T} = \{T^1, \ldots, T^N\}$ and $\mathcal{S} = \{S^1, \ldots, S^N\}$ the two sets of contingency tables, and $\tau_{X|\mathcal{Y}}(\mathcal{T})$ and $\tau_{X|\mathcal{Y}}(\mathcal{S})$ the values of $\tau_{X|\mathcal{Y}}$ computed over $\mathcal{T}$ and $\mathcal{S}$. The following equations hold for t^k_{ij} and s^k_{ij}:

$$\begin{aligned} s^k_{ij} &= t^k_{ij}, \quad \text{if } i \neq b, i \neq e \\ s^k_{bj} &= t^k_{bj} - \mu^k_j \\ s^k_{ej} &= t^k_{ej} + \mu^k_j \end{aligned}$$

Therefore, the variation of $\tau_{X|\mathcal{Y}}$ is:

$$\Delta_{\tau_{X|\mathcal{Y}}}(\mathcal{T}, o, x_b, x_e) = \tau_{X|\mathcal{Y}}(\mathcal{T}) - \tau_{X|\mathcal{Y}}(\mathcal{S}) \; = \; \frac{\sum_k \sum_i \sum_j \frac{\left(t^k_{ij}\right)^2}{t^k_{..} \times t^k_{.j}} - \sum_k \sum_i \frac{\left(t^k_{i.}\right)^2}{\left(t^k_{..}\right)^2}}{N - \sum_k \sum_i \frac{\left(t^k_{i.}\right)^2}{\left(t^k_{..}\right)^2}} - \frac{\sum_k \sum_i \sum_j \frac{\left(s^k_{ij}\right)^2}{s^k_{..} \times s^k_{.j}} - \sum_k \sum_i \frac{\left(s^k_{i.}\right)^2}{(s^k_{..})^2}}{N - \sum_k \sum_i \frac{\left(s^k_{i.}\right)^2}{(s^k_{..})^2}}$$

Setting $\mu^k_. = \sum_j \mu^k_j$, $\Omega = N - \sum_k \sum_i \frac{(t^k_{i.})^2}{(t^k_{..})^2}$, and $\Gamma = N - \sum_k \sum_i \sum_j \frac{(t^k_{ij})^2}{t^k_{..} \times t^k_{.j}}$, we obtain the following updating formula:

$$\Delta_{\tau_{X|\mathcal{Y}}}(\mathcal{T}, o, x_b, x_e) = \\ = \frac{\Omega\left(\sum_k \sum_j \frac{2\mu^k_j}{t^k_{..} \times t^k_{.j}} [t^k_{bj} - t^k_{ej} - \mu^k_j]\right) + \Gamma\left(\sum_k \frac{2\mu^k_.}{(t^k_{..})^2} [t^k_{e.} - t^k_{b.} + \mu^k_.]\right)}{\Omega^2 - \Omega \sum_k \frac{2\mu^k_.}{(t^k_{..})^2} (t^k_{e.} - t^k_{b.} + \mu^k_.)}$$

Thanks to this approach, instead of computing $\tau_{X|\mathcal{Y}}$ at lines 7 and 11 of Algorithm 2 with a complexity in $O(m \cdot \mathcal{N})$, we compute $\Delta_{\tau_{X|\mathcal{Y}}}(\mathcal{T}, o, x_b, x_e)$ once at line 7 with a complexity in $O(\mathcal{N})$ (that is in $O(\sum_k |F^k|)$ in the worst case with the discrete partitions Y^k) and test if it is strictly positive in the first **if** condition or null in the second one. Computing Γ is in $O(m \cdot \mathcal{N})$ and Ω in $O(m \cdot N)$ and is done only once at the beginning of Algorithm 2.

When partition Y^k is modified Let's now consider the variation of $\tau_{X|\mathcal{Y}}$ when one feature f is moved from the cluster y_b to the cluster y_e of partition Y^k. This variation is the result of some changes on the co-occurrence table T^k. Moving f induces, for each cluster of objects x_i, the transfer of a quantity λ_i from t^k_{ib} to t^k_{ie} of T^k, the other contingency tables been unchanged. The overall vector of transferred quantities is called $\boldsymbol{\lambda} = [\lambda_1, \ldots, \lambda_m]$. In the following, we call t^k_{ij} the elements of the co-occurrence table T^k before the update operation, and s^k_{ij} the same elements after the operation. We call $\mathcal{T} = \{T^1, \ldots, T^N\}$ and $\mathcal{S} = \{S^1, \ldots, S^N\}$ the two sets of contingency tables, and $\tau_{X|\mathcal{Y}}(\mathcal{T})$ and $\tau_{X|\mathcal{Y}}(\mathcal{S})$ the values of $\tau_{X|\mathcal{Y}}$ computed over $\mathcal{T}$ and $\mathcal{S}$.

The following equations hold for t^k_{ij} and s^k_{ij}:

$$\begin{aligned} s^\ell_{ij} &= t^\ell_{ij}, \quad \text{if } \ell \neq k \quad \text{or} \quad \ell = k, \quad j \neq b, \quad j \neq e \\ s^k_{ib} &= t^k_{ib} - \lambda_i \\ s^k_{ie} &= t^k_{ie} + \lambda_i \end{aligned}$$

Therefore, the variation of $\tau_{X|\mathcal{Y}}$ is:

$$\Delta_{\tau_{X|\mathcal{Y}}}(\mathcal{T}, f, y_b, y_e) = \tau_{X|\mathcal{Y}}(\mathcal{T}) - \tau_{X|\mathcal{Y}}(\mathcal{S}) = \frac{\sum_k \sum_i \sum_j \frac{\left(t^k_{ij}\right)^2}{t^k_{..} \times t^k_{.j}} - \sum_k \sum_i \frac{(t^k_{i.})^2}{(t^k_{..})^2}}{N - \sum_k \sum_i \frac{(t^k_{i.})^2}{(t^k_{..})^2}} - \frac{\sum_k \sum_i \sum_j \frac{\left(s^k_{ij}\right)^2}{s^k_{..} \times s^k_{.j}} - \sum_k \sum_i \frac{(s^k_{i.})^2}{(s^k_{..})^2}}{N - \sum_k \sum_i \frac{(s^k_{i.})^2}{(s^k_{..})^2}}$$

Setting $\Omega = N - \sum_k \sum_i \frac{(t^k_{i.})^2}{(t^k_{..})^2}$ and $\lambda = \sum_i \lambda_i$, we obtain the following updating formula:

$$\Delta_{\tau_{X|\mathcal{Y}}}(\mathcal{T}, f, y_b, y_e) = \frac{1}{\Omega} \sum_i \left(\frac{(t^k_{ie})^2}{t^k_{.e} t^k_{..}} - \frac{(t^k_{ie} + \lambda_i)^2}{(t^k_{.e} + \lambda) t^k_{..}} + \frac{(t^k_{ib})^2}{t^k_{.b} t^k_{..}} - \frac{(t^k_{ib} - \lambda_i)^2}{(t^k_{.b} - \lambda) t^k_{..}} \right)$$

Thanks to this approach, instead of computing $\tau_{X|\mathcal{Y}}$ at lines 9 and 12 of Algorithm 5 with a complexity in $O(m \cdot \mathcal{N})$, we compute $\Delta_{\tau_{X|\mathcal{Y}}}(\mathcal{T}, f, y_b, y_e)$ once at line 9 with a complexity in $O(m)$ (that is in $O(|O|)$ in the worst case with the discrete partitions X) and test if it is strictly positive in the first **if** condition or strictly negative in the second one. Computing Ω is in $O(m \cdot N)$ and is done only once at the beginning of Algorithm 5.

4.4 Overall complexity

In this subsection, we evaluate the overall complexity of CoStar and compare it to the complexity of competitors used in Sect. 5 (SRC, Long et al. 2006; NMF, Chen et al. 2009; and ComRaf, Bekkerman and Jeon 2007).

CoStar starts by computing N contingency tables T^k. This operation takes $O(m \cdot n_k)$ time for each feature space $F^k, k = 1 \ldots N$. Using the notation introduced beforehand, computing all the contingency tables takes $O(m \cdot \mathcal{N})$. The core of the algorithm is the iterative procedure which updates the object partition and the N feature partitions.

Algorithm 2 has a complexity in $O(m \cdot \mathcal{N})$ for computing Γ and $O(m \cdot N)$ for computing Ω; the loop takes time $O(m \cdot \mathcal{N})$ and Algorithm 3 takes $O(m \cdot n_k)$ in the worst case ($|\mathcal{ND}| = m$). Therefore the whole complexity of the function is in $O(m \cdot \mathcal{N})$.

Concerning Algorithm 4, computing Γ and Ω takes time $O(m \cdot n_k)$ and $O(m)$ respectively; the loop takes $O(m \cdot n_k)$ and Algorithm 5 takes time $O(m \cdot n_k)$ in the worst case ($|\mathcal{ND}| = n_k$). The overall complexity of the function is in $O(m \cdot n_k)$.

Let $\mathcal{I}$ denote the total number of iterations. The overall time complexity of CoStar is $O(\mathcal{I} \cdot m \cdot \mathcal{N})$, i.e., it is linear in the number of objects and features.

We now consider the theoretical complexity of the competitors, as reported by the study (Chen et al. 2010). We use again $\mathcal{I}$ as the number of iterations, N as the number of feature spaces, and m as the number of objects, while n is the maximum feature dimension for all feature spaces and k is the maximum between the number of object clusters and the number of feature clusters. Hence, *SRC* has a complexity in $O(\mathcal{I} \cdot N \cdot (\max(m, n)^3 + kmn))$ and *ComRaf* has a complexity in $O(\mathcal{I} \cdot N \cdot \max(m^3, n^3))$. Notice that the complexity of these two algorithms is cubic in the biggest of the dimensions of objects and all the feature spaces. Complexity of *NMF*, instead, is similar to our method's: this approach takes time $O(\mathcal{I} \cdot N \cdot k \cdot m \cdot n)$. However, this method requires to define the number of clusters as a parameter.

5 Experiments

In this section, we empirically show the effectiveness of our algorithm CoStar in comparison with three state-of-the-art algorithms that deal with star-structured data. All of them are non deterministic, have been presented in recent years and have been summarized in Sect. 2. The first competitor is *Comraf* and it is presented in Bekkerman and Jeon (2007). The second competitor is *SRC* and is described in Long et al. (2006). The third one is *NMF* and is introduced in Chen et al. (2009). In particular, for obtaining all the experiment results of the competitors, we took care to use the same pre- and post-processing methods that have been adopted and described in the original papers. For instance, since the second approach uses the *k-means* algorithm as a post-processing method, we did the same.

All these methods require as input, at least, the number of row clusters. For any given dataset, we set this parameter equal to the number of preexisting classes. To validate our proposal, we use three different groups of datasets (details are given further in the text):

1. a publicly available synthetic dataset (*BBC* and *BBCSport*) for standard analysis,
2. three document–word–category datasets as illustrative cases of imbalanced problems,
3. an image-word-blob dataset (*COREL Benchmark*) as illustrative cases of a balanced problem.

As any of these methods (the three competitors and CoStar) is non deterministic, for any given dataset, we run each algorithm 100 times and compute the average and standard deviation of the evaluation measures used (presented below). Since CoStar does not require to specify the number of clusters, we also study the distribution of the number of row clusters over the 100 trials for this algorithm. The number of iterations for CoStar was set to $10 \times \max(m, \mathcal{N})$ (m and $\mathcal{N}$ being the number of objects and features respectively). As further information, CoStar is written in C++, *Comraf* in C, *SRC* in R, and *NMF* in MATLAB. Finally, all the experiments have been performed on a 2.6GHz Opteron processor PC, with 4GB RAM, running Linux.

5.1 External evaluation measures

We evaluate the performance of CoStar and the competitors using two external validation indices. We denote by $\mathbf{C} = \{C_1 \ldots C_J\}$ the partition built by the clustering algorithm on objects, and by $\mathbf{P} = \{P_1 \ldots P_I\}$ the partition inferred by the original classification. J and I are respectively the number of clusters $|\mathbf{C}|$ and the number of classes $|\mathbf{P}|$. We denote by n the total number of objects.

The first index is the *normalized mutual information* (NMI). NMI provides an information that is impartial with respect to the number of clusters (Strehl and Ghosh 2002). It measures how clustering results share the information with the true class assignment. NMI is computed as the average mutual information between every pair of clusters and classes:

$$\mathbf{NMI} = \frac{\sum_{i=1}^{I} \sum_{j=1}^{J} x_{ij} \log \frac{n x_{ij}}{x_i x_j}}{\sqrt{\sum_{i=1}^{I} x_i \log \frac{x_i}{n} \sum_{j=1}^{J} x_j \log \frac{x_j}{n}}}$$

where x_{ij} is the cardinality of the set of objects that belong to cluster C_j and class P_i; x_j is the number of objects in cluster C_j; x_i is the number of objects in class P_i. Its values range between 0 and 1.

The second measure is the Adjusted Rand Index (Hubert and Arabie 1985). Let a be the number of object pairs belonging to the same cluster in $\mathbf{C}$ and to the same class in $\mathbf{P}$. This metric captures the deviation of a from its expected value corresponding to the hypothetical value of a obtained when $\mathbf{C}$ and $\mathbf{P}$ are two random independent partitions. The expected value of a denoted by $E[a]$ is computed as follows:

$$E[a] = \frac{\pi(C) \cdot \pi(P)}{n(n-1)/2}$$

where $\pi(C)$ and $\pi(P)$ denote respectively the number of object pairs that belong to the same cluster in $\mathbf{C}$ and to the same class in $\mathbf{P}$. The maximum value for a is defined as:

$$\max(a) = \frac{1}{2}(\pi(C) + \pi(P))$$

The agreement between $\mathbf{C}$ and $\mathbf{P}$ can be estimated by the Adjusted Rand Index as follows:

$$\mathbf{ARI}(\mathbf{C}, \mathbf{P}) = \frac{a - E[a]}{\max(a) - E[a]}$$

Notice that this index can take negative values, and when $ARI(\mathbf{C}, \mathbf{P}) = 1$, we have identical partitions.

Name	# obj.	# obj. classes	View	# features
bbc2	2012	5	1st view	6838
			2nd view	6790
bbcsport2	544	5	1st view	3183
			2st view	3203
bbc4	685	5	1st view	4659
			2nd view	4633
			3rd view	4665
			4th view	4684

Fig. 6 Synthetic dataset characteristics

5.2 Datasets

Here, we describe in detail the datasets we employed for our experimental analysis. The first two datasets are multi-view text data from news corpora, downloaded from http://mlg.ucd.ie/datasets/segment.html: *BBC* and *BBCSport*. The multiple views were created by splitting the text corpora into related segments. Each segment is constituted by consecutive textual paragraphs as described in the cited website. Furthermore, as explained by the web site, the standard textual pre-processing steps have been performed on the texts: stemming, stop-words removal and removal of words occurring very few times (lower than 3 times). The *BBCSport* dataset consists in 2 views (*bbcsport2*), while the *BBC* dataset is available both with 2 and 4 views (*bbc2* and *bbc4*). For each collection we selected only the documents that contain at least one segment for each view. The characteristics of the datasets are reported in Fig. 6. Object classes are provided with the datasets and consist in annotated topic labels provided for the news articles (with values such as *business, entertainment, politics, sport* and *tech*).

Also the second group of datasets comes from a textual domain and is described in Forman (2003). In particular, in our experiments we consider combinations of the following datasets:

- *oh15*: is a sample from OHSUMED dataset. OHSUMED is a clinically-oriented MEDLINE subset of abstracts or titles from 270 medical journals over a five-year period (1987–1991).
- *re0*: is a sample from Reuters-21578 dataset. This dataset is widely used as test collection for text categorization research.
- *wap*: is a sample from the WebACE Project, consisting on web pages listed in the subject hierarchy of Yahoo!.

The first view consists in sets of terms contained in the documents, also available on the Weka Website.[1] Also for this data, we performed the same standard pre-processing used for the news corpora: stemming, stop-words removal and removal of words occurring less than 3 times. As a consequence of the pre-processing, we obtained a document–word matrix.

For the second view, we consider some categorization of the documents. To this purpose, we used the semi-supervised technique presented in Chen et al. (2009). Each feature category of this view corresponds to a category provided by the document-set

[1] http://www.cs.waikato.ac.nz/ml/weka/.

Name	Datasets	# terms	# cat.	# obj.	# classes	Classes
T1	*oh15* *re0*	3987	2	833	5	{Aden-Diph, Cell-Mov, Aluminium} {cpi, money }
T2	*oh15* *re0*	3197	2	461	5	{Blood-Coag, Enzyme-Act, Staph-Inf} {jobs, reserves}
T3	*wap* *oh15* *re0*	8282	3	2129	9	{ Film, Television, Health} {Aden-Diph, Cell-Mov, Enzyme-Act} { interest, trade, money}

Fig. 7 Text dataset characteristics

Fig. 8 Classes distributions for text datasets

Class Name	# obj.	Class Name	# obj.
Aden-Diph	56	*Cpi*	60
Cell-Mov	106	*Money*	608
Aluminium	53	*Interest*	219
Blood-Coag	69	*Trade*	319
Enzyme-Act	154	*Film*	196
Staph-Inf	157	*Television*	130
Jobs	39	*Health*	341
Reserves	42		

and depends on the dataset the document belongs to. Then, the value of each feature for each document is compiled as the probability that the document belongs to the related category: it is computed as the portion of the whole vocabulary associated to a specific category which is also present in a specific document. Each element of the document–category matrix has values in the range [0, 1]. The details of these datasets are reported in Figs. 7 and 8. In particular, we generated three new datasets of increasing difficulty (named *T1, T2* and *T3*) in which we created the concept of *super-class* by injecting an increased separation only between some of the classes. The aim is to see if the co-clustering is able to recognize the concept of super-class. We did the following: we took the object classes from the three datasets (*OHSUMED*, *re0* and *wap*) and mixed them as the information in Fig. 7 suggests. For instance, for the generation of dataset *T1* we placed the first three classes of objects indicated in the last column of Fig. 7 (*Aden-Diph, Cell-Mov, Aluminium*) from the original dataset *OHSUMED* (indicated in Fig. 7 by symbol *oh15*) while the remaining 2 classes (*cpi, money*) have been taken from *Reuters-21578* (indicated in figure by *re0*). Analogously for the remaining datasets *T2* and *T3*. Figure 8 reports instead the number of objects for each document class.

The last group of datasets are extracted from the Corel Benchmark.[2] This benchmark has already been used in Bekkerman and Jeon (2007), and we use a similar pre-processing. It consists of 5,000 images from 50 Corel Stock Photo CDs. Each CD contains 100 images on the same topic. The original dataset has 4,500 training images and 500 test images. Here, we consider only the 4,500 training images for our experiments. Every image has a caption and an annotation. The caption is a brief description of the scene and the annotation is a list of objects (entities, or segments) that appear in the image. On overall, 371 words and 42,379 entities are used to describe the collection. We built two different views: the first view is based on the caption words, the second view is based on blobs. A blob is a portion of the image in which a user has recognized a particular entity. It is similar to a concept. In order to obtain blobs, we

[2] http://kobus.ca/research/data/eccv_2002.

Name	# words	# obj.	# classes	Classes
I1	114	630	7	sunsets, tigers, train, swimmers, formula One car, skyscrapers, war airplanes
I2	116	541	6	bears, deers, horses, cliffs, birds, bridges
I3	170	1171	13	sunsets, tigers, train, swimmers, formula One car, skyscrapers, war airplanes, bridges, bears, deers, horses, cliffs, birds,

Fig. 9 Image dataset characteristics

	CoStar	Comraf	SRC	NMF
bbc2	**0.68** ± 0.02	0.58 ± 0.04	0.31 ± 0.11	0.5 ± 0.04
bbcsport2	**0.69** ± 0.06	0.09 ± 0.05	0.24 ± 0	0.57 ± 0.01
bbc4	**0.68** ± 0.03	0.41 ± 0.03	0.33 ± 0.04	0.45 ± 0.0
I1	**0.88** ± 0.04	0.85 ± 0.05	0.80 ± 0.11	0.86 ± 0.06
I2	**0.75** ± 0.04	0.66 ± 0.05	0.61 ± 0.09	0.74 ± 0.04
I3	**0.76** ± 0.03	0.75 ± 0.04	0.61 ± 0.11	0.73 ± 0.06
T1	**0.72** ± 0	0.50 ± 0.03	0.23 ± 0.05	0.49 ± 0.01
T2	**0.73** ± 0.1	0.35 ± 0.03	0.33 ± 0.08	0.58 ± 0.02
T3	**0.71** ± 0.02	0.66 ± 0	0.55 ± 0.06	0.63 ± 0.02

Fig. 10 Normalized mutual information results

clustered (with a *k-means* algorithm) all the entities into 3,000 groups (blobs). Each image is mapped onto the set of blobs or entities represented in the image itself. This model leads to a representation analogous to the bag-of-words model (BOW) often used in text processing. For our experiments, we built three different datasets (details are reported in Fig. 9). Each of the generated dataset has balanced classes represented by an average of 90 objects.

5.3 Results and discussion

We now analyze in detail the different aspects of our experimental evaluation. First, we compare the behavior of our algorithm to the three competitors. Second, we assess the significance of the results statistically. Third, we study the variability of the number of detected clusters. Then, we perform an in-depth qualitative analysis of the specific results. Finally, we provide a scalability study.

5.3.1 CoStar *against competitors*

In Figs. 10 and 11, we report the average and standard deviation of the performance indexes for each dataset and each clustering approach. We can observe that CoStar outperforms all other approaches on NMI index and achieves the best results on all the datasets except *I3* when considering the ARI index. We recall that CoStar does not require any parameter. Despite this, it finds always high-quality partitions, while the other approaches do not take advantage of the additional information of the correct number of clusters.

Notice that CoStar always outperforms *SRC*. On the other hand, *Comraf* achieves comparable results over the Images datasets. This is not surprising, since *Comraf* is suited for this specific application domain. But, when applying *Comraf* on the text

	CoStar	**Comraf**	**SRC**	**NMF**
bbc2	**0.67** ± 0.03	0.42 ± 0.05	0.18 ± 0.08	0.48 ± 0.06
bbcsport2	**0.58** ± 0.14	0.23 ± 0.04	0.14 ± 0.09	0.56 ± 0.02
bbc4	**0.56** ± 0.2	0.34 ± 0.05	0.16 ± 0.06	0.41 ± 0.0
I1	**0.83** ± 0.09	0.80 ± 0.08	0.69 ± 0.11	0.79 ± 0.1
I2	**0.72** ± 0.07	0.55 ± 0.06	0.51 ± 0.09	0.67 ± 0.07
I3	0.50 ± 0.11	**0.62** ± 0.06	0.43 ± 0.13	0.61 ± 0.09
T1	**0.73** ± 0	0.23 ± 0.05	0.09 ± 0.09	0.23 ± 0.02
T2	**0.68** ± 0	0.19 ± 0.04	0.18± 0.09	0.48 ± 0.01
T3	**0.44** ± 0.02	0.42 ± 0.03	0.36 ± 0.08	0.44 ± 0.04

Fig. 11 Adjusted Rand Index results

datasets, it returns very poor results regarding both measures. *NMF* performs better than *Comraf* on text data, and has similar results on *Corel Benchmarks*.

5.3.2 Statistical significance of the results

To assess the statistical quality of our approach we use the Friedman statistic and the Nemenyi test (Demsar 2006). These techniques are usually employed to evaluate the statistical relevance of results of different classifiers over multiple datasets. We briefly summarize the Friedman test:

1. the performance of each method (in terms of a given evaluation measure such as ARI or NMI) is determined on each dataset;
2. the methods are ranked for each dataset according to these results, the best performing algorithm getting the rank of 1, the second best rank 2, and so on;
3. for each method, its average rank R_j w.r.t. the datasets is computed;
4. finally, the Friedman statistic is computed thanks to the following formula:

$$Q = \frac{12N}{k(k+1)} \left[\sum_{j=1}^{k} R_j^2 - \frac{k(k+1)^2}{4} \right]$$

   where N is the number of datasets and k is the number of methods that are compared.

On the null hypothesis, where is no difference among the k sets of measures, the average rank should approximate $\frac{(k+1)}{2}$. When N and k are large (i.e. $N > 15$ and $k > 4$), the probability distribution of Q can be approximated by that of a chi-square distribution with $k - 1$ degrees of freedom. In this case the p-value is given by $P(\chi^2_{k-1} \geq Q)$. If N or k is small, the approximation to chi-square becomes poor and the p-value should be obtained from tables of Q specially prepared for the Friedman test.

We compare ARI and NMI results of CoStar and its competitors (*SRC*, *ComRaf* and *NMF*) on the 9 datasets and compute the associated Q values: we obtain 21.63 and 22.73 for ARI and NMI respectively. As the critical value at significance level $\alpha = 0.01$ is equal to 10.73 (see Zar 1998; Sheskin 2007) and is smaller than the Q values, we decide to reject the null hypothesis for both measures.

	Average ranking	
Methods	**ARI**	**NMI**
CoStar	1.2	1
NMF	2.2	2.4
ComRaf	2.6	2.7
SRC	4	3.9

ARI	*NMF*	*ComRaf*	*SRC*
CoStar	1	**1.4**	**2.8**
NMF	-	0.4	**1.8**
ComRaf	-	-	**1.4**
SRC	-	-	0

NMI	*NMF*	*ComRaf*	*SRC*
CoStar	**1.4**	**1.7**	**2.9**
NMF	-	0.3	**1.5**
ComRaf	-	-	1.2
SRC	-	-	0

Fig. 12 Average ranking results for the Nemenyi test (*top*); Difference between average ranks for ARI and NMI evaluation measures (*bottom*)

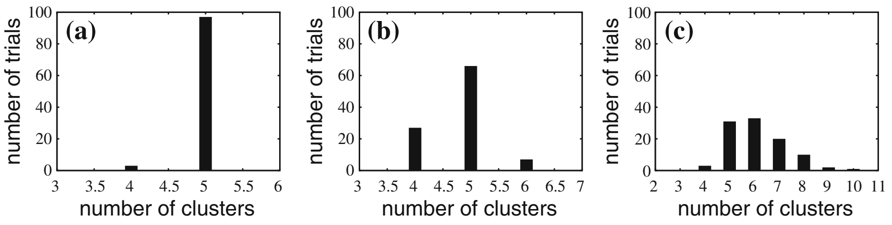


Fig. 13 Distribution of CoStar number of clusters on **a** *bbc2*, **b** *bbcsport2* and **c** *bbc4*

We can now proceed with the post-hoc Nemenyi test. According to this test, the performance of two algorithms is significantly different if the corresponding average ranks differ by at least the critical difference:

$$CD = q_\alpha \sqrt{\frac{k(k+1)}{6N}}$$

where α is the significance level ($\alpha = 0.10$ in our case) and q_α is the critical value for the two tailed Nemenyi test (Demsar 2006). We compare the four methods at the critical value $q_{0.10} = 2.291$. The ranking table is shown in Fig. 12 (top). The critical difference between average ranks is $CD = 1.394$ (for ARI and NMI). Figure 12 (bottom) shows the difference between average ranks for both ARI and NMI. In the first case, CoStar is significantly different from *ComRaf* and *SRC*. In the second case, it performs significantly better than all other competitors.

5.3.3 Number of detected clusters

Since CoStar may find different numbers of clusters at each execution, we ran the algorithm 100 times and evaluate the variability of this result. In Figs. 13, 14 and 15, we report the distributions of the number of row clusters for the three groups of datasets.

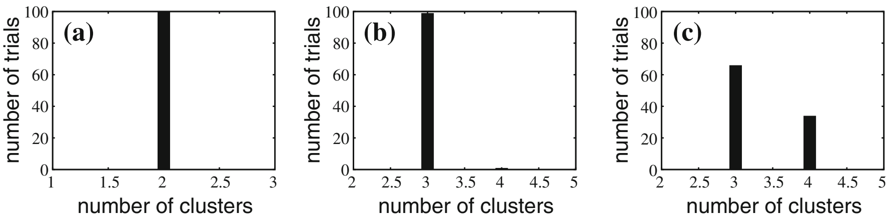


Fig. 14 Distribution of the number of clusters of CoStar on **a** *T1*, **b** *T2* and **c** *T3*

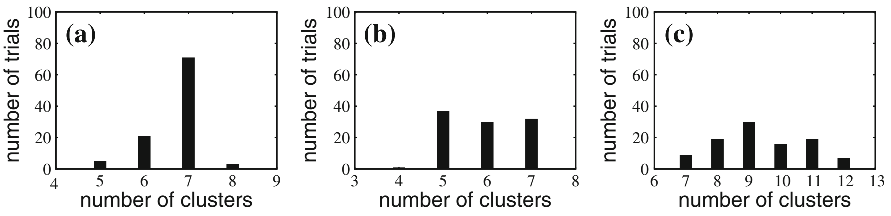


Fig. 15 Distribution of the number of clusters of CoStar on **a** *I1*, **b** *I2* and **c** *I3*

On *bbc2* and *bbcsport2*, CoStar produces most of the times partitions with the correct number of clusters (see Fig. 13). However, the resulting partitions on *bbc4* contain five or six clusters to the same extent.

Now consider the three textual datasets (see Fig. 14). By analyzing the results, we found that CoStar recognizes the "super-classes" corresponding to the originating dataset to which the document belongs to. For instance, for *T1* there are two super-classes: one from *oh15* and one from *re0*. This is reasonable because they represent two types of dictionary: a biology-centered one and a news-centered one. The same phenomenon happens for the other two textual datasets. Furthermore, the found number of clusters remains quite stable, in particular for *T1* and *T2*.

Figure 15 shows the distribution of the number of row clusters for the Corel benchmark. First, CoStar detects most of the time the correct number of clusters (7) when applied to *I1*. When processing *I2*, the algorithm finds three different values of cluster numbers for the row partition: 5, 6 and 7. Notice that in this case, the correct number is 6, which is very close to the average of the detected number of clusters. For *I3*, which is a hard context for a clustering algorithm, the distribution of the number of detected clusters may be approximated with a normal distribution with a mean of 9, while the number of built-in classes is 13.

5.3.4 A qualitative evaluation

We now show an example to provide some insights about the kind of results our algorithm is able to provide. To this purpose, we choose a specific result of the *I2* image dataset. In particular, among the 100 runs of CoStar, we choose the one providing the result with the highest value on the objective function computed over the row space ($\tau_{X|\mathcal{Y}}$).

Fig. 16 Word clusters for image dataset 2

bear, polar, black, snow, antlers, grizzly, elk, ice, tundra, grass
bridge, water, coast, arch, hills, sky, waves, beach, boats, steel
deer, white-tailed, mule, horns, slope, fawn
horses, foals, mare, field, fence, bush
birds, nest, branch, fly, tree, leaf, wood, wings, stick, baby

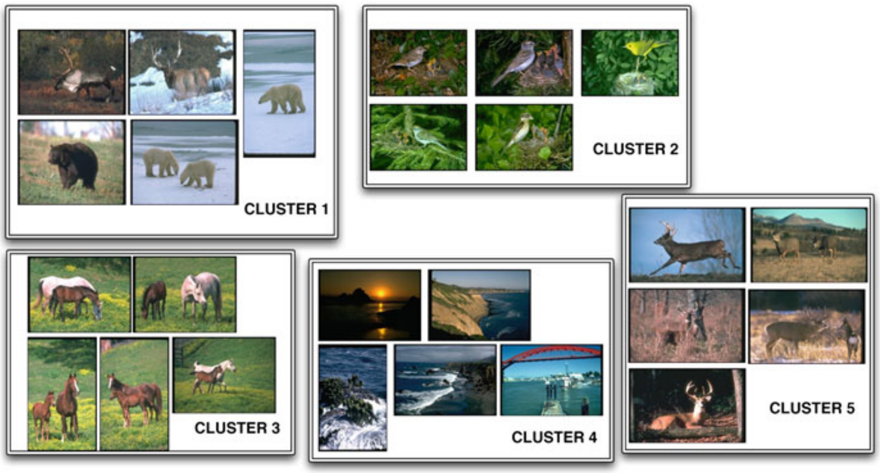
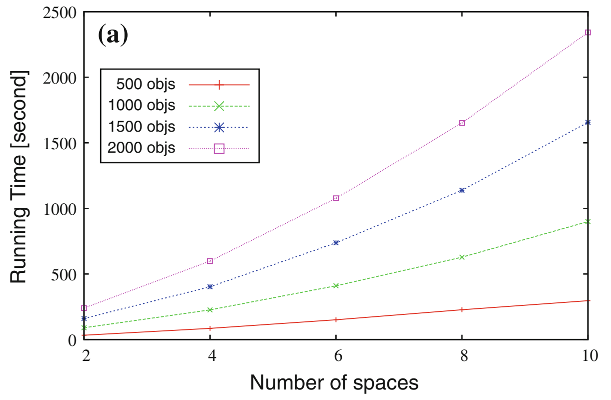


Fig. 17 I2 dataset with five representative images for each cluster

To provide a readable description of each cluster, we perform a similar process to the one used in CoStar when optimizing the objective functions. The goal is to select a subset of elements within each cluster that represents and summarizes the content of the whole cluster. For any given cluster, we try to remove each of its elements and compute the $\Delta\tau$ (see Sect. 4) that corresponds to this removal. In this case we use $\delta\tau$ as an evaluation function of the single cluster elements. We assign to each object the $\Delta\tau$ (maximum in absolute value) that as a consequence of the movement causes the highest reduction of the objective function value. Finally, we rank all the elements by the obtained values of $\Delta\tau$, and select the top k objects as representatives. Notice that we use $\Delta_{\tau_{X|\mathcal{Y}}}$ to rank images, and $\Delta_{\tau_{Y^k|X}}$ to rank words (see Sect. 4). We report the cluster descriptions for the textual view of *I2* in Fig. 16. The reader can observe that the algorithm returns five well-separated clusters of words. The first cluster gathers words around polar animals; the second one is about bridges, coasts and waterways; the topic of the third one is around animal with horns, the fourth one talks of horses and the last one is about birds and places where they live.

Figure 17 resumes the descriptions of the clusters obtained over the images. Using the top 5 ranked images, we distinguish five clusters: $Clusters1$ is about animals (reindeer and bears) that live in tundra or polar climate regions; $Clusters2$ contains pictures of birds; $Cluster3$ is about horses; $Cluster4$ is about coasts and bridges; finally, $Cluster5$ contains pictures of deer. Notice also that in this particular case it is possible to map each cluster of images onto a cluster of words and vice versa.



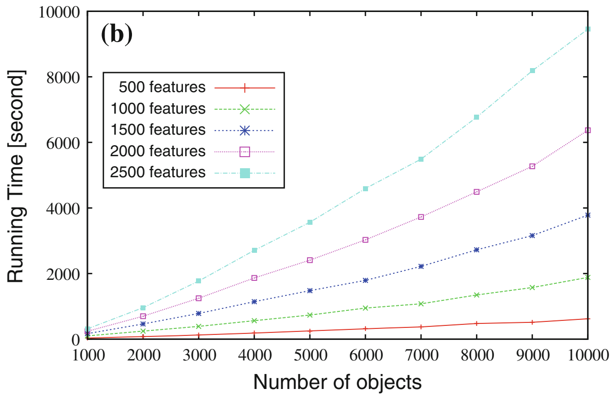


Fig. 18 Running time of CoStar varying the number of feature spaces and objects

5.3.5 Scalability evaluation

We now analyze the time performances of CoStar. We did not perform comparisons with the competitors since all algorithms are written in different programming languages, but the reader may refer to Sect. 4.4 for a theoretical complexity comparison. Here, we focus our analysis on how the number of feature spaces and the number of objects influence the computational time (see Fig. 18).

For the first experiment, we generate several synthetic boolean datasets varying the number of feature spaces. Each feature space contains 1,000 features. We let the number of feature spaces vary from 2 to 10. The number of row objects varies from 500 to 2,000. Data are generated by considering a uniform distribution and an average matrix density of 5%. The related time curves are depicted in Fig. 18a. We observe that CoStar has reasonable time performances on high-order and high dimensional data. As expected, the behavior is linear w.r.t. the number of feature spaces.

The last experiment allows us to study the scalability of our algorithms when applied to large datasets. In this case, the number of spaces is fixed (2), while we let the number of objects vary from 1,000 to 10,000. Furthermore, we repeated this experiment varying the number of features per space from 500 to 2,500. As shown in Fig. 18b, even though the complexity of CoStar is quadratic in the number of objects/features, in practice, the computational time increases linearly with the number of objects.

5.4 Empirical convergence over synthetic data

In this subsection we empirically evaluate the convergence of our method over a synthetic dataset where we know in advance the inner structure. In particular we consider a Boolean dataset $\mathcal{D}$ over $\mathcal{O}$ (1,000 objects) and $\mathcal{F} = \{F^1, F^2\}$ (1,000 features for each views). The synthetic dataset has a perfect block nested structure with perfect partitions $C_{O_1} \times C_{F^1}$ and $C_{O_2} \times C_{F^2}$. We have built $C_{O_1} \times C_{F^1}$ to have two blocks structure (two perfect clusters over the objects and two perfect clusters over the features). In the same way $C_{O_2} \times C_{F^2}$ has a four blocks structure (four perfect clusters over the objects and four perfect clusters over the features).

Let us consider a parameter p that is the probability to flip a Boolean value uniformly at random in $\mathcal{D}$ and denote by $\mathcal{D}_p$ the resulting dataset. In our experiment we vary p over 5, 10, 15 and 20%.

In order to evaluate the quality of the non-dominated solutions, we follow the protocol proposed in Knowles et al. (2006) and Liefooghe et al. (2011). Given a dataset $\mathcal{D}_p$ we perform 30 runs. Then we define Z^{all} as the union of the obtained solutions. Note that Z^{all} could contain both dominated and non-dominated vectors, since a solution may dominate another. For this reason we define $Z^{\star}$ as the set of non-dominated vectors of Z^{all}. Second, we define $z^{\min} = (z_1^{\min}, \ldots, z_{N+1}^{\min})$ and $z^{\max} = (z_1^{\max}, \ldots, z_{N+1}^{\max})$, where $z_k^{\min}$ (resp. $z_k^{\max}$) denotes the lower (resp. upper) bound of the k^{th} objective for all the points contained in Z^{all}. In order to give a roughly equal range to the objective functions, the values are normalized with respect to $z^{\min}$ and $z^{\max}$ ($\frac{value - z^{\min}}{z^{\max} - z^{\min}}$). Note that after the normalization step $z^{\min}$ is equal to $\mathbf{0}^{N+1}$ and $z^{\max}$ is equal to $\mathbf{1}^{N+1}$.

Given a solution S, in order to measure the quality of S in comparison to $Z^{\star}$, we compute the difference between these two sets by using the unary hypervolume metric (Knowles et al. 2006), where the vector $z^{\min} = \mathbf{0}^{N+1}$ is the reference point, as it corresponds to the lower bound of the $N + 1$ normalized objective functions. The hypervolume difference indicator ($I_{\overline{H}}$) computes the portion of the objective space that is weakly-dominated by $Z^{\star}$ and not by S. The closer this measure to 0, the better the approximation S. Figure 19 illustrates this measure. The light gray area corresponds to the hypervolume difference indicator.

At each iteration we compute the average of hypervolume difference indicator $I_{\overline{H}}$ of the objective functions vectors S of the 30 runs. The maximum number of iterations is set to 10,000 using the general rules employed in the experiments ($10 \times \max(m, \mathcal{N})$ where m and $\mathcal{N}$ are the number of objects and features respectively). Using this measure, we evaluate the volume of the dominated portion of the objective space and the convergence to a stable solution of CoStar. This process is repeated for each value of p. The results are reported in Fig. 20.

In Fig. 20, we observe that CoStar converges always to very similar solutions in all the runs with different values of p. As we state in Sect. 4.2, we can notice that at each

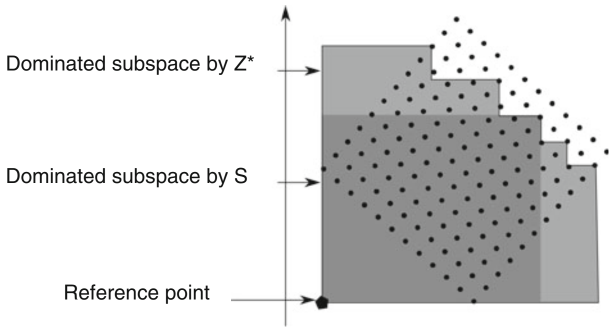


Fig. 19 Illustration of the hypervolume difference indicator

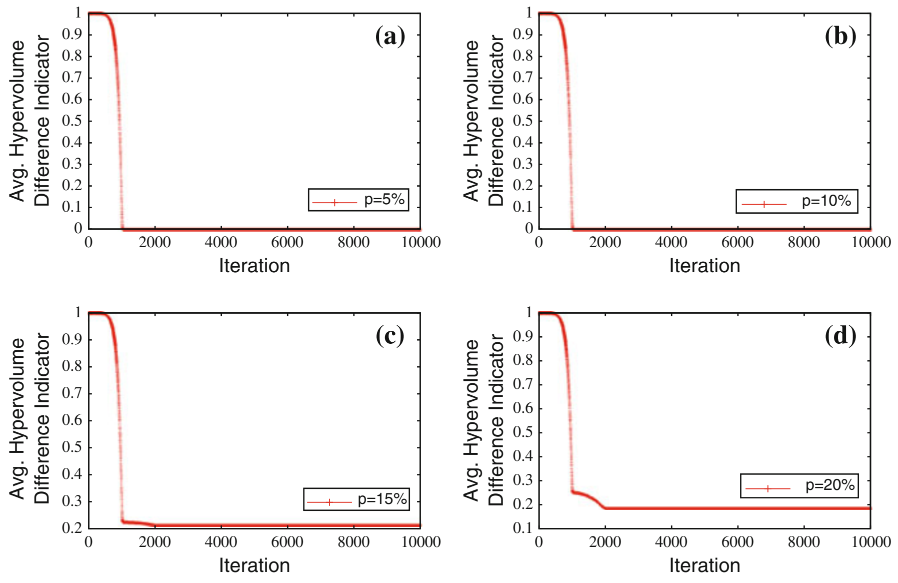


Fig. 20 The average hypervolume difference indicator at each iteration at different values of p. **a** $p = 5\%$, **b** $p = 10\%$, **c** $p = 15\%$ and **d** $p = 20\%$

iteration CoStar converges to a Pareto local optimum with respect the neighborhood. It is interesting to notice that, in this case, the algorithm converges to a stable solution very early w.r.t. the maximum number of iterations N_{iter} foreseen in CoStar. We empirically observe the same phenomenon on the real datasets, too. In particular we have observed that the maximum number of iterations is always an upper bound of the real number of steps needed by CoStar to obtain the final stable solution.

6 Conclusion

Heterogeneous star-structured data are very common in real world applications. In this paper we presented CoStar, a novel algorithm that deals with this kind of data.

Unlike previous approaches, CoStar is totally parameter-less: this means that it does not require either the number of object clusters or the number of feature clusters for each of the feature spaces. Our approach employs a Pareto-dominance approach to optimize a set of measures for cross-association in contingency tables which we extended to a high-dimensional setting. We assessed the performance of CoStar using objective evaluation measures on publicly available datasets over both the textual and image domains. We compared CoStar with three competitors, evaluate the statistical significance of the obtained results, studied the variability of the number of detected clusters and performed an in-depth qualitative analysis on an image dataset. Finally we provided a scalability study. The results show that our approach outperforms state-of-the-art methods for heterogeneous data (co-)clustering.

Acknowledgments Céline Robardet is partially funded by the French Research Agency (ANR) through the FOSTER project (ANR-2010-COSI-012-02).

References

Anagnostopoulos A, Dasgupta A, Kumar R (2008) Approximation algorithms for co-clustering. In: Proceedings of the twenty-seventh ACM SIGMOD-SIGACT-SIGART symposium on principles of database systems, PODS '08. ACM, New York, pp 201–210

Banerjee A, Dhillon I, Ghosh J, Merugu S, Modha DS (2007) A generalized maximum entropy approach to Bregman co-clustering and matrix approximation. J Mach Learn Res 8:1919–1986

Bekkerman R, Jeon J (2007) Multi-modal clustering for multimedia collections. In: Computer vision and pattern recognition, IEEE Computer Society conference on vision and pattern recognition, pp 1–8

Bickel S, Scheffer T (2004) Multi-view clustering. In: ICDM, pp 19–26

Chakrabarti D, Papadimitriou S, Modha DS, Faloutsos C (2004) Fully automatic cross-associations. In: KDD, pp 79–88

Chen Y, Wang L, Dong M (2009) Semi-supervised document clustering with simultaneous text representation and categorization. In: ECML/PKDD (1), pp 211–226

Chen Y, Wang L, Dong M (2010) Non-negative matrix factorization for semisupervised heterogeneous data coclustering. IEEE Trans Knowl Data Eng 22(10):1459–1474

Cho H, Dhillon IS, Guan Y, Sra S (2004) Minimum sum-squared residue co-clustering of gene expression data. In: Proceedings of SIAM SDM 2004

Cleuziou G, Exbrayat M, Martin L, Sublemontier JH (2009) CoFKM: a centralized method for multiple-view clustering. In: ICDM, pp 752–757

Demsar J (2006) Statistical comparisons of classifiers over multiple data sets. J Mach Learn Res 7:1–30

Dhillon IS, Mallela S, Modha DS (2003) Information-theoretic co-clustering. In: Proceedings of the ACM SIGKDD 2003. ACM Press, Washington, pp 89–98

Forman G (2003) An extensive empirical study of feature selection metrics for text classification. J Mach Learn Res 3:1289–1305

Gao B, Liu TY, Ma WY (2006) Star-structured high-order heterogeneous data co-clustering based on consistent information theory. In: ICDM '06: proceedings of the sixth international conference on data mining, pp 880–884

Goodman LA, Kruskal WH (1954) Measures of association for cross classification. J Am Stat Assoc 49:732–764

Greco G, Guzzo A, Pontieri L (2009) Co-clustering multiple heterogeneous domains: linear combinations and agreements. IEEE Trans Knowl Data Eng 22:1649–1663

Hubert L, Arabie P (1985) Comparing partitions. J Classif 2(1):193–218

Ienco D, Pensa RG, Meo R (2009) Parameter-free hierarchical co-clustering by n-ary splits. In: Proceedings of ECML/PKDD 2009. Lecture notes in computer science, vol 5781. Springer, Berlin, pp 580–595

Jaszkiewicz A (2002) Genetic local search for multi-objective combinatorial optimization. Eur J Oper Res 137(1):50–71

Keogh E, Lonardi S, Ratanamahatana CA (2004) Towards parameter-free data mining. In: Proceedings of the tenth ACM SIGKDD international conference on knowledge discovery and data mining, KDD ’04. ACM, New York, pp 206–215

Knowles J, Thiele L, Zitzler E (2006) A tutorial on the performance assessment of stochastic multiobjective optimizers. TIK Report 214, Computer Engineering and Networks Laboratory (TIK), ETH Zurich

Lee DD, Seung DD (2001) Algorithms for non-negative matrix factorization. In: Leen TK, Dietterich TG, Tresp V (eds) Advances in neural information processing systems, vol 13. MIT Press, Cambridge, pp 556–562

Liefooghe A, Humeau J, Mesmoudi S, Jourdan L, Talbi EG (2011) On dominance-based multiobjective local search: design, implementation and experimental analysis on scheduling and traveling salesman problems. J Heuristics 1–36

Long B, Zhang ZM, Wú X, Yu PS (2006) Spectral clustering for multi-type relational data. In: ICML ’06: proceedings of the 23rd international conference on machine learning, pp 585–592

Long B, Zhang ZM, Yu PS (2007) A probabilistic framework for relational clustering. In: KDD ’07: proceedings of the 13th ACM SIGKDD international conference on knowledge discovery and data mining, pp 470–479

Paquete L (2006) Stochastic local search algorithms for multiobjective combinatorial optimization: methods and analysis, vol 295. AKA Verlag/IOS Press, Berlin

Paquete L, Stützle T (2006) A study of stochastic local search algorithms for the biobjective qap with correlated flow matrices. Eur J Oper Res 169(3):943–959

Pensa RG, Boulicaut JF (2008) Constrained co-clustering of gene expression data. In: Proceedings of SIAM SDM 2008, pp 25–36

Ramage D, Heymann P, Manning CD, Garcia-Molina H (2009) Clustering the tagged web. In: WSDM, pp 54–63

Robardet C, Feschet F (2001a) Comparison of three objective functions for conceptual clustering. In: Proceedings PKDD’01, LNAI, vol 2168. Springer, Heidelberg, pp 399–410

Robardet C, Feschet F (2001b) Efficient local search in conceptual clustering. In: Proceedings DS’01, LNCS, vol 2226. Springer, Heidelberg, pp 323–335

Sheskin DJ (2007) Handbook of parametric and nonparametric statistical procedures, 4 edn. Chapman & Hall/CRC, Boca Raton

Strehl A, Ghosh J (2002) Cluster ensembles—a knowledge reuse framework for combining multiple partitions. J Mach Learn Res 3:583–617

Zar JH (1998) Biostatistical analysis, 4th edn. Prentice Hall, Englewood Cliffs