Journal of Global Optimization (2021) 81:967–989
https://doi.org/10.1007/s10898-021-01024-z

ORCA: Outlier detection and Robust Clustering for Attributed graphs

Srinivas Eswar[1] · **Ramakrishnan Kannan**[2] · **Richard Vuduc**[1] · **Haesun Park**[1]

Received: 4 September 2020 / Accepted: 3 April 2021 / Published online: 3 May 2021

Abstract
A framework is proposed to simultaneously cluster objects and detect anomalies in attributed graph data. Our objective function along with the carefully constructed constraints promotes interpretability of both the clustering and anomaly detection components, as well as scalability of our method. In addition, we developed an algorithm called Outlier detection and Robust Clustering for Attributed graphs (ORCA) within this framework. ORCA is fast and convergent under mild conditions, produces high quality clustering results, and discovers anomalies that can be mapped back naturally to the features of the input data. The efficacy and efficiency of ORCA is demonstrated on real world datasets against multiple state-of-the-art techniques.

Keywords Attributed graphs · Robust clustering · Anomaly detection · Joint matrix low rank approximation

✉ Srinivas Eswar
seswar3@gatech.edu

Ramakrishnan Kannan
kannanr@ornl.gov

Richard Vuduc
richie@cc.gatech.edu

Haesun Park
hpark@cc.gatech.edu

[1] School of Computational Science and Engineering, Georgia Institute of Technology, Atlanta, GA 30308, USA

[2] Oak Ridge National Laboratory, 1 Bethel Valley Road, Oak Ridge, TN 37830, USA

1 Introduction

We consider the problem of simultaneously detecting the anomalous objects and producing robust clusters in attributed graph data [1,2,4]. By an "attributed graph" we mean a network with features attached to the objects (the vertices), which has numerous applications in the analysis and mining of social networks, gene regulatory networks, document corpora, image datasets, and infrastructure networks, to name a few [4,13,33,38,39,48]. For example, our case study in Sect. 5.7 considers a corpus of patents represented in an attributed graph. In this graph, the nodes are the patents whose attributes are the keywords associated with each patent, and the edges show the patent-patent citation network information. The analysis task is to (a) cluster the patents, and simultaneously (b) identify patents that are anomalies or outliers in the sense that they do not appear to "fit" within the clusters or corpus. Indeed, the presence of outliers are often detrimental to finding good clusters [41].

There is a variety of previously proposed work on simultaneous clustering and anomaly detection, most notably methods based on robust statistics [6,7,20,46,47]. However, they tend to produce clusters or anomalies (or both) that may be difficult for an end-user analyst to *interpret* in terms of the input data (see Sect. 3.2). For instance, a patent analyst might want to know which keywords or citations raise suspicions about a patent and to what degree. Prior methods often transform the input data into a form that is easier to mine algorithmically in a black-box manner, which cannot be readily translated back into the natural features of the input problem. This transformation makes the detection mechanism, while accurate, difficult to interpret and trust. For example, a particular patent may be flagged as suspicious due to the *absence* of certain keywords rather than highlighting relevant parts of the document. Our patent analyst must then examine the entire document.

We propose a new framework that addresses this limitation and permits interpretable and simultaneous clustering and anomaly detection (Sect. 3). Our work extends nonnegative matrix factorization (NMF) techniques [13,29], which have a similar philosophy of interpretability as their goal. Our key insight lies in our problem formulation. Like prior work, we use a matrix-based formulation that decomposes the input matrices (features and the graph) into a low rank plus sparse form, where the low rank component captures clusters and the sparse component captures anomalies. All the components are nonnegative and computed by solving a suitable optimization problem. However, our novel formulation imposes constraints that give the interpretability of *both* components equal billing. In contrast, the formalisms of prior state-of-the-art methods emphasize the low rank (clustering) component. While they are able to identify anomalies, they fail to do so in a way that can be understood directly in terms of the original features of the input. Consequently, they might, for instance, indicate that a patent is anomalous but trace it to edges that do not exist in the original graph. Our approach mitigates such issues.

In addition to our problem formulation, we develop a fast and convergent algorithm, ORCA (Outlier detection and Robust Clustering for Attributed graphs), in this framework and show both its efficacy and efficiency on real-world data. The main contributions of our work may be summarized as follows.

- We carefully develop a model for simultaneous clustering and anomaly detection, taking care to promote interpretability of the both types of results produced. In particular, we are able to describe why a certain node is flagged as an anomaly (Sect. 3).
- We develop an algorithm ORCA according to this model. ORCA utilizes both attribute and network information in a joint optimization framework that is better than using the different data types individually (Sect. 4).

– ORCA is a fast and convergent algorithm guaranteed to converge to a stationary point under mild conditions (Sect. 4.1).
– We conduct extensive experimental comparisons and demonstrate ORCA's efficacy and efficiency against multiple methods, including state-of-the-art techniques such as Nonnegative Residual Matrix Facotrization (NrMF) [42], Accelerated Local Anomaly Detection [33], Text Outlier Nonnegative Matrix Factorization [22], ANOMALOUS [38], and Robust Principal Components Analysis [46]. We show that ORCA can be faster by a factor up to 100× when compared to ANOMALOUS (Sect. 5).

Taken together, our approach extends simultaneous clustering and anomaly detection with better interpretability and scalability for end-user analysts.

2 Related work

Graph mining methods can largely be categorized into two groups: those involving matrix low rank approximations and others [9]. In matrix approximation methods the adjacency matrix of the graph, or some function of it like the Laplacian [9,43], is approximated via a low rank matrix.

Low rank approximation of the adjacency matrix provides embeddings for every node present in the graph. These embeddings can then be used in various data mining tasks like clustering, anomaly detection, node ranking among others [1,2,4]. Spectral clustering [11,43] based on the eigenvalue decomposition is one the classical methods used to cluster the nodes of a graph. Singular value decomposition (SVD) [12], nonnegative matrix factorization (NMF) [14,26], and the CUR decomposition [35] are other popular matrix approximation techniques used for clustering. Matrix approximation methods have also been used for anomaly detection in graphs by examining the residual obtained by subtracting the low rank matrix from the input [4,41,42]. Tong and Lin impose nonnegativity on both the matrix approximation and the residual to detect anomalous nodes [42].

This matrix-based framework was extended to attributed graphs in a joint optimization formulation over both the attribute and connection matrices [13,33,38]. Du et al. describe a method to cluster attributed graphs using a variation of NMF [13]. Liu et al. have a similar formulation for detecting local anomalies in attributed graphs [33]. Peng et al. use a CUR decomposition based approach forthe same problem [38] Many popular matrix approximation based methods have been extended to tensor based methods for hypergraphs [9,15,45].

With the recent introduction of seminal works on robust principal component analysis by Candes et al. it became possible to decompose a data matrix into a low rank "signal" matrix and a sparse "corruptions" matrix [7,8]. There has been much research on employing this idea in many data mining tasks [6]. Most of the original work in robust methods are theoretical [7,8,46,47] and study conditions for convergence and nature of "corruptions" captured. Many of these results come from computer vision tasks [6] but can be directly translated to graph mining. Kannan et al. impose nonnegativity constraints on the low rank matrix and penalize columns with large norms in the corruptions matrix to detect outliers in text data [22]. The primary differentiating factor of this work from existing methods described above is imposing the same constraints on the residual matrix as those found on the inputs. This allows us to interpret the anomalies detected directly instead of dealing with outlier scores alone.

Apart from the matrix approximation based techniques there exist a lot of other graph mining algorithms for clustering and anomaly detection. Methods often generate node embeddings based on local information, like degree, edge weights, egonet characteristics, etc., and

are commonly used in spam filters, fake review detectors, network intrusion detectors, and web indexers [3,17,19,25,27,44]. Graph clustering has also been studied under different formulations [31] based on network flows [30] and probabilistic graphical models [18,48], among others [16].

3 Model formulation

We assume that the input data is an attributed network with n nodes and m features. It is represented as a nonnegative feature-data matrix $\mathbf{X} \in \mathbb{R}_+^{m \times n}$ and data-data connection matrix $\mathbf{S} \in \mathbb{R}_+^{n \times n}$, where $\mathbb{R}_+$ denotes the nonnegative real numbers. Bold text is used to represent matrices and $\mathbf{I}$, $\mathbf{1}$ and $\mathbf{0}$ refers to the identity matrix, all ones matrix, and all zeros matrix, respectively. Matrix rows and columns are represented using MATLAB notation as $\mathbf{X}(i, :)$ and $\mathbf{X}(:, j)$ for the i-th row and j-th column respectively. The (i, j)-th entry of $\mathbf{X}$ is x_{ij}. For $\mathbf{A} \in \mathbb{R}^{m \times n}$ we define $\|\mathbf{A}\|_{p,q} = \left(\sum_{j=1}^{n} \left(\sum_{i=1}^{m} |a_{ij}|^p\right)^{\frac{q}{p}}\right)^{\frac{1}{q}}$, $\|\mathbf{A}\|_{\max} = \max_{i,j} |a_{ij}|$, and $\|\mathbf{A}\|_* = \sum_i \sigma_i(\mathbf{A})$ where $\sigma_i(\mathbf{A})$ is the ith singular value of $\mathbf{A}$.

3.1 Data model

Empirical observations can be considered to have three separate components, underlying signal, noise, and corruption such as anomalies/outliers [7]. The underlying signal is often assumed to be generated from a known process that can be modelled. Noise is a broad term encompassing other factors like imprecise measuring devices, background factors and random errors. In general, noise can affect all measurements and carry no useful information. Corruptions on the other hand affect a few observations and are more "structural" or "deliberate" in nature in contrast to "random" noise. A key shift from standard models of observation here is to separate the "noise" and "corruptions" components of the data. An intuitive example of such a difference would be from natural images: unsteady handling of the camera may cause some noise via blurring of the entire image whereas a few faulty pixels will cause corruption that will appear black in images.

Formalizing this data model in matrix notation, we are given a data matrix $\mathbf{A}$ and want to decompose it into three components: signal $\mathbf{L}$, corruptions $\mathbf{Z}$, and noise $\mathbf{N}$ such that,

$$\mathbf{A} = \mathbf{L} + \mathbf{Z} + \mathbf{N}. \tag{1}$$

We assume that the true signal is generated with a small number of parameters far less than the number of data items. Therefore, we impose a low rank constraint on $\mathbf{L}$. We would like to highlight some key differences between corruptions and noise. First, the corruptions are rare and affect only a few entries. Second, the corrupted entries can be *grossly* affected and can have large magnitude unlike the small noise term. Little is known about these corrupted entries except that they are rare. Therefore, we can only impose sparsity constraints on $\mathbf{Z}$. Often, we tackle this optimization problem by solving $\mathbf{A} \approx \mathbf{L} + \mathbf{Z}$ by minimizing $\mathbf{N}$.

Our work is concerned with attributed graphs, and the data matrices we deal with are the feature matrix $\mathbf{X}$ and connection matrix $\mathbf{S}$. $\mathbf{X}$ has the feature vector for every data item as its columns and $\mathbf{S}$ is an adjacency matrix representing the connections between the data items. In real applications, often both of these matrices are sparse, as seen in Table 1.

Table 1 Data size, sparsity, and number of outliers returned from RPCA and TONMF

Input	Size	Nonzeros (in %)	RPCA	TONMF
Karate Club (S)	34×34	156 (13.5 %)	156	34
Polblogs (S)	1490×1490	19025 (0.86%)	19025	37250
US Patents-F22 (S)	3040×3040	15954 (0.17%)	15954	9.2×10^6
US Patents-F22 (X)	6387×3040	314215 (1.6%)	2.73×10^5	1.94×10^7
DBLP-CS (S)	6559×8663	14943 (0.03%)	14943	1.79×10^7
DBLP-CS (X)	199147×8663	4.44×10^5 (0.03%)	4.44×10^5	1.71×10^9

3.2 Shortcomings of classical methods

We give a brief overview of two prior methods which deal with data corrupted in the manner described in Eq. (1). Candes et al. have shown that under certain conditions we are able to decompose a data matrix $\mathbf{A}$ into low rank and sparse components using standard convex optimization procedures and developed the seminal method of Robust Principal Component Analysis (RPCA) [7]. The RPCA formulation is as follows,

$$\begin{aligned} \min \quad & \|\mathbf{L}\|_* + \lambda \|\mathbf{Z}\|_{1,1} \\ \text{s.t.} \quad & \mathbf{A} = \mathbf{L} + \mathbf{Z} \,. \end{aligned}$$

Notice that RPCA does not account for noise in the data.

Kannan et al. modified this general formulation and applied the technique to detect anomalous documents in a text corpus via the Text Outlier Nonnegative Matrix Factorization algorithm (TONMF) [22]. Their formulation is,

$$\begin{aligned} \min \quad & \|\mathbf{A} - \mathbf{W}\mathbf{H} - \mathbf{Z}\|_F^2 + \lambda \|\mathbf{Z}\|_{2,1} \\ \text{s.t.} \quad & \{\mathbf{W}, \mathbf{H}\} \geq 0 \,. \end{aligned}$$

Here $\mathbf{W} \in \mathbb{R}_+^{m \times k}$ and $\mathbf{H} \in \mathbb{R}_+^{k \times n}$ are constrained to have low rank by letting $k \ll \min(m, n)$ be an input to the algorithm.

Disentangling the low rank and sparse components can be tricky if the input is both "sparse and low rank". This difficulty results in an identifiability issue for RPCA and in order to make the problem meaningful they impose the general notion of incoherence on the low rank component $\mathbf{L}$ [7]. Conceptually, these conditions ensure that $\mathbf{L}$ is *not sparse*. The rank r of the $m \times n$ matrix $\mathbf{L}$ is inversely proportional to its incoherence condition parameter μ via the following formula with a positive constant C [7],

$$r \leq C \frac{n}{\mu \, (\log m)^2} \,. \tag{2}$$

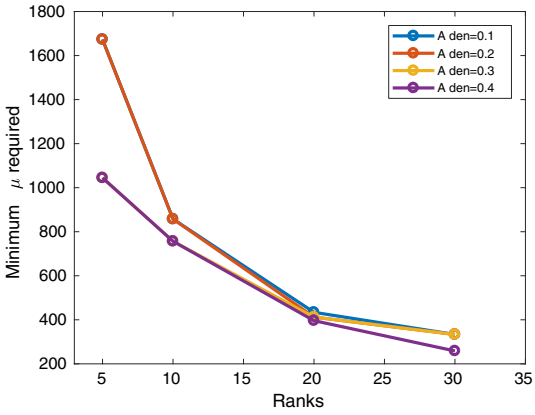


Fig. 1 Incoherence criteria on $\mathbf{A} \in \mathbb{R}^{100\times 100}$ for different ranks and densities. Minimum μ required is often very large for sparse and low-rank matrices

Formally, taking the singular value decomposition of $\mathbf{L} = \mathbf{U}\mathbf{\Sigma}\mathbf{V}^T = \sum_{i=1}^{r} \sigma_i \mathbf{u}_i \mathbf{v}_i^T$ with rank r and $\mathbf{U} \in \mathbb{R}^{m\times r}$, $\mathbf{V} \in \mathbb{R}^{n\times r}$, the incoherence condition with parameter μ states that,

$$\max_i \left\| \mathbf{U}^T \mathbf{e}_i \right\|_2 \leq \sqrt{\frac{\mu r}{m}}$$
$$\max_i \left\| \mathbf{V}^T \mathbf{e}_i \right\|_2 \leq \sqrt{\frac{\mu r}{n}}$$
$$\left\| \mathbf{U}\mathbf{V}^T \right\|_{\max} \leq \sqrt{\frac{\mu r}{mn}}\,.$$

From Eq. (2) large values of μ limit the maximum rank of $\mathbf{L}$ that RPCA can discover. Therefore small values of μ are desired which enables RPCA to separate $\mathbf{L}$ and $\mathbf{Z}$. Smaller μ also causes the singular vectors $\mathbf{u}_i$ and $\mathbf{v}_i$ to spread out their norms over multiple elements and become less "spiky" [7,8].

Unfortunately these conditions do not hold for typical "sparse and low rank" inputs. From Fig. 1 we can see μ is quite large especially in the extremely sparse and low rank cases. This large μ causes of the singular vectors to become concentrated on a few entries making it difficult for RPCA to disentangle the low rank and sparse components. TONMF doesn't suffer from this drawback as the low rank parameter is considered as an input and therefore can be tuned to the problem at hand.

A second major drawback with both RPCA and TONMF is that an inordinate number of outliers may be returned when presented with sparse inputs. Table 1 provides the number of nonzero entries present in $\mathbf{Z}$ from RPCA and TONMF for some standard input datasets. We used the dual RPCA algorithm [46] to generate Table 1. In our tests, often RPCA places the entire input as the sparse component and assigns $\mathbf{L}$ to $\mathbf{0}$ when density is less than 1%. Large μ coupled with the hard constraint of $\mathbf{A} = \mathbf{L} + \mathbf{Z}$ places an extremely low upper limit on permissible ranks for RPCA. Assigning $\mathbf{L}$ to $\mathbf{0}$ defeats the purpose of low rank approximation.

TONMF often returns the corruption matrix $\mathbf{Z}$ with nonzero entries in locations where there are zeros in the input. These corruptions are difficult to interpret as they cannot be traced back to the input. This issue is also present in RPCA albeit to a much lesser degree than TONMF.

The problem of incoherence is hard to solve. One way to get around this is to assume that the low rank is given as an input to algorithm as in TONMF. Imposing nonnegativity constraints on the low rank factors also helps in alleviating this problem. This constrained

WH, like in TONMF, forms a cone which wraps the input data from outside. The nonnegative columns of **W** form the extreme rays of this cone. This "spiky"-ness would violate the incoherence conditions on **U** and **V**. Handling fill-ins for **Z** also involves imposing nonnegativity constraints which forces the nonzeros of **Z** to only appear from the set of nonzeros of **A**. This restriction becomes apparent when we describe the update rules for **Z**.

3.3 Proposed model for attributed graphs

Assume that an attributed graph with m features and n samples is represented in the feature matrix $\mathbf{X} \in \mathbb{R}_+^{m \times n}$ and connection matrix $\mathbf{S} \in \mathbb{R}_+^{n \times n}$. The connection matrix $\mathbf{S}$ is assumed to have proper scaling, for example scaling every edge in the adjacency matrix with the degrees of their end points. We assume that the signal is well approximated by low rank matrices $\mathbf{W} \in \mathbb{R}_+^{m \times k}$ and $\mathbf{H} \in \mathbb{R}_+^{n \times k}$ with $k \ll \min(m, n)$.

Here $\mathbf{W}$ is the cluster basis matrix and $\mathbf{H}$ is the cluster indicator matrix. The location of the maximum entry in each row of $\mathbf{H}$ indicates the sample's cluster. The matrix $\mathbf{H}$ is shared between the low rank approximation of the feature matrix and the low rank approximation of the connection matrix, resulting in the following,

$$\begin{aligned} \mathbf{X} &\approx \mathbf{W}\mathbf{H}^T + \mathbf{Z}_1 \\ \mathbf{S} &\approx \mathbf{H}\mathbf{H}^T + \mathbf{Z}_2 \, . \end{aligned}$$

The sparse matrices $\mathbf{Z}_1$ and $\mathbf{Z}_2$ are expected to capture the outliers in each of the different modalities, respectively. Together, $\mathbf{Z}_1$ and $\mathbf{Z}_2$ form the anomalous attributed subgraph. We impose nonnegativity constraints on $\mathbf{W}$ and $\mathbf{H}$ for two reasons. First, it aids in interpretability of the discovered factors by promoting additive parts based approximations [28]. Second, it alleviates the problem of incoherence as discussed in Sect. 3.2. We also stipulate that $\mathbf{Z}_1$ and $\mathbf{Z}_2$ be nonnegative. Additionally, symmetric constraints are placed on $\mathbf{Z}_2$. These constraints ensure that nonzeros/outliers in $\mathbf{Z}_1$ and $\mathbf{Z}_2$ are only detected from the existing nonzero entries of $\mathbf{X}$ and $\mathbf{S}$ respectively. No spurious fill-ins, like in the case of TONMF (see Sect. 3.2), can occur in the corruption matrices making them easy to interpret. This property is shown mathematically in the next section.

The remaining part of the data is considered as noise and needs to be minimized. We are now left with the following optimization problem,

$$\begin{aligned} \min \quad & \left\| \mathbf{X} - \mathbf{W}\mathbf{H}^T - \mathbf{Z}_1 \right\|_F^2 + \alpha \left\| \mathbf{S} - \mathbf{H}\mathbf{H}^T - \mathbf{Z}_2 \right\|_F^2 \\ & + \lambda_1 \left\| \mathbf{Z}_1 \right\|_{1,1} + \lambda_2 \left\| \mathbf{Z}_2 \right\|_{1,1} \\ \text{s.t.} \quad & \\ & \mathbf{H} \geq 0, \mathbf{W} \geq 0 \\ & \mathbf{Z}_1 \geq 0, \mathbf{Z}_2 \geq 0, \mathbf{Z}_2 = \mathbf{Z}_2^T \, . \end{aligned} \tag{3}$$

The parameter α controls the relative weight given to the connection and feature information [13]. λ_1 and λ_2 control the relative sparsity of the anomalous nodes detected and can be tuned to return fewer or more nodes as needed. Using $\mathbf{Z}_1$ and $\mathbf{Z}_2$ we can define various outlier scores for the data items. In ORCA we simply sum entries of $\mathbf{Z}_1$ and $\mathbf{Z}_2$ column-wise to get the final outlier scores for each data item. More advanced aggregation methods can be designed based on the nature of the data being analyzed.

4 Algorithm

There are multiple ways to solve the optimization problem introduced in Eq. (3). We solve it via block coordinate descent (BCD) using the algorithmic framework introduced by Kim et al. [13,23]. The symmetric nonnegative matrix factorization subproblem arising due to symmetric $\mathbf{S}$ and shared $\mathbf{H}$ is tackled by adding an auxiliary variable $\hat{\mathbf{H}}$ similar to the SymNMF [26] and Joint NMF formulations [13]. Therefore we reformulate Eq. (3) as the following optimization problem.

$$\begin{aligned}
\min \quad & \left\|\mathbf{X}-\mathbf{W}\mathbf{H}^T-\mathbf{Z}_1\right\|_F^2+\alpha\left\|\mathbf{S}-\hat{\mathbf{H}}\mathbf{H}^T-\mathbf{Z}_2\right\|_F^2+\beta\left\|\hat{\mathbf{H}}-\mathbf{H}\right\|_F^2 \\
& +\lambda_1\left\|\mathbf{Z}_1\right\|_{1,1}+\lambda_2\left\|\mathbf{Z}_2\right\|_{1,1} \\
\text{s.t.} \quad & \\
& \mathbf{W}\geq 0, \quad \mathbf{H}\geq 0, \quad \hat{\mathbf{H}}\geq 0 \\
& \mathbf{Z}_1\geq 0, \quad \mathbf{Z}_2\geq 0, \mathbf{Z}_2=\mathbf{Z}_2^T \,.
\end{aligned} \tag{4}$$

Considering the matrices $\mathbf{W}$, $\mathbf{H}$, $\hat{\mathbf{H}}$, $\mathbf{Z}_1$ and $\mathbf{Z}_2$ as the five unknown blocks in the BCD framework, we alternate fixing four other blocks and updating the remaining block.

Solving for Cluster Basis ($\mathbf{W}$): Fixing $\mathbf{Z}_1$, $\mathbf{Z}_2$, $\mathbf{H}$ and $\hat{\mathbf{H}}$ and setting $\mathbf{Y}_1=\mathbf{X}-\mathbf{Z}_1$, we solve the following nonnegative least squares (NLS) problem,

$$\mathbf{W}\leftarrow\underset{\mathbf{W}\geq 0}{\arg\min}\left\|\mathbf{Y}_1-\mathbf{W}\mathbf{H}^T\right\|_F^2 \,. \tag{5}$$

Solving for Cluster Membership ($\mathbf{H}$ **and** $\hat{\mathbf{H}}$): Similarly setting $\mathbf{Y}_2=\mathbf{S}-\mathbf{Z}_2$ and fixing the other blocks, the objective function for $\mathbf{H}$ and $\hat{\mathbf{H}}$ is,

$$\min_{\mathbf{H}\geq 0}\left\|\mathbf{Y}_1-\mathbf{W}\mathbf{H}^T\right\|_F^2+\alpha\left\|\mathbf{Y}_2-\hat{\mathbf{H}}\mathbf{H}^T\right\|_F^2+\beta\left\|\mathbf{H}-\hat{\mathbf{H}}\right\|_F^2 \,.$$

These blocks can be solved as multiple NLS subproblems,

$$\mathbf{H}\leftarrow\underset{\mathbf{H}\geq 0}{\arg\min}\left\|\begin{bmatrix}\mathbf{W}\\ \sqrt{\alpha}\hat{\mathbf{H}}\\ \sqrt{\beta}\mathbf{I}_k\end{bmatrix}\mathbf{H}^T-\begin{bmatrix}\mathbf{Y}_1\\ \sqrt{\alpha}\mathbf{Y}_2\\ \sqrt{\beta}\hat{\mathbf{H}}^T\end{bmatrix}\right\|_F^2 \tag{6}$$

$$\hat{\mathbf{H}}\leftarrow\underset{\hat{\mathbf{H}}\geq 0}{\arg\min}\left\|\begin{bmatrix}\sqrt{\alpha}\mathbf{H}\\ \sqrt{\beta}\mathbf{I}_k\end{bmatrix}\hat{\mathbf{H}}^T-\begin{bmatrix}\sqrt{\alpha}\mathbf{Y}_2\\ \sqrt{\beta}\mathbf{H}^T\end{bmatrix}\right\|_F^2 \,. \tag{7}$$

Each NLS subproblem is a convex optimization problem with efficient algorithmic solutions [23]. We solve each subproblem using the Block Principal Pivoting (BPP) algorithm [24].

Solving for Attribute Outlier Component ($\mathbf{Z}_1$): Denoting $\mathbf{D}=\mathbf{X}-\mathbf{W}\mathbf{H}^T$ and fixing all the other blocks and looking at the objective for $\mathbf{Z}_1$ we get,

$$\min_{\mathbf{Z}_1\geq 0}\left\|\mathbf{D}-\mathbf{Z}_1\right\|_F^2+\lambda_1\left\|\mathbf{Z}_1\right\|_{1,1} \,. \tag{8}$$

This objective can be optimized by using the following elementwise update rule,

$$\mathbf{Z}_1 = \max\left(\mathbf{D} - \frac{\lambda_1}{2}\mathbf{1}, 0\right). \tag{9}$$

This rule can be derived considering two cases. Let us denote Eq. (8) by,

$$f(\mathbf{Z}_1) = \sum_{i,j} f(z_{ij}) = \sum_{i,j} \left(d_{ij} - z_{ij}\right)^2 + \lambda_1 z_{ij}.$$

Then f can be decomposed into an elementwise optimization problem. We can solve this by simply setting the derivative to 0 which results in Eq. (9).

$$f(z_{ij}) = \left(d_{ij} - z_{ij}\right)^2 + \lambda_1 z_{ij}, \quad \frac{df}{dz_{ij}} = -2\left(d_{ij} - z_{ij}\right) + \lambda_1$$
$$\implies z_{ij} = d_{ij} - \frac{\lambda_1}{2}.$$

When the value $d_{ij} - \frac{\lambda}{2}$ is negative, the closest nonnegative solution would be 0. This function $f(z_{ij})$ is a simple quadratic polynomial and if its minima is negative it is monotonically increasing in the positive axis. Therefore we get the projection $\max(d_{ij} - \frac{\lambda}{2}, 0)$ as the solution and the following update rule,

$$\mathbf{Z}_1 = \max\left(\mathbf{X} - \mathbf{W}\mathbf{H}^T - \frac{\lambda_1}{2}\mathbf{1}, \mathbf{0}\right). \tag{10}$$

Solving for Connection Outlier Component ($\mathbf{Z}_2$): Updating $\mathbf{Z}_2$ follows almost exactly the same steps as updating $\mathbf{Z}_1$. With $\mathbf{D} = \mathbf{S} - \hat{\mathbf{H}}\mathbf{H}^T$ we need to optimize the following objective,

$$\min_{\mathbf{Z}_2 \geq 0, \mathbf{Z}_2 = \mathbf{Z}_2^T} \alpha \left\|\mathbf{D} - \mathbf{Z}_2\right\|_F^2 + \lambda_2 \left\|\mathbf{Z}_2\right\|_{1,1}.$$

We must be careful to handle the symmetric constraint on $\mathbf{Z}_2$. Using g to represent the part of Eq. (4) contributed from $\mathbf{Z}_2$, we can split the objective element-wise. Since $\mathbf{Z}_2$ is symmetric we can reduce the number of independent terms,

$$g(\mathbf{Z}_2) = \sum_{i,j} g(z_{ij}) = \sum_{i,j} \alpha\left(d_{ij} - z_{ij}\right)^2 + \lambda_2 z_{ij}$$
$$= \sum_{i \leq j} \alpha\left(\left(d_{ij} - z_{ij}\right)^2 + \left(d_{ji} - z_{ij}\right)^2\right) + 2\lambda_2 z_{ij}.$$

This formulation of the objective handles the symmetric constraint explicitly. Taking the derivative and setting it to 0 gives us the final update rule for $\mathbf{Z}_2$,

$$\mathbf{Z}_2 = \max\left(\mathbf{S} - \left(\frac{\mathbf{H}\hat{\mathbf{H}}^T + \hat{\mathbf{H}}\mathbf{H}^T}{2}\right) - \frac{\lambda_2}{2\alpha}\mathbf{1}, \mathbf{0}\right). \tag{11}$$

The final algorithm combining Eqs. (5)–(7), (10) and (11), called Outlier detection and Robust Clustering for Attributed graphs (ORCA), can be found in Algorithm 1. We initialize $\mathbf{H}, \hat{\mathbf{H}}$ with random nonnegative values and $\mathbf{Z}_1, \mathbf{Z}_2$ as zero matrices. Finally outlier scores for every data item is obtained by summing up the elements of $\mathbf{Z}_1$ and $\mathbf{Z}_2$ in a columnwise manner.

Algorithm 1 Outlier detection and Robust Clustering for Attributed graphs (ORCA)

initialize $\mathbf{H}^{(0)}, \hat{\mathbf{H}}^{(0)}, \mathbf{Z}_1^{(0)}, \mathbf{Z}_2^{(0)}$
procedure ORCA($\mathbf{X}, \mathbf{S}, k, \lambda_1, \lambda_2, \alpha, \beta$)
  $t \leftarrow 0$
  while stopping criteria not satisfied **do**
    ▷ Updating Low-rank components
    $\mathbf{Y}_1 \leftarrow \mathbf{X} - \mathbf{Z}_1^{(t)}$
    $\mathbf{Y}_2 \leftarrow \mathbf{S} - \mathbf{Z}_2^{(t)}$
    $\mathbf{W}^{(t+1)} \leftarrow \text{NLS}(\mathbf{H}^{(t)}, \mathbf{Y}_1)$
    $\mathbf{U}_1 \leftarrow \begin{bmatrix} \mathbf{W} \\ \sqrt{\alpha}\hat{\mathbf{H}} \\ \sqrt{\beta}\mathbf{I}_k \end{bmatrix}$
    $\mathbf{V}_1 \leftarrow \begin{bmatrix} \mathbf{Y}_1 \\ \sqrt{\alpha}\mathbf{Y}_2 \\ \sqrt{\beta}\hat{\mathbf{H}}^T \end{bmatrix}$
    $\mathbf{H}^{(t+1)} \leftarrow \text{NLS}(\mathbf{U}_1, \mathbf{V}_1)$
    $\mathbf{U}_2 \leftarrow \begin{bmatrix} \sqrt{\alpha}\mathbf{H} \\ \sqrt{\beta}\mathbf{I}_k \end{bmatrix}$
    $\mathbf{V}_2 \leftarrow \begin{bmatrix} \sqrt{\alpha}\mathbf{Y}_2 \\ \sqrt{\beta}\mathbf{H}^T \end{bmatrix}$
    $\hat{\mathbf{H}}^{(t+1)} \leftarrow \text{NLS}(\mathbf{U}_2, \mathbf{V}_2)$
    ▷ Updating Sparse components
    $\mathbf{D}_1 \leftarrow \mathbf{X} - \mathbf{W}^{(t+1)}(\mathbf{H}^{(t+1)})^T$
    $\mathbf{D}_2 \leftarrow \mathbf{S} - \hat{\mathbf{H}}^{(t+1)}(\mathbf{H}^{(t+1)})^T$
    $\mathbf{D}_2 \leftarrow \frac{\mathbf{D}_2 + \mathbf{D}_2^T}{2}$
    $\mathbf{Z}_1^{(t+1)} \leftarrow \text{MAX}(\mathbf{D}_1 - \frac{\lambda_1}{2}\mathbf{1}, 0)$
    $\mathbf{Z}_2^{(t+1)} \leftarrow \text{MAX}(\mathbf{D}_2 - \frac{\lambda_2}{2\alpha}\mathbf{1}, 0)$
    $t \leftarrow t + 1$
  end while
  $t \leftarrow t - 1$
  ▷ Calculating outlier scores ($\mathbf{c}$)
  for all $i \leftarrow 1, n$ **do**
    $c_i \leftarrow \sum_{j=1}^{m} \mathbf{Z}_1^{(t)}(j,i) + \sum_{j=1}^{n} \mathbf{Z}_2^{(t)}(j,i)$
  end for
  return $\mathbf{W}^{(t)}, \mathbf{H}^{(t)}, \hat{\mathbf{H}}^{(t)}, \mathbf{Z}_1^{(t)}, \mathbf{Z}_2^{(t)}, \mathbf{c}$
end procedure

4.1 Convergence analysis

We shall examine the convergence of ORCA in the BCD framework [5,23]. Our problem formulation described in Eq. (4) has been divided into 5 blocks of variables, namely $\mathbf{W}$, $\mathbf{H}$, $\hat{\mathbf{H}}$, $\mathbf{Z}_1$ and $\mathbf{Z}_2$. According to Theorem 1 by Kim et al. [23], if the sequence of solutions $\{\mathbf{W}^{(t)}, \mathbf{H}^{(t)}, \hat{\mathbf{H}}^{(t)}, \mathbf{Z}_1^{(t)}, \mathbf{Z}_2^{(t)}\}$ generated by the BCD method attains a unique minimum at all steps t then every limit point of the sequence is a stationary point. We now need to show that every update step of Algorithm 1 achieves both these conditions. The update steps can be separated into NLS updates [Eqs. (5)–(7)] and thresholding updates [Eqs. (10) and (11)].

The NLS problems, of the form $\min_{\mathbf{G} \geq 0} \left\| \mathbf{F}\mathbf{G}^T - \mathbf{C} \right\|_F^2$, are convex and have unique minima when $\mathbf{F}$ is full rank. This computation is denoted by NLS($\mathbf{F}$, $\mathbf{C}$) in Algorithm 1. The full rank condition is trivially true for Eqs. (6) and (7) due to the presence of $\sqrt{\beta}\mathbf{I}_k$ in $\mathbf{F}$. For Eq. (5) we

need to assume $\mathbf{H}$ remains full rank throughout the BCD method, which is a mild assumption on the cluster indicator matrix. The thresholding updates are shown to have a closed form solution that automatically satisfy the conditions for Theorem 1. Thus we can show that ORCA converges to a stationary point under mild assumptions. It should be pointed out that algorithms based on the BCD framework converge to a stationary point unlike popular methods like Multiplicative Updating (MU) which may not converge [23].

4.2 Computational complexity

The runtime of Algorithm 1 can be broken down into NLS and thresholding steps. ORCA performs three NLS solves and two thresholding steps per iteration using the BPP algorithm [24] as the NLS solver. Setting up the normal equations for BPP requires applying $\mathbf{Y}_1$ to $\mathbf{W}^T$ and $\mathbf{H}$ as well as $\mathbf{Y}_2$ to $\mathbf{H}^T$ and $\hat{\mathbf{H}}$ which take $\mathcal{O}\left(mnk + n^2k\right)$ flops. We also need to compute gramians of the factor matrices which takes $\mathcal{O}\left((m+n)\,k^2\right)$ flops. While BPP doesn't have a closed form solution for runtime, it is empirically shown to be faster than algorithms which run in $\mathcal{O}\left((m+n)k^2\right)$ flops [21]. The thresholding steps by themselves are element-wise operations and take $\mathcal{O}\,(mn)$ flops. However creating the intermediate matrices $\mathbf{D}_1$ and $\mathbf{D}_2$ involves an outer product of two low-rank matrices which takes $\mathcal{O}\,(mnk)$ and $\mathcal{O}\left(n^2k\right)$ flops which could become expensive. Therefore the overall expected per iteration runtime is $\mathcal{O}\left(mnk + n^2k + (m+n)k^2\right)$.

5 Experiments

We compare the proposed algorithm ORCA and other existing methods on real world attributed graphs. Our graphs are from the data sets of Cora [34], Disney [37], US Patents [13], Enron [32], Amazon [38] and DBLP-CS [40].

5.1 Data sets

We used the following datasets in our experiments.

- **Cora**: Cora data set contains 2708 machine learning publications classified into 7 Classes [34,36] Citation information among the publications is also present.
- **Disney**: A subset of the Amazon co-purchase network specifically created for anomaly detection [37]. Only Disney products were selected from the network and clustered using a modularity based technique [37].
- **US Patents**: Du, Drake and Park [13] collected and cleaned a subset of US patent data from PatentsView.[1] The data set contains a term-frequency vector for each patent claim as well as the patent citation information. Patents were classified into 13 different categories (e.g., B09, F22 etc.) with each category containing multiple classes. We use the individual categories as separate datasets with classes as labels for individual patents.[2]
- **DBLP-CS**: DBLP scientific publication information is obtained from Aminer[3] and parsed using the pipeline developed by Revelle et al. [40]. The dataset contains term-frequency information extracted from the abstracts for each document as well paper citation informa-

[1] http://www.patentsview.org.

[2] https://github.com/smallk/.

[3] https://aminer.org/

Table 2 Our evaluation datasets vary in size, no. of clusters, and no. of outliers to thoroughly test the different detection methods

Name	Features	Samples	Clusters	Outliers	Outlier (%)
A22	8451	1620	30	25	1.54
B09	10610	1436	10	31	2.16
B68	3487	385	10	10	2.60
C06	8777	1093	20	20	1.83
C13	4728	406	10	10	2.46
D02	7032	1427	40	20	1.40
F22	6387	1306	40	20	1.53
Y04	8142	740	5	45	6.08
Amazon	21	1418	–	28	1.97
Cora	1433	2023	4	10	0.49
DBLP-CS	199147	4196	3	10	0.24
Disney	347	124	8	6	4.84
Enron	18	13533	–	5	0.04

tion. We sampled the papers from the years 1990-2010 and from the conference—FOCS, ICML, IPDPS, ISCA and VLDB. We treat the publication venue as labels for this dataset.

– **Amazon**: This data set contains the Amazon co-purchase network of books with attributes like prices, ratings, number of reviews, etc. [32]. The anomalies are obtained using the amazonfail tag information.
– **Enron**: The Enron dataset[4] is an email network with message content information and connection information built from recipient data [32]. Spam messages are treated as the anomalies. The number of true clusters are unknown, but we estimated the rank of the dataset to be 5 by looking at the singular values of both the message content and connection matrices.

Since Cora, US Patents, and DBLP-CS do not contain outlier samples, we explicitly inject outliers into the datasets. We randomly choose some fixed number of classes from the datasets and treat the subset of the attribute and graph which contain those classes as the non-anomalous attribute and network information. Then we select a few data samples from the remaining classes and inject them into the dataset. These are labelled as the outlier samples. For example, the B09 dataset from US Patents originally contained 38 different classes. We selected the top 10 classes with highest number of documents as our "true" clusters and subsampled both the attribute and graph matrices for these 10 classes. Then we randomly picked 31 patents from other classes and injected them as outliers in the final dataset for B09.[5] We ensure that the injected outliers do not come from the same class since we do not intend to have any correlations between outliers. Since Amazon and Enron data sets do not contain cluster information, we do not include them in the clustering experiments. The details of these networks can be found in Table 2.

All the graphs' adjacency matrices were normalized into the symmetric form $\mathbf{S} = \mathbf{D}^{-\frac{1}{2}}\mathbf{A}\mathbf{D}^{-\frac{1}{2}}$ where $\mathbf{A}$ is the adjacency matrix and $\mathbf{D} = \text{diag}\,(d_1, \ldots, d_n)$ is a diagonal matrix

[4] https://www.cs.cmu.edu/~enron/

[5] Our code and datasets are publicly available at https://gitlab.com/seswar3/orca.

containing the degrees of each vertex along its diagonal. The attributes matrix $\mathbf{X}$ is also normalized so that every column vector has unit ℓ_2-norm.

5.2 Baseline algorithms

We compare our proposed ORCA method to the following baseline algorithms:

- Accelerated Local Anomaly Detection (**ALAD**) uses NMF to jointly cluster both the attribute and network inputs and uses the inverse of cosine similarity between data points and cluster centroids as a score for outlierness [33].
- Nonnegative Residual Matrix Factorization (**NrMF**) approximates the connection matrix as the product of two low rank matrices. The residual matrix, obtained by subtracting the low rank approximation from the input matrix, is constrained to be nonnegative on those entries where the input matrix is positive. We consider Algorithm-3 presented by Tong and Lin [42] as the baseline.
- Robust Principal Components Analysis (**RPCA**) decomposes the input matrix into a low rank component and sparse factor via convex optimization (see Sect. 3.2). We use the dual RPCA algorithm introduced by Wright et al. [46].
- Text Outlier Nonnegative Matrix Factorization algorithm (**TONMF**) is an anomaly detection algorithm designed for text data [22]. It uses a robust formulation of NMF to capture anomalous documents in a column-wise sparse outlier matrix $\mathbf{Z}$. Outlier scores are captured as ℓ_2 norms of columns of $\mathbf{Z}$.
- ANOMALOUS **(ANOM)** is a method that simultaneously filters irrelevant node attributes and performs anomaly detection [38]. It uses a modified form of the CUR decomposition [35] to separate the signal and outlier matrices.

NrMF, RPCA and TONMF are designed to work only on a single view of input, either attribute or connection information. We run each of these algorithms twice using $\mathbf{X}$ and $\mathbf{S}$ as input separately each time, and then aggregate the results to obtain the final outlier and cluster scores. This aggregation ensures that results, for the above algorithms, are produced on the same amount of information for fair comparison even though they were not designed to utilize multiple views of the data.

5.3 Evaluation criteria

We evaluate the methods by ranking every node in descending order of outlier scores. A data item is labelled as an outlier if its score exceeds a certain threshold. We use the Area Under the Precision-Recall Curve (PR-AUC) as the metric for comparison of performance. PR-AUC is a metric that is widely used as an alternate to the Receiver Operator Characteristic curve (ROC-AUC) in cases of binary classification with a large skew in class distribution [10,33] like in anomaly detection. In our test datasets, the range for the number of anomalous data items is from 0.24 to 6.08 % of the total data, which is a clear skew in class distributions. ROC-AUC statistics are also captured. Higher AUC values indicate better anomaly detection.

We utilize the Normalized Mutual Information (NMI) and Adjusted Rand Index (RI) metrics for measuring clustering accuracy. ORCA, TONMF, NrMF and ALAD all provide nonnegative cluster indicator matrices. For TONMF and NrMF we sum up the cluster indicator matrices returned from running the methods on both $\mathbf{X}$ and $\mathbf{S}$. Each of the methods ORCA and ALAD returns only a single cluster indicator matrix. RPCA does not return a cluster indicator matrix but provides embeddings for each node of the graph. We run K-Means on these

embeddings from RPCA in a manner similar to spectral clustering [43]. We exclude ANOM from this comparison as it does not provide node embeddings or clustering assignments to the vertices of the graph.

All our experiments were performed on a server with two Intel(R) Xeon(R) CPU E5-2680 v3 CPUs and 377GB memory.

5.4 Hyperparameter setting

For ORCA, we found that setting $\lambda_1 = 0.7 \max(\mathbf{X})$ and $\lambda_2 = 0.7\alpha \max(\mathbf{S})$ results in good performance, and α, β were selected according to Du et al. [13]. Tuning λ_1, λ_2 to the problem instance can help improve the performance of ORCA, but we used the fixed values throughout our tests as tuning them would give ORCA an unfair advantage. Note that setting $\lambda_1 \geq 2 \max(\mathbf{X})$ or $\lambda_2 \geq 2 \max(\mathbf{S})$ will result in returning $\mathbf{0}$ for the outlier matrices.

For the ALAD and ANOM algorithms we used the default settings present in the algorithm. Since TONMF didn't provide any defaults we did a grid search to tune the penalty term for its sparse component. NrMF and RPCA do not have any hyperparameters. We limit the number of iterations of RPCA to 200, ALAD to 700, ORCA to 50, ANOM to 20, and TONMF to 10. These limits were either specified by the original authors (for ALAD and TONMF) or chosen to roughly let the algorithms all run for a reasonable amount of time (RPCA, ANOM, and ORCA). NrMF only runs for k iterations. RPCA would stop if the relative reconstruction error, i.e., the Frobenius norm of the low rank and sparse component divided by the Frobenius norm of the input matrix, is less than 2×10^{-5}. ALAD, ANOM, and ORCA also stop iterating if the objective increases and return the current iterate as the solution.

5.5 Results

5.5.1 Outlier detection

The ROC-AUC and PR-AUC values for the various real world graphs are presented in Tables 3 and 4 respectively. We average the results over 5 runs with different initializations. The top performing algorithm for each dataset is highlighted in bold text. The second best method is underlined.

We should look at both the PR-AUC and ROC-AUC values in conjunction to evaluate the performance of the methods. Ideally both of the PR-AUC and ROC-AUC values should be high if the detection algorithm is performing as expected. If ROC-AUC is high and PR-AUC is low it indicates that the algorithm is able to classify the non-anomalous items correctly but misses out on the anomalous entries. If the AUC-PR is high and ROC-AUC is low the opposite effect is taking place. This condition implies that the detection algorithm is flagging too many nodes as outliers.

We can observe that ANOM seems to be the best performing algorithm over both metrics with ORCA and ALAD the next best. Looking at both AUC metrics we can observe the shortcomings of ALAD. In our experiments we observed that ALAD would classify a large number of nodes with large outlier scores and thereby perform well on the PR-AUC metric. Algorithms such as TONMF, RPCA, and NrMF, which do not explicitly couple the attribute and connection data, perform significantly worse than others. This poor performance is expected and highlights the importance of fusing and utilising all the available information. Our results further verify it empirically. Finally, the running time of ANOM includes k matrix inversions every iteration which is very expensive and does not scale. For example, it did not

Table 3 ROC-AUC performance for outlier detection. The top performing algorithm is highlighted in bold text and the second best method is underlined

Dataset	ORCA	TONMF	RPCA	NrMF	ALAD	ANOM
A22	0.567	0.493	0.341	0.443	0.491	**0.745**
B09	0.549	0.503	0.483	0.499	0.477	**0.570**
B68	0.774	0.524	0.434	0.501	0.227	**0.782**
C06	0.674	0.515	0.379	0.400	0.501	**0.785**
C13	0.547	0.514	0.419	0.248	0.175	**0.908**
D02	0.642	0.466	0.407	0.436	0.512	**0.750**
F22	0.571	0.450	0.296	0.420	0.503	**0.809**
Y04	0.458	0.515	0.540	0.538	0.321	**0.695**
Amazon	0.497	0.000	0.340	0.554	0.490	**0.627**
Cora	0.602	0.000	0.367	0.467	0.550	**0.859**
DBLP-CS	**0.573**	0.492	0.301	0.359	0.545	-
Disney	**0.712**	0.000	0.556	0.263	0.291	0.596
Enron	**0.579**	0.552	0.571	0.475	0.466	0.457
Average	0.596	0.386	0.418	0.431	0.427	**0.715**

Table 4 PR-AUC performance for outlier detection. The top performing algorithm is highlighted in bold text and the second best method is underlined

Dataset	ORCA	TONMF	RPCA	NrMF	ALAD	ANOM
A22	0.018	0.024	0.011	0.013	**0.456**	0.089
B09	0.025	0.033	0.038	0.021	**0.081**	0.025
B68	**0.118**	0.044	0.023	0.025	0.037	**0.118**
C06	0.058	0.026	0.015	0.014	**0.445**	0.049
C13	0.031	0.026	0.019	0.015	0.014	**0.338**
D02	0.054	0.019	0.012	0.013	**0.492**	0.075
F22	0.018	0.019	0.010	0.039	**0.468**	0.079
Y04	0.072	**0.162**	0.072	0.077	0.041	0.146
Amazon	0.019	NaN	0.014	0.020	**0.114**	0.029
Cora	0.008	NaN	0.004	0.004	0.038	**0.229**
DBLP-CS	0.006	0.005	0.003	0.003	**0.440**	-
Disney	**0.241**	NaN	0.216	0.031	0.032	0.072
Enron	0.001	0.080	0.001	0.000	**0.180**	0.000
Average	0.051	0.044	0.034	0.021	**0.218**	0.104

finish for the DBLP-CS dataset within 8 hours. On the other hand, ORCA is able to scale up to $\sim$ 200,000 features linearly (Sect. 4.2).

After considering these factors we believe that ANOM and ORCA are the most effective anomaly detection methods. ANOM performs very well in terms of accuracy but can become computationally prohibitive for large datasets.

Table 5 Clustering performance measured in NMI. The top performing algorithm is highlighted in bold text and the second best method is underlined

Dataset	ORCA	TONMF	RPCA	NrMF	ALAD
A22	**0.620**	0.067	0.296	0.112	0.124
B09	**0.388**	0.012	0.143	0.028	0.019
B68	**0.809**	0.035	0.256	0.136	0.712
C06	**0.511**	0.045	0.313	0.107	0.090
C13	**0.500**	0.035	0.278	0.080	0.293
D02	**0.515**	0.128	0.297	0.142	0.189
F22	**0.533**	0.139	0.350	0.132	0.191
Y04	**0.441**	0.037	0.188	0.009	0.291
Cora	0.179	0.105	0.015	0.004	**0.301**
DBLP-CS	**0.441**	0.000	0.040	0.008	0.004
Disney	**0.544**	0.222	0.239	0.094	0.508
Average	**0.498**	0.075	0.22	0.077	0.247

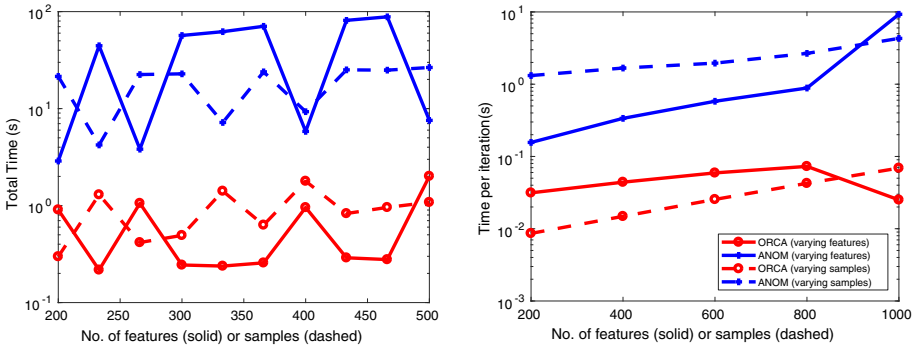
Table 6 Clustering performance measured in Adjusted Rand Index. The top performing algorithm is highlighted in bold text and the second best method is underlined

Dataset	ORCA	TONMF	RPCA	NrMF	ALAD
A22	**0.373**	0.000	0.097	− 0.002	0.009
B09	**0.303**	− 0.000	0.085	− 0.005	0.002
B68	**0.727**	− 0.001	0.103	0.023	0.567
C06	**0.299**	0.000	0.114	0.005	0.009
C13	**0.272**	0.001	0.072	0.001	0.070
D02	**0.241**	− 0.000	0.061	0.001	0.012
F22	**0.215**	0.001	0.074	0.000	0.005
Y04	**0.418**	0.004	0.183	− 0.000	0.127
Cora	0.148	0.054	0.000	0.004	**0.266**
DBLP-CS	**0.448**	− 0.000	0.027	− 0.006	0.002
Disney	**0.419**	0.055	0.042	0.005	0.364
Average	**0.351**	0.01	0.078	0.002	0.13

5.5.2 Clustering

The results based on NMI and RI metrics are presented in Tables 5 and 6 respectively. Similar to the outlier results, the measurements are averaged over 5 runs with different starting points. The top performing algorithm for each dataset is highlighted in bold text. The second best method is underlined.

Unlike the outlier case there is a clear winner in clustering. ORCA performs consistently better than the other methods. RPCA and ALAD are the next best performing method. Both ORCA and RPCA formulations involve an explicit "outlier" matrix $\mathbf{Z}$ which helps the algorithms produce robust clusters less affected by the outliers. ALAD performs better than RPCA on datasets with relatively larger number of outliers (outlier percentage of 2% or more). ORCA is more robust to these varying percentage of outliers and consistently clusters well. ANOM is excluded from this experiment as it does not provide node embeddings or clustering assignments to the vertices of the graph.



(a) $m = 100$ or $n = 100$ and $k = 50$. **(b)** $m = 100$ or $n = 100$ and $k = 50$

Fig. 2 Scaling performance for the two best performing algorithms ORCA and ANOM. The $x-$axis represents the number of features (m) with $n = 100$ for solid graphs and the number of samples (n) with $m = 100$ for dashed graphs. ORCA is significantly faster than ANOM and scales gracefully according to the expected complexity bounds

5.6 Scaling experiments

We run scaling experiments to highlight the runtime differences between ANOM and ORCA as they were the best performing methods for outlier detection and warrant further study. For these experiments we fix either the number of features (m) or number of samples (n) and vary the other dimension. The low rank parameter was fixed to 50. The stopping criteria for the methods were the same as used in the real world experiments (Sect. 5.4).

Figure 2a shows the total time needed for both algorithms to stop. We see that ORCA is consistently faster than ANOM achieving a maximum speedup of nearly a factor of 1000. Since both algorithms are solving different objective functions we measure time per iteration as another runtime characterization. We fix the number of iterations to 5. The stopping criteria for both these methods were primarily based on maximum iterations on the real-world experiments. Figure 2b shows that ORCA is faster than ANOM by a factor of 5 and a factor of 100, respectively. Figure 2b also shows the run times of ORCA grow more gradually than that of ANOM. Even when the inputs to ORCA are 50 times larger than that of ANOM, as shown in Fig. 3, the per iteration times of ORCA are lower than that of ANOM. Figure 3 also shows that ORCA scales better with respect to features than samples. This is in line with the complexity analysis shown in Sect. 4.2. The slowness of ANOM is also expected since it performs k matrix inversions costing $\mathcal{O}\left(mn^2\right)$ flops every outer iteration which is extremely expensive. The lack of an explicit low rank term in their formulation leads to this computational bottleneck.

5.7 Case study

We conduct a case study to illustrate how ORCA operates on an attributed graph. The B68 group of the US Patents dataset [13] is considered, on which ORCA performed well as seen in Tables 3, 4, 5 and 6. This group contains patents pertaining to methods for manufacturing articles from leather such as harnesses and saddles. The patents are divided in to 118 subgroups (for example making upholstery, stirrups, whips, saddles, etc.). Each patent is a technical document that cites other patents. We process the set as described in Sect. 5.1 and

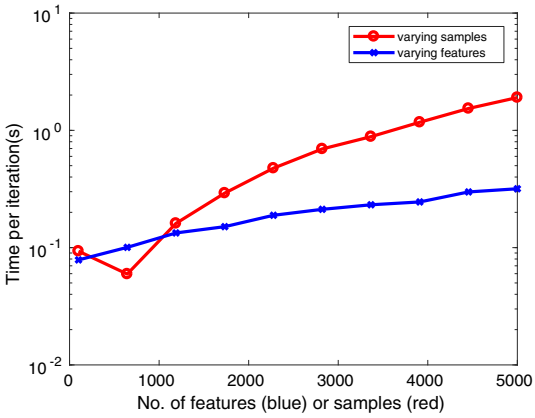


Fig. 3 Scaling plots for ORCA on larger inputs. The x-axis represents the number of features (m) with $n = 100$ for blue graphs and the number samples (n) with $m = 100$ for red graphs. ORCA scales better with respect to features than samples. (Color figure online)

Table 7 Topics present in the B68 group

Topic	Topic top words	Discovered topic
1	Pad, layer, saddle, back, panel, area, cushion	Pad, layer, saddle, panel back, area, cushion
2	Coil, spring, pocket, fabric, row, weld, apparatus	Coil, spring, pocket, fabric, row, weld, string
3	Cinch, girth, buckle, strap, saddle, belt, fastening	Cinch, girth, buckle, strap, saddle, belt, fastening
4	Halter, nose, loop, animal, cheek, strap, lead	Nose, halter, loop, cheek, strap, head, animal
5	Bit, mouthpiece, shank, mouth, cheek, bridle, ring	mouthpiece, bit, shank, bridle, cheek, mouth, ring
6	Seat, cover, cushion, frame, die, prong, machine	Seat, cover, cushion, machine, frame, support, trim
7	Bit, bridle, head, horse, strap, rein, ring	*Composite, lubricant, object, spherical, mixture, fluid, flow*
8	Fiber, aggregate, spherical, mixture, percent, pillow, filling	Fiber, aggregate, spherical, natural, insulation, bonded, filament
9	Rack, frame, saddle, arm, support, vertical, horizontal	Rack, frame, arm, saddle, support, vertical, horizontal
10	Saddle, horn, tree, Substrate, section, aperture, flap	*pillow, viscoelastic, m3, filler, nominal, kilogram, foam*

Each topic is represented by its seven top key words. The left column shows the ground truth topics and the right column shows the topics discovered by ORCA. Topics 7 and 10 found by ORCA are shown in italic as they are different from the ground truth topics

the final dataset includes 10 randomly selected subgroups, which we call topics, and 10 outliers from the remaining subgroups. The top keywords from each of these topics are shown in Table 7.

Table 7 display the top-7 keywords from the true topics and the topics discovered by ORCA. These are generated from the columns of the cluster representative matrix $\mathbf{W}$. ORCA

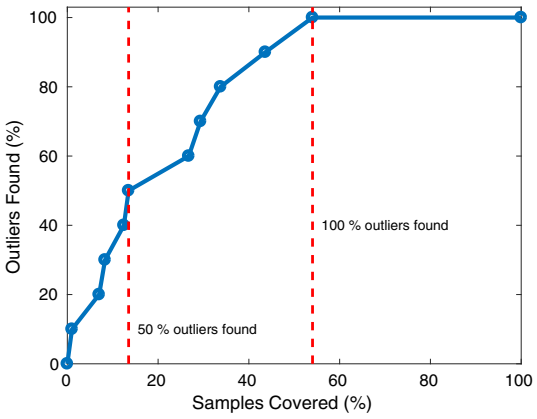


Fig. 4 Outlier ranking in the B68 dataset. Every marker is an outlier found. The samples are arranged in decreasing order of outlier scores and we plot the % of outliers captured. We would expect a perfect ranking algorithm to capture 100 % of the outliers by $\frac{\#\text{outliers}}{\#\text{samples}}$ % mark on the x-axis

Table 8 Example outliers and inliers in the B68 group

Example	Top words	ORCA Output	Neighbours
Outlier (88)	**Fire**, **conductive**, **upholster**, **substance**,	***Outlier***	**8**
	article, resistant, fibrous		
Inlier (304)	**Dee**, **latigo**, **feature**, **billet**,	*Outlier*	**3**
Topic 10	Holding, increment, enclosure	*Topic 9*	
Outlier (370)	**Gel**, stretch, deviation, plot,	*Inlier*	8, 8, 8, 8, 8
	individual, cushion, rebound	*Topic 1*	
Inlier (223)	Ply, retainer, receptacle, block,	***Inlier***	2, −1, 2, 2
Topic 2	Bedding, string, strings	***Topic 2***	

The example outputs for individual patents are italic if they are incorrectly classified as an outlier and bolditalic otherwise. Bold words and edges are the entries in $\mathbf{Z}_1$ and $\mathbf{Z}_2$ responsible for the outlier score of a patent

is able to detect 8 of 10 topics accurately which contributes to the good clustering performance in Tables 5 and 6. The two topics that were not discovered are highlighted in italic.

Next we look at the entries flagged as outliers by ORCA. Figure 4 displays the number of outliers found in descending order of outlier scores. Recall that outlier scores are generated by summing up entries of $\mathbf{Z}_1$ and $\mathbf{Z}_2$ in a columnwise manner. We can see that ORCA is quickly able to detect 5 of the 10 outlier nodes. These are found within the first 52 flagged items (13.5% of data items) but the last 5 are more difficult to capture. The final outlier is flagged only at the 208th position (54% of data items).

Finally we show an example result in Table 8. The Example column provides the index, true topic, and outlier labels for each patent. Top words are the most weighted words in the document vector. Bold words indicate that they were captured in $\mathbf{Z}_1$. The Neighbours column in Table 8 shows the topics of the neighbouring patents in the citation network. For example, data item 370 has connections with 5 other patents all of whom are from Topic 8. Outliers appear as Topic -1. Bold edges indicate that they were captured in $\mathbf{Z}_2$. We consider the patents with the 10 largest outlier scores as the outliers returned by ORCA.

We look at a representative example from each of the categories (true positive, false positive, false negative, and true negative) of flagged results. For the outlier (data item 88) the top words in the document do not seem to correlate with any topic and results in a high score from $\mathbf{Z}_1$ and similarly its connection to a patent from Topic 8 looks arbitrary and is

captured by $\mathbf{Z}_2$. ORCA is able to flag this sample as an outlier successfully. Data item 304 is incorrectly flagged as an outlier. It belongs to Topic 10 which was not discovered by ORCA. The top words look different from the Topic 10 and it has an edge to a patent from Topic 3, which explains some of the difficulty categorizing it. Data item 370 is one of the outliers that was flagged last by ORCA at position 168. The data item has a lot of connections to patents from Topic 8 and is matched to the erroneously discovered Topic 7 by ORCA. It is still considered partially an outlier due to the word "gel". Finally, data item 223 is categorized as an inlier and placed in the correct category by ORCA even though it has a connection to an outlier.

5.8 Discussion

The results from Sect. 5.5 show the importance using both connection and feature information for clustering and anomaly detection in attributed graphs, as well as simultaneous consideration of clustering and anomaly detection rather than in two stages. The best performing algorithms, ORCA, ANOM, and ALAD, effectively use both views of data. Methods that independently run on both types of inputs are not as effective in both clustering and anomaly detection. From the case study conducted on ORCA we are able to see that a poorly discovered or missed topic can cause ORCA to mislabel certain inliers even as we simultaneously optimize with both kinds of data. ORCA is effective in both clustering and anomaly detection, and its performance can be further improved by tuning hyperparameters λ_1 and λ_2 via grid searches. This tuning can become expensive.

ANOM [38], while slow, performs the best at anomaly detection. Instead of performing an exact CUR decomposition on the input they regularize their objective function to promote a CUR like internal representation [35]. This regularization is done by approximating $\mathbf{W} \approx \mathbf{CUR}$ and regularising $\mathbf{W}$ and $\mathbf{W}^T$ via the $\ell_{2,1}$ norm. We noticed that $\mathbf{W}$ is often close to full rank in our experiments. This full rank approximation seems to aid in anomaly detection and might be an interesting addition to our framework. However this flexibility causes the computational complexity to increase dramatically compared to algorithms that use an explicit low rank formulation.

6 Conclusions and future work

In this paper, we develop a general framework for clustering and anomaly detection in attributed graphs. We developed an efficient algorithm (ORCA) within this framework and used this for simultaneously mining clusters and anomalies in text datasets which have connection information. Experiments on real world datasets show that utilizing both attribute and connection modalities outperforms mining on individual inputs. ORCA extends simultaneous clustering and anomaly detection with better interpretability and scalability for end-user analysts. In the future, we would like to implement efficient distributed versions of these algorithms to analyse internet scale graphs. Modeling different norms for penalizing anomalies is also an interesting future extension of this work.

Acknowledgements This material is based in part upon work supported by the U.S. National Science Foundation (NSF) under Grant Nos. OAC-1642410, CCF-1533768, and OAC-1710371. This manuscript has been authored by UT-Battelle, LLC under Contract No. DE-AC05-00OR22725 with the U.S. Department of Energy. This research used resources of the Oak Ridge Leadership Computing Facility, which is a DOE Office of Sci-

ence User Facility. Any opinions, findings, conclusions, or recommendations expressed in this material are those of the authors and do not necessarily reflect the views of NSF or DOE.

References

1. Aggarwal, C.C.: Outlier analysis. In: Data Mining, pp. 237–263. Springer (2015). https://doi.org/10.1007/978-3-319-14142-8_8
2. Aggarwal, C.C.: An introduction to outlier analysis. In: Outlier Analysis, pp. 1–34. Springer (2017). https://doi.org/10.1007/978-3-319-47578-3_1
3. Akoglu, L., McGlohon, M., Faloutsos, C.: Oddball: spotting anomalies in weighted graphs. In: Advances in Knowledge Discovery and Data Mining, pp. 410–421 (2010). https://doi.org/10.1007/978-3-642-13672-6_40
4. Akoglu, L., Tong, H., Koutra, D.: Graph based anomaly detection and description: a survey. Data Min. Knowl. Discov. **29**(3), 626–688 (2015). https://doi.org/10.1007/s10618-014-0365-y
5. Bertsekas, D.P.: Nonlinear Programming. Athena Scientific, Belmont (1999)
6. Bouwmans, T., Sobral, A., Javed, S., Jung, S.K., Zahzah, E.H.: Decomposition into low-rank plus additive matrices for background/foreground separation: a review for a comparative evaluation with a large-scale dataset. Comput. Sci. Rev. **23**, 1–71 (2017). https://doi.org/10.1016/j.cosrev.2016.11.001
7. Candès, E.J., Li, X., Ma, Y., Wright, J.: Robust principal component analysis? J. ACM (JACM) **58**(3), 11 (2011). https://doi.org/10.1145/1970392.1970395
8. Candès, E.J., Recht, B.: Exact matrix completion via convex optimization. Found. Comput. Math. **9**(6), 717 (2009). https://doi.org/10.1007/s10208-009-9045-5
9. Chakrabarti, D., Faloutsos, C.: Graph mining: laws, generators, and algorithms. ACM Comput. Surv. (CSUR) **38**(1), 2-es (2006). https://doi.org/10.1145/1132952.1132954
10. Davis, J., Goadrich, M.: The relationship between precision-recall and roc curves. In: Proceedings of the 23rd International Conference on Machine Learning, pp. 233–240. ACM (2006). https://doi.org/10.1145/1143844.1143874
11. Dhillon, I.S., Guan, Y., Kulis, B.: Kernel k-means: spectral clustering and normalized cuts. In: Proceedings of the Tenth ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, pp. 551–556. ACM (2004). https://doi.org/10.1145/1014052.1014118
12. Drineas, P., Frieze, A., Kannan, R., Vempala, S., Vinay, V.: Clustering large graphs via the singular value decomposition. Mach. Learn. **56**(1), 9–33 (2004). https://doi.org/10.1023/B:MACH.0000033113.59016.96
13. Du, R., Drake, B., Park, H.: Hybrid clustering based on content and connection structure using joint nonnegative matrix factorization. J. Glob. Optim. **74**, 861–877 (2017). https://doi.org/10.1007/s10898-017-0578-x
14. Du, R., Kuang, D., Drake, B., Park, H.: Hierarchical community detection via rank-2 symmetric nonnegative matrix factorization. Computat. Soc. Netw. **4**(1), 7 (2017). https://doi.org/10.1186/s40649-017-0043-5
15. Dunlavy, D.M., Kolda, T.G., Acar, E.: Temporal link prediction using matrix and tensor factorizations. ACM Trans. Knowl. Discov. Data (TKDD) **5**(2), 1–27 (2011). https://doi.org/10.1145/1921632.1921636
16. Fortunato, S.: Community detection in graphs. Phys. Rep. **486**(3–5), 75–174 (2010). https://doi.org/10.1016/j.physrep.2009.11.002
17. Gao, H., Chen, Y., Lee, K., Palsetia, D., Choudhary, A.N.: Towards online spam filtering in social networks. NDSS **12**, 1–16 (2012). https://doi.org/10.1109/ICDM.2011.124
18. Gao, J., Liang, F., Fan, W., Wang, C., Sun, Y., Han, J.: On community outliers and their efficient detection in information networks. In: Proceedings of the 16th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, pp. 813–822. ACM (2010). https://doi.org/10.1145/1835804.1835907
19. Henderson, K., Gallagher, B., Li, L., Akoglu, L., Eliassi-Rad, T., Tong, H., Faloutsos, C.: It's who you know: graph mining using recursive structural features. In: Proceedings of the 17th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, pp. 663–671. ACM (2011). https://doi.org/10.1145/2020408.2020512
20. Huber, P.J.: Robust Statistics, vol. 523. Wiley, Hoboken (2004). https://doi.org/10.1002/9780470434697
21. Kannan, R., Ballard, G., Park, H.: Mpi-faun: an mpi-based framework for alternating-updating nonnegative matrix factorization. IEEE Trans. Knowl. Data Eng. **30**(3), 544–558 (2018). https://doi.org/10.1109/TKDE.2017.2767592

22. Kannan, R., Woo, H., Aggarwal, C.C., Park, H.: Outlier detection for text data. In: Proceedings of the 2017 SIAM International Conference on Data Mining, pp. 489–497. SIAM (2017). https://doi.org/10.1137/1.9781611974973.55
23. Kim, J., He, Y., Park, H.: Algorithms for nonnegative matrix and tensor factorizations: a unified view based on block coordinate descent framework. J. Glob. Optim. **58**(2), 285–319 (2014). https://doi.org/10.1007/s10898-013-0035-4
24. Kim, J., Park, H.: Fast nonnegative matrix factorization: an active-set-like method and comparisons. SIAM J. Sci. Comput. **33**(6), 3261–3281 (2011). https://doi.org/10.1137/110821172
25. Kleinberg, J.M.: Authoritative sources in a hyperlinked environment. J. ACM (JACM) **46**(5), 604–632 (1999). https://doi.org/10.1145/324133.324140
26. Kuang, D., Yun, S., Park, H.: Symnmf: nonnegative low-rank approximation of a similarity matrix for graph clustering. J. Glob. Optim. **62**(3), 545–574 (2015). https://doi.org/10.1007/s10898-014-0247-2
27. Kumar, S., Hooi, B., Makhija, D., Kumar, M., Faloutsos, C., Subrahmanian, V.: Rev2: fraudulent user prediction in rating platforms. In: Proceedings of the Eleventh ACM International Conference on Web Search and Data Mining, pp. 333–341. ACM (2018). https://doi.org/10.1145/3159652.3159729
28. Lee, D.D., Seung, H.S.: (1999) Learning the parts of objects by non-negative matrix factorization. Nature **401**(6755), 788–791. https://doi.org/10.1038/44565
29. Lee, D.D., Seung, H.S.: Algorithms for non-negative matrix factorization. In: Advances in Neural Information Processing Systems, pp. 556–562 (2001). https://doi.org/10.5555/3008751.3008829
30. Leskovec, J., Lang, K.J., Dasgupta, A., Mahoney, M.W.: Statistical properties of community structure in large social and information networks. In: Proceedings of the 17th International Conference on World Wide Web, pp. 695–704. ACM (2008). https://doi.org/10.1145/1367497.1367591
31. Leskovec, J., Lang, K.J., Mahoney, M.: Empirical comparison of algorithms for network community detection. In: Proceedings of the 19th International Conference on World Wide Web, pp. 631–640. ACM (2010). https://doi.org/10.1145/1772690.1772755
32. Li, J., Dani, H., Hu, X., Liu, H.: Radar: Residual analysis for anomaly detection in attributed networks. In: IJCAI, pp. 2152–2158 (2017). https://doi.org/10.24963/ijcai.2017/299
33. Liu, N., Huang, X., Hu, X.: Accelerated local anomaly detection via resolving attributed networks. In: Proceedings of the 26th International Joint Conference on Artificial Intelligence (2017). https://doi.org/10.24963/ijcai.2017/325
34. Lu, Q., Getoor, L.: Link-based classification. In: Proceedings of the 20th International Conference on Machine Learning (ICML-03), pp. 496–503 (2003). https://doi.org/10.5555/3041838.3041901
35. Mahoney, M.W., Drineas, P.: Cur matrix decompositions for improved data analysis. Proc. Natl. Acad. Sci. **106**(3), 697–702 (2009). https://doi.org/10.1073/pnas.0803205106
36. McCallum, A.K., Nigam, K., Rennie, J., Seymore, K.: Automating the construction of internet portals with machine learning. Inf. Retrieval **3**(2), 127–163 (2000). https://doi.org/10.1023/A:1009953814988
37. Muller, E., Sánchez, P.I., Mulle, Y., Bohm, K.: Ranking outlier nodes in subspaces of attributed graphs. In: 2013 IEEE 29th International Conference on Data Engineering Workshops (ICDEW), pp. 216–222. IEEE (2013). https://doi.org/10.1109/ICDEW.2013.6547453
38. Peng, Z., Luo, M., Li, J., Liu, H., Zheng, Q.: Anomalous: a joint modeling approach for anomaly detection on attributed networks. In: IJCAI, pp. 3513–3519 (2018). https://doi.org/10.5555/3304222.3304256
39. Pfeiffer III, J.J., Moreno, S., La Fond, T., Neville, J., Gallagher, B.: Attributed graph models: modeling network structure with correlated attributes. In: Proceedings of the 23rd International Conference on World Wide Web, pp. 831–842. ACM (2014). https://doi.org/10.1145/2566486.2567993
40. Revelle, M., Domeniconi, C., Sweeney, M., Johri, A.: Finding community topics and membership in graphs. In: Machine Learning and Knowledge Discovery in Databases. Lecture Notes in Computer Science, vol. 9285, pp. 625–640. Springer (2015). https://doi.org/10.1007/978-3-319-23525-7_38
41. She, Y., Owen, A.B.: Outlier detection using nonconvex penalized regression. J. Am. Stat. Assoc. **106**(494), 626–639 (2011). https://doi.org/10.1198/jasa.2011.tm10390
42. Tong, H., Lin, C.Y.: Non-negative residual matrix factorization with application to graph anomaly detection. In: Proceedings of the 2011 SIAM International Conference on Data Mining, pp. 143–153. SIAM (2011). https://doi.org/10.1137/1.9781611972818.13
43. Von Luxburg, U.: A tutorial on spectral clustering. Stat. Comput. **17**(4), 395–416 (2007). https://doi.org/10.1007/s11222-007-9033-z
44. Wang, G., Xie, S., Liu, B., Philip, S.Y.: Review graph based online store review spammer detection. In: 2011 IEEE 11th International Conference on Data Mining (ICDM), pp. 1242–1247. IEEE (2011)
45. Whang, J.J., Du, R., Jung, S., Lee, G., Drake, B., Liu, Q., Kang, S., Park, H.: Mega: multi-view semi-supervised clustering of hypergraphs. Proc. VLDB Endowment **13**(5), 698–711 (2020). https://doi.org/10.14778/3377369.3377378

46. Wright, J., Ganesh, A., Rao, S., Peng, Y., Ma, Y.: Robust principal component analysis: exact recovery of corrupted low-rank matrices via convex optimization. In: Advances in Neural Information Processing Systems, pp. 2080–2088 (2009). https://doi.org/10.5555/2984093.2984326
47. Xu, H., Caramanis, C., Sanghavi, S.: Robust PCA via outlier pursuit. In: Advances in Neural Information Processing Systems, pp. 2496–2504 (2010). https://doi.org/10.5555/2997046.2997174
48. Yu, R., He, X., Liu, Y.: Glad: group anomaly detection in social media analysis. ACM Trans. Knowl. Discov. Data (TKDD) **10**(2), 18 (2015). https://doi.org/10.1145/2811268