# Boosting Subspace Co-clustering via Bilateral Graph Convolution

Chakib Fettal, Lazhar Labiod, and Mohamed Nadif

**Abstract**—Subspace clustering seeks to cluster high-dimensional data lying in a union of low-dimensional subspaces. It has achieved state-of-the-art results in image clustering, but text clustering of document-term matrices, has proved more impervious to advances with this approach, even though text data satisfies the assumptions of subspace clustering. We hypothesize that this is because such matrices are generally sparser and higher-dimensional than images. This, combined with the complexity of subspace clustering, which is generally cubic in the number of inputs, makes its use impractical in the context of text. Here we address these issues with a view to leveraging subspace clustering for networked (or not) text data. We first extend the concept of subspace clustering to co-clustering, which is suitable to deal with document-term matrices because of the interplay engendered between the document and word representations. We then address the sparsity problem through bilateral graph convolution, which promotes the grouping effect that has been credited for the effectiveness of some subspace clustering models. The proposed formulation results in an algorithm that is computationally/spatially efficient. Experiments using real-world datasets demonstrate the superior performance, in terms of document clustering, word clustering, and computational efficiency, of our proposed approach over the baselines and comparable methods.

**Index Terms**—Co-clustering, Subspace Clustering, Attributed Graphs

## 1 Introduction

As the datasets become more and more larger and can be in addition more sparse, adaptations to existing clustering algorithms are required to maintain cluster quality. In fact, this quality can greatly suffer in the high dimensional data where many of dimensions are often irrelevant. This appears in many applications including recommendation systems, microarray or even textual data represented as document-term matrices. In the text area, the task of text clustering is always important and has many use cases including fake news detection, sentiment analysis information retrieval and so on; see for instance [1] for more applications. Thereby *subspace clustering* [2] and *biclustering/co-clustering* [3], [4] techniques have shown that can leveraged to uncover the complex relationships found in such data.

Subspace clustering is an unsupervised learning method in which points are to be grouped according to the subspaces in which they lie. A variety of approaches have been used to solve this problem, and a number of these approaches are based on a self-expressive formulation where it is assumed that each element can be written as a linear combination of the elements in the subspace. Based on a self-expressive formulation, subspace clustering methods have been widely used to cluster image datasets, given that image datasets will often be drawn from multiple low-dimensional subspaces, and state-of-the-art clustering results have in many cases been obtained. Regarding text data, however, to the best of our knowledge no self-expressive subspace clustering approaches have been proposed that are specifically tailored to text, although the assumption made in relation to image data applies equally to text data. We argue that this discrepancy between image and text stems from two causes:

- *Complexity.* Document-term datasets are usually very large, much more so than images, which makes it prohibitive to use subspace clustering algorithms whose computational complexity is usually in $\mathcal{O}(n^3)$, and whose spatial complexity is in $\mathcal{O}(n^2)$.
- *Sparsity.* Document-term datasets are sparser than image datasets, and each individual data point may thus potentially lie in a unique subspace, making it difficult for subspace clustering algorithms to group points meaningfully.

In this paper we propose a subspace clustering model tailored for networked (and non-networked) text data based on the principle of *co-clustering* (or *biclustering*), that is to say using the interplay between rows and columns [5]. In the context of text, this consists in harnessing the interplay between the set of documents and the set of terms to jointly generate a partitioning for both of them. This leads to reorganize the initial data matrix into a new one that is reorganized into homogeneous co-clusters/biclusters. The choice of the co-clustering approach is justified for at least four reasons a) co-clustering overcomes the curse of dimensionality and sparsity; by alternatingly updating the row partition given the column partition and vice versa the clustering can be performed in lower dimensional space and therefore more parsimonious than one-sided clustering performed on the row or column sets separately [6] b) the clusters/co-clusters tend to be more easily interpretable, allowing the user to better direct further study [3], [7], [8] c) fuzzy co-clustering can be easily deduced from many co-clustering methods [5] allowing, for instance, to assign a document/term to several classes, and finally d) in terms of topic modelling, the obtained results from co-clustering are in line with the human judgment, outperforming, in general, the conventional LDA (Allocation Dirichlet Allocation) method [9]; see for instance [10]. In this regard, other researchers emphasize that sparse datasets are not suitable

for LDA [11].

In our approach, we address the two major issues that are inherent to subspace clustering on text data: complexity and sparsity. To this end, we use factorized representation matrices and nonnegative kernel feature maps, as well as a bilateral graph convolution that includes a weighted Laplacian smoothing preprocessing step, having similarities with a simple graph convolutional network [12], [13], [14]. Second, combining subspace and co-clustering, we propose an efficient extension to [15] to co-clustering and expand the proposal presented in [16]. This makes our model particularly well suited to clustering attributed graphs whose nodes and/or edges have attributes or features. This leads to tackle networked text data/text attributed graphs that are used to model a wide variety of real-world networks such as in recommender systems [17], citation networks [18] and so on. We summarize our contributions as follows:

- We study how the Laplacian smoothing operation boosts the grouping effect of subspace clustering approaches that possess this property.
- Unlike the iterative process generally adopted in co-clustering, we show here that in our formulation of the self-expressive subspace co-clustering problem the optimal solution can be derived from a single truncated singular value decomposition, which makes it efficient (linear complexity in the number of nodes).
- We carry out extensive experimentation on text attributed graphs where the graphs exist on the one hand and when the graphs are generated from the node features on the other using $k$-nearest neighbor graphs. This allows to demonstrate the flexibility and value of the proposed model, in terms of document /word clustering capability and computational efficiency.
- Making the code available for download to ensure reproducibility of the results [1].

The remainder of this paper is structured as follows. Section 2 discusses related works. Section 3 develops our proposed method SC$^3$, and section 4 discusses the algorithm and its complexity. Section 5 contains a detailed description of our experiments. Finally, we give our conclusion in section 6.

## 2 Related Works

Our contributions can be seen as being at the intersection between subspace clustering, co-clustering, and attributed graph clustering.

### 2.1 Self-expressive Subspace Clustering

Among the earliest approaches was Least Squares Regression (LSR) subspace clustering [19], which leverages a grouping effect based on the correlation of data. More sophisticated approaches that represent today's state of the art were later proposed, such as Elastic-net Subspace Clustering (EnSC) [20], subspace clustering by Orthogonal Matching Pursuit (SSC-OMP) [21], the $\ell^0$-norm regularized subspace clustering ($\ell^0$-SCC) [22] and its recent extension that deals with noisy data (Noisy-DR-$\ell^0$-SCC-LR) [23]. Some recent works have proposed efficient methods such as K-Factorization Subspace Clustering (K-FSC) [24] and others were created to deal with multi-view data [25], [26], [27]. Some models like the GAN-Based Enhanced Deep Subspace Clustering Networks [28] explicitly make the assumption of dealing with image data. Note that deep self-expressive subspace clustering models have received criticism over the necessity of a neural network component [29].

### 2.2 Co-clustering

Co-clustering seeks to form *co-clusters* which are sets of homogeneous sets of rows and columns. It harnesses the inherent duality between the rows and columns of data tables, which can lead to improvements in partitioning for both dimensions. For example, in the case of document-term matrices, co-clustering incorporates term space information that is used in the document partitioning, and vice versa. A popular method is by alternatingly finding a clustering for the rows while taking into consideration the current clustering of columns, and inversely. One of the first co-clustering approaches was the spectral co-clustering algorithm [3]. Directional Co-clustering with Constraints (DCC) [30] is based on a regularized von Mises-Fisher mixture model that makes it suitable for balanced text datasets. Regularized Dual-PPMI Co-clustering (RDPCo) [31] extends DCC to incorporate both word-word semantic relationships and document-document similarities into the procedure (see [5] for a review). Finally, Consensus Factorization for Co-Clustering Networked Data (CFOND) [32] is a consensus factorization model that factorizes information simultaneously from three sources: network topology structures, instance-feature content relationships, and feature-feature correlations.

### 2.3 Attributed Graph Clustering

Attributed Graph clustering involves grouping nodes into clusters depending on the structure of the graph and node-level features. In Graph-InfoClust (GIC) [33] clustering is done by maximizing the mutual information between nodes contained in the same cluster. Simple Spectral Graph Convolution (S$^2$GC) [34] is a method for the aggregation of K-hop neighborhoods that is a trade-off between low- and high-pass filter bands. Graph Convolutional Clustering (GCC) [35] is a procedure that simultaneously clusters and learns clustering-friendly representations of nodes. In addition, CFOND, mentioned above in relation to co-clustering, is also considered to be an attributed graph clustering model.

## 3 Preliminaries and Background

Matrices are denoted using boldface uppercase, and vectors using boldface lowercase letters. Given a matrix $\mathbf{X}$, its $i$-th row is denoted as $\mathbf{x}_i$ and its $j$-th column as $\mathbf{x}'_j$. $\mathbf{I}_n$ is the identity matrix of size $n$. The Frobenius norm is denoted as $\|.\|$. $\texttt{rk}$ gives the rank of a matrix. Function $[\mathbf{U}, \mathbf{\Sigma}, \mathbf{V}] = \texttt{SVD}(\mathbf{X})$ gives the compact singular value decomposition of matrix $\mathbf{X} \in \mathbb{R}^{n \times d}$, where $\mathbf{U} \in \mathbb{R}^{n \times \mathbf{rk}(\mathbf{X})}$ and $\mathbf{V} \in \mathbb{R}^{d \times \mathbf{rk}(\mathbf{X})}$ have the left and right singular vectors in their columns and $\mathbf{\Sigma} \in \mathbb{R}^{\mathbf{rk}(\mathbf{X}) \times \mathbf{rk}(\mathbf{X})}$ is the diagonal matrix containing the singular values, sorted in decreasing order.

1. https://github.com/chakib401/sc3

We also define function $[\mathbf{U}, \mathbf{\Sigma}, \mathbf{V}] = \texttt{TruncatedSVD}(\mathbf{X}, k)$ that returns the first (largest) $k$ singular values, and the left and right singular vectors. Function `diag` creates a diagonal matrix from a vector input, while $\mathbf{1}$ denotes a vector of ones.

### 3.1 Self-Expressive Subspace Clustering

Given a $\mathbf{X} \in \mathbb{R}^{n \times d}$ a matrix of $d$-dimensional data points. The self-expressive subspace clustering is typically formulated as

$$\min_{\mathbf{R}} \quad \| \mathbf{X} - \mathbf{R}\mathbf{X} \|^2 + \Omega(\mathbf{R}) \quad \text{such that} \quad \mathbf{R} \in \mathcal{R} \tag{1}$$

where $\mathbf{R} \in \mathbb{R}^{n \times n}$ is known as the self-representation matrix, $\Omega(\mathbf{R})$ serves as a regularization term designed to establish certain properties for $\mathbf{R}$ so as to avoid trivial solutions (such as $\mathbf{R} = \mathbf{I}$), and $\mathcal{R}$ is the feasible region. After an optimal solution $\mathbf{R}^*$ has been obtained, an affinity matrix is first generated based on the magnitudes of the entries in $\mathbf{R}^*$, usually using $|\mathbf{R}^* + \mathbf{R}^{*\top}|/2$, and a partition of the points is then generated using a graph clustering method such as the spectral clustering algorithm [36].

### 3.2 Block seriation

Co-clustering can be posed as a *block seriation* problem [37] whose objective is finding a block diagonal matrix $\mathbf{R} \in \mathbb{R}^{n \times d}$ that identifies co-clusters.

The block seriation problem can be stated as this integer program:

$$\max_{\mathbf{R}} \quad \sum_{ij} x_{ij} r_{ij} \tag{2}$$

subject to:

$$\begin{array}{lll}
\forall\, i, j & r_{ij} \in \{0, 1\} & \text{Binarity} \\
\forall\, i, j, i', j' & \left.\begin{array}{l} r_{ij} + r_{ij'} + r_{i'j'} - r_{i'j} \leq 2 \\ r_{i'j'} + r_{i'j} + r_{ij} - r_{ij'} \leq 2 \\ r_{i'j} + r_{ij} + r_{ij'} - r_{i'j'} \leq 2 \\ r_{ij'} + r_{i'j'} + r_{i'j} - r_{ij} \leq 2 \end{array}\right\} & \text{No triads} \\
\forall\, j & \left.\begin{array}{l} \sum_i r_{ij} \geq 1 \\ \sum_j r_{ij} \geq 1 \end{array}\right\} & \text{Assignment} \\
\forall\, i &
\end{array}$$

The solution $\mathbf{R}$ is always block diagonal up to a permutation of the rows and columns. An equivalent formulation consists in factorizing $\mathbf{R}$ using two assignment matrices $\mathbf{Z}$ and $\mathbf{W}$:

$$\max_{\mathbf{Z}, \mathbf{W}} \quad \sum_{ij} x_{ij} \mathbf{z}_i^\top \mathbf{w}_j \equiv \text{trace}(\mathbf{Z}^\top \mathbf{X} \mathbf{W}) \tag{3}$$

$$\text{s.t.} \quad \mathbf{Z} \in \{0, 1\}^{n \times k}, \qquad \mathbf{Z}\mathbf{1} = \mathbf{1} \tag{4}$$

$$\mathbf{W} \in \{0, 1\}^{d \times k}, \quad \mathbf{W}\mathbf{1} = \mathbf{1} \tag{5}$$

For example, a simple heuristic for solving this problem consists in using block coordinate descent, to iteratively solve for the row assignments while the column assignments, and vice versa. This approach shows, how given a clustering of the rows and columns, it is possible to obtain a diagonal co-clustering of the data matrix [38].

### 3.3 Neighborhood Propagation & Graph Convolutional Networks

let $\mathcal{G} = (\mathbf{A}, \mathbf{X})$ be an input attributed graph, with $\mathbf{A}$ the adjacency and $\mathbf{X}$ the node features. The graph convolutional network (GCN) consists of the following step repeated for each layer

$$\mathbf{H}^{(l+1)} \leftarrow \sigma\left(\mathbf{S}\mathbf{H}^{(l)}\mathbf{W}\right) \quad \text{such that} \quad \mathbf{H}^{(0)} \leftarrow \mathbf{X}$$

where $\sigma$ is some activation function, $\mathbf{W}$ are learnable weights that depend on the task at hand, and $\mathbf{S}$ is a normalized version of $\mathbf{A}$ with self-loops added. The simplified version, on the other hand, uses a GCN with linear activations and no weights. For example, the simple equivalent of a GCN with $p$-layers is

$$\mathbf{H} \leftarrow \mathbf{S}^p \mathbf{X}$$

These representations can then be used for some downstream task.

## 4 Proposed Method

Here, we derive a model that combines the concepts of self-expressive subspace clustering, co-clustering and neighborhood propagation. The resulting model surpasses the performance of state of the art methods for all three categories on document-term matrices.

### 4.1 Self-Expressive Subspace Co-clustering

Based on the Block Seriation model for co-clustering, given a document-term matrix $\mathbf{X} \in \mathbb{R}_+^{n \times d}$, the self-expressive subspace co-clustering problem can be formulated as

$$\begin{aligned} \min_{\mathbf{R}, \mathbf{C}} \quad & \| \mathbf{X} - \mathbf{R}\mathbf{X}\mathbf{C} \|^2 + \Omega(\mathbf{R}, \mathbf{C}) \\ \text{such that} \quad & \mathbf{R} \in \mathcal{R},\ \mathbf{C} \in \mathcal{C}. \end{aligned} \tag{6}$$

where $\mathbf{R} \in \mathbb{R}^{n \times n}$ and $\mathbf{C} \in \mathbb{R}^{d \times d}$ are respectively the row and column self-representation matrices, $\Omega(\mathbf{R}, \mathbf{C})$ is the regularization term where the regularization of $\mathbf{R}$ and $\mathbf{C}$ can be either independent (i.e., $\Omega(\mathbf{R}, \mathbf{C}) = \Omega_{\mathbf{R}}(\mathbf{R}) + \Omega_{\mathbf{C}}(\mathbf{C})$) or dependent, and $\mathcal{R}$ and $\mathcal{C}$ are the feasible regions. Note that unlike the Block Seriation model, the generic case of our problem does not require $k$ and $g$, the numbers of row and column clusters respectively, to be equal.

### 4.2 Promoting the Grouping Effect Through a Bilateral Graph Convolution

The performance of subspace clustering methods [19], [39], [40] is due to the grouping effect. While the authors in [19], [40] optimized this property implicitly, [39] sought to enforce it explicitly through the regularization term $\Omega(\mathbf{R})$. We use the definition of the grouping effect given in [39].

**Definition 1.** *(Grouping Effect) Given a data matrix $\mathbf{X}$, a self-representation matrix $\mathbf{R}$ has a grouping effect if*

$$\forall i, j,\ \|\mathbf{x}_i - \mathbf{x}_j\| \to 0 \implies \|\mathbf{r}_i - \mathbf{r}_j\| \to 0. \tag{7}$$

In the case of text, the grouping effect will not necessarily be beneficial, because of its high dimensionality and sparsity. Data points may not be sufficiently "close" (as regards the self-expressive property) to be grouped in a meaningful

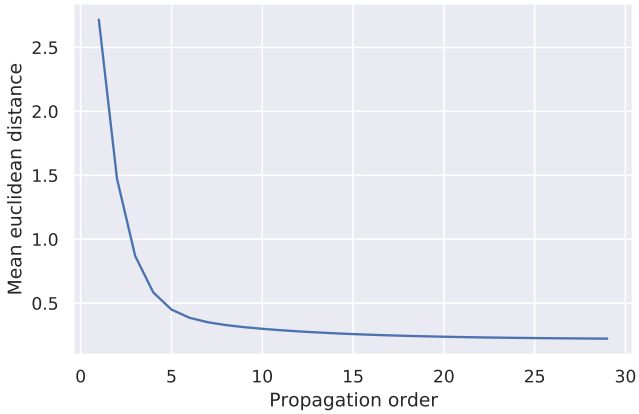



Fig. 1: Mean pairwise euclidean distance of the columns of $\mathbf{S}^p\mathbf{X}$ as $p$ increases on DBLP. The columns get mutually closer as more information propagates over the rows.

way. This means that even if a subspace clustering approach has the grouping effect, in practice $\|\mathbf{x}_i - \mathbf{x}_j\| \to 0$ is not likely, making the property useless. The implication is that data points need some sort of smoothing to make some of the points closer and consequently to help subspace clustering algorithms find common subspaces.

Here, we propose to solve this problem through a bilateral graph convolution preprocessing step, based on simple graph convolution. This requires two similarity matrices to act as graphs on the rows $\mathbf{S_R}$ and columns $\mathbf{S_C}$. These matrices can either be constructed through some similarity measure on the data, e.g. using the $k$-nn approach using the Euclidean or cosine distance, or be provided *a priori* such as in the case of attributed graphs (on the rows at least).

The reasoning, intuitively, is that the rows and columns of $\mathbf{S}_\mathbf{R}^p\mathbf{X}\mathbf{S}_\mathbf{C}^q$, as propagation orders $p, q$ grow, become smoother by being averaged up to their $p$-th and $q$-th neighbors respectively, analogously to Laplacian smoothing. The rows and columns therefore become more similar as powers increase. An illustration of this is shown in figure 1.

**Proposition 1.** *Given a row-normalized adjacency matrix* $\mathbf{S}$ *on the rows of* $\mathbf{X}$*, we have that* $\mathbf{S}$ *is a non-expansive mapping on the columns of* $\mathbf{X}$

$$\forall i \neq j, \quad \|\mathbf{S}\mathbf{x}'_i - \mathbf{S}\mathbf{x}'_j\| \leq \|\mathbf{x}'_i - \mathbf{x}'_j\|$$

*The same holds for the rows of* $\mathbf{X}$ *for* $\mathbf{S}$ *an adjacency matrix over the columns of* $\mathbf{X}$*.*

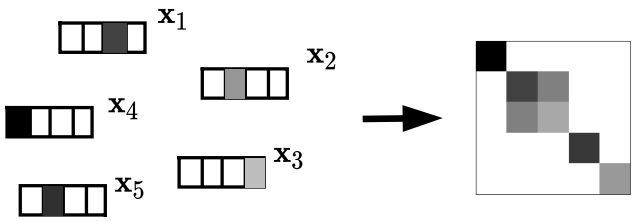
$$\forall i \neq j, \quad \|\mathbf{x}_i\mathbf{S} - \mathbf{x}_j\mathbf{S}\| \leq \|\mathbf{x}_i - \mathbf{x}_j\| \tag{8}$$

*Proof.* Since $\mathbf{S1} = \mathbf{1}$ and $s_{ij} \geq 0$, by the *Gershgorin circle theorem*, the spectral radius of $\mathbf{S}$ is $\rho(\mathbf{S}) = 1$. We therefore have

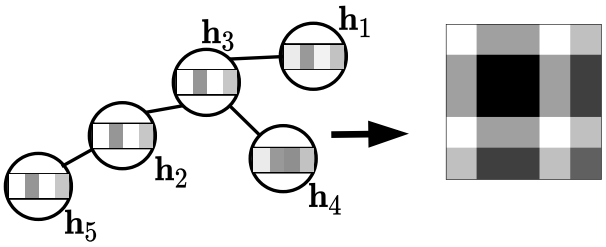
$$\begin{aligned} \|\mathbf{S}\mathbf{x}'_i - \mathbf{S}\mathbf{x}'_j\| &= \|\mathbf{S}(\mathbf{x}'_i - \mathbf{x}'_j)\| \\ &\leq \|\mathbf{S}\|_2\|\mathbf{x}'_i - \mathbf{x}'_j\| \\ &= \|\mathbf{x}'_i - \mathbf{x}'_j\| \end{aligned} \tag{9}$$

since $||\mathbf{S}||_2 = \rho(\mathbf{S})$ where $||.||_2$ is the spectral norm. The proof is the same for the rows. □

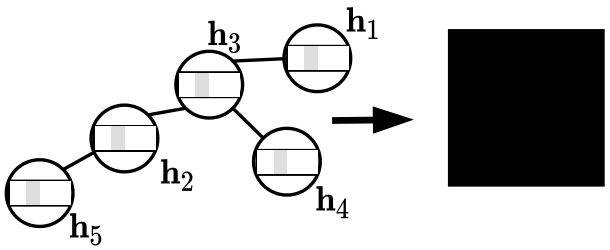
The grouping property also implies that in addition to the features $\mathbf{X}$, the self-representation vectors, i.e., the rows (or, by symmetry, the columns) of $\mathbf{R}$ and $\mathbf{C}$ should also be getting more similar, leading to a more meaningful partitioning when applying spectral clustering on $|\mathbf{R}|$ and $|\mathbf{C}|$ where the absolute value is applied element-wise. In figure 2 we show how propagating the features using a row $k$-nn graph generated from the data using the Euclidean distance impacts the learned self-representation matrix using LSR subspace clustering. We see how the matrix obtained after 10-hop propagation has more nonzero entries than the matrix with no propagation, which indicates, if the self-representation matrix is seen as a graph, that there are more adjacent nodes. The difficulty is identifying the appropriate propagation order, since large values can cause over-smoothing. As we can see from figure 2, the self-representation matrix after 1000-hop propagation is entirely uniform, since node features have converged to uninformative representations.




(a) No propagation.




(b) 10-hop propagation with a $k$-nn graph.




(c) 1000-hop propagation with a $k$-nn graph.

Fig. 2: The resulting self-representation matrices using different levels of propagation over synthetic data with the LSR subspace clustering algorithm.

**Proposition 2.** *Given a row-normalized adjacency matrix with added self-loops* $\mathbf{S}$*,* $\lim_{p\to\infty}\mathbf{S}^p$ *exists and its rank is the same as that of the number of connected components of* $\mathbf{A}$*.*

*Proof.* Since $\mathbf{S}$ is row-stochastic, we have that $\rho(\mathbf{S}) = 1$. Furthermore, since it has added self-loops it cannot be the adjacency matrix of a bipartite graph, and consequently $-1 \notin \sigma(\mathbf{S})$ the spectrum of $\mathbf{S}$.

$$\begin{aligned}\lim_{p\to\infty}\mathbf{S}^p &= \lim_{p\to\infty}\mathbf{U}\mathbf{\Sigma}^p\mathbf{U}^{-1}\\ &= \lim_{p\to\infty}\mathbf{U}\mathtt{diag}([1^p,\dots,1^p,r_1^p,...,r_{n-c}^p])\mathbf{U}^{-1}\\ &= \mathbf{U}\mathtt{diag}([1,\dots,1,0...,0])\mathbf{U}^{-1}\end{aligned} \quad (10)$$

It follows that $\mathtt{rk}(\lim_{p\to\infty}\mathbf{S}^p) = c$, where $\forall i\, r_i \in \sigma(\mathbf{S})$, $|r_i| < 1$ and $c$ is the number of connected components. □

Regarding the bilateral graph convolution of rows and columns of order $p$ and $q$ respectively, we define the generic self-expressive subspace co-clustering with bilateral graph convolution problem as

$$\begin{aligned}\min_{\mathbf{R},\mathbf{C}} \quad & \| \mathbf{S}_\mathbf{R}^p\mathbf{X}\mathbf{S}_\mathbf{C}^q - \mathbf{R}\left(\mathbf{S}_\mathbf{R}^p\mathbf{X}\mathbf{S}_\mathbf{C}^q\right)\mathbf{C}\|^2 + \Omega(\mathbf{R},\mathbf{C})\\ & \text{such that} \quad \mathbf{R}\in\mathcal{R},\ \mathbf{C}\in\mathcal{C}.\end{aligned} \quad (11)$$

In what follows, we will refer to the row- and column-smoothed matrix as

$$\mathbf{H} = \mathbf{S}_\mathbf{R}^p\mathbf{X}\mathbf{S}_\mathbf{C}^q$$

since this operation can be considered as a sort of preprocessing step, independent of the co-clustering model that we are about to introduce.

### 4.3 Subspace Co-clustering through LSR

We propose an initial variant based on the LSR subspace clustering model, where the regularization term is defined as follows: $\Omega(\mathbf{R},\mathbf{C}) = \lambda_\mathbf{R}\|\mathbf{R}\|^2 + \lambda_\mathbf{C}\|\mathbf{C}\|^2$, where $\lambda_\mathbf{R}$ and $\lambda_\mathbf{C}$ are parameters that regulate the trade-off between the reconstruction term and the regularizer. We formulate the LSR subspace co-clustering problem as

$$\min_{\mathbf{R},\mathbf{C}} \quad \|\mathbf{H} - \mathbf{R}\mathbf{H}\mathbf{C}\|^2 + \lambda_\mathbf{R}\|\mathbf{R}\|^2 + \lambda_\mathbf{C}\|\mathbf{C}\|^2. \quad (12)$$

Fixing $\mathbf{R}$ and solving for $\mathbf{C}$ and inversely, a closed form solution can be obtained for both matrices, providing a clear illustration of how our model uses information from the columns for the row space partitioning, and vice versa

$$\begin{aligned}\mathbf{R} &= \mathbf{H}\mathbf{C}^\top\mathbf{H}^\top\left(\mathbf{H}\mathbf{C}\mathbf{C}^\top\mathbf{H}^\top + \lambda_\mathbf{R}\mathbf{I}\right)^{-1}\\ \mathbf{C} &= \left(\mathbf{H}^\top\mathbf{R}^\top\mathbf{R}\mathbf{H} + \lambda_\mathbf{C}\mathbf{I}\right)^{-1}\mathbf{H}^\top\mathbf{R}^\top\mathbf{H}.\end{aligned} \quad (13)$$

However, solving the problem requires an iterative process where, in alternation, one of $\mathbf{R}$ and $\mathbf{C}$ is fixed and the other updated, until convergence. The overall computational complexity is roughly in $\mathcal{O}(n^3 + d^3 + tnd^2 + tn^2d)$ due to the inversion and the necessary spectral clustering step, where $t$ is the number of iterations, and the spatial complexity is $\mathcal{O}(n^2 + d^2)$, which is very often prohibitive for real-world applications, especially those pertaining to text. We therefore try to improve the efficiency of the solving scheme by adding further constraints, as described below.

### 4.4 SC$^3$: A More Efficient Formulation Through Orthogonality Constraints

To address the issue of complexity we introduce factor matrices $\mathbf{Z}\in\mathbb{R}^{n\times k}$ and $\mathbf{W}\in\mathbb{R}^{d\times g}$, which we constrain to be semi-orthogonal, i.e., $\mathbf{Z}^\top\mathbf{Z} = \mathbf{I}_k$ and $\mathbf{W}^\top\mathbf{W} = \mathbf{I}_g$, such that $\mathbf{R} = \mathbf{Z}\mathbf{Z}^\top$ and $\mathbf{C} = \mathbf{W}\mathbf{W}^\top$. With these constraints the LSR co-clustering problem becomes simpler, since $\|\mathbf{Z}\|^2 = \mathtt{rk}(\mathbf{Z})$ and $\|\mathbf{W}\|^2 = \mathtt{rk}(\mathbf{W})$ are constant. The new formulation of the problem, that we have termed SC$^3$, is

$$\begin{aligned}\min_{\mathbf{Z},\mathbf{W}} \quad & \|\mathbf{H} - \mathbf{Z}\mathbf{Z}^\top\mathbf{H}\mathbf{W}\mathbf{W}^\top\|\\ \text{such that} \quad & \mathbf{Z}^\top\mathbf{Z} = \mathbf{I}_k \quad \mathbf{W}^\top\mathbf{W} = \mathbf{I}_g\end{aligned} \quad (14)$$

At first glance this problem also requires an alternating solving scheme using two update rules that we obtain by fixing $\mathbf{W}$ and solving for $\mathbf{Z}$, and vice versa :

$$\mathbf{Z} = \mathbf{U} \quad \text{such that} \quad [\mathbf{U},\mathbf{\Sigma},\mathbf{V}] = \mathtt{TruncatedSVD}(\mathbf{H}\mathbf{W},k)$$
$$\mathbf{W} = \mathbf{U} \quad \text{such that} \quad [\mathbf{U},\mathbf{\Sigma},\mathbf{V}] = \mathtt{TruncatedSVD}(\mathbf{H}^\top\mathbf{Z},g).$$

This entails that $g = k$, making this a block clustering problem reminiscent of the block seriation co-clustering model. The detailed pseudo-code for this method is given in algorithm 1, whose spatial complexity is the same as for LSR, but the computational complexity is in $\mathcal{O}(n^3 + d^3 + tndk)$. Although this method is faster, it remains inefficient owing to bottlenecks, to the iterative nature of the algorithm and, more importantly, because of the spectral clustering step.

**Algorithm 1:** Naive SC$^3$

**Input :** $\mathbf{X}$ feature matrix, $\mathbf{S}_\mathbf{R}$ row propagation matrix, $\mathbf{S}_\mathbf{C}$ column propagation matrix, $k$ number of co-clusters, $p$, $q$ row and column propagation orders, $\epsilon$ tolerance.
**Output:** $\mathbf{Z}$, $\mathbf{W}$ row and column factors, $\pi_\mathbf{R}$, $\pi_\mathbf{C}$ row and columns partitions.

$\mathbf{H} \leftarrow \mathbf{S}_\mathbf{R}^p\mathbf{X}\mathbf{S}_\mathbf{C}^q$;
$[\_,\_,\mathbf{V}] = \mathtt{TruncatedSVD}(\mathbf{H},k)$;
$\mathbf{W}^{(0)} \leftarrow \mathbf{V}$;
**while** $\|\mathbf{Z}^{(k)} - \mathbf{Z}^{(k-1)}\| + \|\mathbf{W}^{(k)} - \mathbf{W}^{(k-1)}\| > \epsilon$ **do**
  $[\mathbf{Z}^{(k)},\_,\_] = \mathtt{TruncatedSVD}(\mathbf{H}\mathbf{W}^{(k-1)},k)$;
  $[\mathbf{W}^{(k)},\_,\_] = \mathtt{TruncatedSVD}(\mathbf{H}^\top\mathbf{Z}^{(k)},k)$;
**end**
$\mathbf{R},\mathbf{C} \leftarrow \mathbf{Z}\mathbf{Z}^\top, \mathbf{W}\mathbf{W}^\top$;
$\pi_\mathbf{R} \leftarrow \mathtt{spectral\_clustering}(|\mathbf{R}|)$;
$\pi_\mathbf{C} \leftarrow \mathtt{spectral\_clustering}(|\mathbf{C}|)$;
Optionally deduce a block diagonal bicluster matrix from $\pi_\mathbf{Z}$ and $\pi_\mathbf{W}$;

#### *Efficiently Solving for $\mathbf{Z}^*$ and $\mathbf{W}^*$*

The problem stated above can be solved efficiently using a single truncated SVD. This is a consequence of the following proposition:

**Proposition 3.** *The alternating process defined in system of equations 4.4 converges to $\mathbf{Z}$ and $\mathbf{W}$ containing the $k$ left and right singular vectors of $\mathbf{H}$ respectively corresponding to the largest $k$ singular values.*

*Proof.* Suppose, without loss of generality, that $k \le \mathtt{rk}(\mathbf{H})$. We have that $k = \mathtt{rk}(\mathbf{Z}) = \mathtt{rk}(\mathbf{W})$, implying that

$$\begin{aligned}\mathtt{rk}\left(\mathbf{Z}\mathbf{Z}^\top\mathbf{H}\mathbf{W}\mathbf{W}^\top\right) &\le \min\{\mathtt{rk}(\mathbf{Z}),\mathtt{rk}(\mathbf{W}),\mathtt{rk}(\mathbf{H})\}\\ &= k.\end{aligned}$$

This means that we are looking for the best $k$-rank approximation of $\mathbf{H}$ for the Frobenius norm. Given $[\mathbf{U},\mathbf{\Sigma},\mathbf{V}] =$

$\texttt{SVD}(\mathbf{H})$, and setting $\mathbf{Z} = \mathbf{U}_k = [\mathbf{u}'_1, \ldots, \mathbf{u}'_k]$ and $\mathbf{W} = \mathbf{V}_k = [\mathbf{v}'_1, \ldots, \mathbf{v}'_k]$, we have

$$\begin{aligned}\|\mathbf{H} - \mathbf{Z}\mathbf{Z}^\top\mathbf{H}\mathbf{W}\mathbf{W}^\top\|^2 &= \|\mathbf{H} - \mathbf{U}_k\mathbf{U}_k^\top\mathbf{U}\boldsymbol{\Sigma}\mathbf{V}^\top\mathbf{V}_k\mathbf{V}_k^\top\|^2 \\ &= \|\mathbf{H} - \mathbf{U}_k[\mathbf{I}_k, \mathbf{0}]\boldsymbol{\Sigma}[\mathbf{I}_k, \mathbf{0}]^\top\mathbf{V}_k^\top\|^2 \\ &= \|\mathbf{H} - [\mathbf{U}_k, \mathbf{0}]\boldsymbol{\Sigma}[\mathbf{V}_k, \mathbf{0}]^\top\|^2 \\ &= \|\mathbf{H} - \mathbf{U}_k\boldsymbol{\Sigma}_k\mathbf{V}_k^\top\|^2.\end{aligned} \tag{15}$$

From the Eckart–Young–Mirsky theorem we have that $\hat{\mathbf{H}} = \mathbf{U}_k\boldsymbol{\Sigma}_k\mathbf{V}_k^\top$ is the best rank-$k$ approximation of $\mathbf{H}$. □

This leads to a more efficient algorithm because the iterative step is circumvented. The interplay between the rows and columns is still implicitly present, however, since this solution is also the analytic solution to the alternating optimization problem referred to above, in which the row-column interaction is explicit.

The above result enables us to show that our approach has a grouping effect.

**Proposition 4.** *Given matrix* $\mathbf{H}$*, the solutions* $\mathbf{R}$ *and* $\mathbf{C}$ *in SC*$^3$ *display a grouping effect on the rows and columns respectively of matrix* $\mathbf{H}$*.*

*Proof.* To this end, we show that $\mathbf{R}$ and $\mathbf{C}$ display a grouping effect on $\mathbf{H}$. We give the proof for $\mathbf{R}$ only since it is similar for $\mathbf{C}$. Let the full singular value decomposition of $\mathbf{H}$ be $\mathbf{H} = \mathbf{U}\boldsymbol{\Sigma}_k\mathbf{V}^\top$. We have that

$$\mathbf{Z} = \begin{pmatrix}\mathbf{I}_k & 0\\ 0 & 0\end{pmatrix}\mathbf{U}$$

and that $\mathbf{u}_i = \boldsymbol{\Sigma}^{-1}\mathbf{V}^\top\mathbf{h}_i$. Given these two equations, it is possible to write

$$\begin{aligned}\|\mathbf{z}_i - \mathbf{z}_j\| &= \left\| \begin{pmatrix}\mathbf{I}_k & 0\\ 0 & 0\end{pmatrix}\mathbf{u}_i - \begin{pmatrix}\mathbf{I}_k & 0\\ 0 & 0\end{pmatrix}\mathbf{u}_j \right\| \\ &= \left\| \begin{pmatrix}\mathbf{I}_k & 0\\ 0 & 0\end{pmatrix}\boldsymbol{\Sigma}^{-1}\mathbf{V}^\top\mathbf{h}_i - \begin{pmatrix}\mathbf{I}_k & 0\\ 0 & 0\end{pmatrix}\boldsymbol{\Sigma}^{-1}\mathbf{V}^\top\mathbf{h}_j \right\| \\ &\leq \|\,\mathbf{h}_i - \mathbf{h}_j\,\| \left\| \begin{pmatrix}\mathbf{I}_k & 0\\ 0 & 0\end{pmatrix}\boldsymbol{\Sigma}^{-1}\mathbf{V}^\top \right\| \\ &= \|\,\mathbf{h}_i - \mathbf{h}_j\,\| \,\text{const.}\end{aligned} \tag{16}$$

We also have that since $\mathbf{R} = \mathbf{Z}\mathbf{Z}^\top$, then $\mathbf{r}_i = \mathbf{Z}\mathbf{z}_i$. Therefore, it holds that

$$\begin{aligned}\|\mathbf{r}_i - \mathbf{r}_j\| &= \|\,\mathbf{Z}(\mathbf{z}_i - \mathbf{z}_j)\| && (17)\\ &= \|\,\mathbf{z}_i - \mathbf{z}_j\,\|. && \text{since } \mathbf{Z}^\top\mathbf{Z} = \mathbf{I}\end{aligned}$$

From equations 16 and 17, we have

$$\forall i, j \quad \|\mathbf{h}_i - \mathbf{h}_j\| \to 0 \implies \|\mathbf{r}_i - \mathbf{r}_j\| \to 0$$

implying that there is a grouping effect on the rows of $\mathbf{H}$ for $\mathbf{R}$. □

This gives us an efficient way to obtain $\mathbf{Z}^*$ and $\mathbf{W}^*$, together with theoretical guarantees regarding the quality of these solutions. The main remaining complexity bottleneck is the spectral clustering step with its cubic computational complexity and its quartic space complexity in the number of rows and columns. However, using the structure of our $\mathbf{R}$ and $\mathbf{C}$ we may obtain a spectral clustering algorithm.

### 4.5 Efficient Spectral Clustering of the Kernel Self Representation Matrices

*Nonnegative feature map*

The optimal self-representation matrix $\mathbf{R}^* = \mathbf{Z}^*\mathbf{Z}^{*\top}$ is symmetric by construction, but its entries are not necessarily nonnegative. An element-wise absolute value would therefore be required in order for us to obtain a valid affinity matrix. However, this would remove all the information already held on the decomposition of $\mathbf{R}^*$ into $\mathbf{Z}^*\mathbf{Z}^{*\top}$, since generally there is no relation between the spectrum of a matrix and its spectrum after applying an entry-wise function. We circumvent this problem by instead considering an affinity matrix constructed through some nonnegative kernel, i.e.,

$$\mathbf{K}_\mathbf{R} = \langle\varphi(\mathbf{Z}^*), \varphi(\mathbf{Z}^*)\rangle \quad \text{such that} \quad k_{ij} \geq 0$$

where $\varphi$ is the feature map of the kernel applied row-wise that we need to calculate explicitly. Two possible approaches are available:

**Exact feature maps.** The kernel trick can provide a means of operating in a high-dimensional, implicit feature space without ever computing the coordinates of the data in that space. Computing explicit feature maps is challenging for most commonly used kernel functions, which tend to increase the dimensionality of the original inputs to such an extent that it becomes impossible to use them in realistic scenarios. For instance, in the case of the second degree polynomial kernel $k(\mathbf{z}_i, \mathbf{z}_j) = (\mathbf{z}_i^\top\mathbf{z}_j + c)^2$ where $c$ is some constant that we can see as a bias term. For example, when $c = 1$ then the feature map is

$$\begin{aligned}\varphi\colon \mathbb{R}^k &\to \mathbb{R}^{\binom{k+2}{2}} \\ \mathbf{z} &\mapsto \langle z_k^2, \ldots, z_1^2, \sqrt{2}z_k z_{k-1}, \ldots, \sqrt{2}z_k z_1, \\ &\quad \sqrt{2}z_{k-1}z_{k-2}, \ldots, \sqrt{2}z_{k-1}z_1, \ldots, \sqrt{2}z_2 z_1, \\ &\quad \sqrt{2}z_k, \ldots, \sqrt{2}z_1, 1\rangle\end{aligned}$$

More generally, for a polynomial kernel of degree $d$ the feature map is a function $\varphi : \mathbb{R}^k \mapsto \mathbb{R}^{\binom{k+d}{k}}$. The other possibility is that the explicit feature map is infinite-dimensional and thus impossible to compute, such as in the case of the Radial Basis Function (RBF) kernel.

**Approximate feature maps** A number of methods using approximations of the feature maps of the desired kernel have been proposed over the years. These include the Nyström method [41], Tensor Sketch [42], and the use of random features [43].

*Efficient Spectral Clustering*

Since the eigenvectors of $\mathbf{K}_\mathbf{R}$ are the same as the left singular vectors of $\varphi(\mathbf{Z}^*)$, the process of spectral clustering becomes much faster, since the affinity matrix does not need to be computed explicitly. For our purposes we adapt the spectral algorithm proposed in [44] as follows:

1) We efficiently compute the diagonal matrix $\mathbf{D}$ containing the sums of the rows of $\mathbf{K}_\mathbf{R}$.
2) We construct matrix $\hat{\mathbf{Z}} = \mathbf{D}^{-1/2}\varphi(\mathbf{Z})$, on which we do an SVD yielding the eigenvectors of $\mathbf{D}^{-1/2}\mathbf{K}_\mathbf{R}\mathbf{D}^{-1/2}$, referred to as $\mathbf{U}$.

3) We obtain the final assignments of the rows of $\mathbf{H}$ according to the assignment of vectors $[\mathbf{u}_1, ...\mathbf{u}_n]^\top$ using the $k$-means algorithm.

This is equivalent to doing the spectral clustering on the normalized Laplacian of $\mathbf{K_R}$ i.e. $\mathbf{L} = \mathbf{I} - \mathbf{D}^{-1/2}\mathbf{K_R}\mathbf{D}^{-1/2}$ rendering the complexity of the spectral clustering linear in the number of samples instead of cubic. To obtain a spectral clustering of the affinity matrix of $\mathbf{C}^*$, the operations are the same.

## 5 Algorithm and Complexity

The pseudo-code for the efficient version of our algorithm is given in algorithm 2. We now discuss the computational complexity in detail.

**Algorithm 2:** Efficient SC$^3$

**Input :** $\mathbf{X}$ feature matrix, $\mathbf{S_R}$ row propagation matrix, $\mathbf{S_C}$ column propagation matrix, $k$ number of co-clusters, $p$, $q$ row and column propagation orders.

**Output:** $\mathbf{Z}$, $\mathbf{W}$ row and column factors, $\pi_{\mathbf{R}}$, $\pi_{\mathbf{C}}$ row and columns partitions.

$\mathbf{H} \leftarrow \mathbf{S}_{\mathbf{R}}^p \mathbf{X} \mathbf{S}_{\mathbf{C}}^q$;
$[\mathbf{Z}, \_, \mathbf{W}] = \texttt{TruncatedSVD}\,(\mathbf{H}, k)$;
$\mathbf{M}_\varphi \leftarrow \varphi(\mathbf{Z})$;
$\mathbf{D} \leftarrow \texttt{diag}(\mathbf{Z}_\varphi(\mathbf{Z}_\varphi^\top \mathbf{1}))$;
$\hat{\mathbf{Z}}_\varphi \leftarrow \mathbf{D}^{-\frac{1}{2}}\mathbf{Z}_\varphi$;
$\mathbf{U} \leftarrow \texttt{TruncatedSVD}\left(\hat{\mathbf{Z}}_\varphi, k\right)$;
$\pi_{\mathbf{Z}} \leftarrow k\texttt{-means}(\mathbf{U})$;
$\mathbf{M}_\varphi \leftarrow \varphi(\mathbf{W})$;
$\mathbf{D} \leftarrow \texttt{diag}(\mathbf{W}_\varphi(\mathbf{W}_\varphi^\top \mathbf{1}))$;
$\hat{\mathbf{W}}_\varphi \leftarrow \mathbf{D}^{-\frac{1}{2}}\mathbf{W}_\varphi$;
$\mathbf{U} \leftarrow \texttt{TruncatedSVD}\left(\hat{\mathbf{W}}_\varphi, k\right)$;
Discard the column of $\mathbf{U}$ corresponding to the largest singular value;
$\pi_{\mathbf{W}} \leftarrow k\texttt{-means}(\mathbf{U})$;
Optionally deduce a block diagonal bicluster matrix from $\pi_{\mathbf{Z}}$ and $\pi_{\mathbf{W}}$;

**Feature Propagation Step** Assuming that the propagation matrix is sparse, the complexity of this operation is in $\mathcal{O}(p\|\mathbf{S_R}\|_0 + q\|\mathbf{S_C}\|_0)$, where $\|.\|_0$ is the 0-norm that gives the number of non-zero entries of its input.

**Truncated SVD on H** The computational complexity of this step using randomized SVD [45] is $\mathcal{O}(nd\log(k))$.

**Truncated SVD on $\varphi(\mathbf{Z})$ and $\varphi(\mathbf{W})$** This depends on the dimensionality of the feature map. Since we consider the affine kernel feature map as the main choice, the complexity using a randomized SVD on it is $\mathcal{O}(nk\log(k) + dk\log(k))$.

**Efficient spectral clustering.** This operation has a complexity in $\mathcal{O}(tnk^2 + tdk^2)$, where $t$ is the number of iterations of $k$-means.

**Overall computational complexity** The overall computational complexity of the proposed SC$^3$ algorithm is thus in $\mathcal{O}\left(p\|\mathbf{S_R}\|_0 + q\|\mathbf{S_C}\|_0 + nd\log(k) + tnk^2 + tdk^2\right)$.

**Overall spatial complexity** The space taken by the created matrices is in $\mathcal{O}\left(nk + dk\right)$.

As seen in Algorithm 2, $k$-means is used to propose a hard clustering. Instead of $k$-means the user can perform any other clustering method including fuzzy clustering method such as soft $k$-means. This implies that each data point can belong to more than one cluster; for each of them a set of coefficients gives the degree of being in each cluster. It is also the case of the EM algorithm [46] where in E-step posterior probabilities of each data point are computed.

## 6 Experiments

In this section we present the experimental setup and results. We consider two tasks which are co-clustering and document clustering. We start by introducing the models used for comparison and the overall experimental settings. Then, for each task we present the datasets and evaluation metrics as well as the results that were obtained. We follow with some comparisons between the different possible nonnegative kernels, and finally present the results for neighborhood propagation when a $k$-nn graph is used instead of the ground truth graph.

### 6.1 Experimental Setup

#### *Baselines*

We compare our model against clustering and co-clustering models that either use only the input node feature matrix $\mathbf{X}$ or that use both $\mathbf{A}$ and $\mathbf{X}$, that is to say *attributed* graph clustering/co-clustering models.

**Clustering models** :

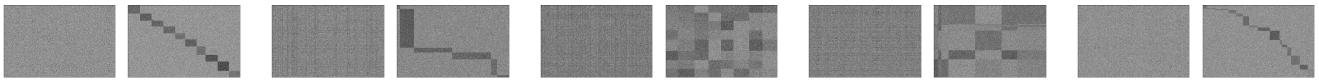
- *Vanilla models*: We take $k$-means as the simplest baseline for our comparison.
- *Subspace Clustering Models*: We compare our model against the subspace clustering models previously mentioned: LSR, EnSC, $\ell^0$-SCC and Noisy-DR-$\ell^0$-SCC-LR, SSC-OMP and the mini-batch version of K-FSC.
- *Attributed graph clustering models*: We use the GIC, S$^2$GC and GCC models.

**Co-clustering models** :

- *Vanilla models*: We compare our model against the spectral co-clustering algorithm (SpecCo), DCC and RDPCo.
- *Attributed graph co-clustering Models*: The only existing model of this kind is CFOND.

#### *Experimental Settings*

For our method, we normalize the word count datasets using tf-idf and unit-normalize across the the rows. We use the official implementation (with the recommended or default parameters) for each of them, apart from CFOND ($\rho = \beta = \alpha = 1$), LSR ($\lambda = 1$), and RDPCo (with $S_r$ and $S_c$), $\ell^0$-SCC and Noisy-DR-$\ell^0$-SCC-LR (see [22], [23] for parameter setting) which, to the best of our knowledge, have not been made publicly available. We also use the recommended parameters for each dataset whenever possible. For models that use a parameter $p$ like ours, we run their selection rule until convergence with a maximum $p$ of 100 for fairness. All models were run on the same machine with 12GB memory GPU and a RAM of 12GB. All experiments for the different models were run on the same machine with 12GB of RAM and a 2.3Ghz Xeon Processor. We perform 20 runs for each model.



(a) A. (b) B. (c) C. (d) D. (e) E.

Fig. 3: Synthetic datasets before and after rearrangement with respect to the true partitions.

**Algorithm 3:** Propagation Order Selection Rule

**Input :** $\mathbf{X}$ document-term matrix, $\mathbf{S_R}$ row propagation matrix, $\mathbf{S_C}$ column propagation matrix, $q$ column propagation order, $k$ number of co-clusters.
**Output:** Propagation order $p^*$.
$p \leftarrow 1$;
**while** $p*$ *not found* **do**
  $\mathbf{Z}, \mathbf{W}, \_, \_ \leftarrow \texttt{SC}^3(\mathbf{X}, \mathbf{S_R}, \mathbf{S_C}, p, q, k)$;
  $\text{loss}_p \leftarrow \left\|\mathbf{S}_\mathbf{R}^p\mathbf{X} - \mathbf{Z}\mathbf{Z}^\top\mathbf{S}_\mathbf{R}^p\mathbf{X}\mathbf{W}\mathbf{W}^\top\right\|$;
  **if** $|\, loss_p - loss_{p-1}\,| < \frac{d}{n\lceil\sqrt{k}\rceil}$ ***or*** $p = 100$ **then**
    $p* \leftarrow p$
  **end**
  $p \leftarrow p + 1$
**end**

### *Choice of Propagation Matrices*

In our model, we use a $k$-nn graph generated from the rows of matrix $\mathbf{X}$ using the euclidean distance with $k = 3$. We use a propagation order of ten, $p = 10$. For the columns, we do not use a $k$-nn graph, but rather a graph based on NNPMI, which is nonnegative version of Pointwise Mutual Information (PMI) traditionally used to quantify word-relatedness. The column propagation order is set to $q = 1$. The column NNPMI graph is defined as

$$\mathbf{S_C} = (s_{\mathbf{C}ij}) = \left(\max\left\{\log\left(\frac{y_{..}}{y_{i.}y_{.j}}y_{ij}\right), 0\right\}_{ij}\right) \tag{18}$$

where $y_{ij} = \mathbf{x}_i^\top\mathbf{x}_j$. We use the nonnegative version so that the resulting matrix after propagation can still be seen as a document-term matrix. Both matrices are then normalized using the normalization proposed in [35].

## 6.2 Co-clustering

### *Synthetic Datasets*

Due to the absence of datasets with labels along both rows and columns. We propose to use synthetic datasets for the evaluation of the co-clustering performance of our model. The five datasets are depicted in figure 3 before and after rearrangement with respect to the ground truth partition. The characteristics of these datasets are available in table 1. These datasets have one of two possible types of structures, checkerboard [47] and block diagonal [3].

TABLE 1: Synthetic datasets' characteristics.

| Dataset | Rows | Columns | Biclusters | Proportions | Structure |
|---|---|---|---|---|---|
| A | 500 | 500 | 10 | equal | Block diagonal |
| B | 800 | 1000 | 6 | unequal | Block diagonal |
| C | 800 | 800 | 8 | equal | Checkerboard |
| D | 2000 | 1200 | 7 | unequal | Checkerboard |
| E | 2500 | 2500 | 15 | unequal | Checkerboard |

### *Evaluation Metrics*

To compare our algorithms we rely on the *clustering accuracy* that is computed by rearranging the rows of the confusion matrix so as to obtain the greatest possible trace using the Hungarian method. Clustering accuracy then corresponds to this trace divided by the number of elements. However, here we use a metric that simultaneously quantifies the clustering accuracy over rows and columns which we call co-clustering accuracy. Given $c(\pi_r)$, the accuracy over the rows; and $c(\pi_c)$, the accuracy over the columns. The co-clustering accuracy $cca$ [6] is given by

$$cca(\pi_r, \pi_c) = c(\pi_r) + c(\pi_r) - c(\pi_r) \times c(\pi_r)$$

TABLE 2: Co-clustering performance on the synthetic datasets averaged over 20 runs of the different co-clustering models. Our model finds the ground truth partition in almost all cases and has the best performance (or is tied for best) over all datasets.

| Dataset | A | B | C | D | E |
|---|---|---|---|---|---|
| SpecCo | 89.53±1.39 | 99.54±0.0 | 94.32±0.0 | 99.95±0.02 | 78.43±1.5 |
| DCC | 98.61±1.0 | 99.96±0.1 | 45.0±15.0 | 72.06±14.1 | 98.87±1.0 |
| RDpCo | 85.1±6.3 | 86.74±6.7 | 28.18±0.0 | 64.19±14.9 | 22.5±0.0 |
| CFOND | **100.0±0.0** | 97.9±2.0 | **100.0±0.0** | 99.25±0.7 | 98.16±0.8 |
| **SC$^3$** | **100.0±0.0** | **100.0±0.0** | **100.0±0.0** | **100.0±0.0** | **99.87±0.2** |

### *Performance*

Table 2 shows the co-clustering results of our algorithm with respect to other co-clustering algorithms. We see that the models that use a topological information generally outperform the other models. Our model finds the ground truth partition in four out of five cases and has the best performance (or is tied for best) over the five datasets.

## 6.3 Document Clustering

### *Datasets*

We use four attributed graph citation networks, which are graphs characterized by an adjacency matrix $\mathbf{A}$ and a node document-term matrix $\mathbf{X}$. The summary statistics are given in table 3. The nodes in ACM and Citeseer correspond to word count vectors, and those in PubMed and Wiki to tf-idf weighted word vectors. We use the ACM dataset, whose graph is not very informative, to compare the robustness of the models that use this information.

### *Evaluation Metrics*

To assess the performance of different algorithms on the document clustering task, we use three commonly used clustering performance metrics: clustering accuracy (Acc), Adjusted Rand Index (ARI) [51], and normalized mutual information (NMI) [52].

TABLE 3: Document datasets' statistics.

| Dataset | #Nodes | #Edges | #Features | #Classes |
|---|---|---|---|---|
| ACM [48] | 3025 | 9150593 | 1870 | 3 |
| CiteSeer [18] | 3327 | 4732 | 3703 | 6 |
| PubMed [18] | 19717 | 44338 | 500 | 3 |
| Wiki [49] | 2405 | 17981 | 4973 | 17 |
| Computers [50] | 13381 | 259159 | 767 | 10 |

*Choice of Propagation Matrices*

Here, we set the row graph to be the adjacency matrix provided in each attributed graph dataset $\mathbf{S_R} = \mathbf{A}$, but later we will test our model with a $k$-nn graph generated from the rows of matrix $\mathbf{X}$. The row propagation order $p$ is selected using the selection rule given in algorithm 3. The column propagation matrix and normalizations used are the same as for the synthetic data.

*Performance*

We consider three versions for our method corresponding to three different nonnegative kernel functions, linear, quadratic and radial basis (rbf) functions. Note that the linear kernel does not really result in a proper affinity matrix but is has a similar behaviour to the other valid kernels when using a bias term, so it can be a more efficient surrogate for them. Tables 4,5 show the row/document clustering performance of the different models. We report results for the power selected using rule 3. Overall, we see that methods using both graph structure and features outperform methods using features only; this is the case for all datasets, with the exception of ACM, where the graph is nearly strongly connected. We included the ACM dataset to show the robustness of our model in the face of an uninformative graph structure when compared with state-of-the-art attributed graph clustering models. Our model is seen to be competitive on all five datasets, and to have the best performances on all five datasets. SC$^3$ has either the best or the second best performance with respect to most metrics on each dataset, while having near-zero standard deviation, which shows that the proposed approach is robust. The performance gap is most striking on Wiki, where our model is seven points ahead of the closest model in terms of accuracy. We also see that the performance of the three variants of our model is similar.

For statistical significance, we perform a Nemenyi post-hoc test [53] with a confidence level of 95% on the ranks of each model in terms of Acc, NMI, and ARI, for each dataset and for each run (20 runs for each dataset) to compare our model with the other best performing models, namely SGC, S²GC, GIC, GCC and CFOND. This allows us to group models with similar performances. The results are shown in figure 4. The best group consisting of the SC$^3$ variants is seen to have the best performances by a wide margin, followed by the group containing only GCC.

Compared to co-clustering methods applied on $\mathbf{X}$, note that sometimes, with certain some methods we have zero values for NMI and ARI. This is explained by the fact that we are facing the recurrent problem of empty classes when dealing with sparse and unbalanced data. The problem is, however, overcome by SC$^3$ as illustrated for ACM.

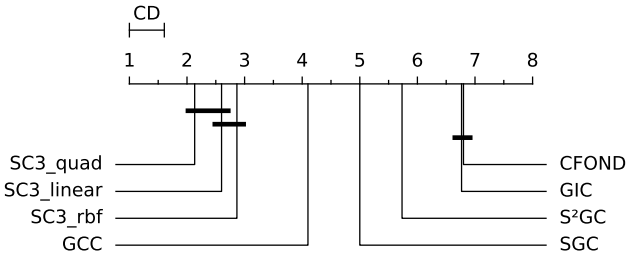



Fig. 4: Visualization of the results of the Nemenyi test with a confidence level of 95%. We see that SC$^3$ variants perform similarly while being better than other models.

*Efficiency*

In figure 5 we plot the accuracy of the subspace clustering models in relation to their training times, comparing their performance on four datasets to that of linear SC$^3$. The results correspond to an average over 20 runs. The time required by the propagation step is included in the overall running time of our algorithm. The fastest model on all datasets is linear SC$^3$, it also achieves the best accuracy score. Note that while the other models give a partition for the rows exclusively, ours gives one for both rows and columns while remaining significantly more efficient.

TABLE 6: The three topics found by SC$^3$ characterized by their top ten most frequent terms, their size and coherence.

| Topic | *a* | *b* | *c* |
|---|---|---|---|
| Most Frequent Terms | patient | cell | rat |
| | insulin | mice | control |
| | glucos | islet | activ |
| | type | iddm | level |
| | group | gene | increas |
| | subject | diseas | respons |
| | lt | develop | signific |
| | risk | nod | effect |
| | associ | children | express |
| | treatment | betacel | plasma |
| Coherence | -403.4 | -365.5 | -387.5 |
| Size | 280 | 63 | 157 |

### 6.4 Term Clustering

Since co-clustering models additionally generate a clustering for the terms. We use the PubMed dataset which is the only dataset for which we managed to find the actual terms. Table 6 presents the most frequent terms for each topic found by our model. The PubMed dataset contains scientific papers concerning diabetes. We see that topic $a$ contains terms that are related to a presentation of diabetes, e.g, *insulin*, *glucos*, *type*, etc. Topic $b$ has terms that are related to the microscopic effects of diabetes such as *cell*, *islet*, *gene*, *betacel*, and so on. Finally, topic $c$ seems to concern terms that are associated with medical experimentation and analysis of results such as *control*, *increas*, *signific*, etc. We note the coherence of these term clusters since they cluster the PubMed paper contents according to three topics. This term clustering can then be used to help characterize document clusters to facilitate interpretation.

In order to quantitatively evaluate the semantic coherence of topics, we use an internal metric [54] to measure the

TABLE 4: Clustering results on ACM, Citeseer and Wiki.

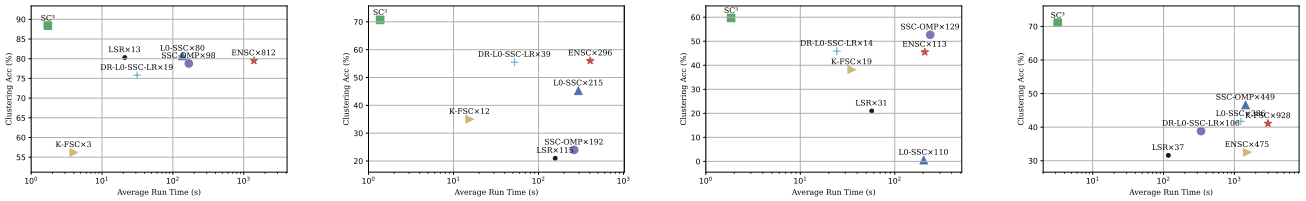
| Method | Input | Co-clustering | Subspace clustering | ACM Acc | ACM NMI | ACM ARI | CiteSeer Acc | CiteSeer NMI | CiteSeer ARI | Wiki Acc | Wiki NMI | Wiki ARI |
|---|---|---|---|---|---|---|---|---|---|---|---|---|
| $k$-means | $\mathbf{X}$ | ✗ | ✗ | 62.8±4.8 | 37.2±9.2 | 34.5±10.4 | 62.5±1.6 | 36.7±1.9 | 35.5±2.5 | 47.3±6.0 | 46.3±6.9 | 26.4±8.1 |
| LSR | $\mathbf{X}$ | ✗ | ✔ | 80.3±0.0 | 47.0±0.0 | 51.9±0.0 | 21.1±0.0 | 0.2±0.1 | 0.0±0.0 | 21.1±3.3 | 9.0±5.9 | 2.6±2.0 |
| EnSC | $\mathbf{X}$ | ✗ | ✔ | 79.5±0.0 | 46.8±0.0 | 50.3±0.0 | 55.6±0.0 | 14.8±0.0 | 14.6±0.0 | 45.5±2.0 | 45.7±1.7 | 28.8±1.3 |
| SSC-OMP | $\mathbf{X}$ | ✗ | ✔ | 78.8±0.1 | 43.4±0.1 | 48.3±0.1 | 24.0±1.1 | 3.5±0.4 | 1.8±0.1 | 52.7±4.4 | 48.1±2.3 | 33.3±1.5 |
| $\ell^0$-SSC | $\mathbf{X}$ | ✗ | ✔ | 80.7±0.0 | 48.4±0.0 | 52.8±0.0 | 55.4±0.0 | 26.3±0.0 | 24.5±0.0 | 45.1±0.0 | 43.8±0.0 | 25.6±0.0 |
| DR-$\ell^0$SCC-LR | $\mathbf{X}$ | ✗ | ✔ | 75.8±2.1 | 42.4±2.7 | 41.5±4.0 | 58.1±0.6 | 27.1±0.3 | 26.2±0.6 | 45.8±0.1 | 45.2±0.4 | 25.3±0.7 |
| K-FSC | $\mathbf{X}$ | ✗ | ✔ | 56.2±7.5 | 17.1±6.3 | 18.0±5.7 | 35.3±6.9 | 12.4±3.5 | 10.6±4.0 | 38.2±5.1 | 35.6±3.9 | 17.7±4.4 |
| SpecCo | $\mathbf{X}$ | ✔ | ✗ | 80.6±0.1 | 48.4±0.1 | 52.3±0.1 | 30.3±1.7 | 10.0±1.3 | 5.5±1.6 | 37.8±1.2 | 38.2±0.3 | 20.8±0.4 |
| DCC | $\mathbf{X}$ | ✔ | ✗ | 40.5±3.3 | 7.8±5.1 | 2.1±2.2 | 35.1±3.8 | 11.5±2.4 | 8.9±2.9 | 48.3±3.6 | 47.5±2.6 | 30.6±3.0 |
| RDPCo | $\mathbf{X}$ | ✔ | ✗ | 35.1±0.0 | 0.0±0.0 | 0.0±0.0 | 46.3±3.2 | 12.6±6.3 | 8.3±4.2 | 18.8±3.1 | 5.6±7.7 | 1.4±2.3 |
| GIC | $\mathbf{A}, \mathbf{X}$ | ✗ | ✗ | 34.3±0.4 | 0.1±0.1 | 0.0±0.0 | 68.8±0.8 | 43.8±1.0 | 44.6±1.0 | 46.5±1.4 | 48.2±0.5 | 30.2±1.4 |
| SGC | $\mathbf{A}, \mathbf{X}$ | ✗ | ✗ | 83.7±0.0 | 55.7±0.0 | 58.8±0.0 | 64.9±0.1 | 39.4±0.0 | 38.8±0.0 | 51.9±0.8 | 49.6±0.2 | 28.6±0.1 |
| S²GC | $\mathbf{A}, \mathbf{X}$ | ✗ | ✗ | 40.5±3.4 | 1.7±1.2 | 1.8±1.3 | 68.1±0.3 | 42.3±0.2 | 43.5±0.3 | 52.7±1.0 | 49.0±0.3 | 29.6±0.9 |
| GCC | $\mathbf{A}, \mathbf{X}$ | ✗ | ✗ | 35.4±0.0 | 0.3±0.0 | 0.0±0.0 | 69.4±0.1 | **45.0±0.2** | 45.4±0.1 | 54.1±0.8 | **55.0±0.2** | **33.3±0.5** |
| CFOND | $\mathbf{A}, \mathbf{X}$ | ✔ | ✗ | 71.8±0.6 | 37.2±0.5 | 38.2±0.7 | 63.0±1.1 | 36.6±1.3 | 36.2±1.2 | 47.8±3.0 | 49.5±2.1 | 30.3±2.5 |
| $\mathbf{SC}^3_{\mathbf{linear}}$ | $\mathbf{A}, \mathbf{X}$ | ✔ | ✔ | **88.4±0.1** | **62.0±0.2** | **68.6±0.1** | 69.3±3.8 | 43.7±2.7 | 43.9±3.8 | **59.7±1.9** | 53.9±1.5 | 31.2±3.1 |
| $\mathbf{SC}^3_{\mathbf{quad}}$ | $\mathbf{A}, \mathbf{X}$ | ✔ | ✔ | **88.4±0.0** | **62.0±0.1** | **68.6±0.1** | **70.7±1.3** | 44.7±1.0 | **45.5±2.0** | 58.2±3.2 | 53.4±1.8 | 30.5±4.2 |
| $\mathbf{SC}^3_{\mathbf{rbf}}$ | $\mathbf{A}, \mathbf{X}$ | ✔ | ✔ | **88.4±0.1** | **62.0±0.2** | **68.6±0.1** | 70.4±1.1 | 44.4±0.9 | 44.8±1.8 | 57.0±3.0 | 52.9±2.1 | 28.4±3.8 |

TABLE 5: Clustering results on PubMed and Amazon Computers.

| Method | Input | Co-clustering | Subspace clustering | PubMed Acc | PubMed NMI | PubMed ARI | Computers Acc | Computers NMI | Computers ARI |
|---|---|---|---|---|---|---|---|---|---|
| $k$-means | $\mathbf{X}$ | ✗ | ✗ | 60.1±0.0 | 31.4±0.0 | 28.1±0.0 | 36.4±1.6 | 7.0±8.5 | -0.2±1.4 |
| LSR | $\mathbf{X}$ | ✗ | ✗ | | OOM | | 31.6±0.5 | 22.3±0.5 | 11.0±0.2 |
| EnSC | $\mathbf{X}$ | ✗ | ✗ | 55.6±0.0 | 14.8±0.0 | 14.7±0.0 | 32.5±1.0 | 32.7±2.6 | 15.9±1.1 |
| SSC-OMP | $\mathbf{X}$ | ✗ | ✗ | 60.4±0.0 | 22.3±0.0 | 19.4±0.0 | 46.6±1.3 | 29.1±0.3 | 30.0±3.1 |
| $\ell^0$-SCC | $\mathbf{X}$ | ✗ | ✗ | | OOM | | 41.2±0.0 | 44.5±0.0 | 22.0±0.0 |
| DR-$\ell^0$SCC-LR | $\mathbf{X}$ | ✗ | ✗ | | OOM | | 38.4±0.4 | 38.1±0.2 | 20.3±0.2 |
| K-FSC | $\mathbf{X}$ | ✗ | ✔ | 49.5±9.8 | 14.8±5.9 | 12.1±7.1 | 41.1±2.5 | 28.9±3.2 | 16.5±3.5 |
| SpecCo | $\mathbf{X}$ | ✔ | ✔ | 61.2±0.0 | 24.7±0.0 | 21.8±0.0 | 31.2±1.2 | 29.9±1.5 | 10.6±1.8 |
| DCC | $\mathbf{X}$ | ✔ | ✔ | 54.3±3.6 | 16.5±2.9 | 13.4±4.1 | 34.3±1.9 | 3.0±1.5 | -2.2±1.4 |
| RDPCo | $\mathbf{X}$ | ✔ | ✔ | 21.4±0.2 | 1.0±0.5 | 0.2±0.1 | 37.1±0.0 | 0.0±0.0 | 0.0±0.0 |
| GIC | $\mathbf{A}, \mathbf{X}$ | ✗ | ✗ | 64.3±0.4 | 26.0±0.5 | 23.6±0.5 | 46.8±2.2 | 47.5±0.9 | 31.3±3.5 |
| SGC | $\mathbf{A}, \mathbf{X}$ | ✗ | ✗ | 64.7±0.0 | 54.3±0.0 | 48.5±0.0 | 65.5±0.0 | 52.2±0.0 | 45.7±0.0 |
| S²GC | $\mathbf{A}, \mathbf{X}$ | ✗ | ✗ | 70.7±0.0 | 32.9±0.0 | 33.5±0.0 | 58.3±0.0 | 54.6±0.0 | 40.1±0.0 |
| GCC | $\mathbf{A}, \mathbf{X}$ | ✗ | ✗ | 70.8±0.0 | 32.3±0.0 | 33.2±0.0 | 67.6±0.0 | 56.0±0.0 | 46.5±0.0 |
| CFOND | $\mathbf{A}, \mathbf{X}$ | ✔ | ✗ | 60.1±0.0 | 31.4±0.0 | 28.1±0.0 | 23.6±0.8 | 13.3±1.3 | 7.0±0.8 |
| $\mathbf{SC}^3_{\mathbf{linear}}$ | $\mathbf{A}, \mathbf{X}$ | ✔ | ✔ | **71.1±0.0** | **33.2±0.0** | **33.9±0.0** | **71.1±0.0** | **59.0±0.0** | **50.6±0.0** |
| $\mathbf{SC}^3_{\mathbf{quad}}$ | $\mathbf{A}, \mathbf{X}$ | ✔ | ✔ | **71.1±0.0** | **33.2±0.0** | **33.9±0.0** | **71.1±0.0** | **59.0±0.0** | **50.6±0.0** |
| $\mathbf{SC}^3_{\mathbf{rbf}}$ | $\mathbf{A}, \mathbf{X}$ | ✔ | ✔ | **71.1±0.0** | **33.2±0.0** | **33.9±0.0** | **71.1±0.0** | **59.0±0.0** | **50.6±0.0** |

TABLE 7: Performance of SC$^3$ with a quadratic kernel using different column propagation matrices averaged over 20 runs. Best results are highlighted in bold font.

| Graph | $k$ | ACM Acc | ACM NMI | ACM ARI | Citeseer Acc | Citeseer NMI | Citeseer ARI | Wiki Acc | Wiki NMI | Wiki ARI | Pubmed Acc | Pubmed NMI | Pubmed ARI | Computers Acc | Computers NMI | Computers ARI |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| k-nn (euclidean) | 3 | 88.7 | 62.8 | 69.2 | 68.0 | 42.8 | 43.8 | 52.9 | 50.3 | 19.0 | **71.3** | **33.4** | **34.2** | 70.5 | 58.7 | 50.3 |
| | 5 | 88.7 | 63.1 | 69.3 | 68.3 | 43.6 | 44.3 | 56.7 | 51.4 | 24.5 | **71.2** | 33.2 | 34.1 | 70.8 | 59.2 | 48.9 |
| | 10 | **88.8** | **63.3** | **69.5** | **68.6** | **44.2** | **44.8** | 58.4 | 52.7 | 26.4 | **71.2** | **33.3** | **34.2** | **72.6** | 60.5 | **52.6** |
| k-nn (cosine) | 3 | 88.0 | 61.1 | 67.7 | 67.3 | 42.3 | 42.7 | 54.7 | 51.7 | 24.0 | **71.2** | 33.0 | 34.0 | 71.3 | 60.2 | 49.7 |
| | 5 | 87.9 | 61.0 | 67.3 | 67.2 | 42.8 | 42.8 | 55.7 | 52.0 | 23.8 | 71.1 | 32.9 | 33.9 | 72.4 | 60.5 | 52.2 |
| | 10 | 87.9 | 60.9 | 67.4 | 67.3 | 42.8 | 43.0 | 55.4 | 51.6 | 25.3 | 71.0 | 32.8 | 33.8 | 72.5 | **60.6** | 52.4 |
| k-nn (correlation) | 3 | 88.6 | 62.7 | 69.0 | 67.3 | 42.4 | 43.1 | **58.8** | **53.3** | **27.3** | **71.2** | 33.1 | **34.2** | **72.6** | 60.5 | **52.6** |
| | 5 | 88.7 | 62.9 | 69.2 | 68.1 | 43.4 | 43.9 | 55.7 | 51.1 | 26.6 | 71.1 | 32.9 | 34.0 | **72.6** | 60.5 | 52.5 |
| | 10 | 88.3 | 62.2 | 68.5 | 68.1 | 43.8 | 44.2 | 57.1 | 52.5 | 26.2 | 71.1 | 32.9 | 34.0 | 72.5 | 60.5 | 52.4 |




(a) ACM. (b) CiteSeer. (c) Wiki. (d) Computers.

Fig. 5: Clustering accuracy plotted against training times on a logarithmic scale of subspace clustering algorithms on the different datasets. Linear SC$^3$ timing is used as the reference; for instance, on ACM, LSR $\times 13$ means that it is approximately 10 times slower than SC$^3$. Linear SC$^3$ consistently gives the best results and training times. PubMed is left out due to OOMs.

level of association between terms within each cluster. Essentially, if a majority of the words within a term cluster are interconnected, it can be inferred that the cluster is semantically coherent from a statistical standpoint. Topic coherence measures the degree of semantic similarity between the top words within a topic or term cluster. It takes into account both the intrinsic coherence of the words themselves, as well as their coherence within the context of the topic or cluster as a whole. This metric has been shown to be effective in assessing the quality of topic models and identifying topics that are most representative of the underlying data. We consider the top 10 terms when computing this metric. We also report the sizes of the topics which are considered as a good metric for assessing topic quality.

### 6.5 Convolution Using $k$-nn Graphs

Unless we are working with attributed graphs, no ground truth adjacency matrix on the rows is provided, but only the feature matrix. Consequently, we need to generate an adjacency matrix on the rows ourselves. A popular choice is the $k$-nearest neighbors graph, based on some metric. In table 7 we compare results on the five datasets using $k$-nn graphs generated based on the Euclidean and cosine dissimilarity with a number of neighbors $k \in \{3, 5, 10\}$. We symmetrize the obtained $k$-nn matrix $\mathbf{A}$ as follows: $(\mathbf{A} + \mathbf{A}^\top)/2$. We see that even without using the ground truth graph, our model continues to outperform subspace clustering and co-clustering models, including CFOND, which uses the actual ground truth graph on all datasets. It also outperforms attributed graph clustering models on Wiki and ACM. On ACM, the results obtained by our model results are better when using the $k$-nn graph than when using the ground truth graph, since the ACM ground truth graph is not very informative. To sum up, we note that whatever the number of neighbors $k \in \{3, 5, 10\}$ and the similarity measure used, SC$^3$ remains profitable for data even without ground truth adjacency matrix on the data points.

## 7 Conclusion

We proposed SC$^3$, a new approach to leverage subspace clustering for text data through co-clustering and a bilateral graph convolution. SC$^3$ circumvents the computational and spatial complexity issues inherent in subspace clustering by using factor matrices and nonnegative kernel feature maps. We showed that the simple model that we propose has a grouping effect, and we demonstrated how the bilateral convolution helps to put this grouping property to good use. Experiments on synthetic and real datasets showed that our model is competitive with the state of the art on the tasks of document and word clustering in the context of document-term attributed graphs, while also being efficient (linear complexity in the number of nodes) and robust in the face of uninformative graph topologies. Even when no ground truth graph is available, the bilateral convolution operation improves performance in comparison to classical subspace clustering approaches.

## References

[1] C. C. Aggarwal and C. Zhai, "A survey of text clustering algorithms," *Mining text data*, pp. 77–128, 2012.
[2] L. Parsons, E. Haque, and H. Liu, "Subspace clustering for high dimensional data: a review," *Acm sigkdd explorations newsletter*, vol. 6, no. 1, pp. 90–105, 2004.
[3] I. S. Dhillon, "Co-clustering documents and words using bipartite spectral graph partitioning," in *International conference on Knowledge discovery and data mining*, 2001, pp. 269–274.
[4] P. Riverain, S. Fossier, and M. Nadif, "Semi-supervised latent block model with pairwise constraints," *Machine Learning*, vol. 111, no. 5, pp. 1739–1764, 2022.
[5] G. Govaert and M. Nadif, *Co-clustering: models, algorithms and applications*. John Wiley & Sons, 2013.
[6] ——, "Block clustering with bernoulli mixture models: Comparison of different approaches," *Computational Statistics & Data Analysis*, vol. 52, no. 6, pp. 3233–3245, 2008.
[7] A. Salah, M. Ailem, and M. Nadif, "Word co-occurrence regularized non-negative matrix tri-factorization for text data co-clustering," in *Proceedings of the AAAI Conference on Artificial Intelligence*, vol. 32, no. 1, 2018.
[8] C. Fettal, lazhar labiod, and M. Nadif, "Efficient and effective optimal transport-based biclustering," in *Advances in Neural Information Processing Systems*, 2022.
[9] H. Jelodar, Y. Wang, C. Yuan, X. Feng, X. Jiang, Y. Li, and L. Zhao, "Latent dirichlet allocation (lda) and topic modeling: models, applications, a survey," *Multimedia Tools and Applications*, vol. 78, pp. 15 169–15 211, 2019.
[10] M. Ailem, F. Role, and M. Nadif, "Model-based co-clustering for the effective handling of sparse data," *Pattern Recognition*, vol. 72, pp. 108–122, 2017.
[11] Y. Chen, H. Zhang, R. Liu, Z. Ye, and J. Lin, "Experimental explorations on short text topic mining between lda and nmf based schemes," *Knowledge-Based Systems*, vol. 163, pp. 1–13, 2019.
[12] M. Defferrard, X. Bresson, and P. Vandergheynst, "Convolutional neural networks on graphs with fast localized spectral filtering," *Advances in neural information processing systems*, vol. 29, 2016.
[13] T. N. Kipf and M. Welling, "Semi-supervised classification with graph convolutional networks," *arXiv preprint arXiv:1609.02907*, 2016.
[14] F. Wu, A. Souza, T. Zhang, C. Fifty, T. Yu, and K. Weinberger, "Simplifying graph convolutional networks," in *International conference on machine learning*. PMLR, 2019, pp. 6861–6871.

[15] C. Fettal, L. Labiod, and M. Nadif, "Scalable attributed-graph subspace clustering," in *Proceedings of the AAAI Conference on Artificial Intelligence*, vol. 37, 2023.
[16] ——, "Subspace co-clustering with two-way graph convolution," in *Proceedings of the 31st ACM International Conference on Information & Knowledge Management*, 2022, pp. 3938–3942.
[17] W. Fan, Y. Ma, Q. Li, Y. He, E. Zhao, J. Tang, and D. Yin, "Graph neural networks for social recommendation," in *The World Wide Web Conference*, 2019, p. 417–426.
[18] P. Sen, G. Namata, M. Bilgic, L. Getoor, B. Galligher, and T. Eliassi-Rad, "Collective classification in network data," *AI magazine*, vol. 29, no. 3, pp. 93–93, 2008.
[19] C.-Y. Lu, H. Min, Z.-Q. Zhao, L. Zhu, D.-S. Huang, and S. Yan, "Robust and efficient subspace segmentation via least squares regression," in *European conference on computer vision*. Springer, 2012, pp. 347–360.
[20] C. You, C.-G. Li, D. P. Robinson, and R. Vidal, "Oracle based active set algorithm for scalable elastic net subspace clustering," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2016, pp. 3928–3937.
[21] C. You, D. Robinson, and R. Vidal, "Scalable sparse subspace clustering by orthogonal matching pursuit," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2016, pp. 3918–3927.
[22] Y. Yang, J. Feng, N. Jojic, J. Yang, and T. S. Huang, "Subspace learning by l0-induced sparsity," *International Journal of Computer Vision*, vol. 126, no. 10, pp. 1138–1156, 2018.
[23] Y. Yang and P. Li, "Noisy l0-sparse subspace clustering on dimensionality reduced data," in *Uncertainty in Artificial Intelligence*. PMLR, 2022, pp. 2235–2245.
[24] J. Fan, "Large-scale subspace clustering via k-factorization," in *Proceedings of the 27th ACM SIGKDD Conference on Knowledge Discovery & Data Mining*, 2021, pp. 342–352.
[25] Y. Wang, W. Zhang, L. Wu, X. Lin, and X. Zhao, "Unsupervised metric fusion over multiview data by graph random walk-based cross-view diffusion," *IEEE transactions on neural networks and learning systems*, vol. 28, no. 1, pp. 57–70, 2015.
[26] Y. Wang, L. Wu, X. Lin, and J. Gao, "Multiview spectral clustering via structured low-rank matrix factorization," *IEEE Transactions on Neural Networks and Learning Systems*, vol. 29, no. 10, pp. 4833–4843, 2018.
[27] Y. Wang and L. Wu, "Beyond low-rank representations: Orthogonal clustering basis reconstruction with optimized graph structure for multi-view spectral clustering," *Neural Networks*, vol. 103, pp. 1–8, 2018.
[28] Z. Yu, Z. Zhang, W. Cao, C. Liu, J. P. Chen, and H. San Wong, "Gan-based enhanced deep subspace clustering networks," *IEEE Transactions on Knowledge and Data Engineering*, 2020.
[29] B. D. Haeffele, C. You, and R. Vidal, "A critique of self-expressive deep subspace clustering," in *International Conference on Learning Representations*, 2021.
[30] A. Salah and M. Nadif, "Model-based von mises-fisher co-clustering with a conscience," in *Proceedings of the 2017 SIAM International Conference on Data Mining*. SIAM, 2017, pp. 246–254.
[31] S. Affeldt, L. Labiod, and M. Nadif, "Regularized dual-ppmi co-clustering for text data," in *Proceedings of the 44th International ACM SIGIR Conference on Research and Development in Information Retrieval*, 2021, pp. 2263–2267.
[32] T. Guo, S. Pan, X. Zhu, and C. Zhang, "Cfond: consensus factorization for co-clustering networked data," *IEEE Transactions on Knowledge and Data Engineering*, vol. 31, no. 4, pp. 706–719, 2018.
[33] C. Mavromatis and G. Karypis, "Graph infoclust: Maximizing coarse-grain mutual information in graphs," in *PAKDD (1)*, 2021, pp. 541–553.
[34] H. Zhu and P. Koniusz, "Simple spectral graph convolution," in *9th International Conference on Learning Representations, ICLR, Virtual Event, Austria, May 3-7, 2021*. OpenReview.net, 2021.
[35] C. Fettal, L. Labiod, and M. Nadif, "Efficient graph convolution for joint node representation learning and clustering," in *Proceedings of the Fifteenth ACM International Conference on Web Search and Data Mining*, 2022, pp. 289–297.
[36] J. Shi and J. Malik, "Normalized cuts and image segmentation," *IEEE Transactions on pattern analysis and machine intelligence*, vol. 22, no. 8, pp. 888–905, 2000.
[37] F. Marcotorchino, "Block seriation problems: A unified approach. reply to the problem of h. garcia and jm proth (applied stochastic models and data analysis, 1,(1), 25–34 (1985))," *Applied Stochastic Models and Data Analysis*, vol. 3, no. 2, pp. 73–91, 1987.
[38] C. Laclau and M. Nadif, "Diagonal latent block model for binary data," *Statistics and Computing*, vol. 27, no. 5, pp. 1145–1163, 2017.
[39] H. Hu, Z. Lin, J. Feng, and J. Zhou, "Smooth representation clustering," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2014, pp. 3834–3841.
[40] C. Lu, J. Feng, Z. Lin, and S. Yan, "Correlation adaptive subspace segmentation by trace lasso," in *Proceedings of the IEEE international conference on computer vision*, 2013, pp. 1345–1352.
[41] P. Drineas, M. W. Mahoney, and N. Cristianini, "On the nyström method for approximating a gram matrix for improved kernel-based learning." *journal of machine learning research*, vol. 6, no. 12, 2005.
[42] N. Pham and R. Pagh, "Fast and scalable polynomial kernels via explicit feature maps," in *Proceedings of the 19th ACM SIGKDD international conference on Knowledge discovery and data mining*, 2013, pp. 239–247.
[43] A. Rahimi and B. Recht, "Random features for large-scale kernel machines," *Advances in neural information processing systems*, vol. 20, 2007.
[44] A. Y. Ng, M. I. Jordan, and Y. Weiss, "On spectral clustering: Analysis and an algorithm," in *Advances in neural information processing systems*, 2002, pp. 849–856.
[45] N. Halko, P.-G. Martinsson, and J. A. Tropp, "Finding structure with randomness: Probabilistic algorithms for constructing approximate matrix decompositions," *SIAM review*, vol. 53, no. 2, pp. 217–288, 2011.
[46] A. P. Dempster, N. M. Laird, and D. B. Rubin, "Maximum likelihood from incomplete data via the em algorithm," *Journal of the royal statistical society: series B (methodological)*, vol. 39, no. 1, pp. 1–22, 1977.
[47] Y. Kluger, R. Basri, J. T. Chang, and M. Gerstein, "Spectral biclustering of microarray data: coclustering genes and conditions," *Genome research*, vol. 13, no. 4, pp. 703–716, 2003.
[48] X. Wang, H. Ji, C. Shi, B. Wang, Y. Ye, P. Cui, and P. S. Yu, "Heterogeneous graph attention network," in *The world wide web conference*, 2019, pp. 2022–2032.
[49] C. Yang, Z. Liu, D. Zhao, M. Sun, and E. Y. Chang, "Network representation learning with rich text information," in *IJCAI*, 2015.
[50] O. Shchur, M. Mumme, A. Bojchevski, and S. Günnemann, "Pitfalls of graph neural network evaluation," *arXiv preprint arXiv:1811.05868*, 2018.
[51] L. Hubert and P. Arabie, "Comparing partitions," *Journal of classification*, vol. 2, no. 1, pp. 193–218, 1985.
[52] D. Cai, X. He, X. Wu, and J. Han, "Non-negative matrix factorization on manifold," in *2008 eighth IEEE international conference on data mining*. IEEE, 2008, pp. 63–72.
[53] P. B. Nemenyi, *Distribution-free multiple comparisons*. Princeton University, 1963.
[54] D. Mimno, H. Wallach, E. Talley, M. Leenders, and A. McCallum, "Optimizing semantic coherence in topic models," in *Proceedings of the 2011 conference on empirical methods in natural language processing*, 2011, pp. 262–272.

**Chakib Fettal** is an industrial Ph.D. Candidate in Centre Borelli at Paris Cité university and Informatique Caisse des Dépôts et Consignations. He works on developing efficient approaches for community detection and representation learning for graphs. Contact at : chakib.fettal@etu.u-paris.fr

**Lazhar Labiod** is an Associate Professor with the IUT de Paris - Rives de Seine and Centre Borelli at Paris Cité university. He received a Ph.D. in statistics from Pierre-et-Marie-Curie university. He has made contributions to data embedding and co-clustering working on bio-medical data, tensor data, attributed graphs, etc. Contact at : lazhar.labiod@u-paris.fr

**Mohamed Nadif** is a Professor at Paris Cité university and researcher at Centre Borelli. He is head of *Artificial Intelligence for Data Science Cybersecurity* team. He teaches data mining, machine learning techniques and multivariate analysis. His current researches interests include cluster analysis, co-clustering, data analysis, data mining, visualization, factorization and latent block models. Contact at : mohamed.nadif@u-paris.fr