ANOMALY DETECTION IN NETWORK USING NON-NEGATIVE MATRIX FACTORIZATION

By

YUNYI KANG

A thesis submitted to the

Graduate School-New Brunswick

Rutgers, the State University of New Jersey

In partial fulfillment of the requirements

For the degree of

Master of Science

Graduate Program in Industrial and Systems Engineering

Written under the direction of

Prof. Myong K. Jeong

And approved by

____

____

____

____

New Brunswick, New Jersey

October 2015

ABSTRACT OF THE THESIS

Anomaly Detection in Network Using Non-negative Matrix Factorization Techniques

By YUNYI KANG

Thesis Advisor: Prof. Myong K. Jeong

Anomaly detection is becoming an important problem in graph mining. This is because people are eager to find out unusual objects or patterns in a network, which may results in possible damages, emerging trends, or even creations in different types of graphs or networks. For example, the transaction occurs in out-of- hometown area with high amount may indicate credit card fraud in a bank transaction network; a substantial high frequency of connectivity in the network may infer possible web attack. The discoveries of these activities are important for people's life and personal information security, and thus are important tasks for relevant institutions. There are mainly three kinds of anomalies in a graph data: node, edge and subgraph anomaly. This thesis presents new graph anomaly detection algorithms that provide scoring functions on detecting both node anomalies and edge anomalies based on non-negative matrix factorization techniques. The experiments on real life data verify that the suggested method could provide better performance in finding meaningful anomalies.

Keywords: Anomaly detection, Graph data, Network, Nonnegative matrix factorization

Acknowledgements

I would like to show my deepest appreciation to Professor Myong.K.Jeong for his continuous encouragement and support on my studies. He encouraged me to generate new ideas and raise constructive feedbacks; he taught me how to improve self-image and set up long term goals; he wrote recommendation letters for me and introduced me to other great researchers. I would like to thank my thesis committee members, Dr. Kang Li and Dr. Honggang Wang. My thesis could not be finished without their inspiring research discussions and suggestions.

I am also grateful to the peers in the research lab- Jeongsub Choi and Ali Tosyali, and the seniors in my research group- Dr. Byung H. Kim and Dr. Andrew Rodriguez. I could not finish the majority of my thesis without the wonderful collaborations. I would also want to thank the former member in the research team, Luquan Li and Gianluca Gazzola, and all the other professors and friends, for helping me in courses and life, and for witnessing my growth.

Most of all, I am grateful to all my family members. I feel very lucky to receive unswerving support at any time for the past twenty three years.

Table of Contents

List of Tables

List of Figures

1. Introduction

Graphs are serving as powerful tools in many situations, containing a group of objects and describing the interdependent relationship or connectivity among the elements in the graphs visually. Therefore, graphic data appears widely in areas such as social networks, brain connectivity graphs, transportation networks, the Internet and other fields. On one hand, the graph data shows explicitly how components are connected and formed, i.e. product flows are shown clearly in the transportation network. On the other hand, with the help of data mining techniques, implicit patterns could also be found to have better understanding of the behaviors in the graphic data, i.e. the connectivity between different nodes could infer how the level of education could influence on friends making in social networks. In the recent years, many research work on mining patterns from graph data (Zou, Li, Gao, & Zhang, 2010; Washio & Motoda, 2003).

Apart from finding frequent patterns and trends, such as clustering, link prediction and frequent data mining, anomaly detection (also called as outlier detection) has become important, aiming at finding abnormal behaviors or irregular information. The anomalous data, according to Hawkins (1980), refers to observations “that deviate so much from other observations as to arouse suspicion that it was generated by a different mechanism”. However, anomaly is only a general concept and no standard concepts or consensus have been reached on defining anomalies. Instead, researchers define anomalies based on the problems they study. Specifically, in graphic data, anomalies could be reflected from the irregular structure compared with other nodes. For example, a new employee in an organization’s human resource network could be treated as an anomaly because she knows a few people in that organization while the edge of such network is built on whether two people in the network get recognized with each other. Generally speaking, people may pick up different anomalies from a graph based on specific searching targets in different situations.

Detecting anomalies in a network becomes very important because it could discover hidden information from graphs, which may contribute to better decision making. For example, the transaction occurs in a strange area with high amount may indicate credit card fraud in a bank transaction network; a substantial high frequency of connectivity in the network may infer possible web attack. The discoveries of these activities are important for people’s life and personal information security, and thus could be applied to some institutions. Moreover, some people would like to find some irregularities from the graph. Such task is called anomaly detection. Detecting graphic anomalies can be applied in many different types of network, such as auction networks (Chau, Pandit, & Faloutsos, 2006), social networks (Gao, Chen, Lee, Palsetia, & Choudhary, 2012), accounting networks (McGlohon, Bay, Anderle, Steier, & Faloutsos, 2009), opinion networks (Xie, Wang, Lin, & Yu, 2012), etc.

Many researchers have successfully developed methods to detect the node anomalies in given graphs; however, seldom have they found effective methods on detecting edge anomalies. Moreover, it is also very difficult for researchers to suggest both node anomaly and edge anomaly detection methods thoroughly and provide meaningful interpretations.

Therefore, in this paper, we introduce new node and edge anomaly detection algorithms using matrix factorization techniques. We will factorize the node adjacency matrix and the edge-node incident matrix generated from the given graph separately in different methods, find relevant subgraphs and weight nodes/edges. By doing so, we could score the nodes or edges, which will help evaluate the likelihood of given nodes or edges compared with others.

The contribution of our work is that we introduce node anomalies scoring function using nonnegative matrix factorization approach in directed graph. Besides, we introduce scoring functions to detect edge anomalies. Our result enables powerful interpretations in real life data set compared to existing anomaly detection methods.

The structure of our work is as the following. Section 1 is the introduction; Section 2 will review the relevant literatures on graphic data anomaly detections considering graph structures; Section 3 will be an introduction to matrix factorization and non-negative matrix factorization; the main methods for edge anomalies will be discussed in Section 4 with examples; Section 5 contains the novel scoring functions and algorithms for node anomalies in directed graph; several experiments, comparisons and discussions will be made in Section 6; finally, in Section 7, conclusion will be reached.

2. Literature Review on Anomaly Detection in Graph Data

The graph anomaly detection problem (GADP) could be divided in many ways with different understandings by researchers. Considering the attributes of graph structures, GADP could be categorized into three parts: node anomaly detection problems, edge anomaly detection, and subgraph anomaly detection; GADP could also be divided into plain graph anomaly detections and attributed graph anomaly detection by considering whether attributes of the nodes are given or not; moreover, if we consider time series, the problem will become dynamic while on the other hand it will be static anomaly detections. Therefore, techniques are developed to meet with the needs under different requirements.

Some researchers have made contributions in doing overall surveys on anomaly detections topics (Aggarwal, 2013; Chandola, Banerjee, & Kumar, 2009; Hodge & Austin, 2004; Akoglu, Tong, & Koutra, 2014). In this thesis, we focus on the literatures related to the topics related to anomaly detections from the graphic data considering only graphic structures. This section consists of three parts: reviewing topics related to node anomaly detection problems, edge anomaly detection, and subgraph anomaly detection.

2.1 Node Anomaly Detection

Different literatures have different definitions on node anomalies. Generally speaking, if nodes are dissimilar from their neighbors, such nodes are likely to be anomalies. The neighbors are not limited to the adjacent nodes; they can be one-step, two steps or multi-steps from given nodes according to different definitions. The dissimilarities could be the differences of occurrence in probabilistic graphs, the distinctions in structural features, or the variance in predefined proximity measures predefined.

2.1.1 Probabilistic Distribution Approach

This approach could be treated as transformations between connectivity and proper distributions so that the nodes with the different structures could be picked out since they have different probabilistic values. Brin and Page (1998) proposed one method of constructing a distribution through random walk. They built up directed graphs showing the directory of webpages: the webpages are nodes and a directed edge exists between nodes *u* and *v* if webpage *u* has a forward link to webpage *v*. The walk is performed randomly and in each step, the probability of jumping from one node to one of its adjacent nodes equals to the inverse value of the node's degree. The stable outcome, or the distribution generated, could reflect the influence among all the nodes. Brin and Page's idea has also been revised by later researchers by considering the different probabilities of jumping and specific requirement in real needs (Broder, et al., 2000; Takács, Pilászy, Németh, & Tikk, 2008; Haveliwala, 2003).

2.1.2 Feature-based Approaches

The intuitive of this approach is simple: find the rules that most nodes follow and filter out the nodes violating the rules.

Many existing researches related to graph theory provided helpful definitions and terms in describing the level of node connectivity in given graphs, which could also be used as measures on node anomaly detection. For example, betweeness of a node is used to describe the number of all paths passing through the node in graph theory (Lewis, 2011). In anomaly detection techniques, the nodes with lower betweeness value are more likely to be anomalies (Freeman, 1997). Similarly, radius (Kang, Tsourakakis, Appel, Faloutsos, & Leskovec, 2011), eigenvectors (Bonacich & Lloyd, 2001), and other concepts (Chandola, Banerjee, & Kumar, 2009; Akoglu, Tong, & Koutra, 2014) could also be found in literatures related to anomaly detections.

Besides, some researchers have defined new structure features to improve anomaly detection performance. Akoglu et al. (2010) developed the algorithm Oddball with a egonet which refers to the expanded subgraph with a node and the neighboring nodes of the corresponding node. Then patterns are selected based on the features that most of the egonets follow and anomalies are the nodes that do not follow the observed patterns.

The others studied on how to extract better features by considering both the node-based (local) features and the egonet-based features. Henderson et al. (2011) created features named recursive structural features through the means and sums of node-based features and egonet features. The collection of recursive structural features and local features are called regional features, through which the information of more nodes other than the neighbors are considered.

2.1.3 Proximity-based Models

The main idea in proximity-based methods is to regard anomalies as points which could not be grouped with other nodes. This type of models perform through either *Clustering method* or *Density-based measurements*.

Clustering method (also refers to *community-based method*) finds anomalies through grouping similar nodes into clusters (or communities). The anomalies come from nodes that do not belong to any cluster. For example, Xu et al. (2007, August) proposed an algorithm, SCAN, with a similarity measure which is calculated as the percentage of common neighbors. For any node *u*, *v* in the node set *V* and the edge set *E*, the number of neighbors node *u* has is defined as $\Gamma(u) = \{\, w \in V | (u, v) \in \mathrm{E}\} \cup \{\mathrm{u}\}$. Then the number of common neighbors two nodes is $|\Gamma(u) \cap \Gamma(v)|$. Then the structural similarity for two nodes is measured as:

$$\sigma(u, v) = \frac{|\Gamma(u) \cap \Gamma(v)|}{\sqrt{|\Gamma(u)||\Gamma(v)|}}$$

If the adjacent nodes meet the predetermined requirement of similarity score, the number of common neighbors is grouped into clusters. Finally, nodes could not be grouped into any cluster and connect to only one other clusters are anomalies.

Sun et al. (2010, December) applied SCAN (Xu et al, 2007) to weighted graph. Instead of using the number of common neighbors, they use the weight of edges between common neighbors to measure the similarity between two nodes.

It is also well-noticed that some researchers detect the anomalies with the algorithms finding clusters first (Akoglu, Tong, & Koutra, 2014). Therefore, the quality of the clusters found could have significance on the quality of anomalies detected. Meanwhile, the other researchers use nonnegative matrix factorization to find communities in graphs and following the similar idea for node anomaly detection, thus creating algorithms called CDNMF (Cao, Wang, Jin, Cao, & He, 2013).

Density-based measurements are used to measure how similar nodes are with each other, and the resulted scores could be ranked to compare such closeness. For one node, if it is "closer" to similar neighbors than others, then it will have high closeness score and is less likely to be an anomaly. Such measurement could be found in (Jeh & Widom, 2002, July), in which they introduced a measure named Simrank, representing the similarity score of a node by adding up the similarity score of all its adjacent nodes and updating through iteration processes. For example, with two nodes *a* and *b* in the given graph, the similarity between these two nodes is defined as:

$$s(a,b) = \frac{C}{I(a)|I(b)|} \sum_{i=1}^{|I(a)|} \sum_{j=1}^{|I(b)|} s(I_i(a), I_j(b)),$$

where $I_i(a)$ and $I_j(b)$ belongs to set *I(a)* and *I(b)*, representing in-neighbors of node a and b respectively, and C is a constant. Initially, *s(a,b)* = 1 if and only if *a* = *b*. Thus the iteration process to calculate the Simrank score for the pair of nodes a and *b*, with the notation of *R(a,b)*, expressed as:

$$R_{k+1}(a,b) = \frac{C}{I(a)|I(b)|} \sum_{i=1}^{|I(a)|} \sum_{j=1}^{|I(b)|} R_k(I_i(a), I_j(b)),$$

with

$$R_0(a,b) = \begin{cases} 1 & when\ a = b \\ 0 & otherwise \end{cases}$$

Such measure then are updated by (Antonellis, Molina, & Chang, 2008) and (Zhao, Han, & Sun, 2009), discussing the weight of in degree nodes and out degree nodes.

2.2 Edge Anomaly Detection

Edges in a given graph could be anomalies. To find the anomalies, two methods could be used, either revising the approaches from node anomaly detections, or using the matrix factorization method.

2.2.1 Intuitive Approaches from Node Anomaly Detections

Some methods applied in node anomaly detection could also be used in detecting edge anomalies, like ODDBALL (Akoglu, McGlohon, & Faloutsos, 2010) and Simrank (Jeh & Widom, 2002, July). In their algorithms, the measure of each edge is represented as the measure of similarity between the nodes on the two sides of that edge. Therefore, the edge could be ranked and compared following the approaches in the node anomaly detection.

Besides, some researchers gave their own understanding on edges anomalies, and specified the edges with the help of found node anomalies or clusters. One way to interpret is that the edges connecting anomalous nodes are treated as edge anomalies. For example, if two nodes are identified as anomalies, then the edge connecting them could be anomaly edge; if the edges connects anomaly parts comparted with regular patterns, they are more likely to be anomaly edges (Aggarwal, Zhao, & Yu, 2011, April; Noble & Cook, 2003); if an edge connects two clusters or one cluster and an anomalous node, such edge could be treated as an anomaly.

2.2.2 Matrix Factorization/Spectral Methods

Tong and Lin (2011) introduced nonnegative residual matrix factorization method to detect link anomalies in a bipartite graph. According to their approach, the residual matrix shows the deviations of low rank factorization from the original matrix. Thus they require the entries of the residual matrix to be nonzero in their matrix factorization algorithm, giving a meaningful and intuitive view on the significance of the deviation. They argued that the edges having large value in the residual matrix are more likely to be anomalies.

Aggarwal (2013) introduced a spectral method to complement the matrix factorization method. The matrix node-link adjacency matrix could be augmented into a positive semi-definite matrix, which could be decomposed further using the singular value decomposition methods. The author then follows the residual matrix idea introduced above and finds the anomaly links.

2.3 Sub-graph/Query-based/Personalized Anomaly Detection

Anomalous subgraphs, according to Aggarwal (2013), are the special or unusual subsets or partitions in the graph. One intuitive thinking with the review in Section 2.1 and 2.2 is that, if anomaly nodes or anomaly edges could be found, then the connected subgraphs containing those anomalies could be the desired the subgraph.

Besides, under one definition of anomalous subgraphs, if a large graph could be partitioned into different subgraphs, among which several patterns could be found with high frequency of occurrence; then the subgraphs with low frequency of occurrence or with patterns other than the selected ones are the candidate of anomaly subgraphs. For example, (Noble & Cook, 2003) created a formula to score the subgraphs by considering the large frequently repeated patterns with the concept Minimum description length (MLD). They finalized the formula with two components, the pattern frequency described by MLD and the nodes appearance frequency in that pattern. The subgraphs with low scores tend to be anomalies, because either the pattern is not frequently reoccurred, or the nodes are not appeared in the same pattern. The method introduced (Noble & Cook, 2003) are helpful in the information network or even dynamic network, where repeated pattern usually occurred.

Anomaly subgraphs could also be customized to meet with users' requirement. Thus, in anomaly detections area, researchers are building algorithms including users' concerns. Gulpta (Gupta, 2013) used query-based techniques to find anomalies defined as Association-Based Clique Outliers (ABCOutliers), which are "rare and surprising associations" in the information network. They found the anomaly cliques according to predefined cliques provided by the customers.

3. Matrix Factorization

In this section, we will introduce background knowledge of matrix factorization. Section 3.1 provides an overview of matrix factorization and its applications. Section 3.2 will focus on non-negative matrix factorization.

3.1 Matrix Factorization

Matrix factorization, or matrix decomposition, is a term from the field of linear algebra. An intuitive understanding of this term is that the original matrix is factorized into two or more matrixes so that the multiplications of factorized matrixes equals to the original matrix. Common types of matrix factorizations include QR factorization, LU factorization, rank factorization, Cholesky factorization, and SVD (Singular Value Decomposition) factorization/eigenvalue decomposition.

One of the most popular matrix factorization methods applied in the literature is the SVD method and/or with PCA. The general expression of SVD is as following. A given q-by-p matrix $\mathbf{Q}$ could be factorized as:

$$\mathbf{Q} = \mathbf{FMR}^{\mathbf{T}} \tag{1}$$

where $\mathbf{U}$ is a $q \times p$ orthogonal matrix ($\mathbf{F}^{\mathrm{T}}\mathbf{F} = \mathbf{I}$), $\mathbf{M}$ is a $p \times p$ diagonal matrix, and $\mathbf{R}$ is a $q \times p$ orthogonal matrix ($\mathbf{R}^{\mathrm{T}}\mathbf{R} = \mathbf{I}$). The diagonal entries of the matrix $\mathbf{M}$ (or M_{ii} for the integer l in [1, p]) are called the singular values for matrix $\mathbf{Y}$. If smaller singular values are removed, leaving the major proportion of the singular values, then lower rank matrix could receive.

The principle component analysis (PCA) uses such idea so that the quality of data representation and the error in misrepresentation are both well-traded.

From the perspective of application, matrix factorization is used to explore the latent features through combining different types of entities. The reason is that matrix factorization is a process of decomposing a matrix into different matrix, indicating linear combinations of entries in different matrices with different representations. For example, the expression in (1) indicates that every element of **Q** is reached from the linear combinations of rows or columns in matrices **F**, **M** and **R**. If let **Q** be an image matrix, **F** be the matrix of different patterns, **M** be a weight matrix, and **R** be the matrix of colors; then the image matrix **Y** could be expressed as combinations of patterns with colors under different weights. Such idea is well applied into the area such as recommender systems (Koren, Bell, & Volinsky, 2009), dictionary Learning (Mairal, Bach, Ponce, & Sapiro, 2010), and Collaborative Prediction (Rennie & Srebro, 2005).

Furthermore, for the situation in which more than two kinds of entities are considered, tensor factorization are applied to reach more complicated while more adaptable matrix factorization (Cichocki, Zdunek, Phan, & Amari, 2009).

3.2 Non-negative Matrix Factorization

Non-negative matrix factorization (NMF) is a specific factorization method under matrix decomposition. A matrix is factorized into usually two lower rank matrices, with the property that the factorized matrices contain no negative elements. One purpose of NMF is to find good approximation to the original matrix (Cai, He, Wu, & Han, 2008).

However, the problem could be difficult to solve and find the global minimum. Lee and Seung (2001) have suggested an iterative algorithm and proven that their approach could find a local optimal value. Their update process have been proved to be non-increasing iterative rules for the Euclidian distance**,** which will turns out local optimal solutions.

The NMF method is in contrast to other algorithms, such as vector quantization (VQ) and principal component analysis (PCA), because NMF enables parts-based representation, not subtractive combinations. Moreover, one advantage of NMF method is that the result it returns is intuitive and thus feasible in interpretation. Since the observations and quantified features in many applications are non-negative, the factorized matrices could be compatible to relevant features or combination of different patterns through linear combinations. The latent features or patterns could be given meanings with regards to the original matrices and the research studying.

Therefore, NMF could be applied into many real world problems such as face analysis (Wang, Jia, Hu, & Turk, 2005), computer vision (Lee & Seung, 1999), document clustering (Kuang & Park, 2013), graph role discovery (2013), and DNA gene expression analysis (Wang, Wang, & Gao, 2013).

Apart from the general expression of the NMF, researchers have been building up different types of matrix factorization methods by adding constraints on the factorized matrices. Symmetric NMF requires that the factorization results should be the multiplication of one matrix and its transpose, thus showing advantage in factorizing symmetric matrices (Cao, Wang, Jin, Cao, & He, 2013); sparse NMF adds regularizations to the objection function, limiting the number of non-zero entries in factorized graph (Kim & Park, 2008); asymmetric nonnegative matrix factorization (ANMF) method has been developed , which shows possible applications in handling the graph problems (Wang, Li, Wang, X., & Ding, 2011). One main purpose of the modification on NMF is to match the characters of certain matrix factorization methods with the characters of the dataset so that useful patterns could be found through matrix factorization.

4. Edge Anomaly Detection Using Non-negative Matrix Factorization

In this section we will introduce our proposed edge anomaly detection techniques. Section 4.1 will introduce the notations and concepts used in designing the new methods, Section 4.2 will focus on scoring node anomaly with non-negative matrix factorization. Section 4.3 will be examples.

4.1 Definitions and Notations

In this thesis, we study static graph G with a set of nodes V and a set of edges E, written as $G = (V; E)$. We use n and m to represent the sizes of V and E respectively, that is, $n = |V|$ and $m = |E|$. There is no requirement on whether the graph is connected or not.

The graph in section is undirected. Besides, the algorithm could also be applied to the graph that are weighted. Weighted graph could be formed to count multiple numbers of connections between two nodes (i.e. the times of emails sending between two people) or using predefined measurements to represent the attributes of nodes on each side of an edge (i.e. the percentage of common friends of two persons in a network).

To describe the graph, we use node-edge incident matrix with notation **B**. Nodes are represented in rows and edges are represented in columns, and the entries of matrix **B** indicate the nodes that an edge connecting to. For unweighted graphs, all entries are either 0 or 1, with 1's in the entries indicating the existence of an edge. For weighted graphs, the entries in the adjacency matrix are the weights.

Besides, there are other symbols used in the matrix factorization process, and we summarize the symbols in *Table 4.1*.

Table 4.1 Table of symbols used in notation

Symbol	**Description**
G	Graph representation of the given datasets
V, n	set of nodes, number of nodes
E, m	set of edges, number of edges
B	n x m node-edge incident matrix
W,H,D	Lower rank factorized matrix used in NMF
H*	Matrix for detecting anomalies
r	Factorization rank
AE(o)	Anomaly score for edge e

4.2 New Scoring Functions for Edge Anomaly Detection

In this section, we propose both node and edge anomaly detection methods through new non-negative matrix factorization approaches, which could provide more stable results. First, further decomposed each of the two factorized matrices using the NMF approaches introduced in Section 3 so that every factorized matrix could be expresses into two new matrices- one normalized matrix and the other diagonal matrix. Then we leave along only the first normalized matrix and multiple the remaining together. The multiplication results could be helpful in determining anomalous scores for edges. In this section, we desire to define edge anomaly score function and pick out anomaly edges through the given node-edge incident matrix **B**.

First, using the NMF idea, our problem could be formulated using as

Given matrix **B** ($m \times n$ dimension), factorization rank *r*

Minimize Euclidean distance $\|\mathbf{B} - \mathbf{WH}\|_{\mathbf{F}}^{\mathbf{2}}$

such that all entries in **W**and **H** are non-negative

Find matrix **W** and **H**

Applying the algorithm proposed by Lee (1999), we could get two factorized matrices, **W** and **H** such that $\mathbf{B} \approx \mathbf{W} * \mathbf{H}$. The integer r meets the requirement that $0 \leq r \leq \min(m, n)$. Therefore, matrix **B** is decomposed into two low-dimension matrix.

Second, we will normalize the **W** matrix in column-wise. To do this, we need to find matrix $\mathbf{W}^{(\mathbf{1})}$ and $\mathbf{D_1}$, satisfying:

(1) $\mathbf{W} = \mathbf{W}^{(\mathbf{1})} \cdot \mathbf{D_1},$

(2) $\mathbf{D_1}$ is a diagonal matrix, and

(3) $\sum_{k=1}^{n} W_{kj}^{(1)} = 1$, for any integer j in the range [0, r].

The diagonal matrix $\boldsymbol{D_1}$ then could also be treated as weight for every column of the matrix $\mathbf{W}^{(\mathbf{1})}$. Similarly, the **H** matrix will be normalized in row-wise. We also need to find matrix $\mathbf{D_2}$ and $\mathbf{H}^{(\mathbf{1})}$ satisfying:

(1) $\mathbf{H} = \mathbf{H}^{(\mathbf{1})} \cdot \mathbf{D_2}$,

(2) $\mathbf{D_2}$ is a diagonal matrix, and

(3) $\sum_{k=1}^{m} H_{jk}^{(1)} = 1$, for any integer j in the range [0, r].

The diagonal matrix $\mathbf{D_2}$ could also be treated as weight for every row of the matrix $\mathbf{H^{(1)}}$. Therefore, matrix $\mathbf{W}$ and $\mathbf{H}$ are factorized into two new matrices respectively, and matrix **B** could be expressed as multiplication of four matrices:

$$\mathbf{B} \approx \mathbf{W^{(1)}} \cdot \mathbf{D_1} \cdot \mathbf{D_2} \cdot \mathbf{H^{(1)}}$$

Third, we define a matrix $H^*_{r\times m}$, which could be expressed as:

$$\mathbf{H}^* = \mathbf{D_1} \cdot \mathbf{D_2} \cdot \mathbf{H^{(1)}}$$

Therefore, we split the **B** matrix into normalized matrix and weight matrix.

Next, we find the anomaly score function for edges in the graph. For each edge *e*, in **B** matrix, it is represented by the number in a certain column, say column *k*. Then the anomaly score equals to the summation of the value in the *kth* column of matrix $\mathbf{H}^*$, or $\mathrm{AE(e)} = \sum_{\mathrm{k=1}}^{\mathrm{r}} \mathrm{H}^*_{\mathrm{rk}}$.

Finally, after finding the $\mathrm{AE(e)}$ value for every edge, we rank the value from low to high. The edges with lower AE(e) scores are more likely to be anomalies because they tend to have lower weights.

The summary of all the steps are shown in the following table (*Table 4.2*):

Table 4.2 Algorithms for detecting edge anomaly

Given:
Node-edge incident matrix **B**, factorization rank *r*

Do:

(1) Use the NMF algorithm to get factorized matrices, $\mathbf{W}$ and $\mathbf{H}$**,** two rank *r* matrices.

(2) Normalize $\mathbf{W}$ to get $\mathbf{W}^{(1)} \cdot \mathbf{D_1}$, where $\mathbf{W}^{(1)}$ is a column-wised normalized matrix and $\mathbf{D_1}$ is a diagonal matrix;

Normalize $\mathbf{H}$ to get $\mathbf{D_2} \cdot \mathbf{H}^{(1)}$, where $\mathbf{H}^{(1)}$ is a row-wised normalized matrix and $\mathbf{D_2}$ is a diagonal matrix.

(3) Find $\mathbf{H}^*$ such that $\mathbf{H}^* = \mathbf{D_1} \cdot \mathbf{D_2} \cdot \mathbf{H}^{(1)}$.

(4) Calculate node anomalous score for node with $AE(e) = \sum_{\mathrm{k=1}}^{\mathrm{r}} \mathrm{H}^*_{\mathrm{rk}}$, where k refers to the column number which edge e is represented in $B_{n\times m}$

(5) Rank the AE(e) scores for all edges from low to high

Find:

Edges with low AE(e) scores are anomalies

Since both $\mathbf{D_1}$ and $\mathbf{D_2}$ are diagonal matrices and used to adjust the weight for the purpose of normalizing **W** and **H**; for interpretation purpose, we may use matrix **D** to represent $\mathbf{D_1} \cdot \mathbf{D_2}$. Thus, the original matrix **B** could be expressed as:

$$\mathbf{B} \approx \mathbf{W}^{(1)} \cdot \mathbf{D_1} \cdot \mathbf{D_2} \cdot \mathbf{H}^{(1)}$$

$$= \boldsymbol{W}^{(1)} \cdot \boldsymbol{D} \cdot \boldsymbol{H}^{(1)}$$

$$= \sum_{k=1}^{r} \widetilde{\boldsymbol{w}}_k \cdot \mathrm{d}_k \cdot \tilde{\mathbf{h}}_k^{\mathrm{T}}$$

where, $\widetilde{\boldsymbol{w}}_k$ is the k^{th} column vector in the matrix $\mathbf{W}$, $\tilde{\mathbf{h}}_k$ is the k^{th} row vector in the matrix **H,** and d_{kk} is the k^{th} diagonal term in matrix **D.**

It can be found that we decompose the incident matrix into three component, weight of nodes in the graph (normalized $\mathbf{W}^{(1)}$), weight of edges in the graph (normalized $\mathbf{H}^{(1)}$), and the coefficient matrix for balancing the equation (diagonal matrix $\mathbf{D}$).

4.3 An Illustrative Example

A numerical example is given to illustrate how the method works. Suppose we are given the following graph (*Figure 4.1*) and the task is to find edge anomalies.

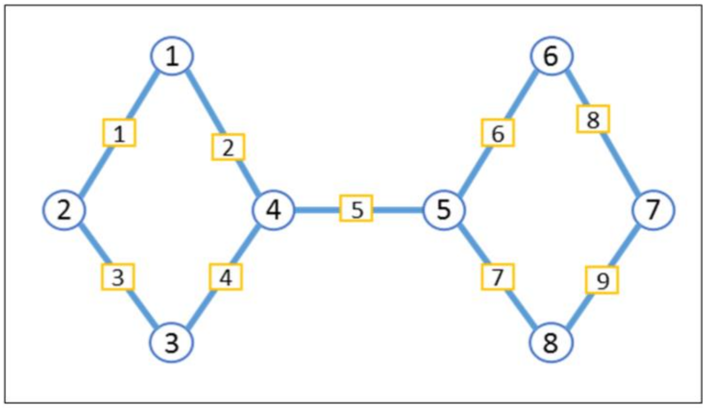

Figure 4.1 Graph example to illustrate the algorithms

Suppose this is a weighted graph, meaning that weight will be given to the edges and the node edge incidence matrix is represented as:

$$\mathbf{B} = \begin{bmatrix} 5 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 5 & 0 & 2 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 2 & 3 & 0 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 3 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 4 & 4 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 4 & 0 & 5 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 5 & 5 \\ 0 & 0 & 0 & 0 & 0 & 0 & 4 & 0 & 5 \end{bmatrix}$$

To factorize the matrix, the variable *r*, the factorization rank, should be determined ahead. The discussion on the using the appropriate *r* value will be discussed in section 5. In this example, to show comprehensive procedures, we will show four situations where *r* = 1, 2, 3, and 8 respectively.

Situation 1: when *r* = 1,

Matrix **B** could be factorized in the following way:

$$\mathbf{B} \approx \begin{bmatrix} .1020 \\ .0702 \\ .1020 \\ .2258 \\ .2258 \\ .1020 \\ .0702 \\ .1020 \end{bmatrix} \times [15.9778] \times \begin{bmatrix} .0702 \\ .1337 \\ .0702 \\ .1337 \\ .1842 \\ .1337 \\ .1337 \\ .0702 \\ .0702 \end{bmatrix}^{\mathrm{T}} = \begin{bmatrix} .1020 \\ .0702 \\ .1020 \\ .2258 \\ .2258 \\ .1020 \\ .0702 \\ .1020 \end{bmatrix} \times \begin{bmatrix} 1.1224 \\ 2.1362 \\ 1.1224 \\ 2.1362 \\ 2.9432 \\ 2.1362 \\ 2.1362 \\ 1.1224 \\ 1.1224 \end{bmatrix}^{\mathrm{T}} = \tilde{\mathbf{w}}_1 \cdot \mathrm{d}_1 \cdot \tilde{\mathbf{h}}_1^{\mathrm{T}}$$

Matrix **B**, based on the factorization result in above, could be treated as combining the weights of nodes and edges. When *r* = 1, the **B** matrix is divided into vectors and such division could be treated as linear combination of weights on nodes and edges.

Situation 2: when *r* = 2,

$$\mathbf{B} \approx \begin{bmatrix} .2215 & 0 \\ .1906 & 0 \\ .2215 & 0 \\ .3240 & .0425 \\ .0425 & .3240 \\ 0 & .2215 \\ 0 & .1906 \\ 0 & .2251 \end{bmatrix} \times \begin{bmatrix} 9.3035 & 0 \\ 0 & 9.3035 \end{bmatrix} \times \begin{bmatrix} .1836 & 0 \\ .2431 & 0 \\ .1836 & 0 \\ .2431 & 0 \\ .1466 & .1466 \\ 0 & .2431 \\ 0 & .2431 \\ 0 & .1836 \\ 0 & .1836 \end{bmatrix}^{\mathrm{T}}$$

$$= \begin{bmatrix} .2215 & 0 \\ .1906 & 0 \\ .2215 & 0 \\ .3240 & .0425 \\ .0425 & .3240 \\ 0 & .2215 \\ 0 & .1906 \\ 0 & .2251 \end{bmatrix} \times \begin{bmatrix} 1.7085 & 0 \\ 2.2614 & 0 \\ 1.7085 & 0 \\ 2.2614 & 0 \\ 1.3637 & 1.3637 \\ 0 & 2.2614 \\ 0 & 2.2614 \\ 0 & 1.7085 \\ 0 & 1.7085 \end{bmatrix}^{\mathrm{T}} = \tilde{\boldsymbol{w}}_1 \cdot \mathrm{d}_1 \cdot \tilde{\mathbf{h}}_1^{\mathrm{T}} + \tilde{\mathbf{w}}_2 \cdot \mathrm{d}_2 \cdot \tilde{\mathbf{h}}_2^{\mathrm{T}}$$

It can be found that in $\tilde{\boldsymbol{w}}_1$, the weight for node 6, 7 and 8 are zero; while in $\tilde{\mathbf{h}}_1^{\mathrm{T}}$, the weight for edge 6, 7, 8 and 9 are all zero. Therefore, in the expression $\tilde{\boldsymbol{w}}_1 \cdot \mathrm{d}_1 \cdot \tilde{\mathbf{h}}_1^{\mathrm{T}}$, the calculation is only related to node 1, 2, 3, 4 and 5, and edge 1, 2, 3, 4, 5. Similarly, expression $\tilde{\mathbf{w}}_2 \cdot \mathrm{d}_2 \cdot \tilde{\mathbf{h}}_2^{\mathrm{T}}$ is also determined by the weight value of partial nodes and edges. Therefore, the matrix factorization of **B** could be treated as factorization of the adjacency matrices of the following subgraphs (*Figure 4.2*).

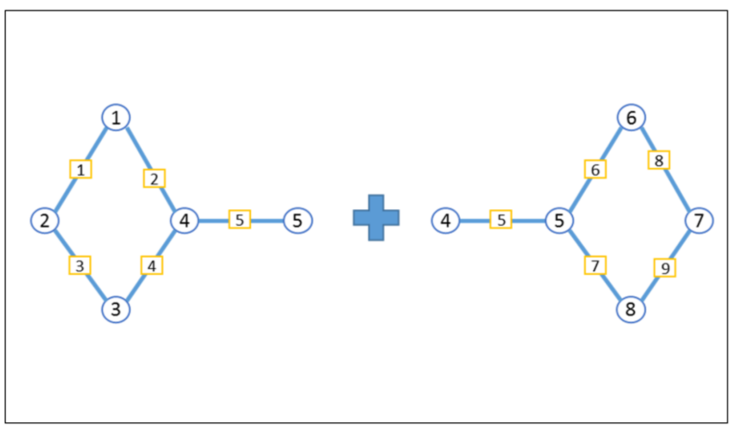

Figure 4.2 Factorized subgraph with r = 2

It may be noticed that the subgraphs are not totally complementary; there are overlapping components. By calculating the column sum of the $H^{*}_{2\times 9}$ matrix,

$$\mathrm{H}^* = \begin{bmatrix} 10.7232 & 1.0873 & 2.4634 & 0.5229 & 0.0523 & 0.0032 & 0.0032 & 0 & 0 \\ 0 & 0.0070 & 0 & 0.0222 & 0.5707 & 5.9633 & 5.9633 & 11.1310 & 11.1310 \end{bmatrix}$$

The edge anomaly score could be found as expressed in AE matrix

$$\mathrm{AE} = [10.7232 \quad 1.0943 \quad 2.4634 \quad 0.5251 \quad 0.6230 \quad 5.9665 \quad 5.9665 \quad 11.1310 \quad 11.1310]$$

Therefore, it can be found that edge 4 tends to be the most likely edge to be anomaly since it has the least weight among the nine edges.

Situation 3: when *r* = 3,

$$\mathbf{B} \approx \begin{bmatrix} .1532 & .2579 & 0 \\ 0 & .4841 & 0 \\ .1532 & .2579 & 0 \\ .5421 & 0 & 0 \\ .1514 & 0 & .2830 \\ 0 & 0 & .2426 \\ 0 & 0 & .2319 \\ 0 & 0 & .2426 \end{bmatrix} \times \begin{bmatrix} 5.8480 & 0 & 0 \\ 0 & 4.6492 & 0 \\ 0 & 0 & 8.7089 \end{bmatrix} \times \begin{bmatrix} 0 & .4344 & 0 \\ .3155 & .0656 & 0 \\ 0 & .4344 & 0 \\ .3155 & .0656 & 0 \\ .3096 & 0 & .0938 \\ .0297 & 0 & .2365 \\ .0297 & 0 & .2365 \\ 0 & 0 & .2166 \\ 0 & 0 & .2166 \end{bmatrix}^{\mathrm{T}}$$

$$= \begin{bmatrix} .1532 & .2579 & 0 \\ 0 & .4841 & 0 \\ .1532 & .2579 & 0 \\ .5421 & 0 & 0 \\ .1514 & 0 & .2830 \\ 0 & 0 & .2426 \\ 0 & 0 & .2319 \\ 0 & 0 & .2426 \end{bmatrix} \times \begin{bmatrix} 0 & 2.0196 & 0 \\ 1.8452 & 0.3051 & 0 \\ 0 & 2.0196 & 0 \\ 1.8452 & 0.3051 & 0 \\ 1.8104 & 0 & 0.8167 \\ 0.1736 & 0 & 2.0599 \\ 0.1736 & 0 & 2.0599 \\ 0 & 0 & 1.8862 \\ 0 & 0 & 1.8862 \end{bmatrix}^{T}$$

$$= \tilde{\boldsymbol{w}}_1 \cdot \mathrm{d}_1 \cdot \tilde{\mathbf{h}}_1 + \tilde{\mathbf{w}}_2 \cdot \mathrm{d}_2 \cdot \tilde{\mathbf{h}}_2 + \tilde{\mathbf{w}}_3 \cdot \mathrm{d}_3 \cdot \tilde{\mathbf{h}}_3$$

Similar to the situation on r =2, the expression $\tilde{\boldsymbol{w}}_1 \cdot \mathrm{d}_1 \cdot \tilde{\mathbf{h}}_1 + \tilde{\mathbf{w}}_2 \cdot \mathrm{d}_2 \cdot \tilde{\mathbf{h}}_2 + \tilde{\mathbf{w}}_3 \cdot \mathrm{d}_3 \cdot \tilde{\mathbf{h}}_3$ suggest that three subgraphs could be found by partitioning the original graph with consideration on the weight in the factorized matrices (*Figure 4.3*).

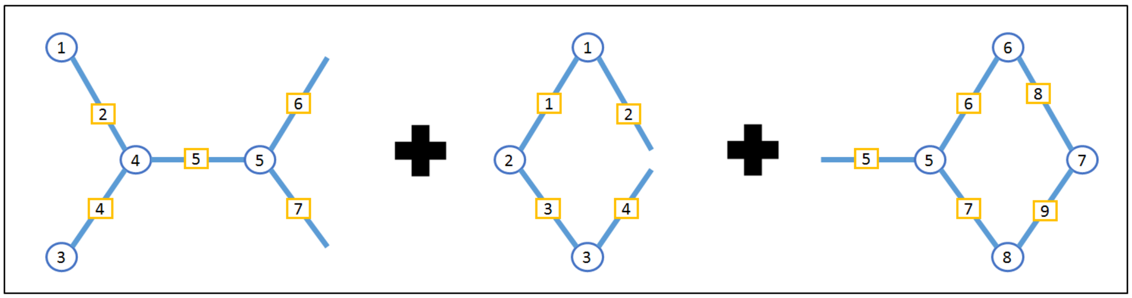

Figure 4.3 Factorized subgraph with r = 3

Situation 4: when r = 8,

What if we factorize the matrix into three matrices with the same size as the original matrix? Under this situation, the NMF factorization results is similar to that with SVD approach.

$$\mathbf{B} \approx \begin{bmatrix} 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 \\ 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 \\ 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 \end{bmatrix} \times \begin{bmatrix} 2 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 3 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 2 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 2 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 2 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 3 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 2 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 2 \end{bmatrix}$$

$$\times \begin{bmatrix} 0 & 0 & 0 & .5000 & .5000 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & .5000 & .3333 & 0 & 0 \\ 0 & 0 & 0 & .5000 & 0 & 0 & 0 & .5000 \\ 0 & 0 & 0 & 0 & 0 & .3333 & 0 & .5000 \\ 0 & .3333 & 0 & 0 & 0 & .3333 & 0 & 0 \\ .5000 & .3333 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & .3333 & 0 & 0 & 0 & 0 & .5000 & 0 \\ .5000 & 0 & .5000 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & .5000 & 0 & 0 & 0 & .5000 & 0 \end{bmatrix}^{T}$$

$$
= \begin{bmatrix} 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 \\ 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 \\ 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 \end{bmatrix} \times \begin{bmatrix} 0 & 0 & 0 & 1 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 & 0 & 1 \\ 0 & 0 & 0 & 0 & 0 & 1 & 0 & 1 \\ 0 & 1 & 0 & 0 & 0 & 1 & 0 & 0 \\ 1 & 1 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 & 0 & 1 & 0 \\ 1 & 0 & 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 & 0 & 1 & 0 \end{bmatrix}^{\mathrm{T}}
$$

Therefore, under this situation, the factorization results could be the incident matrix itself, since the factorization result of $\mathbf{W}^{(\mathbf{1})}$ could be transformed into an identity matrix easily, leaving along the $\mathbf{H}^{*}$ matrix which is exact the incident matrix. Therefore, when r = 8, the decomposition only reflects the original node-link property and may not be helpful in finding anomaly scores.

5. Node Anomaly Detection in Direct Graph Using Asymmetric Non-negative Matrix Factorization

In this section, we are using the Asymmetric Nonnegative Matrix Factorization (ANMF) approach to factorize the adjacency matrix in given graph data and define our anomaly detection functions. Section 5.1 will be the definition used in this section. Section 5.2 will be algorithms for scoring and detecting nodes. Section 5.3 will be examples.

5.1 Definitions and Notations

The graph in this section is directed and unweighted. The matrix used is the adjacency matrix with notation **A**. Nodes are represented in rows and columns, and the entries of **A** indicate the existence of edges. For unweighted graphs, all entries in matrix **A** are either 0 or 1.

$$\mathrm{A_{ij}} = \begin{cases} 1, & \text{if there is a directed link from node } i \text{ to } j \\ 0, & \text{otherwise} \end{cases}$$

In directed graph, the connectivity of two nodes may not be in bi-relationship due to the direction of edges. For the easiness of interpretation, we define two behaviors, "link" and "linked". Node *i* is said to "link" with node j if a path exists from the node *i* to node *j*. One the other hand, we could say node *j* is linked to node *i* vice versa. It is well noticed that such "link" behavior may not be interchangeable, i.e., node *i* links to node *j*, but node *j* may not links to node i due to the direction of edges.

Besides, there are other symbols used in the matrix factorization process, and we summarize the symbols in *Table 5.1*.

Table 5.1 Table of symbols used in notation

Symbol	Description
A	n x n adjacency matrix with entries A_{ij}
X,C,U,V,P	Lower rank factorized matrix used in ANMF
$\mathbf{O_1}$, $\mathbf{O_2}$, **O**	Matrix used to calculate anomaly score
k	Factorization rank
AN(*i*)	Anomaly score for node *i*

5.2 New Scoring Function for Node Anomaly Detection Using ANMF

ANMF is a specific factorization method that can be used on adjacency matrix of directed graph. The formulation of ANMF is as follows:

Given adjacency matrix **A** (n ×n dimension) and factorization k

Minimize Euclidean distance $\left\|\mathbf{A} - \mathbf{XPX^T}\right\|_F^2$

such that all entries in **X** (n ×k dimension) and **P** (k ×k dimension) are non-negative

Find matrix **X** and **P**

Furthermore, matrix **X** could be normalized by columns and $\boldsymbol{X^T}$ normalized by row respectively with the following formula:

$$\mathbf{XPX^T} = (\mathbf{XC^{-1}})(\mathbf{CPC^T})(\mathbf{XC^{-1}})^\mathbf{T} = \mathbf{UVU^T},$$

where $\boldsymbol{U} = \boldsymbol{XC^{-1}}$ and $V = \boldsymbol{CPC^T}$.

The algorithm introduced by Wang et.al (2011) provided application way to solve the ANMF problem using iteration approach, and thus we will use their algorithm to get matrix **U** and **V**.

The **U** matrix could provide k clusters of the original graph and the members of each cluster are decided by the non-zero entries in each column of matrix **U**. However, since this thesis mainly deals with anomaly detection, the exact clustering methods and the quality of clustering are not discussed in details. Rather, we simply group all the nodes with corresponding entries larger than a predetermined value in each column of one column into a cluster.

Moreover, the entries in **U** matrix represent the normalized weight of each node within one cluster, showing the "link" behavior. For example, if U_{ij} is large, then node *i* links to other nodes within cluster j and such "link" behavior is stronger compared with that of other nodes in the same cluster. The $\mathbf{U^T}$ matrix, on the other hand, reflects the "linked" behavior. The **V** matrix value represents connectivity, or the nodes' "link" behavior between two clusters. For example, if $V_{ij} \neq 0$, then some node can be found in cluster i links to some nodes in cluster j. Higher value means that such link is strong, i.e., an edge (1-step path) exits between the nodes of two clusters. Besides, the diagonal terms in **V** matrix could represent the important of a cluster in terms of the whole graph. Therefore, if we only consider such "link" behavior, the importance of one node in the graph is determined by how the node links other nodes both within one cluster and between clusters.

Thus, we could calculate the scores for each from the expression $\mathbf{O_1} = \mathbf{U*V}$. The row element gives the score value of one node in each community, and we will use the row sum as the score function for the "link" behavior, or

$$\mathrm{O}_1\text{\_score } (i) = \sum_{j=1}^{k} [\mathbf{O_1}]_{\boldsymbol{ij}}$$

The nodes with less score values in this scenario tend to be anomalous.

Nevertheless, in directed graph, considering only the “link” behavior is not sufficient because it only describe the behavior of edges that points from one nodes to other nodes. Due to the possibly asymmetric adjacency matrix and the asymmetric edge directions between two nodes in directed graph, may also be necessary to consider the linkage of edges pointing from other nodes to a given node, or the “linked” behavior. Instead of **U** matrix, $\mathbf{U^T}$ matrix will be used to describe how one node is linked to other nodes. Following the similar idea as above, the expression $\mathbf{O_2} = \mathbf{VU^T}$could be used to score the “linked” behavior, and the row sum could be used to represents the scoring function for each nodes, or

$$\mathrm{O}_2\text{\_score } (j) = \sum_{i=1}^{k} [\mathbf{O_2}]_{\boldsymbol{ij}}.$$

To keep the format as in the O_1_score(i), we take the transpose of $\mathbf{O_2}$ matrix,

$$\mathrm{O}_2\text{\_score}(i) = \sum_{j=1}^{k} [\mathbf{O_2}]_{\boldsymbol{ij}}, \text{ where } \mathbf{O_2} = (\mathbf{VU^T})^{\mathbf{T}} = \mathbf{UV^T}.$$

The nodes with less score values in this scenario tend to be anomaly.

Based on the discussion above, the connectivity, or the structure of the original graph could be decomposed into “link” behavior and “linked” behavior. Both behaviors are important for finding detecting anomaly. The sum of the O_1_score(i) and O_2_score(i) could give a simple overall score function. However, the importance of these two behaviors may not be the same in different graphs with the actual needs of anomaly types. For example, in patent citation network, the patents are more likely to be anomalies if they are cited by few other patents so that the "linked" behavior may be more important in this situation; for email contact network, an email address keeping sending high volume of emails to other users may be treated as anomalous/spam source so that the "link" behavior could be more important. To separate the importance, we introduce the weighted scoring function for node anomaly detection, denoted as:

$$\text{AN}(i) = \sum_{j=1}^{k} [\mathbf{O}]_{ij}, \text{ where } \mathbf{O} = \alpha \mathbf{UV} + (1-\alpha)\mathbf{UV^T},\ 0 \le \alpha \le 1.$$

Table 5.2 shows the algorithm for detecting node anomalies and our algorithm is named as "ANMFG".

Table 5.2 Algorithm for ANMFG

Given:

Adjacency matrix **A**, factorization rank k, weight α, threshold δ,

(1) Use ANMF method to factorize **A** matrix and get the output matrix **X** and **P** following the algorithm introduced by Wang et al. (2011).

(2) Normalize the **X** matrix in terms of columns to obtain **U** matrix and **V** matrix

(3) Find the **O** matrix for anomaly scoring using $\mathbf{O} = \alpha \mathbf{UV} + (1-\alpha)\mathbf{U}\boldsymbol{V}^{\boldsymbol{T}}$

(4) Find the anomaly score for each node by sum the row values of **O** matrix, or $\text{AN}(i) = \sum_{j=1}^{k} [\mathbf{O}]_{ij}$

(5) Sort the AN (i) score in ascending order

(6) Lock the score values smaller or equal to a predetermined threshold δ and pick out the respect nodes as anomalies.

Output:

Anomaly nodes

5.3 An Illustrative Example

To better illustrate our idea, consider the following artificial example. The graph is shown as in *Figure 5.1* and the adjacency matrix could be formulated accordingly. By using the ANMF method with factorization rank 4, we could get the following factorization results, matrix **U** and **V**:

$$\boldsymbol{U}_{\mathbf{13}\times\mathbf{4}} = \begin{bmatrix} 0.000 & 0.999 & 0.000 & 0.000 \\ 0.000 & 0.000 & 0.380 & 0.000 \\ 0.000 & 0.000 & 0.000 & 0.145 \\ 0.000 & 0.000 & 0.000 & 0.086 \\ 0.000 & 0.009 & 0.596 & 0.000 \\ 0.000 & 0.000 & 0.019 & 0.276 \\ 0.000 & 0.000 & 0.0058 & 0.392 \\ 0.081 & 0.000 & 0.000 & 0.000 \\ 0.081 & 0.000 & 0.000 & 0.000 \\ 0.020 & 0.000 & 0.000 & 0.000 \\ 0.395 & 0.000 & 0.000 & 0.101 \\ 0.104 & 0.000 & 0.000 & 0.000 \\ 0.318 & 0.000 & 0.000 & 0.000 \end{bmatrix} \quad \boldsymbol{V}_{\mathbf{4}\times\mathbf{4}} = \begin{bmatrix} 1.32 & 0.00 & 0.00 & 0.00 \\ 0.00 & 9.3035 & 1.97 & 0.00 \\ 0.00 & 0.00 & 0.00 & 3.63 \\ 5.81 & 0.00 & 0.00 & 0.00 \end{bmatrix}$$

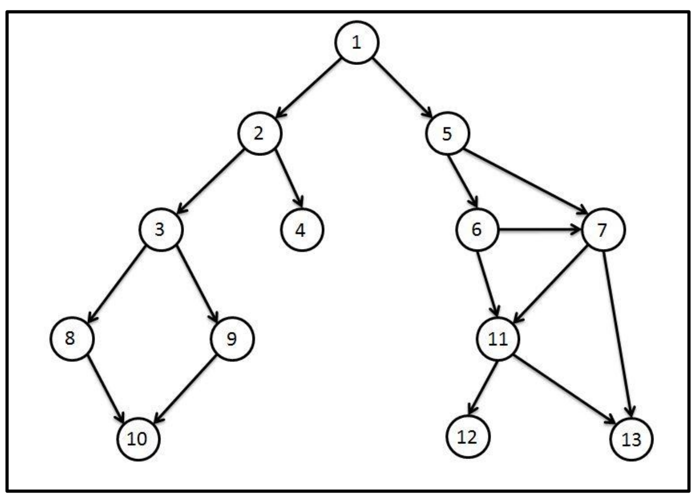

Figure 5.1 Graph of the artificial example

The ANMF method suggests that four clusters can be formulate with regards to the non-zero values, or values larger than a predetermined threshold in each column in the **U** matrix considering the preciseness of matrix factorization. For example, in the first column, the eighth to the thirteenth entries are nonzero values. Thus node eight to thirteen are grouped into one community. Following the same idea, we could find and mark the four clusters in the graph, and the results are shown in *Figure 5.2*. It is worth noticing that even though nodes in one cluster may not be directly connected, they are grouped based on the factorization results of the adjacency matrix, which reflects the structural information of the original graph.

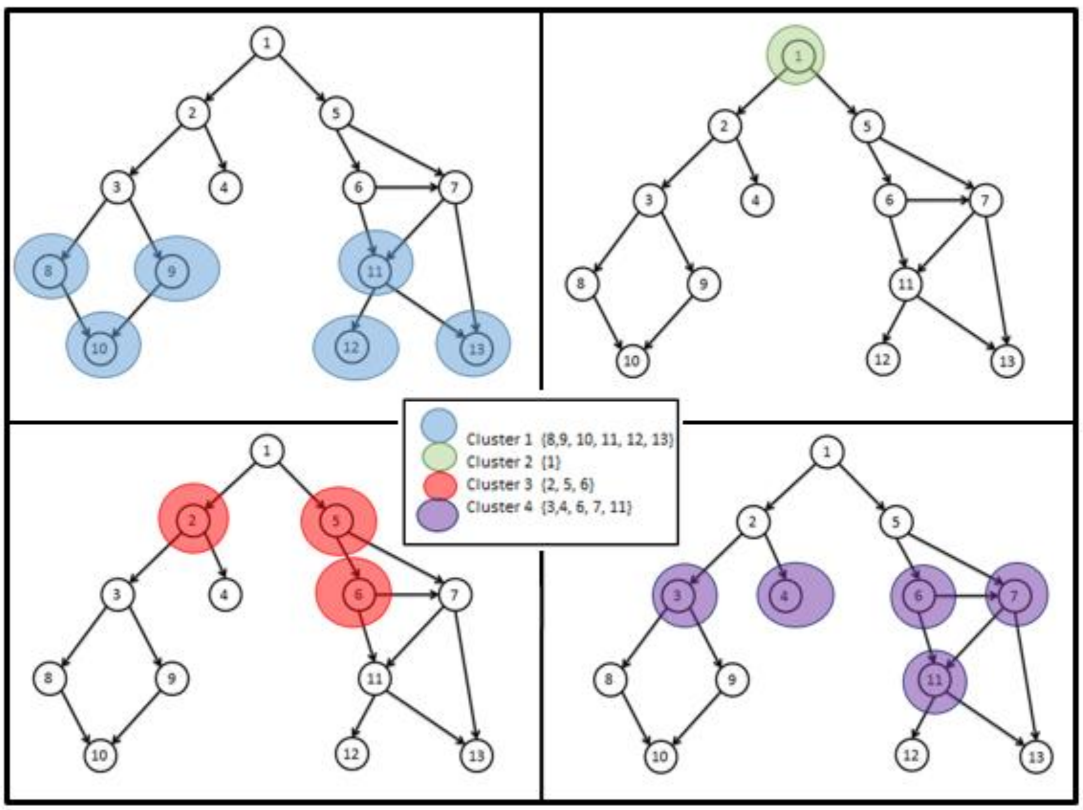

Figure 5.2 Four clusters obtained based on the matrix factorization result

From the four clusters obtained, some interesting relationship could be observed. By looking at the elements of the green cluster and the red cluster, it can be found that there exists paths merely from some nodes in the green cluster to the nodes in the red clusters, i.e., node 1 to node 2 and 5. Similarly, for the red cluster and the purple cluster, if we ignore the overlapping terms, there are only paths from nodes in red cluster to nodes in purple cluster in the directed graph. Such kind of relationship could be explained by the values in H matrix. For example, $\boldsymbol{V_{41}}$ is 5.81, which indicates strong connecting behavior from the purple cluster to the blue cluster. Thus explanation could be tested from the graph, since it can be observed that nodes in purple clusters links to nodes in blue clusters and five direct edges can be found (i.e., line form node 3 to node 8).

The anomaly scores will be calculated using the factorization results. Since the α value in the score function is a predetermined parameter that could influence on the scoring ranking of the nodes, we provide the results with α value at different level, 0, 0.25, 0.5 and 1. The anomaly scores and ranking are shown in *Table 5.3.*

Table 5.3 Anomaly score comparison with different α values in scoring function

α	0		0.25		0.5		0.75		1	
Rank	**Node**	**Score**	**Node**	**Score**	**Node**	**Score**	**Node**	**Score**	**Node**	**Score**
1	1	0.0000	10	0.1144	10	0.0852	10	0.0559	10	0.0266
2	10	0.1437	4	0.3575	8	0.3434	8	0.2254	8	0.1073
3	4	0.3108	8	0.4615	9	0.3434	9	0.2254	9	0.1073
4	3	0.5242	9	0.4615	4	0.4043	12	0.2888	12	0.1375
5	8	0.5795	1	0.4883	12	0.4401	4	0.4510	13	0.4196
6	9	0.5795	12	0.5914	3	0.6817	3	0.7605	4	0.4977
7	12	0.7427	3	0.6030	1	0.9766	13	0.8813	3	0.8392
8	2	0.7479	2	0.9050	2	1.0621	2	1.2192	11	1.1096
9	6	1.0398	6	1.1983	13	1.3429	1	1.4649	2	1.3763
10	5	1.1738	5	1.4247	6	1.3568	6	1.5152	6	1.6737
11	7	1.4322	7	1.6481	5	1.6756	11	1.6274	1	1.9532
12	13	2.2662	13	1.8046	7	1.8639	5	1.9264	5	2.1773
13	11	3.1806	11	2.6629	11	2.1451	7	2.0798	7	2.2956

When $\alpha = 0$, the score shows merely how one node is linked by nodes both within one cluster and among different clusters. Since in the green cluster, node 1 is the only member in the cluster and could not be linked by other nodes because it has edges with directions pointing only to other nodes, thus the score value for node 1 is zero. The value for node 10 is also smaller because the "linked" behavior mainly from the nodes within the same cluster (node 8 and 9) since an edge exists between node 10 and 8 or 10 and 9. Even though a path exists from the nodes from other clusters to node 10, it requires two or more steps, thus the influence on scoring function may be weak.

One clearly phenomenon is that when $\alpha \neq 0$, node 1 will have larger values and drops down in the score ranking. This is because in these scenarios, the "link" behavior is considered. Since node 1 links to other nodes in different clusters closely, i.e. node 1 directly links to node 2 and 5, it will have large score values for "link" behavior. This could also be explained that when $\alpha = 1$, node 1 owns rank 11 in the anomaly score ranking. Besides, since node 10 and its cluster members are at the bottom of the graph with few links to other clusters. Thus the "link" behavior score is also very low.

Thus, by considering both the link and inked behavior with different α value, we may get different rank and score functions. But node 10 have high possibility in these cases as node anomaly.

6. Experiments

In this section, we will test the performance of our algorithms by using both artificial data and the real-world data. Section 6.1 will test the performance of edge anomaly scoring algorithm with NMF and Section 6.2 will test the performance of node anomaly scoring algorithm with ANMF. Moreover, we will discuss the possible updates and challenges of our algorithms for future research in Section 6.3

6.1 Test of Edge Anomaly Detection Algorithms

Two sets of data are provided to evaluate the performance of our algorithms. Section 6.1.1 will use the Amazon Co-purchase data and Section 6.1.2 will use the patent citation network dataset. Moreover, we will discuss the possible updates and challenges of our algorithms for future research in Section 5.3

6.1.1 Amazon Co-purchase Network

In this section, we will use the co-purchase network generated from the data of online book seller Amazon[1]. In the network, the vertices represent the book and the edges represent the behaviors called "co-purchase" by customers. Therefore, if a customer bought one book while at the same time bought another book, then the two nodes representing these two books are connected. The dataset we use is a subgraph with all books categorized into the field "Disney", with 124 nodes and 334 edges, both of which are unweighted. The structure of the graph could be seen in *Figure 6.1*and the histogram of the degree distribution in *Figure 6.2.*

[1]http://www.ipd.kit.edu/~muellere/GOutRank/

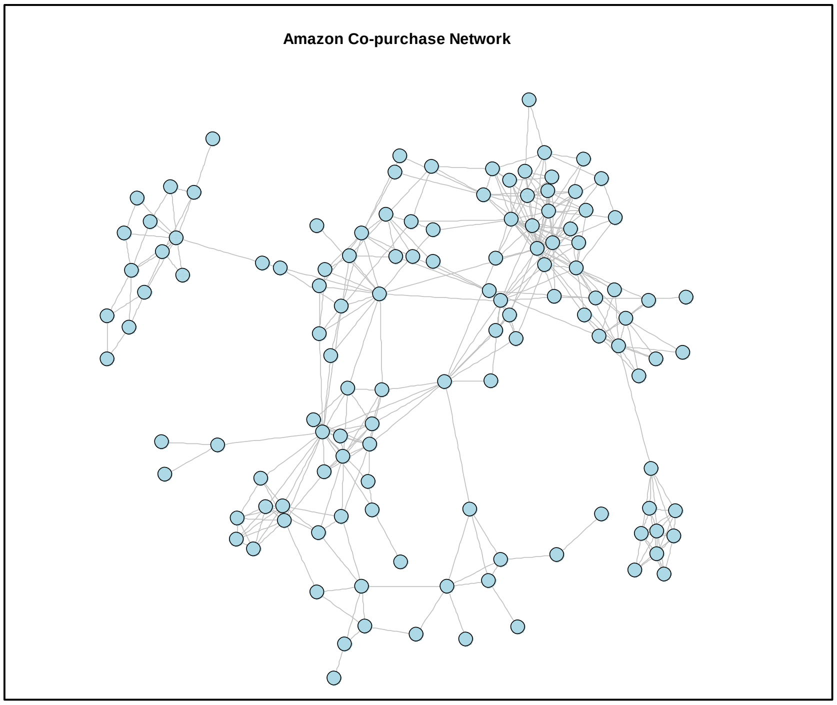

Figure 6.1 Structure of amazon co-purchase network

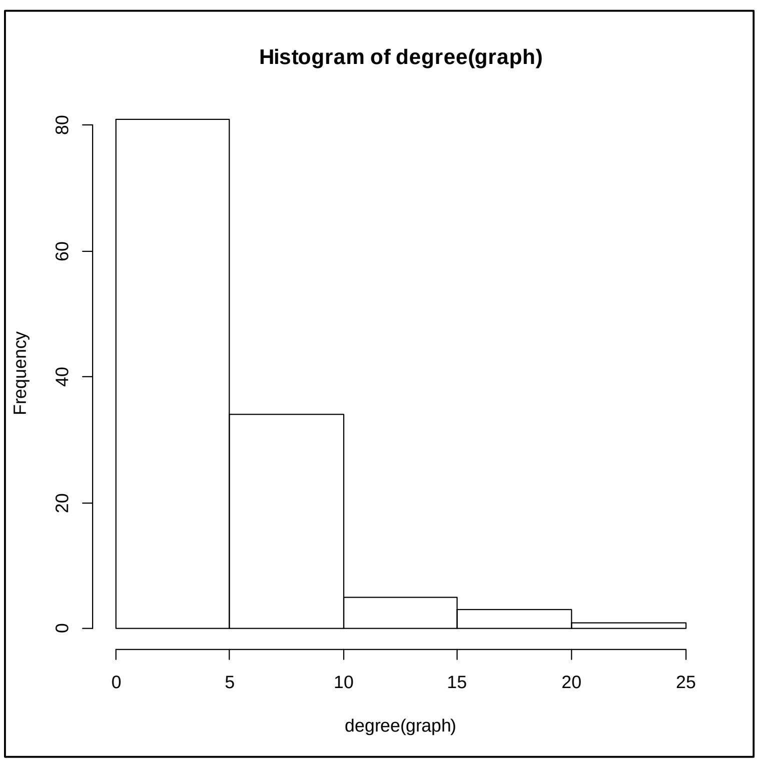

Figure 6.2 The histogram of degree in amazon co-purchase network

We will apply our algorithms first to find the scores of edges that are likely to be anomalous. Here, we decide to select the factorization rank r = 50 for both the adjacency matrix We show up the score for the top 20% edge anomalies with the node-edge adjacency matrix in *Table 6.2.*

Table 6.1 Edge anomaly ranking with scores-amazon data

Ranking	**Edge_Id**	**Score**	**Ranking**	**Edge_Id**	**Score**
1	141	0.2239	25	36	1.5494
2	302	0.5201	26	158	1.5566
3	24	0.6538	27	254	1.5719
4	282	0.6542	28	210	1.5768
5	19	0.6711	29	120	1.5812
6	180	0.7111	30	200	1.5815
7	69	0.7985	31	74	1.5939
8	33	0.8010	32	42	1.5958
9	306	0.8449	33	91	1.6039
10	194	0.8473	34	265	1.6123
11	44	0.8879	35	124	1.6169
12	68	1.0517	36	320	1.6225
13	267	1.0936	37	96	1.6333
14	30	1.1045	38	285	1.6353
15	140	1.1980	39	154	1.6443
16	159	1.2909	40	26	1.6481
17	294	1.3363	41	287	1.6488
18	195	1.3538	42	310	1.6531
19	57	1.3775	43	335	1.6563
20	127	1.4026	44	198	1.6601
21	63	1.4859	45	281	1.6692
22	261	1.5007	46	144	1.6697
23	130	1.5330	47	151	1.6749
24	181	1.5488	48	148	1.6770

Moreover, we could also visualize the position of the anomaly nodes and edges. We will pick out top 20 edge anomalies and plot them onto one figure with red color. The results are shown in *Figure 6.3*.

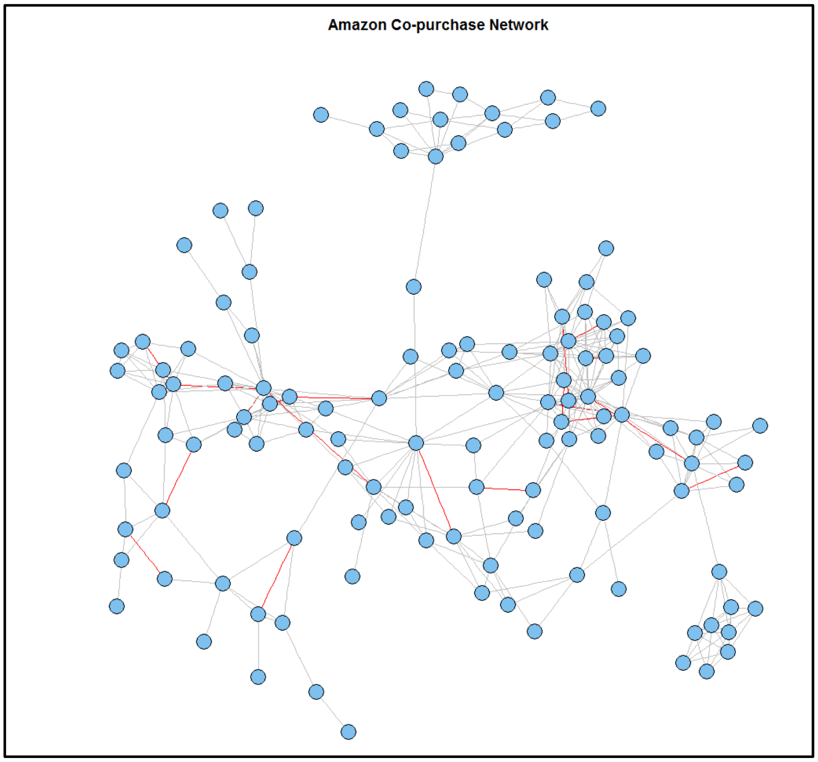

Figure 6.3 Plot of edge anomalies-amazon data

From the plot, it can be found that the edges which connect to two nodes with high degrees are more likely to be anomalous. Since there are many combinations of subgraphs by the two nodes with high degree, therefore the weight for the subgraphs with the edge connecting these two nodes are lower. In real life, such edge anomalies are valuable because it connects two possible groups of nodes which have high average degrees, or highly concentrated. Therefore, by detecting these nodes, people could separate the nodes into different groups and predict the future development of graphs. For example, if book A and book B are very popular in co purchase while seldom do people buy them together, then the manager may have to categorize the books co-purchased with book A into a new group, say group A and recommend the books in that category to customers when customer co-purchase with A. While at the same time, the books co-purchased with book B may not be recommend with books in group A so that the recommendation effectiveness may improve.

Moreover, since the factorization rank r needs to be predetermined before running the algorithm, selection of *r* is necessary to discuss. It has been shown that the selection of r could influence the quality of the matrix factorization; thus some researchers have suggested different methods to find the optimal value of r to achieve the overall matrix factorization performance, such as the residual sum squares (Brunet, Tamayo, Golub, & Mesirov, 2004; Hutchins, Murphy, Singh, & Graber, 2008; Frigyesi & Höglund, 2008). However, for different criteria, the r value may not be the same; on the other hand, there is a big difference in our algorithm that we need only the top lowest value for detecting anomalies. Therefore, it will be important for us to find stable results first to avoid high fluctuation in score ranking, especially among the top lowest scores.

To test the stability of the factorization results, we use the node edge incident matrix as an example. We test the factorization rank r from 1 to 124, with 50 runs for each r. After collecting the scores for 50 runs given an r value, we sort the scores from low to high for each individual run. Then we collect the lowest score from each run, and calculate the variance for these 50 scores. We repeat this process for calculating the variance of the second lowest scores in 50 runs, until getting the variance of the highest scores in the 50 runs. After getting all the variance values, we add them up to represent the variance for the whole variance given an r value. Moreover, since anomaly selection may not involve those nodes with higher score, we also test the variance for the data among the top 20 lowest value. The logic of our testing method is that if the performance of NMF is stable, then the variance of the scores located in the same ranking position should be similar. In other words, there should not be high fluctuations in scoring edges.

The result of our testing is plotted in *Figure 6.4*. The blue dot line shows the sum of variance for scores in all ranking positions, while the red line is only the data with top 20 rankings.

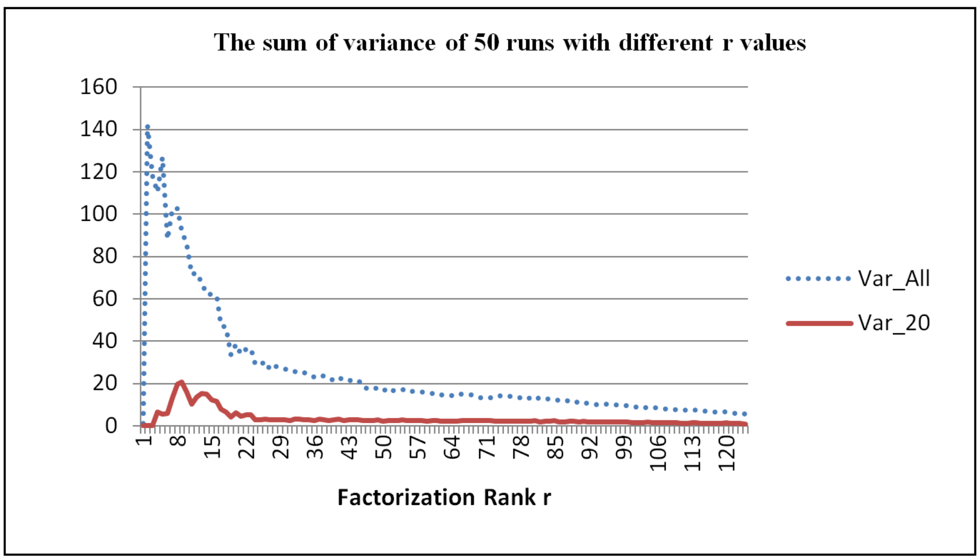

Figure 6.4 Plot of factorization rank with the total variance of 50 runs

It can be found that when r is larger than 25, the results tends to be stable, especially for the score of top 20 nodes. This method may be used a guideline for suggesting proper r value for matrix factorization, combining with other approaches for better judgments.

6.1.2 Patent Citation Network Data

In this section, we test the performance of our algorithms with the patent citation network data[1]. The dataset is a directed and unweighted graph consisting 4241 nodes and 18385 edges. To provide an intuitive understanding, the plot of the graphic data and the histogram of degrees of nodes could be found as the following in

Figure 6.5 and *Figure 6.6.* The nodes represent different patents with a unique patent id. Since for each patent, it may cite another patent as a reference. Therefore, if one patent is cited by another patent, a directed edge connects the two nodes representing these two nodes, pointing from cited patents to the citing patent.

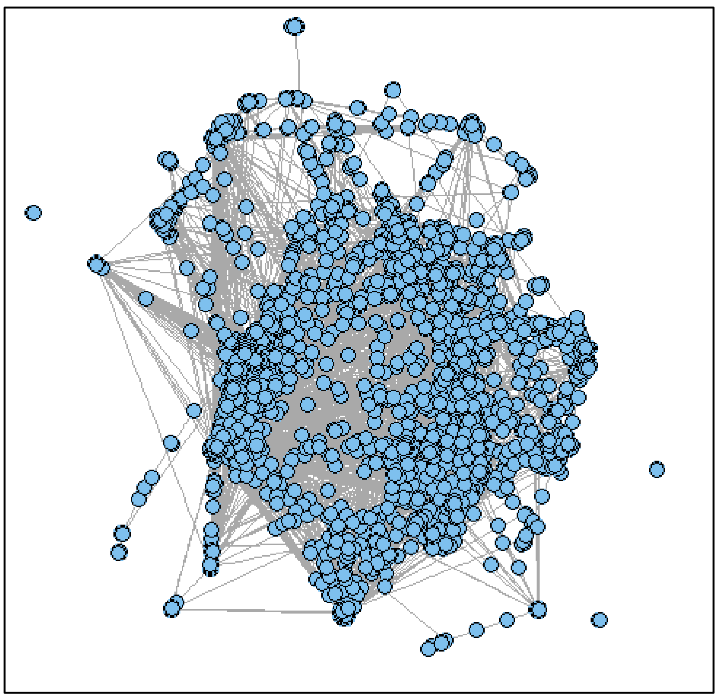

Figure 6.5 Plot of the patent citation network data

[1] http://dme.rwth-aachen.de/de/gamer

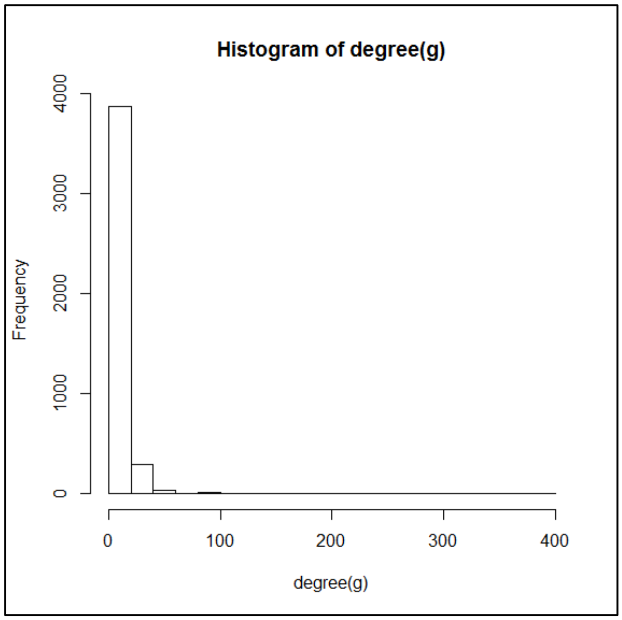

Figure 6.6 Degree distribution of the patent citation network

Different from the dataset in section 6.1.1, the patent citation network is an acyclic graph because only the patent invented later could cite the former patent, or the time order. It can also be found in *Figure 6.6* that the range of degrees is larger than that in section 6.1.1, with extreme amount of nodes having very large degree number. This is a special character of the patent network because the patents at the beginning years in the database tend to have more citations than those in later years.

In this dataset, we will use factorization rank r= 1250 to detect edge anomalies. The scoring results for the top 60 edges anomalies are shown in *Table 6.2*.

Table 6.2 Edge anomalies ranking with scores-patent citation data

Ranking	Edge_Id	Score	Ranking	Edge_Id	Score
1	15144	0.4077	31	8012	1.1738
2	10910	0.4196	32	4875	1.176
3	3191	0.7856	33	1179	1.1772

4	4354	0.7932	34	6729	1.1799
5	13524	0.8274	35	11831	1.1896
6	7346	0.8429	36	9545	1.1900
7	5693	0.8704	37	13124	1.1908
8	9928	0.8892	38	1377	1.1913
9	3172	0.9141	39	3383	1.1972
10	6199	0.9210	40	8218	1.2051
11	11321	0.9927	41	9744	1.2100
12	4878	1.0539	42	1010	1.2118
13	8687	1.0623	43	10915	1.2145
14	11636	1.0636	44	4066	1.2156
15	2990	1.0870	45	11328	1.2173
16	2694	1.1023	46	3743	1.2182
17	12390	1.1215	47	3406	1.2192
18	12254	1.1219	48	786	1.2245
19	1409	1.1239	49	2275	1.2280
20	2730	1.1283	50	2257	1.2284
21	12227	1.1416	51	2206	1.2296
22	3226	1.1515	52	3162	1.2297
23	5089	1.1564	53	13592	1.2297
24	243	1.1570	54	1963	1.2328
25	2627	1.1574	55	13906	1.2366
26	1161	1.1584	56	2932	1.2376
27	9722	1.1617	57	1011	1.2422
28	12961	1.1665	58	8217	1.2427
29	3154	1.1707	59	4710	1.2445
30	7662	1.1718	60	7445	1.2494

For large graph, it is not easy to mark out the edges in the graph visually. However, the edge anomalies can be found between one node having large amount of in-neighbors and another node having a large number of out-neighbors. As shown in *Figure 6.7*, such edge anomalies may provide useful information for the information user and the following situation is one possible explanation. Patent A owns a lot of in-neighbors may indicate that such patent owns a lot of common features among different areas. However, patent A was cited only by a few other patents, i.e., patent B, implying the applicability of patent A. Nevertheless, patent B has been cited by many other patents so that patent B has made a lot of inspirations in terms of invention. Therefore, there may be possible technology updates and from patent A to patent B. Thus, by focusing on the anomaly edges (marked in red), the information user may focus on possible new technology or areas that are beneficial.

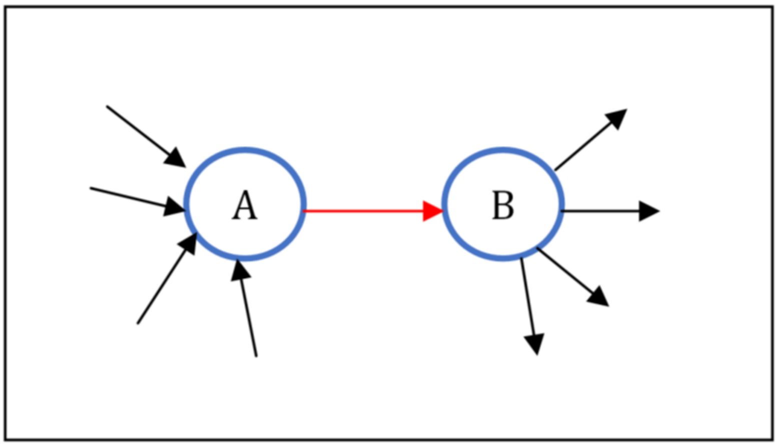

Figure 6.7 Edge anomalies example for patent citation data

6.2 Test of node anomaly detection algorithms ANMFG

In this section, we will use both artificial examples and real world data set to test the performance of our algorithm. To show better performance, we will compare the results of our algorithm with two other types of algorithms, Outrank and Oddball, which could also pick up node anomalies. Three sets of data will be used in the test: the artificial data using as example in section 5; 20 nodes patent citation network data from real data set; 250 nodes patent citation network.

6.2.1 Testing with 13-node Artificial Data

The information of the data has been described in Section 5.3 and in *Figure 5.1*. In this section, we will show the testing results of our algorithm ANMFG with factorization rank k = 4 and α= 0.8. *Table 6.3* shows the scores of different approaches for detecting node anomalies. For the methods OutRankB (Moonesinghe & Tan, 2006), CDNMF (Cao, Wang, Jin, Cao, & He, 2013) and Oddball (Akoglu, McGlohon, & Faloutsos, 2010), we need to transform the adjacency matrix for directed graph into that for undirected graph due to their philosophy of algorithm design.

Table 6.3 Computation results of different algorithms on 13-node artificial data

	OutRankB		**Oddball**		**CDNMF**		**ANMFG**	
Rank	**Node**	**Score**	**Node**	**Score**	**Node**	**Score**	**Node**	**Score**
1	4	0.0434	4	0.0819	12	1	10	0.0500
2	10	0.0460	12	0.0819	1	0	8	0.2017
3	12	0.0483	5	0.1355	2	0	9	0.2017
4	2	0.0604	1	0.2766	3	0	12	0.2585
5	8	0.0646	8	0.2766	4	0	4	0.4603
6	9	0.0646	9	0.2766	5	0	3	0.7762
7	1	0.0734	10	0.2766	6	0	13	0.7889
8	3	0.0851	11	0.3463	7	0	2	1.2506
9	11	0.0899	13	0.7187	8	0	11	1.5238
10	13	0.0903	2	0.7987	9	0	6	1.5469
11	5	0.0934	3	0.7987	10	0	1	1.5626
12	7	0.1203	6	0.9853	11	0	5	1.9766
13	6	0.1203	7	1.0821	13	0	7	2.1229

In Section 5.2, we have discussed that our algorithm could provide meaningful results. In comparisons, the CDNMF method could provide only binary results. It may pick out some anomalies, but show no distinctions between different anomalies. For OutRankB and Oddball methods, the results provided are meaningful if we neglect the direction of the edge. Nodes 4 and nodes 12 are more likely to be anomalous in these methods because node 4 and 12 connect to one node. However, our algorithms enables synthesis scoring of the link nodes and linked nodes, making use of the direction information of the directed graph. Thus, our method is more meaningful in this case.

6.2.2 20-node from Patent Citation Network Data

In this section, we test the performance of our algorithms with a small example from the patent citation network data[1]. The original dataset is a directed and unweighted graph consisting 4241 nodes and 18385 edges, and we pick out 20 nodes from them. The structure of the dataset could be clearly observed from *Figure 6.8*. The nodes refer to different patents with a unique patent id. Since for each patent, it may cite another patent as a reference; thus the edges shows this kind of citation relationship: if one patent is cited by another patent, a directed edge connects the two nodes representing these two nodes, pointing from cited patent to the citing patent.

[1] http://dme.rwth-aachen.de/de/gamer

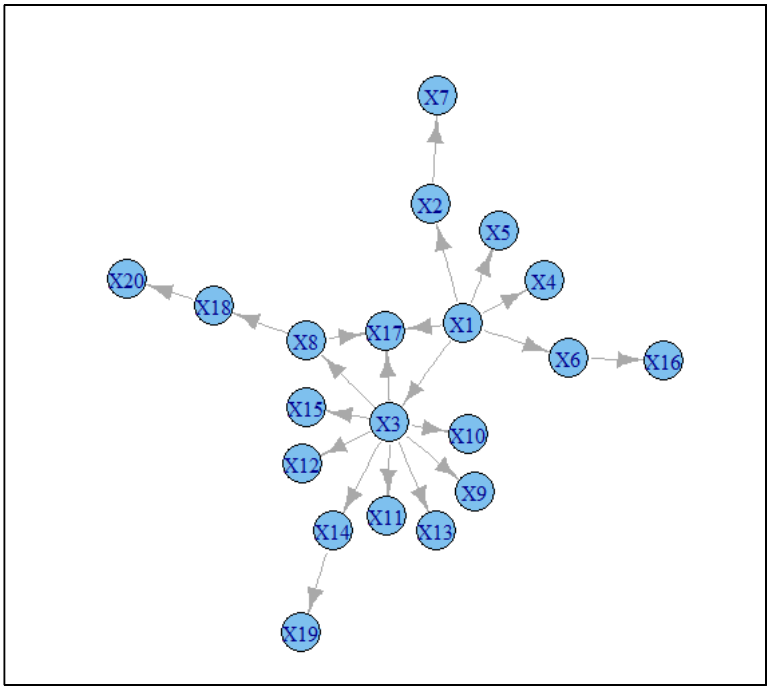

Figure 6.8 Plot of the 20-node patent citation data set

Ahead from testing the performance of using the different algorithms, some ground truth could be observed based on the plot of the dataset in *Figure 6.*. It is found that patent id "20","7","16","19" (We ignore the "X" in front of the id and only use the number) are more likely to be anomalies for two reasons: they only cite one patent, having only one in-degree; they are two or more steps further from the patents "3","1","8","17", which are cited by many different patents and have larger probability to be the center if clustering is performed.

With these understanding of the graph, we perform our algorithms with α = 0.8 and r = 4. We also test the performance of CDNMF with factorization rank 4, Outrank and Oddball. The results are shown in *Table 6.4.*

Table 6.4 Computation results of different algorithms on 13-node artificial data

	OutRankB		Oddball		CDNMF		AMFG	
Rank	**Node**	**Score**	**Node**	**Score**	**Node**	**Score**	**Node**	**Score**
1	20	0.0112	4	0.0018	4	1	7	0.0000
2	7	0.0114	5	0.0018	5	1	16	0.0000
3	16	0.0114	7	0.0018	7	1	19	0.0000
4	19	0.0119	9	0.0018	19	1	20	0.0000
5	18	0.0181	10	0.0018	1	0	18	0.0676
6	2	0.0394	11	0.0018	2	0	9	0.2181
7	4	0.0394	12	0.0018	3	0	10	0.2181
8	5	0.0394	13	0.0018	6	0	11	0.2181
9	6	0.0394	15	0.0018	8	0	12	0.2181
10	9	0.0595	16	0.0018	9	0	13	0.2181
11	10	0.0595	19	0.0018	10	0	14	0.2181
12	11	0.0595	20	0.0018	11	0	15	0.2181
13	12	0.0595	2	0.1796	12	0	2	0.2310
14	13	0.0595	6	0.1796	13	0	4	0.2310
15	14	0.0595	14	0.1796	14	0	5	0.2310
16	15	0.0595	18	0.1796	15	0	6	0.2310
17	3	0.0772	1	0.4099	16	0	17	0.5079
18	8	0.0826	8	0.5230	17	0	8	1.9506
19	1	0.0927	3	0.8909	18	0	1	4.9039
20	17	0.1096	17	1.3727	20	0	3	7.4970

It can be found from the results that our algorithm performs well and pick out the desired anomalies successfully. The results for Oddball and CDNMF method may not be satisfactory because even in the top 10 anomalies among the 20 nodes, they could not include the desires anomalies. Besides, the Outrank algorithm could also pick out the node anomalies correctly. However, it is used only for undirected graph and we have to transform the adjacency matrix and ignore the direct one. The result could show that our method is helpful in detecting anomalies in directed graph.

6.2.3 250-node Patent Citation Network Data

Due to the number of nodes an edges in this case, it is difficult to visualize the results on graph. However, we could still use the score functions and pick out anomalies. The results are shown in *Table 6.5*.

Table 6.5 Computation results of different algorithms on 250-node artificial data

Rank	**Node**	**Score**	**Rank**	**Node**	**Score**
1	58	0.0097	21	67	0.0627
2	11	0.0101	22	210	0.0627
3	95	0.0101	23	21	0.0714
4	141	0.0101	24	48	0.0728
5	89	0.0104	25	53	0.0758
6	26	0.0142	26	213	0.0877
7	227	0.0149	27	109	0.1013
8	146	0.0214	28	125	0.1013
9	22	0.0218	29	212	0.1013
10	127	0.0275	30	232	0.1013
11	62	0.0287	31	9	0.1671
12	31	0.0292	32	4	0.1889
13	126	0.0327	33	160	0.1905
14	184	0.0327	34	118	0.2032
15	231	0.0327	35	128	0.2032
16	30	0.0460	36	139	0.2032
17	32	0.0626	37	42	0.2047
18	52	0.0626	38	168	0.2047
19	173	0.0626	39	6	0.2174
20	55	0.0627	40	18	0.2388

The nodes detected by our algorithms could perform well in the large data set.

6.3 Recommendations for Future Research

In the future, it is suggested that our algorithms could be improved from the following two aspects:

First of all, more real life datasets are needed to test the performance of our algorithms. This may not only help compare the performance of different algorithms with different cases, but also accumulate more experience on the selection of factorization rank r. It is also suggested that a comparison can be made among different approaches of r selection. The relationship between the overall performance and the stability of the top scores could thus be discussed further, which will be benefit to improve the overall performance of the algorithms.

Second, we did not consider the possible known attributes of the nodes and edges in graphs when design the algorithms. However, there are possible ways to make use of the node attributes, like the anomaly detection approaches for continuous or discrete data. Therefore, future discussions can be made on the performance of our algorithm and the updated version by considering the node attributes.

7. Conclusion

In this thesis, we suggest new node and edge anomaly detection algorithms using the non-negative matrix factorization techniques.

Experimental results have shown that our edge anomaly detection algorithm has successfully detected the anomalies with given Amazon book co-purchase network and patent citation network. Moreover, by plotting and studying the graph with anomalies, meaningful interpretations have been found, which have provided real meaning on the results. The selection of factorization rank r has also been discussed, and it is suggested that different criteria should be used on the selection of r to ensure the stability of the factorization result. For the node anomaly detection algorithms, ANMFG, comparison results indicate that our algorithm shows well performance in anomaly scoring and detection on direct graph compared to some existing methods.

The possible improvements on this thesis have also been discussed. It is hoped that in the future, the attribute of the nodes in a graph, if available, could also be considered for detecting the algorithm. It is also suggested more different kinds of datasets are selected to strengthen the computational evidence on factorization rank selection. Besides, in the age of big data, the method should be revised so that it could meet the requirement of large data processing and real time computing.

8. References

Aggarwal, C. C. (2013). *Outlier Analysis.* Springer.

Aggarwal, C. C., Zhao, Y., & Yu, P. S. (2011, April). Outlier detection in graph streams. *Data Engineering (ICDE), 2011 IEEE 27th International Conference on* (pp. 399-409). IEEE.

Akoglu, L., McGlohon, M., & Faloutsos, C. (2010). *Oddball: Spotting anomalies in weighted graphs. In Advances in Knowledge Discovery and Data Mining.* Berlin Heidelberg: Springer.

Akoglu, L., Tong, H., & Koutra, D. (2014). Graph-based Anomaly Detection and Description: A Survey. *Data Mining and Knowledge Discovery*.

Antonellis, I., Molina, H. G., & Chang, C. C. (2008). Simrank++: Query rewriting through link analysis of the click graph. *Proceedings of the VLDB Endowment, 1*(1), 408-421.

Bonacich, P., & Lloyd, P. (2001). Eigenvector-like measures of centrality for asymmetric relations. *Social Networks, 23*(3), 191-201.

Brin, S., & Page, L. (1998). The anatomy of a large-scale hypertextual web search engine. *Computer Networks, 30*(1-7), 107-117.

Broder, A., Kumar, R., Maghoul, F., Raghavan, P., Rajagopalan, S., Stata, R., & Wiener, J. (2000). Graph structure in the web. *Computer networks, 33*(1), 309-320.

Brunet, J. P., Tamayo, P., Golub, T. R., & Mesirov, J. P. (2004). Metagenes and molecular pattern discovery using matrix factorization. *Proceedings of the national academy of sciences, 101*(12), 4164-4169.

Cai, D., He, X., Wu, X., & Han, J. (2008). Non-negative matrix factorization on manifold. . *In Data Mining ICDM'08. Eighth IEEE International Conference on* (pp. 63-72). IEEE.

Cao, X., Wang, X., Jin, D., Cao, Y., & He, D. (2013). Identifying overlapping communities as well as hubs and outliers via nonnegative matrix factorization. *Scientific reports,, 3,* 1-8.

Chandola, V., Banerjee, A., & Kumar, V. (2009). Anomaly Detection: A survey. *ACM Computing Surveys, 41*(3), 15.

Chau, D. H., Pandit, S., & Faloutsos, C. (2006). Detecting fraudulent personalities in networks of online auctioneers. *In Knowledge Discovery in Databases: PKDD 2006* (pp. 103-114). Berlin Heidelberg: Springer .

Cichocki, A., Zdunek, R., Phan, A. H., & Amari, S. I. (2009). *Nonnegative matrix and tensor factorizations: applications to exploratory multi-way data analysis and blind source separation.* John Wiley & Sons.

Freeman, L. C. (1997). A set of measures of centrality based on betweenness. *Sociometry*, 35-41.

Frigyesi, A., & Höglund, M. (2008). Non-negative matrix factorization for the analysis of complex gene expression data: identification of clinically relevant tumor subtypes. *Cancer informatics, 6*, 275.

Gao, H., Chen, Y., Lee, K., Palsetia, D., & Choudhary, A. N. (2012). Towards Online Spam Filtering in Social Networks. *In NDSS.*

Gilpin, S., Eliassi-Rad, T., & Davidson, I. (2013). Guided learning for role discovery (GLRD): framework, algorithms, and applications. *In Proceedings of the 19th*

ACM SIGKDD international conference on Knowledge discovery and data mining (pp. 113-121). ACM.

Gupta, M. (2013). Outlier detection for information networks. *(Doctoral dissertation University of Illinois at Urbana-Champaign).*

Haveliwala, T. H. (2003). Topic-sensitive pagerank: A context-sensitive ranking algorithm for web search. *Knowledge and Data Engineering, IEEE Transactions on, 15*(4), 784-796.

Hawkins, D. M. (1980). *Identification of outliers (Vol. 11).* London: Chapman and Hall.

Henderson, K., Gallagher, B., Li, L., Akoglu, L., Eliassi-Rad, T., Tong, H., & Faloutsos, C. (2011). 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.

Hodge, V. J., & Austin, J. (2004). A survey of outlier detection methodologies. *Atificial Intelligence Review*, *22*(2), 85-126.

Hutchins, L. N., Murphy, S. M., Singh, P., & Graber, J. H. (2008). Position-dependent motif characterization using non-negative matrix factorization. *Bioinformatics, 24*(23), 2684-2690.

Jeh, G., & Widom, J. (2002, July). SimRank: a measure of structural-context similarity. *In Proceedings of the eighth ACM SIGKDD international conference on Knowledge discovery and data mining* (pp. 538-543). ACM.

Kang, U., Tsourakakis, C. E., Appel, A. P., Faloutsos, C., & Leskovec, J. (2011). Hadi: Mining radii of large graphs. *ACM Transactions on Knowledge Discovery from Data (TKDD), 5*(2), 8.

Kim, J., & Park, H. (2008). Sparse nonnegative matrix factorization for clustering.

Koren, Y., Bell, R., & Volinsky, C. (2009). Matrix factorization techniques for recommender systems. *Computer, 42*(8), 30-37.

Kuang, D., & Park, H. (2013). Fast rank-2 nonnegative matrix factorization for hierarchical document clustering. *In Proceedings of the 19th ACM SIGKDD international conference on Knowledge discovery and data mining* (pp. 739-747). ACM.

Lee, D. L., & Seung, H. S. (1999). Learning the parts of objects by nonnegative matrix factorization. *Nature*, *401*(6755), 788-791.

Lee, D. L., & Seung, H. S. (2001). Algorithms for nonnegative matrix factorization. *Advances in Nueral Information Processing Systems 13: Proceedings of the 2000 Conferences* (pp. 556-562). MIT Press.

Lewis, T. G. (2011). *Network science: Theory and applications.* John Wiley & Sons.

Mairal, J., Bach, F., Ponce, J., & Sapiro, G. (2010). Online learning for matrix factorization and sparse coding. *The Journal of Machine Learning Research, 11*, 19-60.

McGlohon, M., Bay, S., Anderle, M. G., Steier, D. M., & Faloutsos, C. (2009). Snare: a link analytic system for graph labeling and risk detection. *In Proceedings of the 15th ACM SIGKDD international conference on Knowledge discovery and data mining* (pp. 1265-1274). ACM.

Moonesinghe, H. D., & Tan, P. N. (2006). Outlier detection using random walks. *In Tools with Artificial Intelligence, 2006. ICTAI'06. 18th IEEE International Conference on IEEE.*, (pp. pp. 532-539).

Noble, C. C., & Cook, D. J. (2003). Graph-based anomaly detection. *In Proceedings of the ninth ACM SIGKDD international conference on Knowledge discovery and data mining* (pp. 631-636). ACM.

Rennie, J. D., & Srebro, N. (2005). Fast maximum margin matrix factorization for collaborative prediction. *In Proceedings of the 22nd international conference on Machine learning.* 713-719: ACM.

Sun, H., Huang, J., Han, J., Deng, H., Zhao, P., & Feng, B. (2010, December). gskeletonclu: Density-based network clustering via structure-connected tree division or agglomeration. *In Data Mining (ICDM), 2010 IEEE 10th International Conference on* (pp. 481-490). IEEE.

Takács, G., Pilászy, I., Németh, B., & Tikk, D. (2008). Matrix factorization and neighbor based algorithms for the netflix prize problem. *In Proceedings of the 2008 ACM conference on Recommender systems* (pp. 267-274). ACM.

Tong, H., & Lin, C. Y. (2011). Non-Negative Residual Matrix Factorization with Application to Graph Anomaly Detection. *SDM*, 143-153.

Wang, F., Li, T., Wang, X., Z. S., & Ding, C. (2011). Community discovery using nonnegative matrix factorization. *Data Mining and Knowledge Discovery*, *22*(3), 493-521.

Wang, J. J., Wang, X., & Gao, X. (2013). Non-negative matrix factorization by maximizing correntropy for cancer clustering. *BMC bioinformatics*, *14*(1), 107.

Wang, Y., Jia, Y., Hu, C., & Turk, M. (2005). Non-negative matrix factorization framework for face recognition. *International Journal of Pattern Recognition and Artificial Intelligence*, *19*(04), 495-511.

Washio, T., & Motoda, H. (2003). State of the art of graph-based data mining. *Acm Sigkdd Explorations Newsletter*, *5*(1), 59-68.

Xie, S., Wang, G., Lin, S., & Yu, P. S. (2012). Review spam detection via temporal pattern discovery. *In Proceedings of the 18th ACM SIGKDD international conference on Knowledge discovery and data mining* (pp. 823-831). ACM.

Xu, X., Yuruk, N., Feng, Z., & Schweiger, T. A. (2007, August). Scan: a structural clustering algorithm for networks. *In Proceedings of the 13th ACM SIGKDD international conference on Knowledge discovery and data mining* (pp. 824-833). ACM.

Zhao, P., Han, J., & Sun, Y. (2009). P-Rank: a comprehensive structural similarity measure over information networks. *In Proceedings of the 18th ACM conference on Information and knowledge management* (pp. 553-562). ACM.

Zou, Z., Li, J., Gao, H., & Zhang, S. (2010). Mining frequent subgraph patterns from uncertain graph data. *Knowledge and Data Engineering,IEEE Transactions on*, *22*(9), 1203-1218.