Data Mining and Knowledge Discovery (2021) 35:2313–2340
https://doi.org/10.1007/s10618-021-00788-y

Implicit consensus clustering from multiple graphs

Rafika Boutalbi[1] · **Lazhar Labiod**[2] · **Mohamed Nadif**[2]

Received: 24 February 2020 / Accepted: 5 August 2021 / Published online: 3 September 2021

Abstract
Dealing with relational learning generally relies on tools modeling relational data. An undirected graph can represent these data with vertices depicting entities and edges describing the relationships between the entities. These relationships can be well represented by multiple undirected graphs over the same set of vertices with edges arising from different graphs catching heterogeneous relations. The vertices of those networks are often structured in unknown clusters with varying properties of connectivity. These multiple graphs can be structured as a three-way tensor, where each slice of tensor depicts a graph which is represented by a count data matrix. To extract relevant clusters, we propose an appropriate model-based co-clustering capable of dealing with multiple graphs. The proposed model can be seen as a suitable tensor extension of mixture models of graphs, while the obtained co-clustering can be treated as a consensus clustering of nodes from multiple graphs. Applications on real datasets and comparisons with multi-view clustering and tensor decomposition methods show the interest of our contribution.

Keywords Three-way data · Multiple graphs · Co-clustering · Consensus

Responsible editor: Hanghang Tong

✉ Rafika Boutalbi
rafika.boutalbi@ipvs.uni-stuttgart.de

Lazhar Labiod
lazhar.labiod@u-paris.fr

Mohamed Nadif
mohamed.nadif@u-paris.fr

[1] Institute for Parallel and Distributed Systems Analytic Computing, University of Stuttgart, Stuttgart, Germany

[2] CNRS, Centre Borelli UMR 9010, Université de Paris, Paris, France

1 Introduction

Relational data are ubiquitous in various fields (web, biology, neurology, sociology, communication, economics, etc.), and their accessibility has kept increasing in recent years. These data, as a whole, form a network formalized by a graph, where each node is an entity, and each edge is a connection between a pair of nodes; this graph can be directed or not. We find this situation in various scientific publications; the relationships between documents can often be described as multiple graphs with different types of links. In fact, several relationships, such as co-terms, co-authors, co-keywords, and co-references between documents can be used. The objective of this work is to address the clustering of multiple graphs. This is a graph mining task of clustering vertices into several groups in the presence of multiple types of proximity relations. We could hypothesize that the combination of different information that arises from multiple graphs may improve the clustering results. For instance, two documents which share a number of words and/or have one or more authors in common and/or quote each other, are likely to deal with the same topic. Incorporating this additional information leads us to consider a tensor representation of the data.

To deal with multiple graphs, various models and methods under different approaches are proposed to analyze these networks. In Banerjee et al. (2007) and Tang et al. (2009), the authors proposed a multi-way clustering framework for relational data, where different types of entities are simultaneously clustered, based not only on their intrinsic attribute values, but also on the multiple relations between the entities. Other works use a spectral decomposition-based approach relying on the combination of adjacency matrices (Tang et al. 2009; Chen et al. 2017; Nie et al 2017). In these works, the clustering is not the main objective of the proposed approaches, nevertheless it can be deduced from decomposition results.

On the other hand, one of the most used methods in this context is the *Stochastic Block Model* (SBM) (Nowicki and Snijders 2001) which is a probabilistic approach. SBM is commonly used for network modeling and discovering the latent community structures from a graph. It provides a statistical approach able to model data matrix, symmetric or not, into homogeneous blocks. This leads to consider SBM (Daudin et al. 2008) as a particular case of the *Latent Block Model* (LBM) proposed by Govaert and Nadif (2003, 2005, 2006) and extended in (Shan and Banerjee 2008; Govaert and Nadif 2013), which models any kind of data matrices not necessarily square or symmetric. In other words, the clustering of the graph directed or not, is in fact, a particular case of co-clustering (Dhillon et al. 2003; Labiod and Nadif 2014; Salah and Nadif 2019; Affeldt et al. 2021). In this work, we consider graphs represented by adjacency matrices assimilated to contingency tables. Thus, considering the previous example of document clustering, the relations between documents (co-terms, co-authors, etc.) are count data and can be represented by particularly sparse contingency tables. Many works in the literature show the interest of Poisson distribution for graph theory and clustering of random graphs (Janson 1987; Daudin et al. 2008).

To the best of our knowledge, this is the first attempt to formulate a model-based co-clustering for sparse three-way data. To this end, we rely on the latent block model (Govaert and Nadif 2013) for its flexibility to consider any data matrices. Figure 1 presents a binary three-way dataset constructed from multiple graphs, and the expected

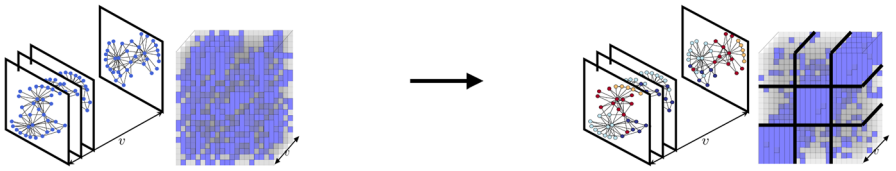


Fig. 1 Goal of clustering of multiple graphs

results in terms of co-clustering. This leads us to consider our objective as a problem of consensus clustering from different sources.

Consensus clustering, also called cluster ensembles, refers to the situation in which different clusterings have been obtained from a dataset, and it is desired to find a single consensus clustering that is a better fit in some sense than the existing clusterings. Thereby, consensus clustering aims to reconcile clustering information about the same data set coming from different runs of the same algorithm or different algorithms. This kind of consensus is referred to, in the paper, as *explicit consensus* clustering. On the other hand, we will aim to obtain a consensus clustering from different sources (slices) with the same algorithm as in our case. We refer to this type of consensus clustering as *implicit consensus* clustering. The key contributions of this work are:

- We first establish the links between *Poisson Latent Block Model* (PLBM) and *Poisson Stochastic Block Model* (PSBM). Then we show the interest of considering PLBM rather than PSBM.
- We propose a *Sparse* PLBM (SPLBM), a suitable probabilistic model for clustering of multiple graphs. Then we derive an EM-type learning algorithm.
- We perform extensive numerical experiments and compare our proposal with multi-view and tensor decomposition methods.
- Finally, using the ensemble method, we prove that the proposed algorithm, which can be viewed as an *implicit* consensus clustering for multiple graphs, is more effective than *explicit* clustering obtained by traditional consensus clustering methods.

The remainder of this paper is organized as follows. In Sect. 2, we present related work and show the strong points of our approach. Section 3 reviews PLBM, shows the limits of traditional PSBM and describes Sparse PLBM (SPLBM). Sect. 4 discusses the extension of SPLBM to consider multiple graphs. In Sect. 5, we present a variational Expectation-Maximization algorithm. Sect. 6 is devoted to evaluating our approach. Finally, Sect. 7 concludes the paper and gives some directions for future research.

2 Related work

Although SBM is popular in social networks analysis, dealing with the count data and due to the degree of heterogeneity, the traditional SBM fail to detect relevant clusters of edges to adress community detection problem (Qiao et. al 2017). Thereby, several authors have developed a degree-corrected SBM. In Karrer and Newman (2011), using a Poisson SBM, they introduced a parameter θ_i controlling the degree of expected degrees of vertices i. They consider that each x_{ij} with $i \neq j$ is distributed accord-

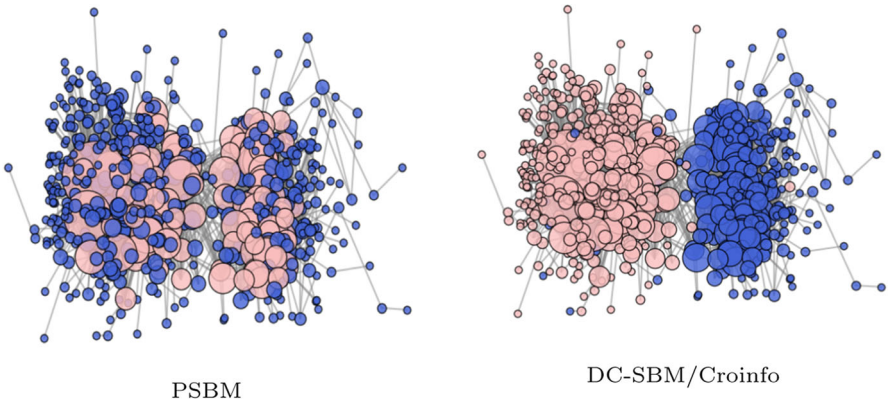


Fig. 2 Political blogs dataset: clustering with `PSBM` and `DC-SBM/Croinfo`

ing to Poisson($\theta_i\theta_j\delta_{k\ell}$), where $\delta_{k\ell}$ is the expected value of the adjacency matrix for the vertices i and j lying in block (k, ℓ) while x_{ii} is distributed according to Poisson($\frac{1}{2}\theta_i^2\delta_{kk}$). Doing so and under some constraints on the θ_i's, they proposed the DC-SBM (Degree-Corrected SBM) clustering algorithm (`DC-SBM`[1]) from an undirected graph on n vertices, possibly including self-edges. Furthermore, they established the equivalence between the maximization of the log-likelihood and the maximization of mutual information used as an objective function for clustering bipartite graphs (Dhillon et al. 2003). It is important to emphasize that the model proposed in Karrer and Newman (2011) is similar to that proposed by Nadif and Govaert (2005), where the authors also showed this connection with the maximization of mutual information; they proposed the `Croinfo` algorithm as illustrated in Fig. 2. In fact, the objective function maximized by `DC-SBM`, which can also be used for the co-clustering of an undirected graph, is associated with a *constrained* Poisson LBM commonly used in the co-clustering context; see e.g.; Ailem et al. (2017a, b) and Role et al. (2019). To sum up, considering `DC-SBM` which implies that the data are generated according to a Poisson LBM with $\mathcal{P}(x_{ij}, x_{i.}x_{.j}\gamma_{k\ell})$ where $\mathcal{P}(x_{ij}; \lambda) = \frac{e^{-\lambda}\lambda^{x_{ij}}}{x_{ij}!}$, the proportions of the classes of the nodes are assumed to be equal. In addition, although both algorithms `DC-SBM` or `Croinfo` are different, the objective is the same, and the clustering considered is based on an approach similar to that of the traditional hard clustering algorithms; for more detail, the reader can refer to recent works (Govaert and Nadif 2013, 2018).

In our contribution, we structured graphs as three-way data where the clustering is the principal objective. We propose an extension of LBM to tackle the co-clustering of multiple undirected/directed graphs where each cell of the diagonal is not necessarily equal to an even number as conventionally considered in community detection. To do this, we adopt an EM-type approach to refer to the Expectation-Maximization algorithm (Dempster et al. 1977; McLachlan and Peel 2000) and not Classification

[1] In the paper, to distinguish between a model and its derived algorithm we use *typewriter font* for an algorithm, thereby DC-SBM is the model and `DC-SBM` its derived algorithm.

EM (Celeux and Govaert 1992). Furthermore, we will show that this purpose can be viewed as an implicit consensus clustering from Multiple Graphs.

3 Poisson latent and stochastic block models

Given an $n \times d$ data matrix $\mathbf{X} = (x_{ij}, i \in I = \{1, \ldots, n\}; j \in J = \{1, \ldots, d\})$, it is assumed that there exists a partition on I and a partition on J. A pair of partitions $(\mathbf{Z}, \mathbf{W})$ will represent a partition of $I \times J$ into $g \times m$ blocks. The partition $\mathbf{Z}$ for rows can be represented by a label vector $(z_1, \ldots, z_n)$ where $z_i \in \{1, \ldots, g\}$ or a binary matrix $\mathbf{Z} = (z_{ik}) \in \{0, 1\}^{n \times g}$ satisfying $\sum_{k=1}^{g} z_{ik} = 1$. In the same manner the partition $\mathbf{W}$ for columns can be represented by a label vector $(w_1, \ldots, w_d)$ where $w_j \in \{1, \ldots, m\}$ or a binary matrix $\mathbf{W} = (w_{j\ell}) \in \{0, 1\}^{d \times m}$ satisfying $\sum_{\ell=1}^{m} w_{j\ell} = 1$.

3.1 Poisson latent block model (PLBM)

Denoting $\mathcal{Z}$ and $\mathcal{W}$ the sets of possible labels $\mathbf{Z}$ for I and $\mathbf{W}$ for J, the marginal density function $f(\mathbf{X}; \boldsymbol{\Omega})$ of the *Poisson Latent Block Model* (PLBM) (Govaert and Nadif 2018) can be written

$$f(\mathbf{X}, \boldsymbol{\Omega}) = \sum_{(\mathbf{z},\mathbf{w}) \in \mathcal{Z} \times \mathcal{W}} \prod_{i,k} \pi_k^{z_{ik}} \prod_{j,\ell} \rho_\ell^{w_{j\ell}} \prod_{i,j,k} \mathcal{P}(x_{ij}; x_{i.} x_{.j} \gamma_{k\ell})^{z_{ik} w_{j\ell}} \tag{1}$$

where $\boldsymbol{\Omega} = (\boldsymbol{\pi}, \boldsymbol{\rho}, \boldsymbol{\gamma})$, with $\boldsymbol{\pi} = (\pi_1, \ldots, \pi_g)$ and $\boldsymbol{\rho} = (\rho_1, \ldots, \rho_m)$ where $(\pi_k = P(z_{ik} = 1), k = 1, \ldots, g)$, $(\rho_\ell = P(w_{j\ell} = 1), \ell = 1, \ldots, m)$ are the mixing proportions of row and column clusters respectively, and $\boldsymbol{\gamma} = (\gamma_{k\ell}; k = 1, \ldots g, \ell = 1, \ldots, m)$. For this model, the complete data are taken to be the vector $(\mathbf{X}, \mathbf{Z}, \mathbf{W})$ where unobservable $\mathbf{Z}$ and $\mathbf{W}$ lead to the labels, the resulting complete data log-likelihood can be written as follows:

$$\begin{aligned} L_C(\mathbf{Z}, \mathbf{W}, \boldsymbol{\Omega}) &= \log f(\mathbf{X}, \mathbf{Z}, \mathbf{W}; \boldsymbol{\Omega}) \\ &= \sum_{i,k} z_{ik} \log \pi_k + \sum_{j,\ell} w_{j\ell} \log \rho_\ell + \sum_{i,j,k,\ell} z_{ik} w_{j\ell} \log \mathcal{P}(x_{ij}; x_{i.} x_{.j} \gamma_{k\ell}). \end{aligned}$$

To estimate $\boldsymbol{\Omega}$, we consider the EM algorithm (Dempster et al. 1977). However, the E-step using the log-likelihood of (1) directly is intractable due to the dependence structure among the rows and columns. Govaert and Nadif (2005) suggest a variational approximation in relying on the interpretation of EM due to Neal and Hinton (1998). This leads to maximize the following lower bound of the log-likelihood criterion:

$$L_C(\widetilde{\mathbf{Z}}, \widetilde{\mathbf{W}}, \boldsymbol{\Omega}) + H(\widetilde{\mathbf{Z}}) + H(\widetilde{\mathbf{W}}) \tag{2}$$

where $L_C(\widetilde{\mathbf{Z}}, \widetilde{\mathbf{W}}, \boldsymbol{\Omega})$ is the fuzzy complete-data log-likelihood. $H(\widetilde{\mathbf{Z}}) = -\sum_{i,k} \widetilde{z}_{ik} \log \widetilde{z}_{ik}$ with $P(z_{ik} = 1|\mathbf{X}) = \widetilde{z}_{ik}$, and $H(\widetilde{\mathbf{W}}) = -\sum_{j,\ell} \widetilde{w}_{j\ell} \log \widetilde{w}_{j\ell}$ with $P(w_{j\ell} = 1|\mathbf{X}) = \widetilde{w}_{j\ell}$ are the entropies.

3.2 Poisson stochastic block model

As we mentioned earlier, Poisson SBM, even DC-SBM, are particular cases of Poisson LBM insofar as the latter can model matrices, symmetric or not, oriented or non-oriented graphs, numbers of row clusters and columns clusters not necessarily equal ($g \neq m$) and finally with proportions of clusters equal or not. Therefore the transition from LBM to SBM is easy to show. Thereby, for undirected graph, the maximization of (2) leads to maximizing

$$L_C(\widetilde{\mathbf{Z}}, \boldsymbol{\Omega}) + 2H(\widetilde{\mathbf{Z}})$$

which is proportional to

$$\sum_{i,k} \widetilde{z}_{ik} \log \pi_k + \frac{1}{2} \sum_{i \neq j, k \neq \ell} \widetilde{z}_{ik} \widetilde{w}_{j\ell} \log \mathcal{P}(x_{ij}; x_{i.} x_{.j} \gamma_{k\ell}) + \frac{1}{2} \sum_{i,k} \widetilde{z}_{ik} \log \mathcal{P}(x_{ii}; x_{i.} x_{i.} \gamma_{kk}) - \sum_{i,k} \widetilde{z}_{ik} \log \widetilde{z}_{ik}.$$

The main differences between both models are a) with the Poisson SBM, the third term which concerns the diagonal of $\mathbf{X}$ is ignored and it does not take into account the degree of nodes unlike LBM, b) with the Poisson LBM, $x_{ij}|z_{ik}w_{j\ell} = 1 \sim \mathcal{P}(x_{i.}x_{.j}\gamma_{k\ell})$, while with SBM $x_{ij}|z_{ik}w_{j\ell} = 1 \sim \mathcal{P}(\gamma_{k\ell})$.

Notice that $\gamma_{k\ell}$ depends only on the block $k\ell$ and not on the margins. Thereby, starting from PLBM, we will see next how to take into account the sparsity often present in the graphs.

3.3 PLBM for sparse data: sparse PLBM (SPLBM)

Recently, in Ailem et al. (2017b), the authors proposed a generative mixture model for co-clustering document-term matrices referred to as SPLBM. With this model, they assume that for each diagonal block kk the values $x_{ij} \sim \text{Poisson}(\lambda_{ij})$ where

$$\lambda_{ij} = x_{i.}x_{.j} \sum_{k} [z_{ik}w_{jk}]\gamma_{kk} \quad or \quad x_{ij}|z_{ik}w_{jk} = 1 \sim \mathcal{P}(x_{i.}x_{.j}\gamma_{kk})$$

and for each block $k\ell$ with $k \neq \ell$, $x_{ij} \sim \text{Poisson}(\lambda_{ij})$ where the parameter λ_{ij} takes the following form:

$$\lambda_{ij} = x_{i.}x_{.j} \sum_{k,\ell \neq k} [z_{ik}w_{j\ell}]\gamma \quad or \quad x_{ij}|z_{ik}w_{j\ell} = 1 \sim \mathcal{P}(x_{i.}x_{.j}\gamma).$$

Assuming $\forall \ell \neq k$, $\gamma_{k\ell} = \gamma$ leads to suppose that all blocks outside the diagonal share the same parameter. SPLBM has been designed from the ground up to deal with data sparsity problems. As a consequence, in addition to seeking homogeneous blocks,

it also filters out homogeneous but noisy ones due to the sparsity of the data. The pdf of SPLBM can be written as follows:

$$f(\mathbf{X}, \mathbf{\Omega}) = \sum_{(\mathbf{z},\mathbf{w})\in\mathcal{Z}\times\mathcal{W}} \prod_{i,k} \pi_k^{z_{ik}} \prod_{j,k} \rho_k^{w_{jk}} \prod_{i,j,k} \times \left(\mathcal{P}(x_{ij};\lambda_{kk})\right)^{z_{ik}w_{jk}} \prod_{i,j,k,\ell\neq k} \left(\mathcal{P}(x_{ij};\lambda)\right)^{z_{ik}w_{j\ell}}.$$

Assuming that the complete data are $(\mathbf{X}, \mathbf{Z}, \mathbf{W})$, the complete data log-likelihood $L_C(\mathbf{Z}, \mathbf{W}, \mathbf{\Omega})$ takes the following form :

$$\log\left(\prod_{i,k}\pi_k^{z_{ik}}\prod_{j,\ell}\rho_k^{w_{jk}}\prod_{i,j,k}\left(\frac{e^{-x_{i.}x_{.j}\gamma_{kk}}(x_{i.}x_{.j}\gamma_{kk})^{x_{ij}}}{x_{ij}!}\right)^{z_{ik}w_{jk}} \prod_{i,j,k,\ell\neq k}\left(\frac{e^{-x_{i.}x_{.j}\gamma}(x_{i.}x_{.j}\gamma)^{x_{ij}}}{x_{ij}!}\right)^{z_{ik}w_{j\ell}}\right). \tag{3}$$

To estimate the parameters $\mathbf{\Omega}$, $\mathbf{Z}$ and $\mathbf{W}$, a variationnel EM has been proposed (Ailem et al. 2017b) to maximize (2).

Note that although SPLBM is a co-clustering model, we can derive a graph clustering algorithm from an adjacency matrix (symmetric or not). Thereby, when we are dealing with undirected graphs; strating with the same initialization of $\mathbf{z}$ and $\mathbf{w}$ ($\mathbf{z}^{(0)} = \mathbf{w}^{(0)}$), we obtain the same row and column clusters, that is essential for the undirected graph clustering problem.

3.4 PSBM, PLBM and SPLBM for graphs

Although PLBM can deal with sparse matrices, SPLBM can be more suitable for sparse matrices (Fig. 3). It is designed to seek a diagonal block structure and capture the most reliable associations between the rows and columns object clusters. SPLBM assumes that each diagonal block (or co-cluster) is generated according to the Poisson distribution with some specific parameters, and each non-diagonal co-cluster representing noise data is generated according to Poisson distribution with identical parameters. In Fig. 4 we report the graphical models of Poisson models discussed in the paper.

To clarify expectations and the impact of this parameterization, on the political blogs dataset,[2] we applied the clustering algorithms derived from SBM, PLBM, and SPLBM, using 30 random initializations and measured the clustering accuracy. Figure 5 shows the interest of SPLBM, which takes into account the sparsity often present in a graph network.

The properties of this parameterization prompt us to adopt it for co-clustering with multiple graphs, as illustrated in Fig. 1. Next, to avoid confusion between all the rows and columns that are identical in our case, we still keep the notations using the z_{ik}'s and $w_{j\ell}$'s.

[2] https://dl.acm.org/citation.cfm?id=1134277.

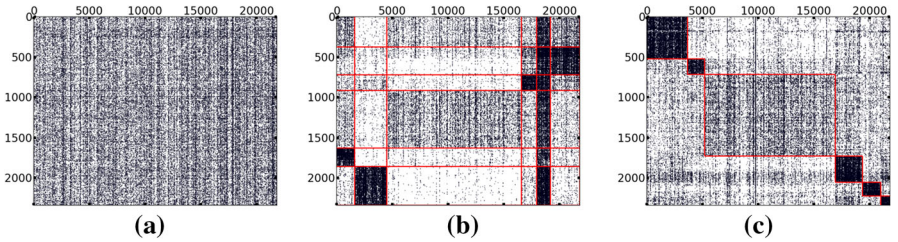


Fig. 3 **a** Original data, **b** co-clustering according PLBM and **c** co-clustering according SPLBM

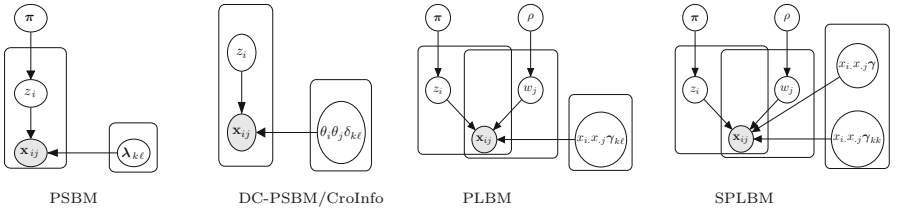


Fig. 4 Graphical models: z_i is the label of row i, w_j is the label of column j

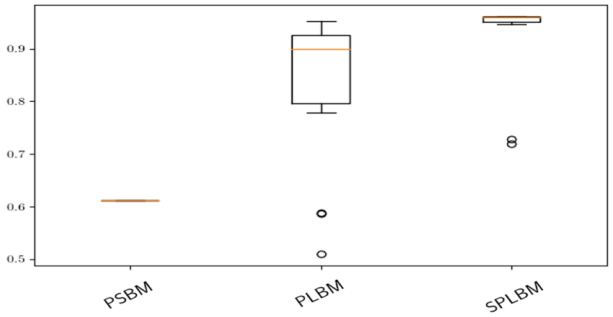


Fig. 5 Political blogs dataset: comparison of PSBM, PLBM, and SPLBM in terms of accuracy

The presented models PSBM, PLBM, and SPLBM deal with adjacency matrices (2D data matrix) to tackle the problem of graph clustering. In the sequel, we deal with multiple graphs organised as 3D data matrix; each matrix depicts a graph.

4 SPLBM with multiple graphs

4.1 Three-way tensor characteristics

A tensor is a multidimensional array, which is also known as the N-way, Nth-order tensor. A tensor can be viewed as an element product of N vector spaces (Kolda and Bader 2009). This notion of tensors should not be confused with tensors in physics and mathematics fields such as stress and strain tensors (Frankel 2012).

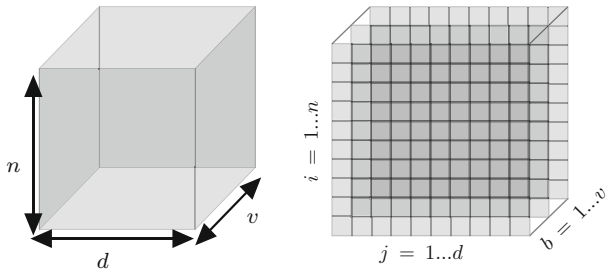
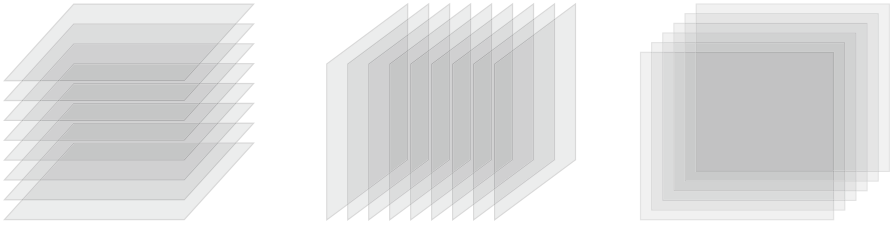


Fig. 6 Third-way tensor data representation



(a) Horizontal slices. **(b)** Lateral slices. **(c)** Frontal slices.

Fig. 7 Slices representations of the three-way tensor

A three-way tensor or third-order tensor has three dimensions and then has three indices, as shown in Fig. 6. A first-order tensor is a vector, a second-order tensor is a matrix, and tensors of order three or higher are called higher-order tensors.

The notation used here is very close to that introduced by Kiers (2000) for third-order tensor. Notice that scalars are represented by lowercase letters e.g. x, and vectors are expressed by a bold lowercase letter e.g. $\mathbf{x}$. The matrices are denoted by bold capital letters e.g. $\mathbf{X}$. And finally, tensors are indicated by bold capital Euler letters e.g. $\mathcal{X}$. The ith element of vector $\mathbf{x}$ is denoted as x_i, the element (i, j) of a matrix is expressed by x_{ij}, and x_{ij}^b represents the element (i, j, b) of a tensor.

The order of tensor is referred to as the number of dimensions, also called ways or modes. One-mode tensor is a vector, second-order tensor is a matrix, and third-order tensor is a cuboid. In the case of matrix $\mathbf{X}$, a row and column can be denoted by $\mathbf{x}_{i:}$ and $\mathbf{x}_{:j}$, respectively. In the case of three-way tensor $\mathbf{x}_{ij:}$, $\mathbf{x}_{i:b}$, and $\mathbf{x}_{:jb}$ represents the vector of the three different modes respectively. As we consider *frontal slices*, the tensor can be represented by $\{\mathbf{X}^b, b = 1 \ldots, v\}$ (Fig. 7c); this is the most often chosen representation. For convenience, in the following, we will denote the tensor entry $\mathbf{x}_{ij:}$ by $\mathbf{x}_{ij} = (x_{ij}^1, \ldots, x_{ij}^b, \ldots, x_{ij}^v)$ (Fig. 8); then $x_{i.}^b = \sum_j x_{ij}^b$ and $x_{.j}^b = \sum_i x_{ij}^b$. In this sequel, we aim to extract homogeneous sub-tensors from three-way data.

4.2 Definition of the proposed model

We extend SPLBM to Three-way tensor data leading to *Tensor* SPLBM (or TSPLBM). The proposed model seeks not only to discover homogeneous tube co-clusters (a three

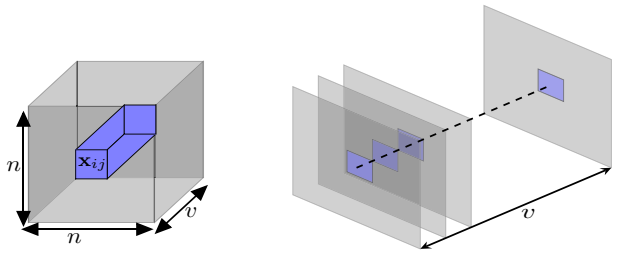


Fig. 8 The three-way tensor structure

dimensional co-clusters) but also discover important blocks and ignore noisy ones. Thereby, TSPLBM allows to discover a diagonal co-clusters structure, which are tubes (through all slices) from the three-way tensor. It makes it more useful for sparse tensor with high sparsity close to 90%, as shown in the experiments. TSPLM provides a better partitioning than the classical co-clustering algorithm applied on each slice of tensor separately or a consensus clustering used on these independent results.

Our proposal Tensor SPLBM considers 3D data matrix $\mathcal{X} = [\mathbf{x}_{ij}] \in \mathbb{R}^{n \times n \times v}$ where n is the number of nodes, and v the number of graphs (slices). Figure 1 presents a tensor data with v graphs. As $\mathcal{X}$ is symmetric per slice b, when $i = j$ we have $z_{ik} = w_{jk}$ and for $k = 1, \ldots, g$ we have $\pi_k = \rho_k$. This leads to deduce the fuzzy complete data log-likelihood $\mathcal{L}_C(\widetilde{\mathbf{Z}}, \mathbf{\Omega})$ from (3)

$$\mathcal{L}_C(\widetilde{\mathbf{Z}}, \mathbf{\Omega}) = 2 \sum_{i,k} \widetilde{z}_{ik} \log \pi_k + \sum_{i,j,k} \widetilde{z}_{ik} \widetilde{z}_{jk} \sum_{b=1}^{v} \log \mathcal{P}(x_{ij}^b; x_{i.}^b x_{.j}^b \gamma_{kk}^b) + \sum_{i,j,k,\ell \neq k} \widetilde{z}_{ik} \widetilde{z}_{j\ell} \sum_{b=1}^{v} \log \mathcal{P}(x_{ij}^b; x_{i.}^b x_{.j}^b \gamma^b).$$

and the lower bound of log-likelihood criterion noted $\mathcal{F}_C(\widetilde{\mathbf{Z}}, \mathbf{\Omega})$ ("Appendix A" for more details). Thus, to estimate Ω and $\widetilde{\mathbf{Z}}$, from which we can deduce $\mathbf{Z}$, we optimize

$$\frac{1}{2}\mathcal{F}_C(\widetilde{\mathbf{Z}}, \mathbf{\Omega}) = \frac{1}{2}\mathcal{L}_C(\widetilde{\mathbf{Z}}, \mathbf{\Omega}) + H(\widetilde{\mathbf{Z}}) \tag{4}$$

where $H(\widetilde{\mathbf{Z}}) = -\sum_{i,k} \widetilde{z}_{ik} \log \widetilde{z}_{ik}$ is the entropy.

After some algebraic calculations, we can simplify the criterion (up a constant) that takes the following form ("Appendix B" for more details)

$$\sum_{i,k} \widetilde{z}_{ik} \log \pi_k + \frac{1}{2} \sum_b \left(\sum_k \left[x_{kk}^b \log\left(\frac{\gamma_{kk}^b}{\gamma^b}\right) - [x_{k.}^b]^2 (\gamma_{kk}^b - \gamma^b) \right] + N_b (\log(\gamma^b) - N_b \gamma^b) \right) + H(\widetilde{\mathbf{Z}}) \tag{5}$$

where $x_{k.}^b = \sum_i \widetilde{z}_{ik} x_{i.}^b = \sum_j \widetilde{z}_{jk} x_{.j}^b = x_{.k}^b, x_{kk}^b = \sum_{i,j} \widetilde{z}_{ik} \widetilde{z}_{jk} x_{ij}^b$, and $N_b = \sum_{i,j} x_{ij}^b$.

5 Variational inference

To estimate the parameters of the model, we rely on the Variational EM algorithm (Govaert and Nadif 2005), and we extend it to multiple graphs. In the sequel, the proposed algorithm is referred to as TSPLBM.

E-step It consists in computing, for all i, j, k the posterior probabilities $\widetilde{z}_{ik}$ and $\widetilde{z}_{jk}$ given the estimated parameters $\boldsymbol{\Omega}$. As $\sum_k \widetilde{z}_{ik} = \sum_k \widetilde{z}_{jk} = 1$, using the corresponding Lagrangians, up to terms which are not function of $\widetilde{z}_{ik}$, leads to

$$\log \widetilde{z}_{ik}^{(t+1)} \propto \log \pi_k + \frac{1}{2}\left(\sum_j \widetilde{z}_{jk}^{(t)} \sum_{b=1}^{v} \mathcal{P}_{kk}^{ijb} + \sum_{j\neq i, k\neq \ell} \widetilde{z}_{j\ell}^{(t)} \sum_{b=1}^{v} \mathcal{P}_{k\ell}^{ijb} \right), \tag{6}$$

where $\mathcal{P}_{kk}^{ijb} = \log \mathcal{P}(x_{ij}^b; x_{i.}^b x_{.j}^b \gamma_{kk}^b)$ and with $k \neq \ell$, $\mathcal{P}_{k\ell}^{ijb} = \log \mathcal{P}(x_{ij}^b; x_{i.}^b x_{.j}^b \gamma^b)$. The update of $\widetilde{z}_{ik}^{(t+1)}$, in a simple form (Algorithm 1), is described in "Appendix C" where $\widetilde{z}_{ik}^{(t)}$ represents the value of $\widetilde{z}_{ik}$ in the previous iteration (t).

M-step Given the previously computed posterior probabilities $\widetilde{\mathbf{Z}}$, the M-step consists in updating, $\forall k$, the parameters π_k, $\boldsymbol{\gamma}_{kk}^b$ and $\boldsymbol{\gamma}^b$. The estimated parameters are defined as follows. First, taking into account the constraints $\sum_k \pi_k = 1$, it is easy to show that $\pi_k = \frac{\sum_i \widetilde{z}_{ik}}{n}$. Secondly, it is easy to obtain for all b, k ("Appendix C")

$$\boldsymbol{\gamma}_{kk}^b = \frac{\sum_{i,j} \widetilde{z}_{ik}\widetilde{z}_{jk} x_{ij}^b}{\sum_i \widetilde{z}_{ik} x_{i.}^b \sum_j \widetilde{z}_{jk} x_{.j}^b} = \frac{x_{kk}^b}{[x_{k.}^b]^2}$$

$$\text{and, } \boldsymbol{\gamma}^b = \frac{N_b - \sum_{i,j,k} \widetilde{z}_{ik}\widetilde{z}_{jk} x_{ij}^b}{N_b^2 - \sum_k \sum_i \widetilde{z}_{ik} x_{i.}^b \sum_j \widetilde{z}_{jk} x_{.j}^b} = \frac{N_b - \sum_k x_{kk}^b}{N_b^2 - \sum_k [x_{k.}^b]^2}. \tag{7}$$

The TSPLBM algorithm (Algorithm 1) for multiple graphs alternates the two previously described Expectation-Maximization steps until the objective function value (4) change is small or there is no change. At the convergence, a hard co-clustering where each data point either belongs to a cluster completely or not is deduced from $\widetilde{z}_{ik}$'s using the maximum a posterior principle defined by $\forall i, z_i \in \{1, \ldots, g\}$ is given by $z_i = \arg\max_{k=1..g} \widetilde{z}_{ik}$.

The computational complexity of the TSPLBM algorithm scales linearly with the number of non-zero entries. Let us denote nz the number of non-zero entries in $\mathcal{X}$, it the number of iterations, g the number of clusters and v the number of slices; the computational complexity is given in $O(it \cdot g \cdot v \cdot nz)$.

6 Experiments

The objective of our experiments is fivefold. First, we discuss some connections between TSPLBM and multiview clustering (Sect. 6.2). Secondly, we evaluate the

Algorithm 1: TSPLBM

Input: $\mathcal{X}$, g.
Initialization: $\mathbf{Z}^{(0)}$ randomly and compute $\mathbf{\Omega}^{(0)}$, $t = 0$
repeat
E-Step: Compute $\tilde{z}_{ik}^{(t+1)}$
$\tilde{z}_{ik}^{(t+1)} \propto \pi_k \exp\left(\frac{1}{2}\sum_j \tilde{z}_{jk}^{(t)} \sum_{b=1}^{v} x_{ij}^b \log(\frac{\gamma_{kk}^b}{\gamma^b})\right)$
M-Step: Update $\mathbf{\Omega}^{(t+1)} = (\pi_k^{(t+1)}, (\boldsymbol{\gamma}_{kk}^b)^{(t+1)}, (\boldsymbol{\gamma}^b)^{(t+1)})$ given by
$\pi_k = \frac{\sum_i \tilde{z}_{ik}^{(t+1)}}{n}$, $\boldsymbol{\gamma}_{kk}^b = \frac{x_{kk}^b}{[x_{k.}^b]^2}$, and $\boldsymbol{\gamma}^b = \frac{N_b - \sum_k x_{kk}^b}{N_b^2 - \sum_k [x_{k.}^b]^2}$
until *the objective function value (4) change is small, or there is no change*;
return Z, Ω

Table 1 Characteristics of datasets

Datasets	Type	#Graphs	#Nodes	#Clusters	Sparsity (%)
UC-digits	Images	6	2000	10	98
100leaves	Images	3	1600	100	98
3sources	Text	3	169	6	80
BBC	Text	4	685	5	88
DBLP1	Text	3	2223	3	96
Nus-Wide-8	Text + Images	6	2738	8	83
DBLP3	Text	3	12,550	10	99
Amazon-products-10	Text + Images	7	9897	10	98

interest to consider multiple graphs simultaneously by TSPLBM, unlike tensor decomposition methods that consider a reduced matrix arising from multiple graphs (Sect. 6.3). Thirdly, we evaluate the impact of considering multiple graphs (Sect. 6.4). Fourthly, we show how we can harness the results obtained by TSPLBM (Sect. 6.5). Finally, we show that TSPLBM can be viewed as an implicit consensus clustering and propose a solution to increase its clustering performance in an *ensemble method* framework (Sect. 6.6).

6.1 Datasets description and pre-processing

We used eight datasets with a different number of graphs (slices) and clusters. Table 1 shows the characteristics of datasets in terms of the type of instances (image or image+text), the number of graphs/slices (#Graphs), the number of instances (#Nodes), the number of clusters (#Clusters) and the rate of sparsity.

Fig. 9 Amazon-products-10 dataset reorganized according to the true clusters

We selected four benchmark datasets[3] commonly used to compare multi-view clustering methods, namely UC-digits, 3sources, BBC, 100leaves. Further, we constructed four datasets for multiple graphs clustering, namely DBLP1, DBLP3, Nus-Wide-8, and Amazon-products-10. Hereafter, we give in detail the description of each dataset

- UC-digits consists of 2000 images of handwritten digits (including ten classes correspond to the number 0–9) described by six views Fourier coefficients of the character shapes, profile correlations, Karhunen-Love coefficients, pixel averages, Zernike moments, and morphological features.
- *3sources* consists of 169 news texts reported by three newspaper sources BBC, Reuters, and The Guardian.
- BBC consists of 658 documents from BBC news splited into four segments and addressing five different topics.
- 100leaves consists of 1600 images from one hundred plant species and described by shape descriptor, fine-scale margin, and texture histogram features.
- DBLP1 consists of 2223 papers published in three different journals and described by words from title, words from abstract, and authors.
- DBLP3 is similar to the DBLP1 dataset but including 12,550 papers from ten journals.
- Nus-Wide-8 consists of 2738 images from *Flickr* addressing eight topics and described by tags, Color Histogram (CH), Color Correlogram (CORR), Edge direction histogram (EDH), Wavelet texture (WT), and block-wise color moments (CW55).
- Amazon-products-10 consists of 9897 product images from ten product categories and is described by words of product title, words of the product description, LBP features, Haralick features, and Gabor features, co-viewed and co-purchased products. Figure 9 shows all graphs (slices) reorganized according to the true partition into 10 classes.

In these tensor datasets, each (slice) graph can be assimilated to adjacency matrices representing similarities between nodes (objects). Note that the `TSPLBM` model considers count or binary adjacency matrices. Thereby, in order to apply `TSPLBM` for image datasets where graphs represent similarities between images according to each type of feature, we had to convert these matrices into binary adjacency matrices (1 if the similarity is higher than ninety-seven percent quantile and 0 otherwise). In this way, we were able to study the robustness of our algorithm even when one or many slices in original data do not respect the expected structure –binary or count data–.

6.2 TSPLBM versus multi-view clustering

The multi-view clustering (MvC) (Bickel and Scheffer 2004) aims to perform clustering from diverse sources or domains, where each object (instance) is described by several sets of features (or views). The MvC methods are used in several applications, such as image clustering, where we can have different kinds of features. They allow to taking into account the information arising from each view. Because of the diversity of feature sets, each view can be converted to a symmetric instances $\times$ instances similarity/dissimilarity matrix. This brings us back to a tensor representation of these views

[3] https://github.com/KunyuLin/Multi-view-Datasets.

Table 2 Mutiview clustering performance comparison

Datasets	MultiNMF		SwMV			TSPLBM		
	ACC	NMI	ACC	NMI	Purity	ACC	NMI	Purity
UC-digits	0.88	0.80	**0.94**	**0.91**	**0.95**	0.74	0.80	0.76
3sources	0.48	0.46	0.35	0.10	0.36	**0.66**	**0.54**	**0.66**
BBC	0.48	0.33	0.33	0.05	0.33	**0.66**	**0.66**	**0.66**
100leaves	**0.67**	0.86	0.59	**0.87**	**0.61**	0.46	0.81	0.46
DBLP1	–	–	NA	NA	NA	**0.83**	**0.57**	**0.85**
Nus-Wide-8	–	–	0.28	0.004	0.28	**0.56**	**0.41**	**0.56**

– means that we could not retrieve the results for MultiNMF for these datasets
NA means that the SwMV algorithm could not find clustering solution
Bold values indicate the best performances in terms of ACC, NMI and Purity

where each of them is a graph where the edges are continuous. Thereby, even though each view is not a count matrix, we compared TSPLBM—after binarisation—with two recent and effective algorithms SwMV (Nie et al 2017) and MultiNMF (Liu et al. 2013). We consider 6 bases from the 8 ones for which we have or can apply these two algorithms.

We performed the same experimentation procedure as TSPLBM with 30 runs, and we compute the average of ACC, NMI, and Purity (Sripada and Rao 2011). For the MultiNMF, we pricked up the results in terms of ACC and NMI that are available in Wang et al. (2020) and Wang et al. (2015).

Table 2 are reported the obtained results on the six multi-view datasets. Thereby SwMV does a better job than MultiNMF; it achieves good results on UC-digits and 100Leaves. However, SwMV could not give the clustering for DBLP1. On the other hand, TSPLBM achieves highly better results than SwMV on the four datasets.

Overall, from these experiments, even with binary edges, we observe that TSPLBM gives encouraging results compared with SwMV and MultiNMF applied on graphs with continuous edges.

6.3 TSPLBM versus and tensor decomposition approaches

Undoubtedly and for a long time, to deal with tensor data $\mathcal{X} \in \mathbb{R}^{n \times n \times v}$, the tensor decomposition methods are the most popular (Kolda and Bader 2009). Even if they are not devoted to clustering, they allow to contribute to this task. Actually, these methods return a factor matrix $\in \mathbb{R}^{n \times r}$ (r is a given rank) that can be used for clustering. Thus, we used a list of suitable algorithms for the clustering: Kmeans++ (Arthur and Vassilvitskii 2007), Spectral clustering (SC) (Ng et al. 2001), and the EM algorithm (Dempster et al. 1977) derived from *diagonal Gaussian Mixture Model* (GMM) available in the Scikit-Learn package. Thereby, we compared the sparse tensor co-clustering algorithm TSPLBM with PARAFAC (Harshman and Lundy 1994) and

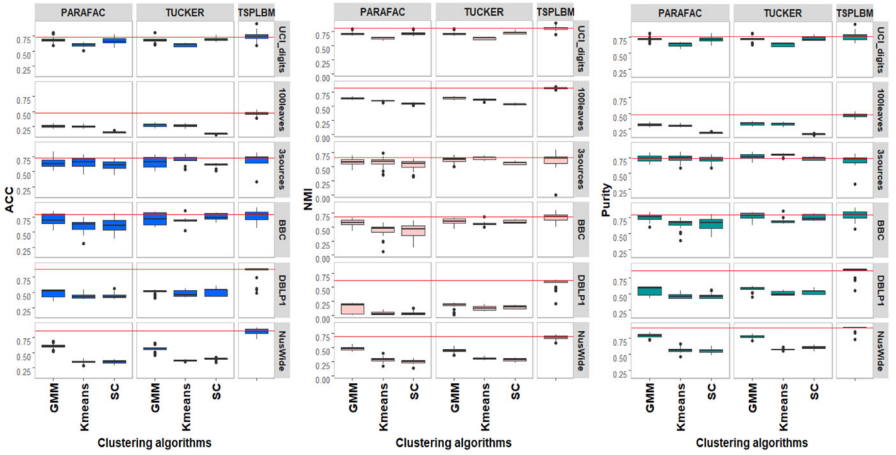


Fig. 10 Comparison between TSPLBM and tensor decomposition approaches based on clustering performances (ACC, NMI, and purity)

`Tucker decomposition` (Tucker 1966) on the six datasets presented in the previous section. We used different ranks (10, 20, and 50) and performed 30 runs with random initialization. Thus, we computed ACC, NMI, and purity by averaging all runs.

In Fig. 10 are reported the obtained clustering results for the six datasets according to the different tensor-based algorithms (PARAFAC, TUCKER decomposition, and `TSPLBM`) and the clustering algorithmsapplied on the obtained tensor decomposition. The results concern tensor decomposition approaches with rank number equal to 10 (The results for rank 20 and 50 are similar to those using rank equal to 10). We observe that in most of the cases `TSPLBM` does a better job than PARAFAC and Tucker decomposition methods. For the 3sources and Caltech-7 datasets, PARAFAC and TUCKER decomposition with GMM obtain close results in terms of Purity and Accuracy but `TSPLBM` achieves higher performances in terms of NMI.

To compare the computing time of `TSPLBM` and tensor decomposition approaches, we represent in Fig. 11 the time execution in seconds. We notice that for the four datasets 3sources, BBC, DBLP1, and UCI-digits, `TSPLBM` is close to all other approaches in terms of time execution. However, with Nus-wide-8 and 100Leaves, the time execution is more important, this is due to the dataset size and the number of clusters for Nus-wide-8 and 100Leaves. Note however, in Fig. 10, we observe that `TSPLBM` outperforms tensor decomposition approaches with approximately 25 points of ACC for both datasets.

6.4 `TSPLBM` versus `PSBM`, `PLBM`, and `SPLBM`

In this section, we aim to evaluate the impact of considering multiple graphs simultaneously in terms of clustering. To this end, we compare `TSPLBM` with `PSBM`, `PLBM`, and `SPLBM` that consider the slices separately (Sect. 3.4).

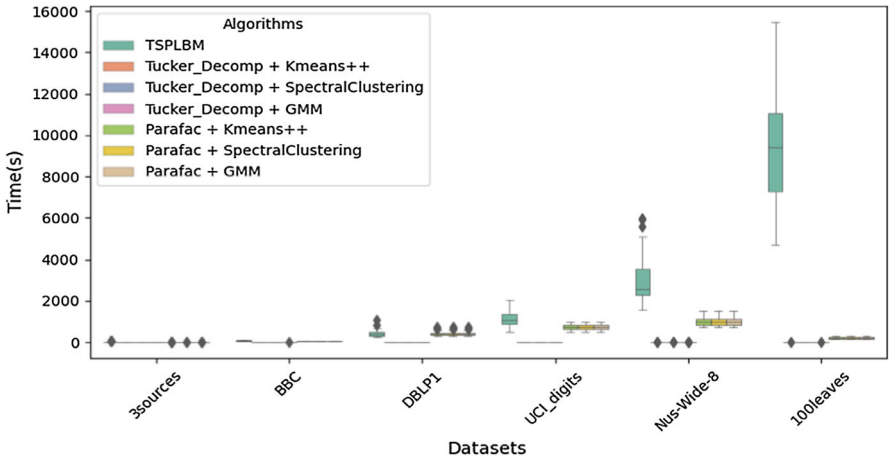


Fig. 11 Time complexity analysis

We performed 30 random initializations and computed Accuracy and Normalized Mutual Information (NMI) (Strehl and Ghosh 2002) metrics by averaging all runs. The clustering accuracy noted (ACC) discovers the one-to-one relationship between two partitions and measures the extent to which each cluster contains data points from the corresponding class. However, NMI is based on Mutual Information (MI) and measures the amount of retrieved information considering our knowledge about the clusters and the obtained results by a clustering method while respecting the proportions of clusters. For lack of space, we are focus on DBLP1, DBLP3, Nus-Wide and Amazon-Products-10. In Fig. 12, are reported the performances of the four algorithms PSBM, PLBM, SPLBM, and TSPLBM. PSBM, PLBM, and SPLBM are applied on each slice $\mathbf{x}^b$ separately unlike TSPLBM which is applied on $\mathcal{X}$ considering all graphs simultaneously. We notice that, in most cases, TSPLBM is better than other algorithms applied to each graph and allows us to achieve the best trade-off. TSPLBM includes all graphs and also the graphs with a very complex structure. DBLP3 obtains the lowest results due to the complex structure of dataset composed of 12K papers with very close or complementary topics on computer science. We observe that PLBM and SPLBM do a better job than PSBM for all datasets on the more informative slices. It is also worth noting that PLBM does good performances in terms of Accuracy on DBLP1 and in terms of NMI on DBLP3. TSPLBM performs a natural consensus when considering all slices and allows us to obtain a unique partition at the end with good clustering results.

6.5 Interpretation of multiple graph clustering results

This part aims to analyze the obtained topics and demonstrate how the proposed model can help the user interpret the obtained clusters using a visualization method. To illustrate this, we rely on the Nus-Wide-8 dataset.

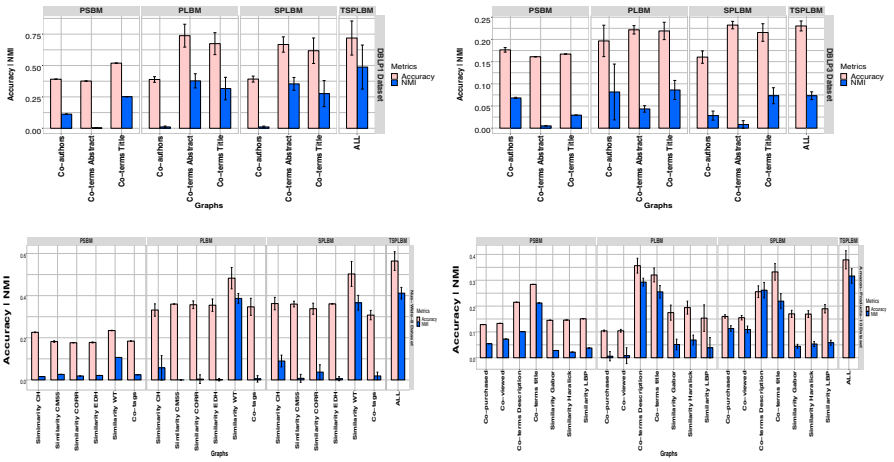


Fig. 12 Comparison in terms of Accuracy and NMI for all datasets with PSBM, PLBM, SPLBM and TSPLBM

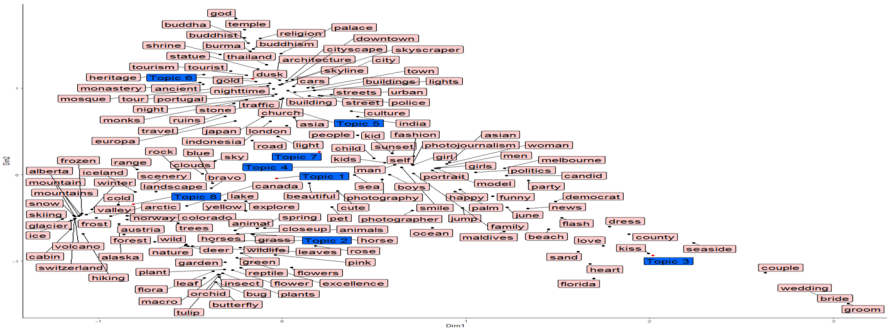


Fig. 13 CA applied on topic-tags matrix

On the topics-tags matrix, we performed the *Correspondence Analysis* (CA) method (Benzecri 1973; Nenadic and Greenacre 2007). The choice of CA is due to the connection between Poisson distribution, mutual information, and chi-square on which CA is based, see, e.g., Govaert and Nadif (2018). The matrix topic-tags $\mathbf{Z}^T \mathcal{M}$ is constructed from *image-tags* $\mathcal{M}$ based on obtained topics (or partition) $\mathbf{Z}$ obtained by TSPLBM. In Fig. 13, are projected the tags and topics on the two first dimensions of CA including the top tags in terms of contribution[4] on the CA results.

We can notice that there are some close topics and other very different one. For instance, topic 3 about weddings is opposed to topics 8 and 6 about *snow* and *temple* considering the first and the second dimension respectively. On the other hand, we can see that topics 1 and 2 about plants and animals are close.

Figure 14 presents the tags whose contribution is important. We show the frequencies of each term for each topic. For topics 2 and 5 (pink and purple color respectively),

[4] With CA each tag contributes to the inertia of each axis. The contribution of a tag to axis α is expressed as a percent of the inertia for axis α.

Fig. 14 Frequency matrix of subject tags whose contribution is important

we can see that the four top tags are *Nature, Green, Macro, and Flower* related to Plants topic and *Street, City, Night and Architect* related to Town topic.

Based on the *Co-tags* graph and the obtained topics, we construct a graph of image clusters linked by edges representing the intensity of joint tags between all topics, this can be computed by $\mathbf{Z}^\top \mathcal{H} \mathbf{Z}$ where $\mathbf{Z}$ is obtained by TSPLBM, and $\mathcal{H}$ is the co-tags matrix. We can notice that there are some topics with a strong relationship like *plants-snow* and *town-persons*. On the other hand, some topics with a weak link like *animals-town* and *animals-temple*. This representation highlights that there are some tags used with confused meaning. In this context, it is possible to use tensor models for tags completion and tags correction (Tang et al. 2017; Veit et al. 2017).

6.6 Discussion: implicit consensus versus explicit consensus

In the first part of our experiments, we observed that TSPLBM applied on all slices simultaneously is, in most of the cases, better than the other algorithms. As we are in an unsupervised context, we have found it helpful to run the calculation with several different random initial conditions and take the best result in terms of maximum log-likelihood, overall runs. This is the usual procedure in clustering. Next we study why and how we can improve this task.

6.6.1 Towards a consensus clustering

Figure 15 shows the 30 performed runs sorted according to Normalized log-likelihood (NL), which is the objective function of TSPLBM. We also draw the ACC and NMI curve according to the 30 runs. We observe that for DBLP1, the best runs leading to maximal NL are the best runs in terms of clustering (ACC and NMI). However, this observation is not noticed in all datasets; for instance, some best runs can achieve less good results in terms of ACC and NMI. This problem is recurrent with all unsupervised methods where the best runs in terms of the objective function are not necessarily the best ones in terms of clustering. On the other hand, we may see the proposed model as an implicit consensus model for graphs clustering, and it is tempting to compare the proposed model to ensemble-based clustering methods.

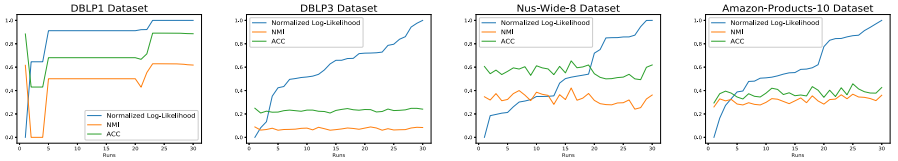


Fig. 15 Normalized Log-likelihood versus NMI and ACC for all runs

6.6.2 Ensemble method

The first works about consensus or ensemble classification have emerged in the context of supervised learning; see for instance (Maclin and Opitz 1997; Schapire 2003; Dietterich 2000). However, only the majority voting type algorithms work on the model output level, and the most well-known classification ensembles approaches are based on different variants of voting (Bauer and Kohavi 1999; Crammer et al. 2008; Gao et al. 2009). This approach has been extended to unsupervised learning (Strehl and Ghosh 2002; Vega-Pons and Ruiz-Shulcloper 2011). A clustering ensemble, also known as a consensus clustering or clustering aggregation, is defined in the same manner as for classification (Hanczar and Nadif 2012; Alqurashi and Wang 2019; Yu et al. 2019). It consists in combining multiple clustering models (partitions) into a single consolidated partition that we refer to as *explicit* consensus clustering. In other words, from r partitions $\{\mathbf{Z}_1, \mathbf{Z}_2, \mathbf{Z}_3, \ldots, \mathbf{Z}_r\}$, a consensus clustering leads to a unique partition $\mathbf{Z}^*$. Based on consensus functions, many approaches exist; see for instance (Strehl and Ghosh 2002; Hanczar and Nadif 2012; Affeldt et al. 2020a, b). In Strehl and Ghosh (2002), the authors introduced three ensemble clustering methods that can produce a consensus partition. All of them consider the consensus problem on a hypergraph representation of the set of partitions. More specifically, each partition is a binary classification matrix (with objects in rows and clusters in columns) where the concatenation of all the set defines the hypergraph. Figure 16 presents this matrix and different steps to construct a combination of these different graphs of clusters, emerged from different partitions, to obtain a unique graph. To this end, we rely on the three hypergraph clustering-based approaches proposed by Strehl and Ghosh (2002), namely Cluster-based Similarity Partitioning Algorithm (`CSPA`), HyperGraph Partitioning Algorithm (`HGPA`), and Meta-CLustering Algorithm (`MCLA`).

To improve clustering results of `TSPLBM` we will adopt the ensemble approach. We explore in the next part, how *implicit* consensus clustering through `TSPLBM` behaves compared to *explicit* consensus through cluster ensembles of multiple graphs. In Fig. 17, we report the proposed approach to compare `TSPLBM` with the clustering ensemble methods proposed by Strehl and Ghosh (2002). To do this, we used the implementation of python package `Cluster_Ensembles`.[5] It relies on `CSPA`, `HGPA`, and `MCLA` and returns the best results in terms of the mean of NMI between the obtained consensus clustering $\mathbf{Z}^*$ and the different clustering solutions $\{\mathbf{Z}_1, \mathbf{Z}_2, \mathbf{Z}_3, \ldots, \mathbf{Z}_r\}$. Therebey, with `TSPLBM`, we select the top ten runs maximizing log-likelihood then we carry out the consensus by using the cluster-ensembles methods. With `SPLBM`,

[5] https://pypi.org/project/Cluster_Ensembles/.

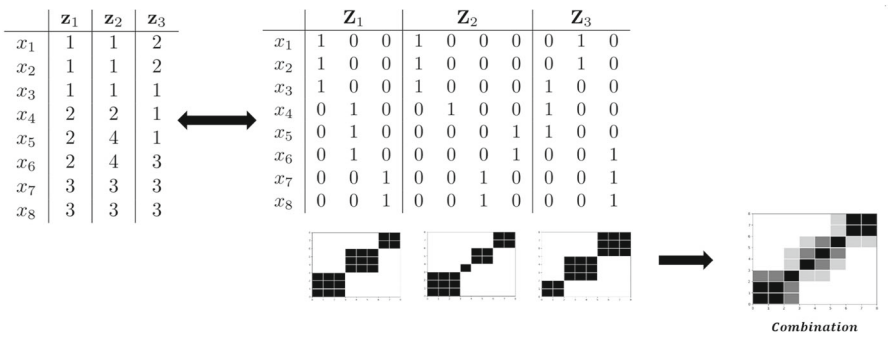


Fig. 16 Process of the transition from clustering to consensus clustering

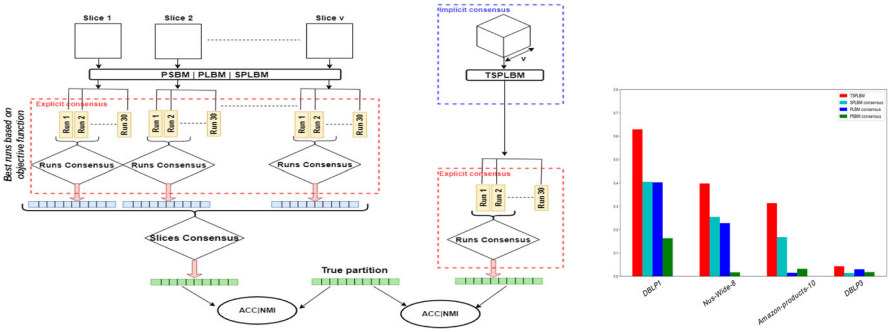


Fig. 17 Ensemble methods with PSBM, PLBM, SPSBM and TSPBLM. Description of the assessment process of all algorithms in terms of ACC and NMI (left). Comparison between PSBM, PLBM, SPSBM and TSPBLM (right)

PLBM, and PSBM, we consider two steps. The first step is the same as that used with TSPLBM to select the top ten runs and apply the cluster-ensembles methods. The second one consists in applying another clustering consensus between graphs to obtain a unique partition. Note that the consensus clustering information is implicitly provided by the TSPLBM algorithm.

In Fig. 17 (right) are reported the obtained results in terms of NMI using the comparison approach described above. We can notice that TSPLBM achieves the highest NMI for all datasets. SPLBM does a better or similar job than PLBM on three datasets, while PSBM obtains the lowest NMI measures on all datasets. These results can be explained by the fact that the implicit consensus achieved by TSPLBM is optimized within the objective function of the algorithm, unlike the explicit consensus, where the partitions are obtained separately.

7 Conclusion

It is well known that the traditional Poisson SBM fails to detect relevant clusters of edges, this requires a degree-corrected SBM (DC-SBM). Drawing on this, we first

established some connections between Poisson SBM and the corrected version DC-SBM with Poisson LBM commonly used for the co-clustering of contingency tables. We justified the extension of the latter to deal with multiple graphs clustering. To take into account the sparsity of the tensor, we modified the parametrization of the model and proposed a Tensor SPLBM (TSPLBM). We derived, thereby, an EM-like learning algorithm called `TSPLBM` capable of performing clustering from a tensor data. On real datasets of text and image graphs, we have shown that `TSPLBM`, is better than the cited baselines algorithms in terms of clustering.

On the other hand, we can note that the proposed clustering algorithm `TSPLBM` can be seen as an implicit consensus clustering for multiple graphs. To reinforce our idea that `TSPLBM` can be used in this sense, a comparative study with explicit consensus through ensemble clustering methods was realized. Experiments on several real graphs datasets highlight the effectiveness of `TSPLBM`. Thereby, this work gives an extra dimension to LBM as an ensemble method. Our approach has made it possible to propose a like-EM learning algorithm. It is possible to develop a like-Classification EM version. To do this, all that is needed is to insert a classification step between E and M steps. This could lead to propose an extension of `DC-SBM` for multiple graphs.

Our work opens different avenues for future research. First, in our proposal, we have considered a Poisson model. However, other distributions and other model variants can be developed compared to recent approaches relying on the mixture models and applied on image clustering (Zhang et al. 2021). When a data point has different representations, the authors propose to maximize a joint probability with multiple representations that can be generated by diverse methods such as kernel functions or data embedding methods. The model incorporates the prior information about data and utilizes it to set preferences for these representations. Second, in order to go further, the proposed model can be extended in incorporating Must Link and Cannot Link relationships in the model based on Hidden Markov Random Fields to deal with semi-supervised learning problems as those already dealt in Wu et al. (2021); Li et al. (2021). Finally, in our proposal, the number of clusters has been assumed to be known. It would be interesting to propose an extension of some criteria, such as the Integrated Completed Likelihood (ICL) criterion, already used with SBM (Daudin et al. 2008).

Funding Open Access funding enabled and organized by Projekt DEAL. Our work is funded by the German Federal Ministry for Economic Affairs and Energy (BMWi) under Grant Agreement Number 01MK20008F (Service-Meister).

A. Appendix: Proof of (4)

The marginal density function $f(\mathbf{X}; \boldsymbol{\Omega})$ of TSPLBM can be written as:

$$f(\mathbf{X}; \boldsymbol{\Omega}) = \sum_{(\mathbf{z},\mathbf{w})\in\mathcal{Z}\times\mathcal{W}} \prod_{i,k} \pi_k^{z_{ik}} \prod_{j,k} \rho_k^{w_{jk}} \prod_{i,j=1}^{n} \prod_{k=1}^{g} \left\{ \prod_{b=1}^{v} \mathcal{P}(x_{ij}^b; x_{i.}^b x_{.j}^b \gamma_{k\ell}^b) \right\}^{z_{ik}w_{jk}} \times \prod_{k,\ell\neq k}^{g} \left\{ \prod_{b=1}^{v} \mathcal{P}(x_{ij}^b; x_{i.}^b x_{.j}^b \gamma^b) \right\}^{z_{ik}w_{j\ell}}.$$

Thus, the complete-data log-likelihood function is given by:

$$\mathcal{L}_C(\mathbf{Z}, \mathbf{W}, \boldsymbol{\Omega}) = \sum_{i,k} z_{ik} \log \pi_k + \sum_{j,k} w_{jk} \log \rho_k + \sum_k \mathcal{L}_C^k$$

where

$$\mathcal{L}_C^k = \sum_{i,j} z_{ik} w_{jk} \left\{ \sum_{b=1}^{v} \mathcal{P}(x_{ij}^b; x_{i.}^b x_{.j}^b \gamma_{k\ell}^b) \right\} + \sum_{i,j,\ell\neq k} z_{ik} w_{j\ell} \left\{ \sum_{b=1}^{v} \mathcal{P}(x_{ij}^b; x_{i.}^b x_{.j}^b \gamma^b) \right\}.$$

Hence, the aim is to maximize the following lower bound of the log-likelihood criterion:

$$\mathcal{F}_C(\widetilde{\mathbf{Z}}, \widetilde{\mathbf{W}}, \boldsymbol{\Omega}) = \mathcal{L}_C(\widetilde{\mathbf{Z}}, \widetilde{\mathbf{W}}, \boldsymbol{\Omega}) + H(\widetilde{\mathbf{Z}}) + H(\widetilde{\mathbf{W}})$$

where $\mathcal{L}_C(\widetilde{\mathbf{Z}}, \widetilde{\mathbf{W}}, \boldsymbol{\Omega})$ is the fuzzy complete-data log-likelihood function. As $\mathcal{X}$ is symmetric per slice b, when $i = j$ we have $z_{ik} = w_{jk}$ and for $k = 1, \ldots, g$ we have $\pi_k = \rho_k$ and $H(\widetilde{\mathbf{Z}}) = H(\widetilde{\mathbf{W}})$. Then the objective function to optimize takes the following form:

$$\mathcal{F}_C(\widetilde{\mathbf{Z}}, \boldsymbol{\Omega}) = \mathcal{L}_C(\widetilde{\mathbf{Z}}, \boldsymbol{\Omega}) + 2H(\widetilde{\mathbf{Z}}) \quad \text{with} \quad \mathcal{L}_C(\widetilde{\mathbf{Z}}, \boldsymbol{\Omega}) = 2\sum_{i,k} \widetilde{z}_{ik} \log \pi_k + \sum_k \mathcal{L}_C^k,$$

or $\frac{1}{2}\mathcal{F}_C(\widetilde{\mathbf{Z}}, \boldsymbol{\Omega})$ leading to optimizing $\frac{1}{2}\mathcal{L}_C(\widetilde{\mathbf{Z}}, \boldsymbol{\Omega}) + H(\widetilde{\mathbf{Z}})$. □

B. Appendix: Proof of (5)

The simplified optimization criterion can be written as:

$$
\begin{aligned}
&\sum_{i,k} \widetilde{z}_{ik} \log \pi_k + \frac{1}{2} \sum_{i,j,k} \widetilde{z}_{ik} \widetilde{z}_{jk} \sum_{b=1}^{v} \log \mathcal{P}(x_{ij}^b; x_{i.}^b x_{.j}^b \gamma_{kk}^b) \\
&\quad + \frac{1}{2} \sum_{i,j,k,\ell \neq k} \widetilde{z}_{ik} \widetilde{z}_{j\ell} \sum_{b=1}^{v} \log \mathcal{P}(x_{ij}^b; x_{i.}^b x_{.j}^b \gamma^b) + H(\widetilde{\mathbf{Z}}) \\
&= \sum_{i,k} \widetilde{z}_{ik} \log \pi_k + \frac{1}{2} \Bigg[\sum_{i,j,k} \widetilde{z}_{ik} \widetilde{z}_{jk} \sum_b \log \left(\frac{e^{-x_{i.}^b x_{.j}^b \gamma_{kk}^b} (x_{i.}^b x_{.j}^b \gamma_{kk}^b)^{x_{ij}^b}}{x_{ij}^b!} \right) \\
&\quad + \sum_{i,j,k,\ell \neq k} \widetilde{z}_{ik} \widetilde{z}_{j\ell} \sum_b \left(\frac{e^{-x_{i.}^b x_{.j}^b \gamma^b} (x_{i.}^b x_{.j}^b \gamma^b)^{x_{ij}^b}}{x_{ij}^b!} \right) \Bigg] \\
&\quad + H(\widetilde{\mathbf{Z}}) \\
&= \sum_{i,k} \widetilde{z}_{ik} \log \pi_k + \frac{1}{2} \Bigg[\sum_{i,j,k} \widetilde{z}_{ik} \widetilde{z}_{jk} \sum_b \left(-x_{i.}^b x_{.j}^b \gamma_{kk}^b + x_{ij}^b \log \gamma_{kk}^b \right) \\
&\quad + \sum_{i,j,k,\ell \neq k} \widetilde{z}_{ik} \widetilde{z}_{j\ell} \sum_b \left(-x_{i.}^b x_{.j}^b \gamma^b + x_{ij}^b \log \gamma^b \right) \Bigg] \\
&\quad + \sum_{i,j,b} x_{ij}^b \log(x_{i.}^b x_{.j}^b) - \log(x_{ij}^b!) + H(\widetilde{\mathbf{Z}})
\end{aligned}
$$

Note that $\sum_{i,j,b} x_{ij}^b \log(x_{i.}^b x_{.j}^b) - \log(x_{ij}^b!)$ does not depend on $\widetilde{\mathbf{Z}}$, and $\boldsymbol{\Omega}$ and therefore can be ignored for optimization purpose. To keep formulas uncluttered we therefore ignore this term in the subsequent development. Thus, we obtain:

$$
\begin{aligned}
&\sum_{i,k} \widetilde{z}_{ik} \log \pi_k + \frac{1}{2} \sum_b \Bigg[\sum_k \left(x_{kk}^b \log(\gamma_{kk}^b) - x_{k.}^b x_{.k}^b \gamma_{kk}^b \right) \\
&\quad + \left((N_b - \sum_k x_{kk}^b) \log(\gamma^b) - (N_b^2 - \sum_k x_{k.}^b x_{.k}^b) \gamma^b \right) \Bigg] + H(\widetilde{\mathbf{Z}})
\end{aligned}
$$

where $x_{k.}^b = \sum_i \widetilde{z}_{ik} x_{i.}^b = \sum_j \widetilde{z}_{jk} x_{.j}^b = x_{.k}^b, x_{kk}^b = \sum_{i,j} \widetilde{z}_{ik} \widetilde{z}_{jk} x_{ij}^b$, and $N_b = \sum_{i,j} x_{ij}^b$. This leads to optimize

$$
\sum_{i,k} \widetilde{z}_{ik} \log \pi_k + \frac{1}{2} \sum_{b,k} \mathsf{L}_c^{kb} + H(\widetilde{\mathbf{Z}}) \tag{8}
$$

where

$$\mathsf{L}_c^{kb} = x_{kk}^b \log(\frac{\gamma_{kk}^b}{\gamma^b}) - [x_{k.}^b]^2 \left(\gamma_{kk}^b - \gamma^b\right) + \frac{N_b}{g}\left(\log(\gamma^b) - N_b\gamma^b\right). \quad \square \qquad (9)$$

C. Appendix: E-step and M-step

E-step To obtain the expression of $\widetilde{z}_{ik}$, we maximize (4) with respect to $\widetilde{z}_{ik}$, subject to the constraint $\sum_k \widetilde{z}_{ik} = 1$. The corresponding Lagrangian, up to terms which are not a function of $\widetilde{z}_{ik}$, is given by:

$$L(\widetilde{\mathbf{Z}}, \beta) = \sum_{i,k} \widetilde{z}_{ik} \log \pi_k + \frac{1}{2} \sum_{i,j,k} \widetilde{z}_{ik}\widetilde{z}_{jk} \left(\sum_{b=1}^{v} \mathcal{P}_{kk}^{ijb}\right) + \frac{1}{2} \sum_{i\neq j, k\neq \ell} \widetilde{z}_{ik}\widetilde{z}_{j\ell} \left(\sum_{b=1}^{v} \mathcal{P}_{k\ell}^{ijb}\right) - \sum_{i,k} \widetilde{z}_{ik} \log(\widetilde{z}_{ik}) + \beta(1 - \sum_k \widetilde{z}_{ik}).$$

Taking derivatives with respect to $\widetilde{z}_{ik}$, we obtain:

$$\frac{\partial L(\widetilde{\mathbf{Z}}, \beta)}{\partial \widetilde{z}_{ik}} = \log \pi_k + \frac{1}{2} \sum_j \widetilde{z}_{jk} \left(\sum_{b=1}^{v} \mathcal{P}_{kk}^{ijb}\right) + \frac{1}{2} \sum_{j\neq i, k\neq \ell} \widetilde{z}_{j\ell} \left(\sum_{b=1}^{v} \mathcal{P}_{k\ell}^{ijb}\right) - \log \widetilde{z}_{ik} - 1 - \beta.$$

Setting this derivative to zero yields:

$$\widetilde{z}_{ik} = \frac{\pi_k \exp \frac{1}{2} \left(\sum_j \widetilde{z}_{jk} \sum_{b=1}^{v} \mathcal{P}_{kk}^{ijb} + \sum_{j\neq i, k\neq \ell} \widetilde{z}_{j\ell} \sum_{b=1}^{v} \mathcal{P}_{k\ell}^{ijb}\right)}{\exp(\beta + 1)}.$$

Summing both sides over all k' yields $\exp(\beta+1) = \sum_{k'} \pi_{k'} \exp \frac{1}{2} \left(\sum_{j,k'} \widetilde{z}_{jk} \sum_{b=1}^{v} \mathcal{P}_{k'k'}^{ijb} + \sum_{j\neq i, k'\neq \ell} \widetilde{z}_{j\ell} \sum_{b=1}^{v} \mathcal{P}_{k'\ell}^{ijb}\right)$. Plugging $\exp(\beta + 1)$ in $\widetilde{z}_{ik}$ leads to:

$$\widetilde{z}_{ik} \propto \pi_k \exp \frac{1}{2} \left(\sum_j \widetilde{z}_{jk} \sum_{b=1}^{v} \mathcal{P}_{kk}^{ijb} + \sum_{j\neq i, k\neq \ell} \widetilde{z}_{j\ell} \sum_{b=1}^{v} \mathcal{P}_{k\ell}^{ijb}\right) \quad \square \qquad (10)$$

or,

$$\log \widetilde{z}_{ik} \propto \log \pi_k + \frac{1}{2} \left(\sum_j \widetilde{z}_{jk} \sum_{b=1}^{v} \mathcal{P}_{kk}^{ijb} + \sum_{j\neq i, k\neq \ell} \widetilde{z}_{j\ell} \sum_{b=1}^{v} \mathcal{P}_{k\ell}^{ijb}\right).$$

M-step Given $\widetilde{\mathbf{Z}}$ the maximization step consists in computing π_k's, γ_{kk}^b's and γ^b maximizing $\frac{1}{2}\mathcal{F}_C(\widetilde{\mathbf{Z}}, \Omega)$.

– Computation of $\gamma_{kk}, \forall k, b$. It is easy to show that this can be computed separately. This leads to derivate (9)

$$\frac{\partial \mathsf{L}_c^{kb}}{\partial \gamma_{kk}^b} = \frac{x_{kk}^b}{\gamma_{kk}^b} - [x_{k.}]^2 = 0 \Longrightarrow \hat{\gamma}_{kk}^b = \frac{x_{kk}^b}{[x_{k.}]^2}. \quad \square$$

As $\frac{\partial^2 \mathsf{L}_c^{kb}}{\partial^2 \gamma_{kk}^b} \leq 0$ then $\hat{\gamma}_{kk}^b$ is a maximum and it is easy to verify that $\hat{\gamma}_{kk}^b \leq 1$.

– Computation of $\gamma^b, \forall b$. It suffices to derivate $\sum_k \mathsf{L}_c^{kb}$.

$$\frac{\partial \sum_k \mathsf{L}_c^{kb}}{\partial \gamma^b} = -\frac{\sum_k x_{kk}^b}{\gamma^b} + \sum_k [x_{k.}^b]^2 + \frac{N_b}{\gamma^b} - N_b^2 = 0 \Longrightarrow \hat{\gamma}^b = \frac{N_b - \sum_k x_{kk}^b}{N_b^2 - \sum_k [x_{k.}]^2}. \quad \square$$

A simple expression of z_{ik} Note that, plugging the estimation of γ_{kk}^b's and γ^b's in (8) yields to:

$$\sum_{i,k} \widetilde{z}_{ik} \log \pi_k + \frac{1}{2} \sum_b \Bigg(\sum_k x_{kk}^b \log(\frac{\gamma_{kk}^b}{\gamma^b}) - [x_{k.}^b]^2 \frac{x_{kk}^b}{[x_{k.}^b]^2} - \frac{N_b - \sum_k x_{kk}^b}{N_b^2 - \sum_k [x_{k.}^b]^2} (N_b^2 - [x_{k.}^b]^2) + N_b \log(\gamma^b) \Bigg) + H(\widetilde{\mathbf{Z}})$$

Since $N_b = \sum_k x_{kk}^b$, this leads to

$$\sum_{i,k} \widetilde{z}_{ik} \log \pi_k + \sum_b \Bigg(\sum_k x_{kk}^b \log \left(\frac{\gamma_{kk}^b}{\gamma^b} \right) + N_b (\log(\gamma^b) - 1) \Bigg) + H(\widetilde{\mathbf{Z}})$$

As $x_{kk}^b = \sum_{i,j} \widetilde{z}_{ik} \widetilde{z}_{jk} x_{ij}^b = \sum_i \widetilde{z}_{ik} x_{i.}^b$ with $x_{ik}^b = \sum_j \widetilde{z}_{jk} x_{ij}^b$, after algebraic calculations as in E-step it is easy to show that

$$\widetilde{z}_{ik} \propto \pi_k \exp \frac{1}{2} \sum_b \left(x_{ik}^b \log \left(\frac{\gamma_{kk}^b}{\gamma^b} \right) \right)$$

which becomes

$$\widetilde{z}_{ik} \propto \pi_k \exp \frac{1}{2} \sum_b \left(\sum_j \widetilde{z}_{jk} x_{ij}^b \log \left(\frac{\gamma_{kk}^b}{\gamma^b} \right) \right)$$

and we obtain a simple update of $\widetilde{z}_{ik}$ as follows

$$\widetilde{z}_{ik}^{(t+1)} \propto \pi_k \exp\left(\frac{1}{2}\sum_j \widetilde{z}_{jk}^{(t)} \sum_{b=1}^{v} x_{ij}^b \log\left(\frac{\gamma_{kk}^b}{\gamma^b}\right)\right). \quad \square$$

References

Affeldt S, Labiod L, Nadif M (2020a) Ensemble block co-clustering: a unified framework for text data. In: Proceedings of the 29th ACM international conference on information and knowledge management, pp 5–14

Affeldt S, Labiod L, Nadif M (2020b) Spectral clustering via ensemble deep autoencoder learning (SC-EDAE). Pattern Recognit 108:107522

Affeldt S, Labiod L, Nadif M (2021) Regularized bi-directional co-clustering. Stat Comput 31(3):32

Ailem M, Role F, Nadif M (2017a) Model-based co-clustering for the effective handling of sparse data. Pattern Recognit 72:108–122

Ailem M, Role F, Nadif M (2017b) Sparse Poisson latent block model for document clustering. IEEE Trans Knowl Data Eng 29(7):1563–1576

Alqurashi T, Wang W (2019) Clustering ensemble method. Int J Mach Learn Cybern 10(6):1227–1246

Arthur D, Vassilvitskii S (2007) K-means++: the advantages of careful seeding. In: Proceedings of the eighteenth annual ACM-SIAM symposium on discrete algorithms, Society for Industrial and Applied Mathematics, USA, SODA '07, pp 1027–1035

Banerjee A, Basu S, Merugu S (2007) Multi-way clustering on relation graphs. In: SIAM international conference on data mining, pp 145–156

Bauer E, Kohavi R (1999) An empirical comparison of voting classification algorithms: bagging, boosting, and variants. Mach Learn 36(1–2):105–139

Benzecri JP (1973) L'analyse des données, tome 2: l'analyse des correspondances. Dunod, Paris

Bickel S, Scheffer T (2004) Multi-view clustering. ICDM 4:19–26

Celeux G, Govaert G (1992) A classification EM algorithm for clustering and two stochastic versions. Comput Stat Data Anal 14(3):315–332

Chen C, Ng MK, Zhang S (2017) Block spectral clustering methods for multiple graphs. Numer Linear Algebra Appl 24(1):e2075

Crammer K, Kearns M, Wortman J (2008) Learning from multiple sources. J Mach Learn Res 9:1757–1774

Daudin JJ, Picard F, Robin S (2008) A mixture model for random graphs. Stat Comput 18(2):173–183

Dempster AP, Laird NM, Rubin DB (1977) Maximum likelihood from incomplete data via the EM algorithm. J R Stat Soc B 39:1–38

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

Dietterich TG (2000) Ensemble methods in machine learning. In: International workshop on multiple classifier systems. Springer, pp 1–15

Frankel T (2012) The geometry of physics: an introduction. Cambridge University Press, Cambridge

Gao J, Liang F, Fan W, Sun Y, Han J (2009) Graph-based consensus maximization among multiple supervised and unsupervised models. In: Advances in neural information processing systems, pp 585–593

Govaert G, Nadif M (2003) Clustering with block mixture models. Pattern Recogn 36:463–473

Govaert G, Nadif M (2005) An EM algorithm for the block mixture model. IEEE Trans Pattern Anal Mach Intell 27(4):643–647

Govaert G, Nadif M (2006) Fuzzy clustering to estimate the parameters of block mixture models. Soft Comput 10(5):415–422

Govaert G, Nadif M (2013) Co-clustering: models, algorithms and applications. Wiley, Hoboken

Govaert G, Nadif M (2018) Mutual information, phi-squared and model-based co-clustering for contingency tables. Adv Data Anal Classif 12(3):455–488

Hanczar B, Nadif M (2012) Ensemble methods for biclustering tasks. Pattern Recogn 45(11):3938–3949

Harshman RA, Lundy ME (1994) Parafac: parallel factor analysis. Comput Stat Data Anal 18:39–72

Janson S (1987) Poisson convergence and Poisson processes with applications to random graphs. Stoch Process Appl 26:1–30
Karrer B, Newman ME (2011) Stochastic blockmodels and community structure in networks. Phys Rev E 83(1):016107
Kiers HA (2000) Towards a standardized notation and terminology in multiway analysis. J Chemom 14:105–122
Kolda TG, Bader BW (2009) Tensor decompositions and applications. J Math Psychol 51(3):455–500
Labiod L, Nadif M (2014) A unified framework for data visualization and coclustering. IEEE Trans Neural Netw Learn Syst 26(9):2194–2199
Li X, Zhang Y, Zhang R (2021) Semisupervised feature selection via generalized uncorrelated constraint and manifold embedding. In: IEEE transactions on neural networks and learning systems
Liu J, Wang C, Gao J, Han J (2013) Multi-view clustering via joint nonnegative matrix factorization. In: Proceedings of the 2013 SIAM international conference on data mining. SIAM, pp 252–260
Maclin R, Opitz D (1997) An empirical evaluation of bagging and boosting. AAAI/IAAI 1997:546–551
McLachlan GJ, Peel D (2000) Finite mixture models. Wiley, New York
Nadif M, Govaert G (2005) Block clustering of contingency table and mixture model. In: International symposium on intelligent data analysis. Springer, pp 249–259
Neal RM, Hinton GE (1998) A view of the EM algorithm that justifies incremental, sparse, and other variants. In: Learning in graphical models. Springer, pp 355–368
Nenadic O, Greenacre M (2007) Correspondence analysis in R, with two-and three-dimensional graphics: the CA package. J Stat Softw 20(3)
Ng A, Jordan M, Weiss Y (2001) On spectral clustering: analysis and an algorithm. Adv Neural Inf Process Syst 14:849–856
Nie F, Li J, Li X et al (2017) Self-weighted multiview clustering with multiple graphs. In: IJCAI, pp 2564–2570
Nowicki K, Snijders TAB (2001) Estimation and prediction for stochastic blockstructures. J Am Stat Assoc 96(455):1077–1087
Qiao M, Yu J, Bian W, Li Q, Tao D (2017) Improving stochastic block models by incorporating power-law degree characteristic. In: Proceedings of the twenty-sixth international joint conference on artificial intelligence (IJCAI-17), pp 2620–2626
Role F, Morbieu S, Nadif M (2019) Coclust: a python package for co-clustering. J Stat Softw 88(7):1–29
Salah A, Nadif M (2019) Directional co-clustering. Adv Data Anal Classif 13(3):591–620
Schapire RE (2003) The boosting approach to machine learning: An overview. In: Nonlinear estimation and classification, Springer, pp 149–171
Shan H, Banerjee A (2008) Bayesian co-clustering. In: Eighth IEEE international conference on data mining. IEEE, pp 530–539
Sripada SC, Rao MS (2011) Comparison of purity and entropy of k-means clustering and fuzzy c means clustering. Indian J Comput Sci Eng 2(3):343–346
Strehl A, Ghosh J (2002) Cluster ensembles: a knowledge reuse framework for combining multiple partitions. J Mach Learn Res 3:583–617
Tang J, Shu X, Qi G, Li Z, Wang M, Yan S, Jain R (2017) Tri-clustered tensor completion for social-aware image tag refinement. IEEE Trans Pattern Anal Mach Intell 39(8):1662–1674
Tang W, Lu Z, Dhillon IS (2009) Clustering with multiple graphs. In: Ninth IEEE international conference on data mining. IEEE, pp 1016–1021
Tucker LR (1966) Some mathematical notes on three-mode factor analysis. Psychometrika 31(3):279–311
Vega-Pons S, Ruiz-Shulcloper J (2011) A survey of clustering ensemble algorithms. Int J Pattern Recognit Artif Intell 25(03):337–372
Veit A, Nickel M, Belongie S, Maaten L (2017) Separating self-expression and visual content in hashtag supervision. In: IEEE/CVF conference on computer vision and pattern recognition (CVPR)
Wang H, Yang Y, Liu B (2020) GMC: graph-based multi-view clustering. IEEE Trans Knowl Data Eng 32(6):1116–1129
Wang Z, Kong X, Fu H, Li M, Zhang Y (2015) Feature extraction via multi-view non-negative matrix factorization with local graph regularization. In: IEEE international conference on image processing (ICIP), pp 3500–3504
Wu T, Zhang R, Jiao Z, Wei X, Li X (2021) Adaptive spectral rotation via joint cluster and pairwise structure. In: IEEE transactions on knowledge and data engineering

Yu X, Yu G, Wang J, Domeniconi C (2019) Co-clustering ensembles based on multiple relevance measures. In: IEEE transactions on knowledge and data engineering pp 1–1 https://doi.org/10.1109/TKDE.2019.2942029

Zhang R, Zhang H, Li X (2021) Maximum joint probability with multiple representations for clustering. In: IEEE transactions on neural networks and learning systems