Community Distribution Outlier Detection in Heterogeneous Information Networks

Manish Gupta[1], Jing Gao[2], and Jiawei Han[3]

[1] Microsoft, India (gmanish@microsoft.com)
[2] SUNY, Buffalo, NY (jing@buffalo.edu)
[3] UIUC, IL (hanj@illinois.edu)

Abstract. Heterogeneous networks are ubiquitous. For example, bibliographic data, social data, medical records, movie data and many more can be modeled as heterogeneous networks. Rich information associated with multi-typed nodes in heterogeneous networks motivates us to propose a new definition of outliers, which is different from those defined for homogeneous networks. In this paper, we propose the novel concept of *Community Distribution Outliers (CDOutliers)* for heterogeneous information networks, which are defined as objects whose community distribution does not follow any of the popular community distribution patterns. We extract such outliers using a type-aware joint analysis of multiple types of objects. Given community membership matrices for all types of objects, we follow an iterative two-stage approach which performs pattern discovery and outlier detection in a tightly integrated manner. We first propose a novel outlier-aware approach based on joint non-negative matrix factorization to discover popular community distribution patterns for all the object types in a holistic manner, and then detect outliers based on such patterns. Experimental results on both synthetic and real datasets show that the proposed approach is highly effective in discovering interesting community distribution outliers.

1 Introduction

Heterogeneous information networks are omnipresent. In such networks, the nodes are of different types and relationships between nodes are encoded using multi-typed edges. For example, bibliographic networks consist of authors, conferences, papers and title keywords. Edges in such a network represent relationships like “an author collaborated with another author”, “an author published in a conference”, and so on. Analysts often perform community detection on such networks with an aim of understanding the hidden structures more deeply. Although methods designed for homogeneous networks can be applied by extracting a set of homogeneous networks from the heterogeneous network, such a transformation causes inevitable information loss. For example, when converting bibliographic networks to co-authorship networks, some valuable connectivity information, e.g., paper title or conference an author published in, is lost. As objects of different types interact strongly with each other in the network, analysis on heterogeneous information networks at various levels must be conducted simultaneously from multiple types of data. Such an analysis will help in exploiting the shared hidden structure of communities across object types, i.e., the common patterns across types that can

explain the generation of these community distributions. For example, in bibliographic networks, when grouping authors based on their "research area" distributions, one must use the knowledge of the grouping of "research area" distributions for related conferences and keywords. This is because (1) the community space (research areas) is the same across different object types, and (2) all these objects interact strongly with each other in the network.

Although most of the objects in a heterogeneous network follow common community distribution patterns which can be uncovered by joint analysis of community membership of multiple heterogeneous object types, certain objects deviate significantly from these patterns. It is important to detect such outliers in heterogeneous information networks for de-noising data thereby improving the quality of the patterns and also for further analysis. Therefore, in this paper, we propose to detect such anomalous objects as *Community Distribution Outliers* (or *CDOutliers*) given the community distribution of each object of every type. In the following, we present a few *CDOutlier* examples and discuss the importance of identifying such outliers in real applications.

CDOutlier Examples Consider a bibliographic network where the research area label associated with an author node depends on the community labels of the conferences where he publishes, keywords he uses in the title of the papers, and the other authors he collaborates with. There may exist some popular community distribution patterns extracted by analysis across various object types, which majority of the objects follow. For example, say there are four communities: data mining (DM), software engineering (SE), compilers (C) and machine learning (ML). Then popular distribution patterns could be (DM:1, SE:0, C:0, ML:0), (DM:0, SE:1, C:0, ML:0), (DM:0, SE:0, C:1, ML:0), (DM:0, SE:0, C:0, ML:1), and (DM:0.7, SE:0, C:0, ML:0.3). Then, an author who contributes to DM and C (with a distribution like (DM:0.5, SE:0, C:0.5, ML:0)) would be considered as a *CDOutlier*. Furthermore, there could be subtle patterns like (DM:0.8, Energy:0.2), i.e., 80% probability belonging to DM and 20% probability in Energy, which is followed by majority of the objects. If an author's community distribution is (DM:0.2, Energy:0.8), which deviates from the majority pattern, then he is considered as a *CDOutlier*. Similarly, one could compute outliers among other types of objects, such as conferences and title keywords, based on the popular distribution patterns derived by holistic analysis across all object types.

Besides these examples, applications of *CDOutliers* can be commonly observed in real-life scenarios, and we briefly mention a few here. (1) In the Delicious network, most users who tag pages about "Tech and Science" do not tag pages about "Arts and Design". A user doing so (user with unusual skill combinations) can be considered as a *CDOutlier*. (2) In the Youtube network, most of the users would be interested in videos of a particular category. However, certain users who act as middlemen in publishing and uploading videos may interact with videos of many different categories and would be detected as *CDOutliers*.

CDOutlier distributions should not be confused with "hub" distributions (i.e., distributions with high entropy) over communities. Certain "hub" distributions could be frequent patterns, but only those that are very rare should be labeled as *CDOutliers*. On the other hand, not all *CDOutlier* distributions have high entropy.

Brief Overview of CDOutlier Detection Given the *soft* community distributions for each object of every type, one can compute distribution patterns. *CDOutliers* are objects that defy the trend, and the trend must be obtained from accurate pattern discovery. However, pattern discovery suffers from the presence of *CDOutliers* itself. Therefore, given community detection results, we design an iterative two-stage procedure to identify *CDOutliers*, which integrates community distribution pattern discovery and *CDOutlier* detection. First, we discover popular distribution patterns for all the object types together by performing a joint nonnegative matrix factorization (NMF) on the community distribution matrices, such that it ignores the outliers discovered in the previous iteration. At the second step, the outlierness score for an object is computed based on its distance from its nearest distribution pattern. The algorithm iterates until the set of outliers discovered do not change. Thus, distribution pattern discovery and outlier detection are improved through iterative update procedures, and upon convergence, meaningful outliers are output.

Summary Our contributions are summarized as follows.

- We introduce the notion of identifying *CDOutliers* from heterogeneous networks based on the discovery of community distribution patterns.
- We propose a unified framework based on joint-NMF formulation, which integrates the discovery of distribution patterns across multiple object types and the detection of *CDOutliers* based on such patterns together.
- We show interesting and meaningful outliers detected from multiple real and synthetic datasets.

Our paper is organized as follows. In Sec. 2, we introduce the notion of distribution patterns and develop our method to extract heterogeneous community trends for objects of different types in the form of popular distribution patterns. In Sec. 3, we present discussions related to practical usage of the algorithm. We discuss datasets and results with detailed insights in Sec. 4. Finally, related work and conclusions are presented in Sec. 5 and 6 respectively.

2 CDOutlier Detection Approach

Notation	Meaning
τ_k	k^{th} object type
k, l	Index for a type of objects
N_k	Number of objects of type k
K	Number of types of objects
C	Number of communities
C'	Number of distribution patterns
$T_k^{N_k \times C}$	Membership matrix for objects of type k
$W_k^{N_k \times C'}$	Distribution pattern indicator matrix for objects of type k
$H_k^{C' \times C}$	Distribution patterns matrix for objects of type k
O_k	Outlier objects set for type k
α	Regularization Parameter

Fig. 1. Table of Notations

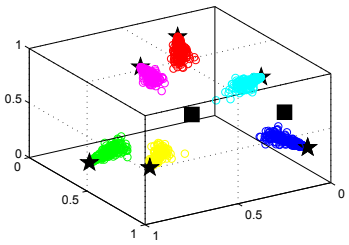

Fig. 2. Distribution Patterns in 3D Space

In this section, we will present our iterative two-stage approach for *CDOutlier* detection. Table 1 shows the important notations we will use in this paper. We denote an

element (i, j) of a matrix A by $A_{(i,j)}$. More details about the notations will be found in the following problem definition.

2.1 Problem Definition

We start with introduction to a few basic concepts.

Community Consider a heterogeneous network with K types of objects $\{\tau_1, \tau_2, \ldots, \tau_K\}$. A community is a probabilistic collection of similar objects, such that similarity between objects within the community is higher than the similarity between objects in different communities. For example, a research area is a community in a bibliographic network. For heterogeneous networks, one is often interested in identifying heterogeneous communities which contain objects of different types. We will use C to denote the number of communities.

Membership Matrix Membership matrix T is a matrix such that the element T_{oi} corresponds to the probability with which an object o belongs to a community i. The rows of the matrix correspond to objects while the columns correspond to communities. Let $N_1, N_2, \ldots, N_K$ be the number of objects of each type. Let $T_1, T_2, \ldots, T_K$ denote the membership matrices for the objects of types $\tau_1, \tau_2, \ldots, \tau_K$ respectively. Thus, the membership matrix T_k is of size $N_k \times C$.

Distribution Patterns The rows of a membership matrix can be grouped into clusters. To be able to capture inter-type interactions, such clusters should be obtained using a joint analysis of membership matrices of all types. The cluster centroid of each such cluster denotes a representative distribution in the community space. We call these cluster centroids as distribution patterns. For example, in Figure 2, we plot a membership matrix with C=3. Each axis represents probability of membership for the corresponding community. Different colors represent objects following different patterns. Black stars (★) are the representatives (cluster centroids) used to represent the distribution patterns.

Community Distribution Outlier An object o in a heterogeneous network, is a *CD-Outlier* if its distance to the closest distribution pattern, which is obtained by a joint analysis of all the object types, is very high. For example, in Figure 2, the *CDOutlier* points are marked as black squares (■).

Communities and hence distribution patterns discovered from a heterogeneous network are very different from those obtained by processing a homogeneous projection of a heterogeneous network. Thus, *CDOutliers* are quite different from the community outliers obtained using homogeneous network analysis [6].

Community Distribution Outlier Detection Problem

Input: Community membership matrices $T_1, T_2, \ldots, T_K$ for the types $\tau_1, \tau_2, \ldots, \tau_K$.

Output: Top κ outlier objects of each type that deviate the most from distribution patterns for that type.

For example, for DBLP, the types are τ_1 =author, τ_2 =conference and τ_3 =keywords, and research areas are communities. T_1 will then be a matrix where each row denotes the probability with which an author belongs to various research areas. The expected output is top few authors (conferences, keywords) that deviate the most from the popular research area distribution patterns for the author (conference, keyword) type.

We will solve this problem using an iterative two-stage approach. In the first stage, distribution patterns are discovered ignoring the outliers detected in the previous iteration. In the second stage outliers are detected based on the patterns discovered at the first stage within the same iteration. The proposed pattern discovery step is a joint Non-negative Matrix Factorization (NMF) process, and thus we will first discuss basics about NMF in the next section. We then introduce the two stages in Sections 2.3 and 2.4, and finally present the complete algorithm.

2.2 Brief Overview of NMF

Given a non-negative matrix $T \in \mathbb{R}^{N \times C}$ (each element of T is ≥ 0), the basic NMF problem formulation aims to compute a factorization of T into two factors $W \in \mathbb{R}^{N \times C'}$ and $H \in \mathbb{R}^{C' \times C}$ such that $T \approx WH$. Both matrices W and H are constrained to have only non-negative elements in the decomposition.

It has been shown earlier ([4]) that NMF is equivalent to a relaxed form of *KMeans* [16] clustering. NMF can be considered as a form of clustering over the matrix T. Each row of H represents a cluster centroid (or a distribution pattern) in the C-dimensional space. Thus, H contains the information about the C' cluster centroids obtained by clustering T. Each element of row r of W represents the probability with which the object corresponding to row r belongs to the different clusters. Generally, the loss function used to represent the error between T and WH is the element-wise Euclidean distance. Thus the typical NMF can be expressed as the following optimization problem.

$$\min_{W,H} ||T - WH||^2 \tag{1}$$

$$subject\ to\ the\ constraints$$

$$W \geq 0, H \geq 0 \tag{2}$$

where $||A||$ is the sum of the square of each element in the matrix A.

2.3 Discovery of Distribution Patterns

In this sub-section, we will discuss how to learn distribution patterns from community membership matrices. These patterns will form the basis for outlier detection which we will discuss in Section 2.4.

For a homogeneous network, any clustering algorithm could be run over the community membership matrix to obtain distribution patterns. However, the case of heterogeneous networks is challenging. Each of the membership matrices T_k can be clustered individually (using the basic NMF) to obtain distribution patterns for that type. However, since all the membership matrices are defined for objects that are connected to each other, the hidden structures that can explain these objects' communities should be consistent across types. Also, the membership matrices represent objects in the same space of C components. Hence the clustering of matrix T_i should correspond to the clustering of matrix T_j for all $1 \leq i, j \leq K$. In other words, the divergence between any pair of clusterings should be low.

This intuition can be encoded in the form of an optimization problem, which conducts Non-negative Matrix Factorization (NMF) over multiple matrices together. In the

proposed problem setting, each of the matrices in the set $T = \{T_1, T_2, \ldots, T_K\}$ needs to be factorized, and we expect them to share a lot of common factors or have factors which are quite similar to each other. We will factorize each matrix $T_k \in \mathbb{R}^{N_k \times C}$ into two factors $W_k \in \mathbb{R}^{N_k \times C'}$ and $H_k \in \mathbb{R}^{C' \times C}$. Also, we need to ensure that clustering across different types is somewhat related. We achieve this by introducing a new term $||H_k - H_l||^2$ to the basic NMF optimization objective function, and a parameter α which decides what degree of correspondence should be obtained across clusterings.

Problem Formulation Based on the above discussion, the problem can be formulated as an optimization problem as follows. Let W and H represent the set of matrices $\{W_1, W_2, \ldots, W_K\}$ and $\{H_1, H_2, \ldots, H_K\}$ respectively.

$$\min_{W,H} \sum_{k=1}^{K} \{||T_k - W_k H_k||^2\} + \alpha \sum_{\substack{k=1 \\ l=1 \\ k<l}}^{K} \{||H_k - H_l||^2\} \tag{3}$$

$$subject\ to\ the\ constraints$$

$$W_k \geq 0 \quad \forall k = 1, 2, \ldots, K \tag{4}$$

$$H_k \geq 0 \quad \forall k = 1, 2, \ldots, K \tag{5}$$

For example, for DBLP, τ_1=author, T_1 is the research-area distribution matrix for the author type. Each row of H_1 represents a distribution pattern for the author type and each row of W_1 denotes the probability with which the author belongs to the C' author distribution patterns.

The objective function in Eq. 3 is quadratic with respect to W_k or H_k when the other variable matrices are fixed. Converting to Lagrangian form by introducing the Lagrangian multiplier matrix variables $P = \{P_1, P_2, \ldots, P_K\}$ and $Q = \{Q_1, Q_2, \ldots, Q_K\}$, we obtain the following.

$$\min_{W,H,P,Q} \sum_{k=1}^{K} \{||T_k - W_k H_k||^2\} + \alpha \sum_{\substack{k=1 \\ l=1 \\ k<l}}^{K} \{||H_k - H_l||^2\} + \sum_{k=1}^{K} \{tr(P_k W_k^T) + tr(Q_k H_k^T)\} \tag{6}$$

KKT optimality conditions require the following.

$$\frac{\partial \left[||T_k - W_k H_k||^2 + \alpha \sum\limits_{\substack{l=1 \\ k \neq l}}^{K} ||H_l - H_k||^2 \right]}{\partial H_{k_{(i,j)}}} = Q_{k_{(i,j)}} \forall k = 1, 2, \ldots, K \tag{7}$$

$$\frac{\partial \left[||T_k - W_k H_k||^2 \right]}{\partial W_{k_{(i,j)}}} = P_{k_{(i,j)}} \quad \forall k = 1, 2, \ldots, K \tag{8}$$

Also, the complementary slackness conditions can be expressed as follows.

$$Q_{k_{(i,j)}} \times H_{k_{(i,j)}} = 0 \quad \forall i, j, k \tag{9}$$

$$P_{k_{(i,j)}} \times W_{k_{(i,j)}} = 0 \quad \forall i, j, k \tag{10}$$

Substituting Eqs. 7 and 8 into Eqs. 9 and 10 respectively, we get the following.

$$\left[W_k^T W_k H_k - W_k^T T_k + \alpha \sum_{\substack{l=1 \\ k \neq l}}^{K} \left(I^{C' \times C'} H_k - I^{C' \times C'} H_l \right) \right]_{(i,j)} \times H_{k_{(i,j)}} = 0 \quad \forall i, j, k \tag{11}$$

$$\left[W_k H_k H_k^T - T_k H_k^T \right]_{(i,j)} \times W_{k_{(i,j)}} = 0 \quad \forall i, j, k \tag{12}$$

These set of equations can be solved using the following iterative equations.

$$W_k \leftarrow W_k \odot \frac{T_k H_k^T}{W_k H_k H_k^T} \quad \forall k = 1, 2, \ldots, K \tag{13}$$

$$H_k \leftarrow H_k \odot \frac{W_k^T T_k + \alpha \sum\limits_{\substack{l=1 \\ k \neq l}}^{K} I^{C' \times C'} H_l}{W_k^T W_k H_k + \alpha \sum\limits_{\substack{l=1 \\ k \neq l}}^{K} I^{C' \times C'} H_k} \quad \forall k = 1, 2, \ldots, K \tag{14}$$

Here $\odot$ denotes the Hadamard product (element-wise product) and $\frac{A}{B}$ denotes the element-wise division, i.e. $\left(\frac{A}{B}\right)_{i,j} = \frac{A_{ij}}{B_{ij}}$.

2.4 Community Distribution Outlier Detection

Using the joint-NMF formulation described in the previous sub-section, we obtain the matrices $\{W_k\}_{k=1}^{K}$. Each row of H_k is a distribution pattern (a cluster centroid) and each element (i, j) of W_k denotes the probability with which object i belongs to the distribution pattern j. We define the outlier score of an object as the distance of the object i of type T_k from the nearest cluster centroid. Thus, the outlier score for an object i, $OS(i)$ can be written as follows.

$$OS(i) = \underset{j}{\operatorname{argmin}}\ Dist(T_{k_{(i,\cdot)}}, H_{k_{(j,\cdot)}}) \tag{15}$$

An object which is far away from its nearest cluster centroid gets a high outlier score. Using this outlier definition, one can find outlier scores for all objects of all types. Top κ objects with highest outlier scores for each type can be marked as outliers.
Iterative Refinement If the input data contains outliers, the distribution patterns will try to overfit to those outliers and hence will be distorted compared to the actual hidden structure of the clean data, so the distribution pattern discovery needs to be outlier-aware. Similarly, if the distribution patterns are accurate, outlier detection will be of a high quality. Therefore, we propose to perform the steps of pattern discovery and outlier detection iteratively until convergence. At each iteration, while performing pattern discovery we ignore the set of top-κ outliers from each type. For outlier detection, we use the patterns discovered during the same iteration, to compute outlier scores for all the objects of all types. Empirically we observed that such an iterative refinement always converges. However in case the algorithm oscillates (i.e., enters a loop where the set of outliers detected repeats), the algorithm can be terminated when the set of outliers detected after any iteration is the same as the one detected in any previous iteration.

We summarize the outlier detection algorithm in Algorithm 1. We initialize the set of outliers of each type to an empty set (Step 1). The set of outliers is updated iteratively and the algorithm terminates when the outliers detected across two consecutive iterations are the same. Within every iteration, we first obtain T_k for that iteration by removing the rows corresponding to the current outliers from the original membership matrix (Step 6). NMF is sensitive to initialization and hence we initialize W_k's and H_k's using clusters discovered by running *KMeans* [16] on T_k (Step 7). Steps 6 to 13 correspond to pattern discovery using joint-NMF. Steps 14 to 17 correspond to outlier detection based on the discovered patterns. Finally, the outlier objects are returned.

Algorithm 1 *CDOutlier* Detection Algorithm (CDODA)

Input: (1) Cluster membership matrices $T = \{T_1, T_2, \ldots, T_K\}$ corresponding to objects of types $\tau = \{\tau_1, \tau_2, \ldots, \tau_K\}$, (2) α, (3)κ.
Output: Top κ *CDOutlier* objects of each type ($\{O_1, O_2, \ldots, O_K\}$).
1: Initialize each element of $currOutliers = \{O_1, O_2, \ldots, O_K\}$ to ϕ.
2: Initialize each element of $prevOutliers = \{O'_1, O'_2, \ldots, O'_K\}$ to $null$.
3: $\{origT_k \leftarrow T_k\}_{k=1}^{K}$
4: **while** checkForChange($currOutliers, prevOutliers$) **do**
5: $prevOutliers \leftarrow currOutliers$
6: $\{T_k \leftarrow origT_k -$ rows corresponding to $O_k\}_{k=1}^{K}$ ▷ Pattern Discovery
7: Initialize $\{W_k\}_{k=1}^{K}$ and $\{H_k\}_{k=1}^{K}$ using $\{KMeans(T_k)\}_{k=1}^{K}$.
8: **while** NOT converged **do**
9: **for** $k = 1$ to K **do**
10: Update W_k using Eq. 13.
11: Update H_k using Eq. 14.
12: **end for**
13: **end while**
14: **for** $k = 1$ to K **do** ▷ Outlier Detection
15: Compute outlier scores for all objects of type τ_k.
16: $O_k \leftarrow$ top κ objects of type τ_k with highest outlier scores.
17: **end for**
18: **end while**

3 Discussions

In this section, we analyze the time complexity of the proposed *CDOutlier* detection method. We also discuss several important issues in implementing the method.

Initialization The joint-NMF formulation will converge to a *local* optimum, and thus it could be sensitive to initialization. Therefore, it is very important to choose an appropriate initialization for the algorithm. To initialize the matrix H_k, we run *KMeans* [16] on the matrix T_k. W_k is then computed by finding the nearest cluster for each object and setting the corresponding entry in W_k to 1.

Computational Complexity The time required for an update to a W_k or H_k matrix is $O(NKC'^2)$. Thus, the pattern discovery phase has a complexity of $O(K^2INC'^2)$, where I is the number of iterations for joint-NMF and N is the average number of objects per type. The outlier detection phase consists of finding top κ outliers per type which can be done in $O(KNlog(\kappa))$ time. Let the number of iterations for the external While loop (Steps 4 to 18) be I'. Thus, the overall complexity of the algorithm is $O(NI'K(KIC'^2 + log(\kappa)))$. Note that $I'K(KIC'^2 + log(\kappa))$ becomes a small constant when N is large. Thus the algorithm is linear in the number of objects.

Selecting Parameters (α and κ) α determines the amount of regularization applied when performing the joint-NMF. If we set α to 0, it is as good as performing NMF sep-

arately. A high value of α will favor a solution where there are many shared distribution patterns across various types, while a low value of α will try to fit the NMF for each of the types individually without trying to discover any shared distribution patterns. Hence, the setting of the parameter α is important and domain dependent. If we believe that the objects of different types interact a lot all across the network, we should use a higher value for α for better results. κ can be selected based on the percentage of outliers expected. Another way of principled thresholding is to set the variance level, for example, consider any point as an outlier if it is at least two standard deviations away from the nearest cluster centroid.

4 Experiments

Evaluation of outlier detection algorithms is quite difficult due to lack of ground truth. We generate multiple synthetic datasets by injecting outliers into normal datasets, and evaluate outlier detection accuracy of the proposed algorithms on the generated data. We also conduct case studies by applying the method to real data sets. We perform comprehensive analysis to justify that the top few outliers returned by the proposed algorithm are meaningful. The code and the data sets are available at: `http://dais.cs.uiuc.edu/manish/CDOutlier/`

4.1 Baselines

Community Distribution Outlier Detection Algorithm (*CDO*) is the proposed method. The baseline methods (*SI* and *Homo*) are explained as follows.

SingleIteration (SI) As described in Algorithm 1, *CDO* performs community pattern discovery and outlier detection iteratively until the set of top κ outliers for each type do not change. *SI* is a simpler version of *CDO*, which performs only one iteration. Thus the pattern discovery phase in *SI* suffers from the presence of *CDOutliers*. This baseline will help us evaluate the importance of ignoring the *CDOutlier* noise when computing the distribution patterns.

Homogeneous (Homo) *CDO* performs pattern discovery using joint-NMF across multiple types. In contrast to this, the baseline *Homo* treats all objects to be of the same type and then performs distribution pattern discovery using a single matrix NMF. This baseline will help us evaluate the importance of modeling heterogeneous data types rather than reducing them to homogeneous ones in heterogeneous information networks.

4.2 Synthetic Datasets

Dataset Generation

We generate our synthetic dataset as follows. The dataset is represented by the matrices T_k for $1 \leq k \leq K$. We start by generating H_k and W_k and then obtain $T_k = W_k H_k$. We first generate a single matrix $H^{C' \times C}$ which we consider as a template for generating the distribution patterns. It appears across different types in a slightly perturbed form. H is generated as follows. We first fix $C' = 2C$. Next, each cluster

centroid (a row of H) could be an impulse probability distribution function at different dimensions or could have non-zero random probability value for 2 dimensions. Perturb H randomly such that all objects of the same type follow the same fixed perturbation to get $H_1, \ldots, H_K$ (Recall K=Number of types). Such a perturbation captures the fact that clusters across different types of objects deviate slightly from each other. Then $\{W_k\}_{k=1}^{K}$ are generated such that one element in every row is close to 1, and the remaining probability mass is distributed uniformly among other elements. These W_k's and H_k's could then be used to generate $\{T_k = W_k H_k\}_{k=1}^{K}$.

Outliers are injected as follows. First we set an outlierness factor Ψ and choose a random set of objects, R_k with $N_k \times \Psi$ objects of type k. For each object o in R_k, we choose either a pattern randomly from some other type $k' \neq k$ or a pattern quite different from any pattern in H_k's. We use this pattern to define the row in T_k corresponding to the object o, i.e., $T_{k_{(o,.)}}$. Note that patterns in different types are reasonably different from each other. Hence, such an object which follows a pattern from some other type, or a completely different pattern from H itself, can be considered as an outlier for type k.

Results on Synthetic Datasets

We generate a variety of synthetic datasets capturing different experimental settings. For each setting, we perform 20 experiments and report the average values. We fix the threshold for NMF objective function convergence to 0.01. We vary the number of objects as 1000, 2000 and 5000. We also study the accuracy with respect to variation in number of object types (2, 3, 4) and variation in the number of communities (4, 6, 8, 10). We also vary the percentage of injected outliers as 1%, 2% and 5%. We fixed α=0.5 for our experiments. Using these settings, we compare the actual outlier objects with the top outliers returned by various algorithms. For each algorithm, we show the accuracy with respect to matches in the set of detected outliers and the set of injected outliers, in Table 1 (False Positives(%)=100-accuracy). Results for $C = 6, 8$ are also similar and we omit them for lack of space. For each experimental setting, we show the best accuracy obtained in bold. Each of the accuracy values is obtained by averaging the accuracy across all types of objects for that experimental setting (across 20 runs). Average standard deviations are 3.07% for *CDO*, 3.48 % for *SI* and 2.19% for *Homo*. As the table shows, the proposed algorithm outperforms both of the other algorithms for most of the settings by a wide margin. On an average across all experimental settings, *CDO* is 2.85% better than *SI* and 21.5% better than *Homo*. In general, the accuracy of the proposed algorithm decreases slightly as the amount of outlierness increases to 5%.

4.3 Running Time and Convergence

The experiments were run on a Linux machine with 4 Intel Xeon CPUs with 2.67GHz each. The code was implemented in Java. *KMeans* [16] implementation of Weka [11] was used for initialization of the H_k and W_k matrices. Figure 3 shows the execution time for *CDO* algorithm for different number of object types. Note that the algorithm is linear in the number of objects. These times are averaged across multiple runs of the algorithm across different settings for degree of outlierness and number of communities.

Figure 4 shows the decrease in the objective function value with respect to the number of iterations for different dataset sizes (for K=3 and C=10). The figure shows that

the joint-NMF algorithm converges well. The average number of iterations for convergence of joint-NMF are 118, 173 and 242 for datasets of sizes 1000, 2000 and 5000 respectively.

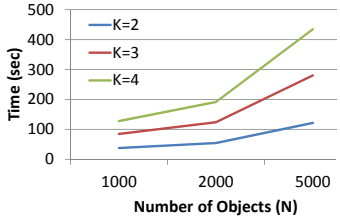

Fig. 3. Running Time (sec) for *CDO* (Scalability)

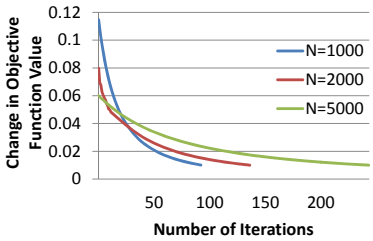

Fig. 4. Convergence of joint-NMF

On an average across all experimental settings, the proposed algorithm *CDO* takes the following number of external iterations (I') of pattern discovery and outlier detection: 6.21 for N=1000, 6.98 for N=2000 and 7.66 for N=5000.

Table 1. Synthetic Dataset Results (*CDO*=The Proposed Algorithm CDODA, *SI*= Single Iteration Baseline, *Homo*=Homogeneous (Single NMF) Baseline) for C=4 (left) and C=10 (right)

N	Ψ (%)	$\|K\|=2$ CDO	SI	Homo	$\|K\|=3$ CDO	SI	Homo	$\|K\|=4$ CDO	SI	Homo
1000	1	**92**	91.5	52	**81.3**	80	53.7	73.8	**75**	54.2
	2	**94.2**	85.8	60	**83.3**	83	57.3	**76.1**	75.4	56.4
	5	**86.5**	70.5	59.5	**74.7**	67.8	57.2	**71**	64.4	55.6
2000	1	**95**	91	56.5	81.2	**81.3**	54.8	73.1	**74.5**	52.1
	2	**90.4**	86.1	57.1	**81.8**	78.3	55.2	**74.2**	73.8	52.3
	5	**91.7**	72.8	58	**73.4**	65.4	57.2	**74**	67.7	55.4
5000	1	**92.1**	86.4	52.3	**80.9**	78.4	56.3	**72.8**	69.1	51.6
	2	**95.4**	94.4	56	**79.9**	77.2	54.6	**74.6**	74	53.8
	5	**88.5**	68	60.7	**80.4**	66.7	57.9	**74.8**	65.9	56.8

N	Ψ (%)	$\|K\|=2$ CDO	SI	Homo	$\|K\|=3$ CDO	SI	Homo	$\|K\|=4$ CDO	SI	Homo
1000	1	**97**	90.5	51	**78**	74.3	51.3	**69.5**	68.2	52.8
	2	**81.8**	81.2	55	**67.3**	66.8	56.8	**65.9**	65.6	59
	5	**78.6**	77.2	59.4	**69.2**	69.1	58.3	68.8	**69**	56
2000	1	**79.2**	78	55.5	**72.7**	71.5	58.2	71.9	**72.2**	56.6
	2	**79**	78.2	55.8	68.1	**68.2**	59.2	65.4	**65.9**	56.1
	5	**74.4**	72.4	61.5	73.1	**73.4**	58.4	66.4	**67.2**	56.2
5000	1	**97.1**	85.7	54.3	**77.8**	71.2	54.9	**69.3**	69	58.3
	2	**75.8**	74.4	57.1	68.9	**69.3**	56.9	69.3	**70.8**	57.3
	5	**75**	72.1	61.2	**70.2**	69.5	57.9	68.2	**69.9**	56.3

4.4 Regularization Parameter Sensitivity

The joint-NMF optimization problem (Eq. 3) includes a regularization parameter α. We study the sensitivity of the algorithm with respect to this parameter. Table 2 shows the accuracy of the proposed *CDO* algorithm for K=3 and C=6. Across different settings of the number of objects (N) and the degree of outlierness (Ψ), the table shows that the accuracy is not sensitive to the value of α. We observe that the algorithm provides good accuracy for any value of α between 0 and 1. Note that α decides how much importance the algorithm gives to the quality of object clustering within one type versus matches between clusters obtained across types. Thus, α should be decided for any dataset based on the size of the dataset and its inter-type cluster structure similarity.

4.5 Real Datasets

Dataset Generation

We perform experiments using 2 real datasets: *DBLP* and *Delicious*. We use *NetClus* [20] to perform community detection on the datasets since it uses both data and

Table 2. Regularization Parameter Sensitivity for K=3,C=6

N	Ψ (%)	α					
		0	0.2	0.5	0.8	1	10
1000	1	85.0	86.3	86.0	86.3	86.3	86.0
	2	81.5	83.5	83.3	82.8	82.7	82.2
	5	64.2	67.2	66.9	66.4	68.1	66.7
2000	1	82.1	85.5	85.8	85.5	85.3	83.5
	2	74.7	78.6	81.0	80.2	80.3	77.7
	5	62.3	70.6	70.5	70.5	70.4	69.9
5000	1	80.1	84.5	84.6	84.5	84.5	83.3
	2	79.6	82.3	82.7	83.9	83.9	84.1
	5	65.6	72.1	71.9	72.0	71.8	71.4

link information for clustering and is specifically designed to handle heterogeneous networks. *NetClus* outputs the matrices $T_1, \ldots, T_K$ which we use as input for the proposed outlier detection algorithm. We found that the proposed method provides much more interesting top outliers compared to the *Homo* baseline and we provide case studies using *CDO* only, for lack of space.

DBLP: The *DBLP* network consists of papers, authors, keywords and conferences. We considered a temporal subset of DBLP[4] for 2001-2010. We removed authors with $<$10 papers during that time period. Our dataset consists of $\sim$650K papers, $\sim$480K authors, 3900 conferences, $\sim$107K keywords and 14 research areas. We obtained a list of conferences from the Wikipedia Computer Science Conferences page[5] which labels conferences by research areas. By associating keywords from these conferences with research areas, we obtained term priors which were used as input for *NetClus*. We consider each research area as a community, and thus the number of communities is 14. We experimented with C'=28 (twice the number of communities), α=0.5 and κ =1%.

Delicious: The *Delicious* network consists of tagging events, users, URLs and tags. The dataset consists of all tagging events performed by a randomly chosen list of $\sim$73K users from July 1 to July 28, 2010. The tagging events were obtained as RSS feeds[6] and were processed to obtain the desired network. Delicious provides a basic categorization on the home page[7]. We scrap category pages linked from home page to associate keywords with the categories. We consider these categories as communities and hence use the number of communities as 10 when running *NetClus* on the Delicious data. The categorized keywords are used to supply term priors for *NetClus*. Our Delicious dataset consists of $\sim$73K users, $\sim$1.3M tagging events, $\sim$902K URLs, $\sim$273K tags and 10 categories. We experimented with C'=20 (twice the number of communities), α=0.5 and κ =1%.

Results on Real Datasets

Running time for the algorithm is about 1.5 hours for both the datasets. Here, we will discuss case studies obtained from these datasets. We analyze the top 2 outliers of each type from the 2 datasets in terms of their community distribution. Objects that have very small frequency of occurrence may not have an appropriate community distribution. Hence, we analyze objects with at least 10 links in the network. Note that the outliers

[4] `http://www.informatik.uni-trier.de/~ley/db/`

[5] `http://en.wikipedia.org/wiki/List_of_computer_science_conferences`

[6] `http://feeds.delicious.com/v2/rss/`

[7] `http://delicious.com/`

for each type have been obtained using a joint hidden structure analysis across multiple types, and hence are quite different from outliers obtained using homogeneous network analysis [6].

DBLP

In *DBLP*, we observe specialization in one of the 14 categories as clear patterns. Multiple types of objects share a few patterns, which combine several areas, for example, ("Databases":0.8, "Computational Biology":0.2). However, some of the other patterns with combinations of research areas are specific to particular types. For example, the pattern ("Software engineering":0.3, "Operating systems":0.6, "others":0.1) is observed for conferences but not for other types. Similarly, the pattern ("Concurrent Distributed and Parallel Computing":0.5, "Security and privacy":0.45, "others":0.05) is observed specifically for authors while ("Security and privacy":0.8,"Education":0.2) is observed specifically for title keywords. Thus some patterns are shared across types while others are slightly different. This stresses the need for a joint-NMF-based clustering.

Authors: Most of the authors publish frequently in such "commonly-paired" categories or in a single category of their expertise. However our top outliers show interesting combinations as follows. (Note that the community membership probabilities are shown in brackets and may not add up to 1; the residual is spread across other communities.)

(1) Giuseppe de Giacomo: Algorithms and Theory (0.25), Databases (0.47), Artificial Intelligence (0.13), Human Computer Interaction (0.06). Note that the combination of Algorithms and Theory, Databases and Artificial Intelligence with small contributions to HCI is rare and hence interesting.

(2) Guang R Gao: Concurrent Distributed and Parallel Computing (0.41), Computer Architecture (0.3), Computational Biology (0.27). Similar to the case above, this combination of the research areas is quite rare.

Conferences: Among the top conference outliers are conferences that span across multiple streams of computer science. The top 2 conference outliers are as follows.

(1) From integrated publication and information systems to virtual information and knowledge environments[8]: Databases (0.5), Artificial Intelligence (0.09), Human Computer interaction (0.4). This conference is special because it celebrates an *occasion* (65^{th} birthday of Erich J. Neuhold). From the name itself the reader can guess the wide nature of this conference.

(2) International Conference on Modelling and Simulation: Programming languages (0.18), Security and privacy (0.29), Databases (0.39), Computer Graphics (0.13). Again, this combination is quite rare.

Keywords: Finally, we also list the top 2 paper title keywords with high outlierness scores.

(1) military: Algorithms and theory (0.02), Security and Privacy (0.37), Databases (0.22), Computer Graphics (0.37). Lots of military sponsored research and paper motivations containing military scenarios results in such a diverse distribution for "military".

(2) inventory: Security and Privacy (0.29), Databases (0.31), Computer Graphics (0.34), Computational Biology (0.03). The nearest matching pattern for this one was (Databases:

[8] `http://dblp.dagstuhl.de/db/conf/birthday/neuhold2005.html`

0.8, Computational Biology: 0.2). But usually computer graphics and security and privacy are not associated with these.

Delicious

In *Delicious*, we observe specialization in one of the 10 categories as clear patterns, as expected. Different types of objects share a few patterns, which corresponds to combinations of categories, for example, "Education" and "Tech and Science". However, some of the other patterns with combinations of categories are specific to particular types. For example, "Arts and Design" and "Tech and Science" is observed for URLs but not for other types. Similarly, the pattern "Arts and Design" and "Entertainment" is observed specifically for users and "Lifestyle" and "Sports" is observed specifically for tags. Thus even in the Delicious dataset, some patterns are shared across types while others are slightly different.

Users: Most of the users (who tag a sizeable number of pages) tag pages related to a particular category only. However, there are some users who are experts across multiple categories. Sometimes their interests are quite diverse and do not follow patterns of other users. Here, we report top 2 users that the proposed algorithm reported as outliers, along with the probabilistic categories they belong to. Usually lifestyle and travel are highly correlated with food, unlike for the user "saassaga".

(1) saassaga: Arts and Design (0.25), Food (0.04), Lifestyle (0.35), Travel (0.34)

(2) lbbrad: Food (0.24), Lifestyle (0.37), News and Politics (0.37)

Tags: Top 2 tags detected as outliers by our algorithm along with the community distributions are as follows. It is interesting to note that people often mention "canoeing" as a sport that they perform often when they travel (e.g., on group outings).

(1) canoeing: Sports (0.62), Travel (0.38). Though there are other sports which people feel interesting in while traveling, canoeing seems to be a clear exception wrt number of travel pages it is mentioned on. The closest distribution pattern is (Sports: 1).

(2) rosary: Arts and Design (0.38), Education (0.02), Sports (0.6)

URLs: We find that not many web-pages belong to the Lifestyle and Travel categories together. As a result the pages that belong partially to the Travel and Lifestyle categories get marked as top outliers.

(1) `http://globetrooper.com/`: Lifestyle (0.35), Travel (0.38)

(2) `http://vandelaydesign.com/blog/galleries/travel-websites/`: Lifestyle (0.33), Travel (0.48)

In conclusion, our algorithm is effective at finding interesting outliers from real datasets.

5 Related Work

Outlier detection has been studied in the context of a large number of application domains [1, 2, 5, 6, 13, 15]. Chandola et al. [3] and Hodge et al. [12] provide extensive overview of outlier detection techniques. Different from these studies, we perform community outlier detection for heterogeneous network data.

Individual, Global and Community Contexts Outlier Detection can be performed at different levels of context. (1) Individual Context: For example, Type I and Type II Outliers [5] in time series are defined based on values observed for the same object across

different time points. (2) Global Context: Stream Outliers [2], DB Outliers [13], Sub-Structure Outliers [18] are defined based on comparison with all the other objects in the dataset. (3) Community Context: Different from existing community outlier detection approaches (Community Outliers [6], CTOutliers [9], ECOutliers [10]), we model multiple data types in a *heterogeneous* network simultaneously to find outliers.

Homogeneous versus Heterogeneous Networks Recently there has been work on outlier detection for homogeneous networks [2, 6, 7, 10]. While previous work on outlier detection for heterogeneous networks [14, 17] models the anomaly detection problem in heterogeneous networks as a tensor decomposition problem, we model the problem using a joint-NMF model to extract distribution patterns, which are further used to detect outliers. Also compared to our previous work (ABCOutliers [8]) which identified outlier cliques, this work focuses on finding outlier objects.

6 Conclusions

We introduced the notion of outliers with respect to latent communities for heterogeneous networks, i.e., *CDOutliers*. Such outliers represent objects that disobey the frequent community distribution patterns. The challenge in detecting such outliers is twofold: (1) correlation between patterns across different types of objects in the network should be considered; and (2) patterns need to be learned by ignoring the outliers, while outlier detection depends on effective discovery of patterns. To tackle such challenges, we proposed a joint-NMF optimization framework to learn distribution patterns across multiple object types, that uses a regularizer for distance between the cluster centroid matrices of different object types. We derive the update rules to learn the joint NMF model, which alternately updates the cluster membership and the cluster centroid matrices. Experiments on a series of synthetic data show the proposed algorithm's capability of detecting outliers under various levels of outlierness, data dimensionality, and number of types. Case studies on *DBLP* and *Delicious* datasets reveal some interesting and meaningful outliers. In the future, we plan to extend the framework to handle multiple temporal network snapshots in a stream scenario.

Acknowledgments

We thank the anonymous reviewers for their insightful comments. Research was sponsored in part by the U.S. Army Research Laboratory under Cooperative Agreements W911NF-11-2-0086 (Cyber-Security) and W911NF-09-2-0053 (NS-CTA), U.S. National Science Foundation grants IIS-0905215, CNS-0931975, CCF-0905014, IIS-1017362, DTRA, and NASA NRA-NNH10ZDA001N. The views and conclusions contained in this document are those of the authors and should not be interpreted as representing the official policies, either expressed or implied, of the Army Research Laboratory or the U.S. Government. The U.S. Government is authorized to reproduce and distribute reprints for Government purposes notwithstanding any copyright notation here on.

References

1. Charu C. Aggarwal and Philip S. Yu. Outlier Detection for High Dimensional Data. *SIGMOD Records*, 30:37–46, May 2001.
2. Charu C. Aggarwal, Yuchen Zhao, and Philip S. Yu. Outlier Detection in Graph Streams. In *ICDE*, pp. 399–409, 2011.
3. Varun Chandola, Arindam Banerjee, and Vipin Kumar. Anomaly Detection: A Survey. *ACM Surveys*, 41(3), 2009.
4. Chris H. Q. Ding and Xiaofeng He. On the Equivalence of Nonnegative Matrix Factorization and Spectral Clustering. In *SDM*, pp. 606–610, 2005.
5. A. J. Fox. Outliers in Time Series. *Journal of the Royal Statistical Society.*, 34(3):350–363, 1972.
6. Jing Gao, Feng Liang, Wei Fan, Chi Wang, Yizhou Sun, and Jiawei Han. On Community Outliers and their Efficient Detection in Information Networks. In *KDD*, pp. 813–822, 2010.
7. Amol Ghoting, Matthew Eric Otey, and Srinivasan Parthasarathy. LOADED: Link-Based Outlier and Anomaly Detection in Evolving Data Sets. In *ICDM*, pp. 387–390, 2004.
8. Manish Gupta, Jing Gao, and Jiawei Han. On Detecting Association-Based Clique Outliers in Heterogeneous Information Networks. In *ASONAM*, To appear, 2013.
9. Manish Gupta, Jing Gao, Yizhou Sun, and Jiawei Han. Community Trend Outlier Detection using Soft Temporal Pattern Mining. In *ECML PKDD*, pp. 692–708, 2012.
10. Manish Gupta, Jing Gao, Yizhou Sun, and Jiawei Han. Integrating Community Matching and Outlier Detection for Mining Evolutionary Community Outliers. In *KDD*, pp. 859–867, 2012.
11. Mark Hall, Eibe Frank, Geoffrey Holmes, Bernhard Pfahringer, Peter Reutemann, and Ian H. Witten. The WEKA Data Mining Software: An Update. *SIGKDD Explorations*, 11(1):10–18, Nov 2009.
12. Victoria J. Hodge and Jim Austin. A Survey of Outlier Detection Methodologies. *AI Review*, 22(2):85–126, 2004.
13. Edwin M. Knorr, Raymond T. Ng, and Vladimir Tucakov. Distance-Based Outliers: Algorithms and Applications. *VLDBJ*, 8:237–253, Feb 2000.
14. Danai Koutra, Evangelos E. Papalexakis, and Christos Faloutsos. TensorSplat: Spotting Latent Anomalies in Time. In *Panhellenic Conference on Informatics*, pp. 144-149, 2012.
15. Hans-Peter Kriegel, Matthias Schubert, and Arthur Zimek. Angle-based Outlier Detection in High-Dimensional Data. In *KDD*, pp. 444–452, 2008.
16. J. B. MacQueen. Some Methods for Classification and Analysis of MultiVariate Observations. In *Berkeley Symposium on Mathematical Statistics and Probability*, volume 1, pp. 281–297, 1967.
17. Koji Maruhashi, Fan Guo, and Christos Faloutsos. MultiAspectForensics: Pattern Mining on Large-Scale Heterogeneous Networks with Tensor Analysis. In *ASONAM*, pp. 203–210, 2011.
18. C. C. Noble and Diane J. Cook. Graph-Based Anomaly Detection. In *KDD*, pp. 631–636, 2003.
19. Yizhou Sun, Jiawei Han, Xifeng Yan, and Philip S. Yu. Mining Knowledge from Interconnected Data: A Heterogeneous Information Network Analysis Approach. *PVLDB*, 2012.
20. Yizhou Sun, Yintao Yu, and Jiawei Han. Ranking-based Clustering of Heterogeneous Information Networks with Star Network Schema. In *KDD*, pp. 797–806, 2009.
21. Xiaowei Xu, Nurcan Yuruk, Zhidan Feng, and Thomas A. J. Schweiger. SCAN: A Structural Clustering Algorithm for Networks. In *KDD*, pp. 824–833, 2007.