F-CAG: Fuzzy Clustering for Attributed Graphs

Lazhar Labiod [1], **Mohamed Nadif** [1]

[1]Centre Borelli UMR 9010
Université Paris Cité
Paris, France
firstname.lastname@u-paris.fr

Abstract

Fuzzy clustering and perception are two fundamental concepts in the field of ML that complement each other to enhance the understanding and processing of complex data. In this paper, we focus on attributed networks and introduce a novel framework for clustering networked data using joint embedding and clustering. Our approach, based on entropy-based regularization in the fuzzy clustering criterion, employs low-rank subspaces and fuzzy clusters to better capture complex relationships between *content and structure* information, enhancing clustering robustness. Experiments on various benchmark datasets for document clustering, using both bag-of-words and large language model (LLM) representations, demonstrate that our algorithm surpasses state-of-the-art clustering methods, including task-specific deep learning approaches.

Introduction

Fuzzy clustering, also known as fuzzy grouping, is a data partitioning technique that allows a data point to belong to multiple clusters with varying degrees of membership. Unlike traditional clustering methods where each point is strictly assigned to a single cluster, fuzzy clustering better reflects the reality of complex and ambiguous data by drawing inspiration from human perception. For instance, fuzzy clustering allows a medical document to belong to multiple categories or clusters with varying degrees of membership, thereby reflecting the multifaceted nature of medical information. For instance, a document on 'cardiovascular diseases' and 'diabetes management' can be 60% in 'Cardiology' and 40% in 'Endocrinology' with fuzzy clustering, reflecting its dual relevance better than a single category. By integrating fuzzy clustering into ML systems, more robust and adaptable models can be created, capable of handling uncertainty and more accurately replicating human cognitive processes, thereby improving artificial perception of data.

In this paper we focus on *Attributed Networks* (AN) (Qi et al. 2012) which have been used to model real-world networks, including academic and health care networks. These networks offer node links and attributes for analysis, unlike plain networks with only node links. In AN, each node is linked to a set of features, leading to two matrices. The first is a square matrix $\mathbf{W}$ of size $n \times n$; $\mathbf{W}$ is constructed from a graph represented by an adjacency matrix $\mathbf{A}$. The second is $\mathbf{X}$ of size $n \times d$, the graph feature matrix, where each of the n nodes is described by d features, as shown in Figure 1.

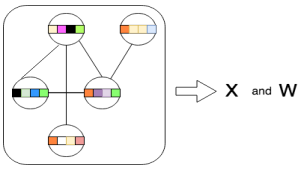

Figure 1: Information from AN can be split into $\mathbf{X}$ and $\mathbf{W}$.

Recently, representation learning has become important in fields like social and academic networks, and protein interactions. ANE (Cai, Zheng, and Chang 2018) seeks to create a compact node representation that preserves network topology and node proximity by attributes. While NE (Yu et al. 2019) has fostered several methods (Chang et al. 2015), ANE has been less explored. Unlike NE, ANE integrates nodes' proximity and attribute similarity, which distinguishes it from existing NE algorithms (Labiod and Nadif 2024).

Learned representations are beneficial for tasks like network clustering (Wang et al. 2017), node visualization (Dai et al. 2018), node classification (Huang, Li, and Hu 2017), and link prediction (Pan et al. 2018). Consequently, tackling high-dimensionality, sparsity, and nonlinearity is now a critical research focus. However, these challenges are particularly pronounced in network clustering techniques. Often, clustering techniques disappoint for two reasons: the continuous embedding solution usually diverges from precise discrete clustering, and information loss occurs between continuous embedding generation and discretization stages.

This paper introduces an objective function integrating embedding and fuzzy clustering, unlike the separate considerations of $\mathbf{X}$ and $\mathbf{W}$. Our algorithm, based on this function, employs low-rank subspace and fuzzy clustering to better capture complex relationships between $\mathbf{X}$ and $\mathbf{W}$, enhancing clustering robustness. The paper's key contributions are as follows:

1. First, we introduce a new ANE approach that integrates data embedding and clustering into a unified framework,

incorporating information from $\mathbf{X}$ and $\mathbf{W}$ via entropy-based regularization in the fuzzy clustering criterion.

2. Second, we introduce an optimization approach that efficiently manages large datasets by simultaneously performing embedding and fuzzy clustering. This significantly reduces computational load while maintaining high-quality clustering, making it ideal for real-world applications.

Related Work

Subspace clustering methods based on the self-expressive property are commonly used on image data and have set state-of-the-art results on the task of image clustering. One of the earlier approaches was least-squares regression subspace clustering (LSR), which leverages a grouping effect in the data. Newer models that make up the state-of-the-art include Elastic-Net Subspace Clustering (EnSC) (You et al. 2016) that uses $\ell 1$- and $\ell 2$-norm regularization, and the Subspace Clustering through Orthogonal Matching Pursuit (SSC-OMP) (You, Robinson, and Vidal 2016) which possesses a subspace-preserving affinity under broad conditions. More recently, a new efficiency trend has appeared, and some scalable models have also been proposed, e.g., k-Factorization Subspace Clustering (k-FSC) (Fan 2021) which was put forward as a scalable subspace clustering model that factorizes data into subsets via structured sparsity.

Regarding attributed-graph clustering, which refers to the process of grouping nodes into clusters according to the graph topology and node features, we can classify attributed graph clustering models into two subsets. A first one, where the goal is to learn graph representations and then use traditional clustering models such as k-means. Examples of models that use this approach include Simplified Graph Convolution (SGC) (Wu et al. 2019) which proposes a neighborhood averaging process that corresponds to a fixed low-pass filter, and Simple Spectral Graph Convolution (S²GC) which uses a new method for the aggregation of K-hop neighborhoods that is a trade-off of low- and high-pass filter bands (Zhu and Koniusz 2021). On the other hand, the second class of attributed graph clustering models proposes to include the clustering objective into the representation learning process to learn better results, e.g., Graph InfoClust (GIC) (Mavromatis and Karypis 2021) which generates clusters by maximizing mutual information between nodes contained in the same cluster, and Graph Convolutional Clustering (GCC) (Fettal, Labiod, and Nadif 2022) that performs clustering by minimizing the difference between convolved node representations and their reconstructed cluster representatives.

Notation. The dimensions of the matrices and the parameters are explicitly described below.

- Let n represent the number of nodes/objects, d the number of features, g the number of clusters, c the number of retained components, λ the regularization parameter, and p the power parameter.
- Let $\mathbf{X} \in \mathbb{R}^{n \times d}$ denote n data points described by d features and let $\mathbf{A} \in \{0,1\}^{n \times n}$ be an adjacency matrix on the n data points.
- Let $\mathbf{U} = (\hat{\mathbf{U}}_1 | \dots | \hat{\mathbf{U}}_k | \dots | \hat{\mathbf{U}}_g) \in \mathbb{R}^{n \times g}$. We note $\mathbf{U}_k$ a diagonal matrix of $\hat{\mathbf{U}}_k$;

$$\mathbf{U}_k = \text{Diag}(\hat{\mathbf{U}}_k) = \begin{bmatrix} u_{1k} & & \\ & \ddots & \\ & & u_{nk} \end{bmatrix}.$$

  Each u_{ik} denotes a membership value for node $(i = 1, \dots, n)$ in the k-th cluster $(k = 1, \dots, g)$, we have $\sum_{k=1}^{g} u_{ik} = 1$.
- Let $\mathbf{B} \in \mathbb{R}^{n \times c}$ be the embedding of samples and $\mathbf{Q} \in \mathbb{R}^{d \times c}$ be the embedding of attributes.
- Let $\mathbf{Z} = (\mathbf{Z}_1 | \dots | \mathbf{Z}_k | \dots | \mathbf{Z}_g)$ be a $c \times g$ matrix of the centroids of the clusters; each $\mathbf{Z}_k \in \mathbb{R}^{c \times 1}$ represents the k-th centroid. Let $\mathbb{1}$ be a real $n \times 1$ vector of ones, we note $\widetilde{\mathbf{Z}}_k = \mathbb{1}\mathbf{Z}_k^\top \in \mathbb{R}^{n \times c}$; the rows of $\widetilde{\mathbf{Z}}_k$ are the $\mathbf{Z}_k^\top$ duplicated n times.
- The symbol 'Tr' represents the Trace of a matrix, and we have $||K||_{\mathbf{H}}^2$ =Tr$(K^\top \mathbf{H} K)$.

Content and Structure information

Combining data in attributed graphs is crucial in the clustering process. The quality of the results, or more specifically, the ability to obtain relevant classes, fundamentally depends on the pre-processing step. In the following section, we will detail and justify the various steps involved in constructing new matrices $\mathbf{M}$ and $\mathbf{S}$ from $\mathbf{X}$, $\mathbf{A}$ and $\mathbf{W}$.

Construction of $\mathbf{M}$

An attributed network $\mathcal{G} = (\mathcal{V}, \mathcal{E}, \mathbf{X})$ consists of the set of nodes V, the set of links $\mathcal{E} \subseteq \mathcal{V} \times \mathcal{V}$, and $\mathbf{X} = [\mathbf{x}_1, \mathbf{x}_2, \dots, \mathbf{x}_n]$ where $n = |\mathcal{V}|$ and $\mathbf{x}_i \in \mathbb{R}^d$ is the feature/attribute vector of the node v_i. Formally, the graph can be represented by two types of information, namely content information $\mathbf{X} \in \mathbb{R}^{n \times d}$ and structure information $\mathbf{A} \in \mathbb{R}^{n \times n}$, where $\mathbf{A}$ is an adjacency matrix of $\mathcal{G}$ and $a_{ij} = 1$ if $e_{ij} \in \mathcal{E}$ otherwise 0; we consider that each node is a neighbor of itself, then we set $a_{ii} = 1$ for all nodes. We therefore model the proximity of the nodes using an $(n \times n)$ transition matrix $\mathbf{W}$ given by $\mathbf{W} = \mathbf{D}^{-1}\mathbf{A}$, where $\mathbf{D}$ is the degree matrix of $\mathbf{A}$ defined by $d_{ii} = \sum_{i'=1}^{n} a_{i'i}$. Since $\mathbf{W}$ is a stochastic matrix (transition matrix), $\forall p \in \mathbb{N}^*$, $\mathbf{W}^p$ is stochastic and the construction of $\mathbf{M}$ such as

$$\mathbf{M} = \mathbf{W}^p \mathbf{X} \tag{1}$$

can be viewed as an iterative process $\mathbf{W}^p\mathbf{X}$; when $p = 0$ we have $\mathbf{M} = \mathbf{X}$. This process will converge to the approximated data $\mathbf{W}^p\mathbf{X}$ where each row moves towards its prototype. In other words, this process converges to an equilibrium state. With g denoting the number of eigenvalues of $\mathbf{W}^p$ equal to 1, the matrix $\mathbf{W}^p\mathbf{X}$ is composed of $g << n$ quasi-similar rows where each row is represented by its prototype. Thus, with this type of multiplicative smoothing, the original data is transformed into a set of representative prototypes. These prototypes capture the main characteristics of the data, and the process converges to a stable state where each data point is well represented by one of these prototypes.

Construction of S

To utilize additional information about node similarity from $\mathbf{X}$, we first preprocess the above dataset $\mathbf{X}$ to produce input from a similarity graph $\mathbf{W_X}$ of size $(n \times n)$; then we construct a K-Nearest-Neighbor (KNN) graph. To this end, we use the heat kernel and L_2 distance, KNN neighborhood mode with a given K and we set the width of the neighborhood $\sigma = 1$. Note that any appropriate distance or dissimilarity measure can be used. Finally, we combine the proximity of the nodes from both content information $\mathbf{X}$ and structure information $\mathbf{W}$ in an $(n \times n)$ matrix $\mathbf{S}$. Thus, we propose perturbing the similarity $\mathbf{W}$ by adding the similarity from $\mathbf{W_X}$; we choose to define $\mathbf{S}$ by

$$\mathbf{S} = \mathbf{W} + \mathbf{W_X}. \tag{2}$$

By integrating $\mathbf{W_X}$, we overcome the problem of sparsity. Later, we will see the interest in using $\mathbf{W_X}$ in $\mathbf{S}$; note that $\mathbf{S} = \mathbf{S}^\top$.

How to combine the information from M and S?

From two types of smoothing, we have constructed the two matrices $\mathbf{M}$ and $\mathbf{S}$. We now propose to exploit these two types of information in a clustering objective. To do this, we propose to consider an objective function based on approximations of both $\mathbf{M}$ and $\mathbf{S}$ while sharing a common information $\mathbf{B}$ and integrating the clustering objective given by

$$\Phi(\mathbf{M}, \mathbf{B}\mathbf{Q}^\top) + \lambda \sum_{k=1}^{g} \Psi_k(\mathbf{S}, \widetilde{\mathbf{Z}}_k \mathbf{B}^\top)$$

where Φ denotes the deviation between $\mathbf{M}$ and $\mathbf{B}\mathbf{Q}^\top$ while Ψ_k denotes the deviation between $\mathbf{S}$ and $\widetilde{\mathbf{Z}}_k\mathbf{B}^\top$. In other words, we choose to regularize the approximation of $\mathbf{M}$ by a term seeking to approximate $\mathbf{S}$ while taking into account the structure into classes; λ is regularized parameter. Note that if $\lambda = 0$, it is possible to consider only $\mathbf{M}$. In the following, we use the Frobenius norm as a deviation measure for the functions Φ and Ψ_k.

Model and algorithm

Fuzzy clustering, or soft clustering, offers a more flexible and nuanced approach to grouping data compared to traditional hard clustering. It handles uncertainty and ambiguity by allowing data points to belong to multiple clusters with varying degrees of membership, making it robust to noise and outliers. This flexibility enhances interpretability and is particularly useful in fields like pattern recognition, image processing, and bioinformatics, where cluster boundaries are often not clear-cut. Algorithms such as Fuzzy C-means (FCM) are widely used for fuzzy clustering, providing an efficient and optimizable method for analyzing complex datasets. For all these reasons, we adopt this approach in the context of attributed graphs.

Objective function

With $\mathbf{M}$ and $\mathbf{S}$, the F−CAG method aims to obtain the maximally informative embedding with respect to the clustering structure in the attributed network data. On the other hand, in (Gao et al. 2019) for instance, the authors have shown that exploiting the maximum entropy principle, as applied to fuzzy clustering, provides a new perspective on facing the problem of fuzzifying the clustering of the objects, whilst ensuring the maximum compactness of the obtained clusters. Thereby we propose the following objective function to be minimized

$$\mathcal{F}(\mathbf{B}, \mathbf{Z}, \mathbf{Q}, \mathbf{U}) = \|\mathbf{M} - \mathbf{B}\mathbf{Q}^\top\|^2 + \lambda\Big(\sum_{k=1}^{g} \Big\|\mathbf{S} - \widetilde{\mathbf{Z}}_k\mathbf{B}^\top\Big\|^2_{\mathbf{U}_k} + \sum_{k=1}^{g}\sum_{i=1}^{n} u_{ik}\log u_{ik}\Big) \tag{3}$$

with subject to $\mathbf{B}^\top\mathbf{B} = \mathbf{I}_c$ where $\mathbf{I}_c$ is a c by c identity matrix. The first term represents a smoothed PCA criterion, the second and third term represent a fuzzy clustering model with an entropy regularization which makes the optimization problem more numerically tractable and λ denotes a regularized parameter. Our proposed method is solved by minimizing these two terms simultaneously. The intuition behind the factorization of $\mathbf{M}$ and $\mathbf{S}$ is to encourage the nodes with similar proximity, those with higher similarity in both matrices, to have closer representations in the latent space given by $\mathbf{B}$. In doing so, the optimisation of (3) leads to a fuzzy clustering of the nodes into g clusters given by $\mathbf{U}$. Finally $\mathbf{Q}$ can be viewed as an embedding matrix of features.

Note that, both tasks –embedding and clustering– are performed simultaneously and supported by $\mathbf{B}$; it is the key to attaining good embedding while taking into account the clustering structure. Before tackling the estimation of the unknown matrices $\mathbf{B}$, $\mathbf{Z}$, $\mathbf{U}$ and $\mathbf{Q}$, we will first detail the second term of (3).

Proposition 1. *Given* $\mathbf{S} \in \mathbb{R}^{n\times n}$, $\mathbf{U}_k \in [0,1]^{n\times n}$, $\widetilde{\mathbf{Z}}_k \in \mathbb{R}^{n\times c}$, $\mathbf{B} \in \mathbb{R}^{n\times c}$ *and* $\mathbf{B}^\top\mathbf{B} = \mathbf{I}$ *then we have for each* k

$$\|\mathbf{S} - \widetilde{\mathbf{Z}}_k\mathbf{B}^\top\|^2_{\mathbf{U}_k} = \|\mathbf{S} - \mathbf{S}\mathbf{B}\mathbf{B}^\top\|^2_{\mathbf{U}_k} + \|\mathbf{S}\mathbf{B} - \widetilde{\mathbf{Z}}_k\|^2_{\mathbf{U}_k} \tag{4}$$

Proof. First, we have

$$\begin{aligned}\|A\|^2_{\mathbf{U}_k} &= Tr(A^\top \mathbf{U_k} A)\\ &= Tr(A^\top \mathbf{U}_k^{0.5}\mathbf{U}_k^{0.5} A)\\ &= Tr((\mathbf{U}_k^{0.5}A)^\top(\mathbf{U}_k^{0.5}A))\\ &= \|\mathbf{U}_k^{0.5}A\|.\end{aligned}$$

Using transition to the Frobenius norm and for convenience taking $\mathbf{\Sigma}_k = \mathbf{U}_k^{0.5}$, the equality (4) to prove becomes

$$\Big\|\mathbf{\Sigma}_k\mathbf{S} - \mathbf{\Sigma}_k\widetilde{\mathbf{Z}}_k\mathbf{B}^\top\Big\|^2 = \Big\|\mathbf{\Sigma}_k\mathbf{S} - \mathbf{\Sigma}_k\mathbf{S}\mathbf{B}\mathbf{B}^\top\Big\|^2 + \Big\|\mathbf{\Sigma}_k\mathbf{S}\mathbf{B} - \mathbf{\Sigma}_k\widetilde{\mathbf{Z}}_k\Big\|^2 \tag{5}$$

First, since $\mathbf{B}^\top\mathbf{B} = \mathbf{I}$ and $\mathrm{Tr}(AB) = \mathrm{Tr}(BA)$ we have

$$\begin{aligned}\|\mathbf{\Sigma}_k\widetilde{\mathbf{Z}}_k\mathbf{B}^\top\|^2 &= \mathrm{Tr}(\mathbf{B}\widetilde{\mathbf{Z}}_k^\top\mathbf{\Sigma}_k^\top\mathbf{\Sigma}_k\widetilde{\mathbf{Z}}_k\mathbf{B}^\top)\\ &= \mathrm{Tr}(\widetilde{\mathbf{Z}}_k^\top\mathbf{\Sigma}_k^\top\mathbf{\Sigma}_k\widetilde{\mathbf{Z}}_k)\\ &= \|\widetilde{\mathbf{Z}}_k^\top\mathbf{\Sigma}_k\|^2.\end{aligned}$$

Similarly we have $\|\mathbf{\Sigma}_k\mathbf{S}\mathbf{B}\mathbf{B}^\top\|^2 = \|\mathbf{\Sigma}_k\mathbf{S}\mathbf{B}\|^2$; this leads to

$$(a)\ \Big\|\mathbf{\Sigma}_k\mathbf{S} - \mathbf{\Sigma}_k\widetilde{\mathbf{Z}}_k\mathbf{B}^\top\Big\| = \|\mathbf{\Sigma}_k\mathbf{S}\|^2 + \|\widetilde{\mathbf{Z}}_k^\top\mathbf{\Sigma}_k\|^2 - 2\mathrm{Tr}(\mathbf{\Sigma}_k\mathbf{S}\mathbf{\Sigma}_k\widetilde{\mathbf{Z}}_k\mathbf{B}^\top).$$

$$(b)\|\boldsymbol{\Sigma}_k\mathbf{S}-\boldsymbol{\Sigma}_k\mathbf{S}\mathbf{B}\mathbf{B}^\top\|^2 = \|\boldsymbol{\Sigma}_k\mathbf{S}\|^2+\|\boldsymbol{\Sigma}_k\mathbf{S}\mathbf{B}\mathbf{B}^\top\|^2 - 2\mathrm{Tr}(\boldsymbol{\Sigma}_k\mathbf{S}\mathbf{B}\mathbf{B}^\top\boldsymbol{\Sigma}_k^\top\mathbf{S}^\top) = ||\boldsymbol{\Sigma}_k\mathbf{S}||^2+||\boldsymbol{\Sigma}_k\mathbf{S}\mathbf{B}||^2-2||\boldsymbol{\Sigma}_k\mathbf{S}\mathbf{B}||^2 = ||\boldsymbol{\Sigma}_k\mathbf{S}||^2-||\boldsymbol{\Sigma}_k\mathbf{S}\mathbf{B}||^2.$$

$$(c)\|\boldsymbol{\Sigma}_k\mathbf{S}\mathbf{B}-\boldsymbol{\Sigma}_k\widetilde{\mathbf{Z}}_k\|^2 = \|\boldsymbol{\Sigma}_k\mathbf{S}\mathbf{B}\|^2+\|\widetilde{\mathbf{Z}}_k^\top\boldsymbol{\Sigma}_k\|^2 - 2\mathrm{Tr}(\boldsymbol{\Sigma}_k\mathbf{S}\boldsymbol{\Sigma}_k\widetilde{\mathbf{Z}}_k\mathbf{B}^\top).$$

Summing (b) and (c) (terms on the right of (5)) for a given k) leads to (a); Q.E.D. □

Finally summing over k all the terms of eq.(5) leads to eq.(4). Hence the second term of (3) (without λ) takes the following form

$$\sum_k \left\|\mathbf{S}-\mathbf{S}\mathbf{B}\mathbf{B}^\top\right\|^2_{\mathbf{U}_k}+\sum_k\left\|\mathbf{S}\mathbf{B}-\widetilde{\mathbf{Z}}_k\right\|^2_{\mathbf{U}_k}.$$

Given that $\mathbf{SB}=((sb)_{ij})$ is of reduced size $n\times c$, note that since $\forall i$, $\sum_{k=1}^g u_{ik}=1$ the first term does not depend on $\mathbf{U}_k$ and the second term takes the following form

$$\sum_{i,k}u_{ik}\sum_{j=1}^d\left((sb)_{ij}-\widetilde{z}_{kj}\right)^2.$$

Thus, we can identify the criterion for regularized fuzzy clustering in the objective function (3)

$$\sum_{i,k}u_{ik}\sum_{j=1}^d\left((sb)_{ij}-\widetilde{z}_{kj}\right)^2+\sum_{i,k}u_{ik}\log(u_{ik}).$$

Optimization and algorithm

To infer the latent factor matrices $\mathbf{Z}$, $\mathbf{B}$, $\mathbf{Q}$ and $\mathbf{U}$ from $\mathbf{M}=\mathbf{W}^p\mathbf{X}$ and $\mathbf{S}=\mathbf{W}+\mathbf{W}_\mathbf{X}$, we derive an alternating optimization algorithm. In the following, we detail the different steps involved in inferring $\mathbf{B}$,$\mathbf{Q}$, $\mathbf{Z}$, and $\mathbf{U}_k$.

Compute $\mathbf{B}$. Update $\mathbf{B}$ for fixed $\mathbf{Q}$,$\mathbf{Z}_k$ and $\mathbf{U}_k$. This is equivalent to maximizing $tr(\mathbf{M}\mathbf{Q}+\lambda\mathbf{S}\sum_{k=1}^g\mathbf{U}_k\widetilde{\mathbf{Z}}_k)$ as follows: Let

$$\mathtt{SVD}(\mathbf{M}\mathbf{Q}+\lambda\mathbf{S}\sum_{k=1}^g\mathbf{U}_k\widetilde{\mathbf{Z}}_k)=\mathbf{A}\mathbf{D}\mathbf{V}^\top.$$

Then

$$\mathbf{B}^*=\mathbf{A}\mathbf{V}^\top. \tag{6}$$

Compute $\mathbf{Q}$**.** Given $\mathbf{U}$, $\mathbf{Z}$ and $\mathbf{B}$, (3) is reduced to $\min_\mathbf{Q}\left\|\mathbf{M}-\mathbf{B}\mathbf{Q}^\top\right\|^2$, and we get

$$\mathbf{Q}=\mathbf{M}^\top\mathbf{B}. \tag{7}$$

It is therefore possible to consider $\mathbf{Q}$ as an embedding of attributes.

Compute $\mathbf{Z}$. $\mathbf{Z}_k$ is updated for fixed $\mathbf{B}$, $\mathbf{Q}$ and $\mathbf{U}_k$. Let $d_{ik}=\|(\mathbf{SB})_i-\mathbf{Z}_k\|^2$, this is equivalent to minimizing $\sum_{k=1}^g\sum_{i=1}^n u_{ik}d_{ik}$ with respect to $\mathbf{Z}_k$. This yields to

$$\mathbf{Z}_k^\top=(\mathbb{1}^\top\mathbf{U}_k\mathbb{1})^{-1}\mathbb{1}^\top\mathbf{U}_k\mathbf{SB} \tag{8}$$

Compute $\mathbf{U}$. u_{ik} is updated for fixed $\mathbf{B}$, $\mathbf{Q}$ and $\mathbf{Z}_k$. Let $d_{ik}=\|(\mathbf{SB})_i-\mathbf{Z}_k\|^2$, then, u_{ik} is updated by

$$u_{ik}=\frac{\exp(-\frac{d_{ik}}{\lambda})}{\sum_{\ell=1}^g\exp(-\frac{d_{i\ell}}{\lambda})}. \tag{9}$$

The update of u_{ik} is derived as follows. Minimizing (3) under the membership constraint $\sum_{k=1}^g u_{ik}=1$ is equivalent to minimizing

$$\sum_{k=1}^g\sum_{i=1}^n u_{ik}d_{ik}+\lambda\sum_{k=1}^g\sum_{i=1}^n u_{ik}\log u_{ik}-\alpha(\sum_{k=1}^g u_{ik}-1) \tag{10}$$

Note that at the convergence of the algorithm that we describe below (Algorithm 1), we can deduce hard clustering from $\mathbf{U}$ by applying the maximum a posteriori principle. It is important to emphasize that at each step $\mathbf{B}$ makes use of the information from the matrices $\mathbf{Q}$, $\mathbf{U}$, and $\mathbf{Z}$. This highlights one of the aspects of simultaneous embedding and clustering.

Algorithm 1: : F−CAG$_\mathbf{S}$ algorithm

Input: $\mathbf{M}$ and $\mathbf{S}$ from structure matrix $\mathbf{W}$ and content matrix $\mathbf{X}$, g, c, p and λ ;
Initialize: $\mathbf{B}$, $\mathbf{Q}$ and $\mathbf{Z}$ with arbitrary matrix;
repeat
(a) - Compute $\mathbf{U}$ using (9)
(b) - Compute $\mathbf{B}$ using (6)
(c) - Compute $\mathbf{Q}$ using (7)
(d) - Compute $\mathbf{Z}_k$ using (8)
until convergence
Output: $\mathbf{U}$: cluster matrix, $\mathbf{Z}$: representation matrix, $\mathbf{B}$: node embedding matrix and $\mathbf{Q}$: attribute embedding matrix.

Relationships between F−CAG and other methods

We will now look at how our proposed F-CAG approach is related to some other clustering and data embedding methods.

Smoothed PCA The new data representation referred to as $\mathbf{M}=\mathbf{WX}$ of size $(n\times d)$ can be viewed as a multiplicative way of encoding information from both $\mathbf{W}$ and $\mathbf{X}$. Then the objective of F−CAG, defines a Graph-regularized PCA on the smoothed matrix $\mathbf{M}$. The first term in (3) thus performs a PCA, on the centroid computed on the neighborhood (barycenter) of each node. In fact, from $\min_{\mathbf{B},\mathbf{Q}}\left\|\mathbf{M}-\mathbf{B}\mathbf{Q}^\top\right\|^2$ plugging the optimal solution $\mathbf{Q}=\mathbf{M}^\top\mathbf{B}$ leads to the following equivalent trace maximization problem of a smoothed PCA

$$\max_\mathbf{B}\mathrm{Tr}(\mathbf{B}^\top(\mathbf{M}\mathbf{M}^\top)\mathbf{B}). \tag{11}$$

Smoothed PCA with Fuzzy graph clustering regularization From (3) and (11), the objective function of F−CAG with respect to $\mathbf{B}$ has this form

$$\max_{\mathbf{B}} \mathrm{Tr}(\mathbf{B}^\top(\mathbf{M}\mathbf{M}^\top)\mathbf{B} + \lambda(\mathbf{B}^\top\mathbf{S}\sum_{k=1}^{g}\mathbf{U}_k\widetilde{\mathbf{Z}}_k)). \quad (12)$$

This shows that the first term is related to a Smoothed PCA and the second term represents a *fuzzy graph clustering* regularization that enriches the Smoothed PCA by plugging the graph structure via $\mathbf{S}$ and its clustering structure via $\sum_{k=1}^{g}\mathbf{U}_k\widetilde{\mathbf{Z}}_k$. The role of the representation matrix $\mathbf{Z}$ is to closely maps out the memberships clustering matrix $\mathbf{U}$ to the embedding matrix $\mathbf{B}$ and vice versa.

Convolutional Networks Graph Convolutional Networks (GCNs) have experienced significant attention and have become the popular methods for learning graph representations. Below, we establish the connection between the graph convolution operator of GCN and the closed-form embedding solution of the F−CAG formulation. We demonstrate that the F−CAG embedding is SVD of the GCN embedding and then can can achieve better or similar results to GCN over several benchmark datasets.

Similar to other neural networks stacked with repeated layers, GCN contains multiple graph convolution layers; each of which is followed by a nonlinear activation (Wu et al. 2019; Chen et al. 2020). Let $\mathbf{H}^{(\ell)}$ be the ℓ-th layer hidden representation, then GCN follows:

$$\mathbf{H}^{(\ell+1)} = \sigma(\mathbf{W}\mathbf{H}^{(\ell)}\mathbf{Q}^{(\ell)}) \quad (13)$$

where $\mathbf{Q}^{(\ell)}$ is the ℓ-th layer parameter (to be learned), $\mathbf{H}^{(0)} = \mathbf{X}$ and σ is the nonlinear activation function. Graph convolution operation is defined as the formulation before activation in the above formulation of $\mathbf{H}^{(\ell+1)}$. The graph convolution (parameterized with $\mathbf{Q}$) mapping the feature matrix $\mathbf{X}$ to a new representation $\mathbf{Y}$ defined as $\mathbf{Y} = \mathbf{W}\mathbf{X}\mathbf{Q}$. The embedding solution of F−CAG is given by the closed form solution of the following problem

$$\max_{\mathbf{B}} \ \mathrm{Tr}\Big((\mathbf{M}\mathbf{Q} + \lambda\mathbf{S}\sum_{k=1}^{g}\mathbf{U}_k\widetilde{\mathbf{Z}}_k)\mathbf{B}^\top\Big) \quad \text{s.t.} \quad \mathbf{B}^\top\mathbf{B} = \mathbf{I}.$$

Let $\hat{\mathbf{U}}\hat{\Sigma}\hat{\mathbf{V}}^\top$ be SVD of

$$\begin{aligned}(\mathbf{M}\mathbf{Q} + \lambda\mathbf{S}\sum_{k=1}^{g}\mathbf{U}_k\widetilde{\mathbf{Z}}_k)) &= (\mathbf{W}\mathbf{X}\mathbf{Q} + \lambda\mathbf{S}\sum_{k=1}^{g}\mathbf{U}_k\widetilde{\mathbf{Z}}_k)) \\ &= (\mathbf{Y} + \lambda\mathbf{S}\sum_{k=1}^{g}\mathbf{U}_k\widetilde{\mathbf{Z}}_k)).\end{aligned}$$

Then, by setting $\lambda = 0$ $\mathbf{B} = \hat{\mathbf{U}}\hat{\mathbf{V}}^\top$ is derived from the singular value decomposition (SVD) of $\mathbf{Y}$.

Numerical experiments

ANE Clustering is evaluated on datasets with predefined clusters. Experiments involved datasets: ACM, DBLP, and Wiki, with sparse bag-of-words vectors and citation links; nodes are documents with class labels, edges are citation links. These datasets are frequently utilized for the evaluation of ANE clustering methods.The balance coefficient is defined as the ratio of the number of documents in the smallest class to the number of documents in the largest class. while *nz* denotes the percentage of sparsity.

Table 1: The dataset statistics include the imbalance ratio between the majority and minority classes, and *nz%* denotes the percentage of sparsity.

Dataset	Nodes	Edges	Features	Classes	nz%	Imbalance
ACM	3025	16,153	1870	3	96.52	1.1
DBLP	4057	2,502,276	334	4	96.4	1.6
Wiki	2405	14,001	4973	17	86.46	45.1

Parameter settings and compared algorithms We compare F−CAG with embedding-based methods and with other methods that are explicitly for graph clustering. In our comparison, we include standard methods and also recent deep learning methods; these differ in the way they use available information. Some of them (such as K−means) use only $\mathbf{X}$ as the baseline, while others use more recent algorithms based on $\mathbf{X}$ and $\mathbf{W}$. In Table 2 the following algorithms are the baselines, described in (Fettal, Labiod, and Nadif 2023), we used in our experiments:

- **K-Means** will be used as the simplest baseline.
- **LSR** is a subspace clustering model that incorporates the $\ell 2$-norm regularization.
- **EnSC** is a subspace clustering model that incorporates elastic net regularization (mix of $\ell 1$ and $\ell 2$ norm regularization).
- **SSC-OMP** has a subspace-preserving affinity under broad conditions.
- **k-FSC** is a scalable subspace clustering model that factorizes model in subsets via structured sparsity.
- **SC** refers to the classical spectral clustering algorithm applied on the original adjacency matrix of the graph.
- **SGC** proposes a neighborhood averaging process that corresponds to a fixed low-pass filter.
- **GIC** generates clusters by maximizing mutual information between nodes contained in the same cluster.
- **S²GC** proposes a new method for the aggregation of K-hop neighborhoods that is a trade-off of low- and high-pass filter bands.
- **GCC** performs clustering by minimizing the difference between convolved node representations and their reconstructed cluster representatives.

Clustering To evaluate F−CAG$_\mathbf{S}$ against other methods, we need to derive a hard clustering from the convergence of F−CAG by applying the principle of maximum a posteriori. Thus, beyond accuracy, we retain Normalized Mutual Information (NMI) (Strehl and Ghosh 2002) and Adjusted Rand Index (ARI) (Steinley 2004); they are more effective especially when clusters are imbalanced or numerous. ARI measures similarity between data groups and relates to precision. NMI indicates how well estimated clustering reflects

true clustering, while ARI measures the agreement with the reference partition. Both NMI and ARI reach 1 for perfect clustering. Higher ACC/NMI/ARI values indicate better performance, which is evaluated against true clusters. A clustering algorithm performing well on these measures is likely the best for the evaluated scenario.

Results and discussion Regarding the datasets ACM, DBLP, and Wiki, we present in Table 2 the results of the various clustering algorithms used in the literature for their evaluation (Li et al. 2021). Compared to the true available clusters, in our experiments the clustering performance is evaluated by ACC, NMI and ARI. To obtain the values of these metrics, we repeat the experiments 50 times and the averages (mean) are reported in Table 2; the best performance for each dataset is highlighted in bold and the second is underlined. In our experiments we took $g = c$ and and tested a set of (p, λ) values that will be discussed in Algorithm 2. First, we observe the high performance of methods that integrate information from $\mathbf{W}$. On the other hand, all methods, including deep learning algorithms based on $\mathbf{X}$ and $\mathbf{W}$ are even better. However, regarding F−CAG with both versions relying on $\mathbf{W}$, referred to as F−CAG$_{\mathbf{W}}$ or $\mathbf{S}$ referred to as F−CAG$_{\mathbf{S}}$, we note high performances for all datasets, and with F−CAG$_{\mathbf{S}}$, we note the impact of $\mathbf{W}_{\mathbf{X}}$; it learns low-dimensional representations while suiting the clustering structure. To go further in our investigation and given the sparsity of $\mathbf{X}$ we proceeded to standardization tf-idf followed by ℓ_2, as it is often used to process document-term matrices; see e.g., (Affeldt, Labiod, and Nadif 2021, 2020; Salah and Nadif 2017; Salah, Ailem, and Nadif 2018; Salah and Nadif 2019; Febrissy et al. 2022), while in the construction of $\mathbf{W}_{\mathbf{X}}$ we used the cosine metric. From the results in Table 2, we see the benefit of F−CAG$_{\mathbf{S}}$.

Assessing of p and λ To evaluate clustering quality, internal validity criteria are often used to rank solutions, known as relative validity criteria. We propose using the Silhouette Width Criterion (SWC) (Rousseeuw 1987) to estimate hyperparameters (p, λ). SWC measures how well an object fits its own cluster (cohesion) versus others (separation). It ranges from -1 to 1, with higher values indicating better fit within its cluster. For different (p, λ) values, we run F−CAG and select the pair maximizing SWC (Algorithm 2).

Algorithm 2: : Estimation of p and λ

Input: $\mathbf{M}$ and $\mathbf{S}$ from structure matrix $\mathbf{W}$ and content matrix $\mathbf{X}$, g
for $p = 1$ to 20 **do**
  for $\lambda \in \{0, 10^{-6}, 10^{-3}, 10^{-1}, 10^{0}, 10^{1}, 10^{3}\}$ **do**
    (a) - Run F−CAG
    (b) - Compute SWC
  end for
end for
$(p^*, \lambda^*) = \max_{(p,\lambda)} SWC$
Output: (p^*, λ^*)

Document clustering without $\mathbf{W}$ In the domain of *natural language processing* (NLP), unsupervised learning is prevalent. When dealing with unlabeled datasets, techniques like clustering and visualization can enhance the usefulness of textual data. The primary obstacle in document clustering is often how to represent the documents, with common methods including Bag-Of-Words (BOW), like the implementation found in $\mathbf{X}$ for ACM, DBLP, and Wiki. Nonetheless, Large Language Models (LLMs) and perception are two crucial elements in the field of ML that, together, are revolutionizing the way machines understand and interact with human language. LLMs, with their ability to process and generate text contextually, mimic human perception by interpreting the nuances and ambiguities of natural language. This synergy enables the creation of systems capable of grasping complex concepts, adapting their responses based on context, and continuously improving their understanding through learning. In the following we retain the MiniLM model (Wang et al. 2020) which is a pre-trained language model that uses knowledge distillation to compress larger models, such as BERT, into smaller, more efficient versions. Through the technique of "Deep Self-Attention Distillation," MiniLM transfers knowledge from a "teacher" model to a "student" model, creating lighter models that retain much of the original models' performance. These models are designed to be faster and more computationally efficient, making them suitable for real-time applications and resource-constrained environments.

To evaluate F−CAG$_{\mathbf{S}}$ we consider two coprus Classic4 and BBC. These collections are often used as benchmarks for evaluating text classification and clustering techniques. BBC News [1] is a dataset sourced from the BBC News, encompasses a collection of 2225 articles labeled across five categories: business, entertainment, politics, sport, and tech. The Classic4 dataset is a well-known collection used in the field of text mining. It is a collection of 7095 articles and consists of four distinct document sets: CACM, CISI, CRAN, and MED. According the MiniLM model we obtain both matrices Classic4 (7095×384) and BBC (2225×384). Without $\mathbf{W}$, we construct a K-Nearest-Neighbor graph (KNN with $K = 15$) to create the similarity graph $\mathbf{W}_{\mathbf{X}}$ of size $(n \times n)$, simplifying the objective function to (3) with $\mathbf{M} = \mathbf{W}_{\mathbf{X}}^{p}\mathbf{X}$ and $\mathbf{S} = \mathbf{W}_{\mathbf{X}}$. Table 3 highlights the benefits of introducing $\mathbf{W}_{\mathbf{X}}$, even in the absence of $\mathbf{W}$.

The UMAP algorithm achieves dimensionality reduction through manifold learning techniques and concepts from topological data analysis. Similar to the construction of $\mathbf{W}_{\mathbf{X}}$, the number of neighbors selected with UMAP is 15. Figure 2 displays the visualizations obtained for $\mathbf{X}$, $\mathbf{M}$, and $\mathbf{B}$. This can be assessed using the Trustworthiness score (Kaski et al. 2003), a metric that evaluates the quality of dimensionality reduction techniques. It measures how well the local structure of the data is preserved when transitioning from a high-dimensional to a lower-dimensional space. Additionally, to evaluate the compactness of clusters, we employ the silhouette metric. Table 4 showcases the effectiveness of F−CAG$_{\mathbf{S}}$, particularly in terms of SWC.

[1] http://mlg.ucd.ie/datasets/bbc.html

Table 2: Clustering performance of the different models over ACM, DBLP and Wiki. The best results are highlighted in bold font and the second best results are underlined.

Method	**Input**	**ACM**			**DBLP**			**Wiki**		
		ACC	NMI	ARI	ACC	NMI	ARI	ACC	NMI	ARI
K-means	$\mathbf{X}$	87.8 ±0.9	61.7 ±1.5	67.4 ±2.1	67.9 ±0.0	37.3 ±0.0	31.5 ±0.1	47.6 ±1.4	48.6 ±0.2	26.6 ±0.2
LSR	$\mathbf{X}$	78.6 ±0.0	43.1 ±0.0	48.3 ±0.0	69.4 ±0.1	34.7 ±0.1	36.4 ±0.2	17.8 ±0.5	2.8 ±1.7	0.3 ±0.2
EnSC	$\mathbf{X}$	83.8 ±0.0	53.0 ±0.0	58.6 ±0.0	30.0 ±0.1	0.8 ±0.2	0.1 ±0.0	47.5 ±0.0	45.2 ±0.2	30.2 ±0.1
SSC-OMP	$\mathbf{X}$	82.1 ±0.0	49.4 ±0.1	55.3 ±0.0	29.4 ±0.1	0.4 ±0.1	-0.1 ±0.0	37.8 ±8.5	34.4 ±9.1	21.2 ±7.9
k-FSC	$\mathbf{X}$	59.7 ±7.2	25.2 ±7.1	27.2 ±7.2	51.3 ±11.1	17.4 ±7.3	17.3 ±9.6	38.2 ±5.1	35.6 ±3.9	17.7 ±4.4
SC	$\mathbf{W}$	36.5 ±0.2	1.0 ±0.2	0.7 ±0.1	91.0 ±0.0	73.0 ±0.1	78.3 ±0.1	30.7 ±1.1	24.0 ±0.8	6.0 ±0.2
SGC	$\mathbf{W}, \mathbf{X}$	83.7 ±0.0	55.7 ±0.0	58.8 ±0.0	88.8 ±0.0	69.5 ±0.0	73.2 ±0.0	51.9 ±0.8	49.6 ±0.2	28.6 ±0.1
GIC	$\mathbf{W}, \mathbf{X}$	90.1 ±0.3	68.2 ±0.6	73.2 ±0.6	90.2 ±0.2	72.4 ±0.4	77.4 ±0.3	48.0 ±0.7	48.4 ±0.3	31.0 ±0.3
S²GC	$\mathbf{W}, \mathbf{X}$	84.1 ±0.1	56.8 ±0.1	59.6 ±0.2	88.3 ±0.0	69.2 ±0.0	71.9 ±0.0	52.1 ±1.0	52.2 ±0.1	33.0 ±0.4
GCC	$\mathbf{W}, \mathbf{X}$	91.3 ±0.0	**71.2 ±0.1**	76.0 ±0.1	91.8 ±0.0	74.5 ±0.0	80.5 ±0.0	53.7 ±1.4	**53.5 ±0.5**	31.6 ±1.1
F-CAG$_{\mathbf{W}}$	$\mathbf{W}, \mathbf{X}$	90.4 ±0.0	68.4 ±0.01	73.9 ±0.01	93.7 ±0.0	78.9 ±00.2	**84.7 ±0.02**	53.0 ±0.01	49.4 ±0.02	36.0 ±0.01
F-CAG$_{\mathbf{S}}$	$\mathbf{S}, \mathbf{X}$	**91.5 ±0.1**	71.04 ±0.02	**76.7 ±0.1**	**93.76 ±0.1**	**79.1 ±0.02**	84.6 ±0.02	**56.7 ±0.01**	52.4 ±0.02	**39.0 ±0.01**

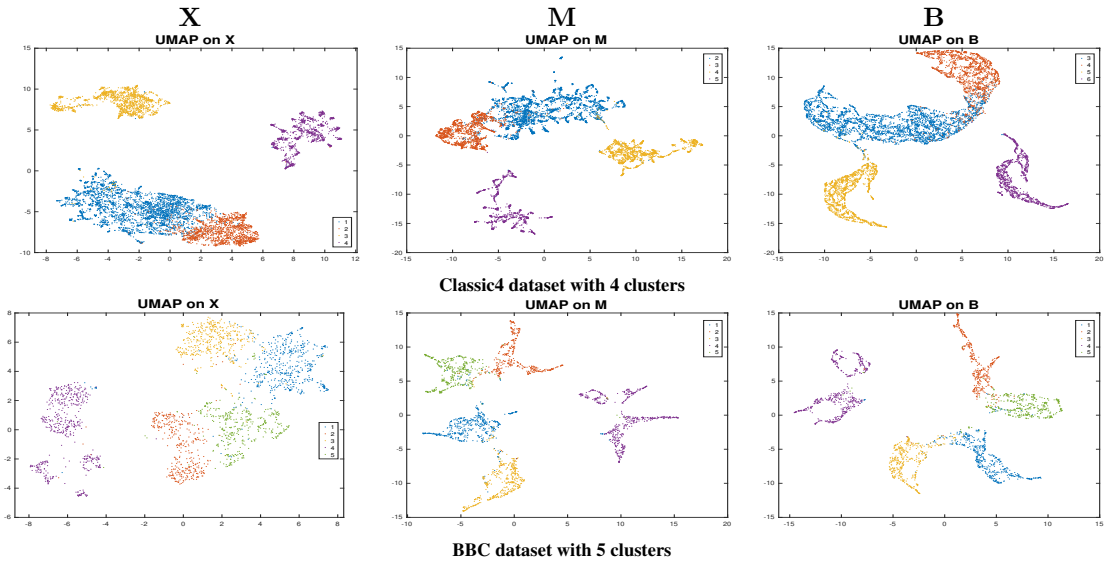

Figure 2: From top to bottom and from left to right, clusters projection using UMAP applied on $\mathbf{X}$, $\mathbf{M} = \mathbf{W}_{\mathbf{X}}^{p}\mathbf{X}$ and $\mathbf{B}$.

Table 3: Clustering performances of F−CAG in terms of ACC % , NMI % and ARI % on Classic4 and BBC datasets.

Input	**Classic4** $(p^*, \lambda^*) = (6, 10^3)$			**BBC** $(p^*, \lambda^*) = (7, 10^1)$		
	ACC	NMI	ARI	ACC	NMI	ARI
$\mathbf{X}$	79.18	68.93	46.07	92.58	80.19	82.90
$\mathbf{X}, \mathbf{W}_{\mathbf{X}}$	**93.77**	**83.53**	**82.20**	**94.97**	**85.07**	**88.14**

Table 4: Separability and compacteness of F−CAG in terms of Trustworthiness ($\mathbf{T}$) and SWC on Classic4 and BBC.

Input	**Classic4**			**BBC**		
	$\mathbf{X}$	$\mathbf{M}$	$\mathbf{B}$	$\mathbf{X}$	$\mathbf{M}$	$\mathbf{B}$
SWC	0.077	0.213	**0.600**	0.067	0.241	**0.577**
$\mathbf{T}$	0.988	0.995	**0.999**	0.982	0.993	**0.997**

Conclusion and perspectives

We introduced a novel framework for clustering networked data using joint embedding and clustering. Our approach improves embedding and clustering performance while significantly speeding up the process by clustering on a reduced matrix. Experiments on real-world benchmark networks show our algorithm F-CAG$_\mathbf{S}$ outperforms state-of-the-art methods. Additionally, if $\mathbf{W}$ is unavailable, using a KNN approach to construct $\mathbf{W_X}$ effectively enhances clustering. Thereby, the integration of fuzzy clustering and perception in machine learning, enhanced by Large Language Models (LLMs), marks a significant advancement in processing complex data. Fuzzy clustering, by allowing data points to belong to multiple clusters, handles ambiguity and uncertainty. LLMs contribute to this framework by providing context-aware text processing, enabling more nuanced and accurate data representations. This can also help in interpreting clusters, which is our goal for future work.

References

Affeldt, S.; Labiod, L.; and Nadif, M. 2020. Ensemble block co-clustering: a unified framework for text data. In *CIKM*, 5–14.

Affeldt, S.; Labiod, L.; and Nadif, M. 2021. Regularized bi-directional co-clustering. *Statistics and Computing*, 31(3): 32.

Cai, H.; Zheng, V. W.; and Chang, K. C. 2018. A Comprehensive Survey of Graph Embedding: Problems, Techniques, and Applications. *IEEE TKDE*, 30(9): 1616–1637.

Chang, S.; Han, W.; Tang, J.; Qi, G.-J.; Aggarwal, C. C.; and Huang, T. S. 2015. Heterogeneous Network Embedding via Deep Architectures. In *SIGKDD*, 119–128.

Chen, M.; Wei, Z.; Huang, Z.; Ding, B.; and Li, Y. 2020. Simple and deep graph convolutional networks. In *ICML*, 1725–1735.

Dai, Q.; Li, Q.; Tang, J.; and Wang, D. 2018. Adversarial Network Embedding. In *AAAI*, 2167–2174.

Fan, J. 2021. Large-scale subspace clustering via k-factorization. In *SIGKDD*, 342–352.

Febrissy, M.; Salah, A.; Ailem, M.; and Nadif, M. 2022. Improving NMF clustering by leveraging contextual relationships among words. *Neurocomputing*, 495: 105–117.

Fettal, C.; Labiod, L.; and Nadif, M. 2022. Efficient graph convolution for joint node representation learning and clustering. In *WSDM*, 289–297.

Fettal, C.; Labiod, L.; and Nadif, M. 2023. Scalable attributed-graph subspace clustering. In *AAAI*, volume 37, 7559–7567.

Gao, Y.; Wang, D.; Pan, J.; Wang, Z.; and Chen, B. 2019. A novel fuzzy c-means clustering algorithm using adaptive norm. *International Journal of Fuzzy Systems*, 21: 2632–2649.

Huang, X.; Li, J.; and Hu, X. 2017. Label Informed Attributed Network Embedding. In *WSDM*, 731–739.

Kaski, S.; Nikkilä, J.; Oja, M.; Venna, J.; Törönen, P.; and Castrén, E. 2003. Trustworthiness and metrics in visualizing similarity of gene expression. *BMC bioinformatics*, 4: 1–13.

Labiod, L.; and Nadif, M. 2024. Power attributed graph embedding and clustering. *IEEE TNNLS*, 35(1): 1439–1444.

Li, Q.; Zhang, X.; Liu, H.; Dai, Q.; and Wu, X. 2021. Dimensionwise Separable 2-D Graph Convolution for Unsupervised and Semi-Supervised Learning on Graphs. In *KDD*, 953–963.

Mavromatis, C.; and Karypis, G. 2021. Graph infoclust: Maximizing coarse-grain mutual information in graphs. In *PAKDD*, 541–553.

Pan, S.; Hu, R.; Long, G.; Jiang, J.; Yao, L.; and Zhang, C. 2018. Adversarially Regularized Graph Autoencoder for Graph Embedding. In *IJCAI*, 2609–2615.

Qi, G.; Aggarwal, C. C.; Tian, Q.; Ji, H.; and Huang, T. S. 2012. Exploring Context and Content Links in Social Media: A Latent Space Method. *IEEE TPAMI*, 34(5): 850–862.

Rousseeuw, P. J. 1987. Silhouettes: a graphical aid to the interpretation and validation of cluster analysis. *Journal of computational and applied mathematics*, 20: 53–65.

Salah, A.; Ailem, M.; and Nadif, M. 2018. Word co-occurrence regularized non-negative matrix tri-factorization for text data co-clustering. In *AAAI*, volume 32.

Salah, A.; and Nadif, M. 2017. Model-based von Mises-Fisher Co-clustering with a Conscience. In *SDM*, 246–254.

Salah, A.; and Nadif, M. 2019. Directional co-clustering. *Advances in Data Analysis and Classification*, 13(3): 591–620.

Steinley, D. 2004. Properties of the hubert-arable adjusted rand index. *Psychological methods*, 9(3): 386.

Strehl, A.; and Ghosh, J. 2002. Cluster ensembles—a knowledge reuse framework for combining multiple partitions. *Journal of machine learning research*, 3: 583–617.

Wang, C.; Pan, S.; Long, G.; Zhu, X.; and Jiang, J. 2017. MGAE: Marginalized Graph Autoencoder for Graph Clustering. In *CIKM*, 889–898. ISBN 978-1-4503-4918-5.

Wang, W.; Wei, F.; Dong, L.; Bao, H.; Yang, N.; and Zhou, M. 2020. Minilm: Deep self-attention distillation for task-agnostic compression of pre-trained transformers. *Neurips*, 33: 5776–5788.

Wu, F.; Souza, A.; Zhang, T.; Fifty, C.; Yu, T.; and Weinberger, K. 2019. Simplifying graph convolutional networks. In *ICML*, 6861–6871.

You, C.; Li, C.-G.; Robinson, D. P.; and Vidal, R. 2016. Oracle based active set algorithm for scalable elastic net subspace clustering. In *CVPR*, 3928–3937.

You, C.; Robinson, D.; and Vidal, R. 2016. Scalable sparse subspace clustering by orthogonal matching pursuit. In *CVPR*, 3918–3927.

Yu, W.; Cheng, W.; Aggarwal, C.; Zong, B.; Chen, H.; and Wang, W. 2019. Self-Attentive Attributed Network Embedding Through Adversarial Learning. In *ICDM*, 758–767.

Zhu, H.; and Koniusz, P. 2021. Simple Spectral Graph Convolution. In *ICLR*.