MixedAD: A Scalable Algorithm for Detecting Mixed Anomalies in Attributed Graphs

Mengxiao Zhu, Haogang Zhu
State Key Laboratory of Software Development Environment Lab, Beihang University, China
Beijing Advanced Innovation Center for Biomedical Engineering, China
{zhumx, haogangzhu}@buaa.edu.cn

Abstract

Attributed graphs, where nodes are associated with a rich set of attributes, have been widely used in various domains. Among all the nodes, those with patterns that deviate significantly from others are of particular interest. There are mainly two challenges for anomaly detection. For one thing, we often encounter large graphs with lots of nodes and attributes in the real-life scenario, which requires a scalable algorithm. For another, there are anomalies *w.r.t.* both the structure and attribute in a mixed manner. The algorithm should identify all of them simultaneously. State-of-art algorithms often fail in some respects. In this paper, we propose the scalable algorithm called MixedAD. Theoretical analysis is provided to prove its superiority. Extensive experiments are also conducted on both synthetic and real-life datasets. Specifically, the results show that MixedAD often achieves the F_1 scores greater than those of others by at least 25% and runs at least 10 times faster than the others.

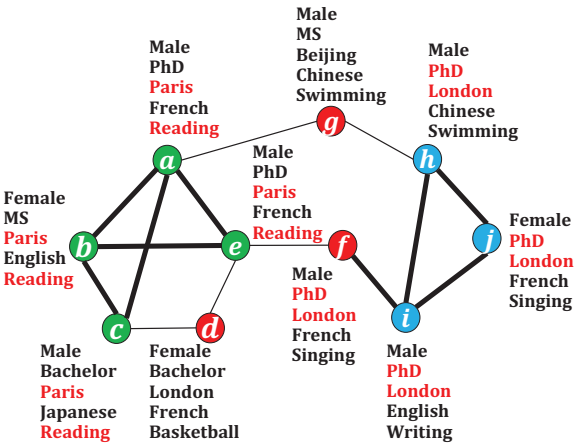


Figure 1: Three types of anomalies.

Introduction

Networks are ubiquitous nowadays; examples include social networks, where links represent the friendship between users, the web where hyperlinks connect different hypertext documents, or paper citation networks where some papers are cited by the others, and so on. They are often modeled by attributed graphs where nodes (*i.e.*, the entities of the network) are associated with a rich set of attributes, such as the profiles of the users in social networks. In contrast to the majority of nodes on the graph, those with patterns or behaviors that deviate significantly from others are of great interest, as detecting such anomalies has a wide range of applications, such as finding the spammers in social networks (Yang et al. 2012; Castillo et al. 2007), detecting the cyber-attacks in computer networks (Ding et al. 2012), and spotting fraudulent activities in financial trading networks (Chau, Pandit, and Faloutsos 2006).

Most common networks naturally have community structure; the nodes of the network can be easily divided into communities such that those from the same community are densely connected with each other. We claim that there are three types of anomalies defined with respect to the communities. For example, a spammer may connect with as many communities as possible for advertising. As it clearly does not belong to any community, we call such anomalies global anomalies (GA). A gene, though interacting with others in its community, may have already mutated. After comparing the patterns of their principal attribute values (*i.e.*, those possessed by most nodes in the community) with those of others in the community, we can detect such local attribute anomalies (LAA). The topic of a paper fits well in an area but it is cited by many papers from various areas due to its great influence. Such nodes are named local structural anomalies (LSA). We use a toy example to illustrate them on the graph.

Example 1. *Imagine a social network with communities representing circles of friends, as showed in Figure 1. There are two communities* $\{a, b, c, d, e\}$ *and* $\{f, h, i, j\}$ *in this attributed graph. The set of attributes is* {*gender, degree, nationality, language skill, hobby*}*. Node* g *is a GA because it does not belong to any community. Node* d *has many friends in the first community, whereas it shows abnormal attribute values "London" and "Basketball", in contrast to the principal values "Paris" and "Reading" possessed by the others, so it is an LAA. For node* f*, though it is a "PhD" and*

Copyright © 2020, Association for the Advancement of Artificial Intelligence (www.aaai.org). All rights reserved.

Table 1: Comparison of related work

	Unknown $\lvert\mathcal{C}\rvert$	Principal Values	LSA	LAA	GA	Scalability
CODA (Gao et al. 2010)				✓		
GOutRank (Müller et al. 2013)	✓	✓		✓		
FocusCo (Perozzi et al. 2014)	✓	✓		✓		✓
Radar (Li et al. 2017)	✓			✓	✓	
ANOMALOUS (Peng et al. 2018)	✓	✓		✓	✓	
PAICAN (Bojchevski and Günnemann 2018)		✓	✓	✓	✓	
MixedAD [this paper]	✓	✓	✓	✓	✓	✓

from "London", it only has one friend in the second community, hence an LSA.

Lots of efforts have been made on anomaly detection in attributed graphs (Gao et al. 2010; Müller et al. 2013; Perozzi et al. 2014; Li et al. 2017; Peng et al. 2018; Bojchevski and Günnemann 2018) (which will be reviewed later). The main challenge, however, is not just to use the elaborate approaches to detect them, but in a scalable way. For one thing, any single pass on the large-scale networks (with millions of nodes (Catanese et al. 2011; Ley 2002)) would incur high computational costs, which requires the algorithms with time complexity basically linear in the size of the network (*i.e.*, the number of edges). For another, high-dimensional attributes often hinder the recognition of patterns and hence the anomaly detection, because lots of attributes are irrelevant ones (*e.g.*, the gender).

Therefore, in this paper, we propose a scalable algorithm called MixedAD, which jointly learns the community structure and mines the principal attribute values in a single pass. Since we need at least $\Omega(mD)$ (where m is the number of edges and D is the number of attributes) time to inspect the whole network, we prove that our proposed approach indeed takes $\mathcal{O}(mD)$.

Related Work

Anomaly detection in attributed graphs receives much attention (Gao et al. 2010; Müller et al. 2013; Perozzi et al. 2014; Li et al. 2017; Peng et al. 2018; Bojchevski and Günnemann 2018) and is mostly related to our work. The comparison is made in Table 1, where $|\mathcal{C}|$ represents the number of communities. Some of them only focus on the attribute anomaly *w.r.t.* the community but fail to spot the structural anomaly (Gao et al. 2010; Müller et al. 2013; Perozzi et al. 2014). For recent works (Li et al. 2017; Peng et al. 2018; Bojchevski and Günnemann 2018), they do notice multiple types of anomalies, but their models usually incur high computational costs in the learning processes. (Li et al. 2017; Peng et al. 2018) all use a linear model to analyze the residual (which can be seen as the degree of difference to the model) of the attribute values. As the operations on matrices need at least $\Omega(n^2D)$ time, the methods are impractical for large data. (Bojchevski and Günnemann 2018) adopts a probabilistic generative model to jointly perform clustering and anomaly detection. The learning process requires several runs to converge, which is shown to be efficient only on data of moderate size (demonstrated in experiments). Besides, it assumes that the model knows the number of communities $|\mathcal{C}|$ in prior. We believe we are the first to detect all three types of anomalies in a scalable way.

Anomaly detection in plain graphs (*i.e.*, without the attributes) is also widely studied. See (Akoglu, Tong, and Koutra 2015) for a survey. Some extract graph-centric features to find the normal patterns (Akoglu, McGlohon, and Faloutsos 2010; Henderson et al. 2011). Others propose the proximity-based methods (Moonesinghe and Tan 2008; Sun et al. 2005). There are also methods which detect anomalies when finding the communities (Chakrabarti 2004; Xu et al. 2007; Shiokawa, Fujiwara, and Onizuka 2015; Tong and Lin 2011; Hu et al. 2016).

Apart from anomaly detection, there is also a considerable amount of work for graph mining (*e.g.*, the clustering) in attributed graphs (Akoglu et al. 2012; Tsourakakis, Pachocki, and Mitzenmacher 2017; Yang, McAuley, and Leskovec 2013; Xu et al. 2012; Günnemann et al. 2013; Bojchevski, Matkovic, and Günnemann 2017; Perozzi et al. 2014; Bojchevski and Günnemann 2018; Huang et al. 2011; Silva, Jr., and Zaki 2012). Among them, (Günnemann et al. 2013; Bojchevski, Matkovic, and Günnemann 2017; Perozzi et al. 2014; Bojchevski and Günnemann 2018; Huang et al. 2011) even consider the corruption of data; some anomalies may hinder the clustering and the algorithms must be robust.

Problem Statement

In this section, we introduce several basic concepts and then define the problem of anomaly detection.

Definition 1 (Attributes)**.** *There is one set of D categorical attributes (features) $F = \{f_1, f_2, \ldots, f_D\}$. Each $f \in F$ has a domain $dom(f) = \{a_1, a_2, \ldots\}$, which is a finite set of possible values for the attribute f. Assume without loss of generality $dom(f_i) \cap dom(f_j) = \emptyset$.*

Definition 2 (Attributed Graph)**.** *The attributed graph $G = (V, E, F)$ consists of the node set V and the edge set E with their cardinalities n and m, respectively. Each node $v \in V$ is described by the set of attributes (features) F. If we use $g_f(v)$ to denote v's attribute value* w.r.t. *attribute f, v has attribute values $\{g_{f_1}(v), \ldots, g_{f_D}(v)\}$.*

The node set V can be partitioned into communities $\mathcal{C} = \{C_1, C_2 \ldots, \}$ and global anomalies (GA), the definition of which is deferred.

Definition 3 (Communities)**.** *Each community C_i consists*

of nodes densely connected with each other, and these nodes also have some common attribute values.

We assume that each node belongs to only one community at most. This assumption is reasonable for real applications with some abnormal nodes. And how to decide whether the community is densely connected or not would be defined later. For convenience, if node v belongs to a community C_i, we give it a label $L(v) = i$.

On the other hand, because of the consistency among some attribute values, these nodes belong to the same community. We call these values the principal attribute values.

Definition 4 (Frequency)**.** *For the community C_i and some attribute $f \in F$, the frequency for each value $a \in dom(f)$ is defined as*

$$p_{C_i}(a) = \frac{|\{v \in C_i \mid g_f(v) = a\}|}{|C_i|}. \quad (1)$$

Definition 5 (Mode)**.** *The mode (*w.r.t. *C_i and f) is defined as the value with the largest frequency* i.e.*,* $\arg\max_a p_{C_i}(a)$.

Definition 6 (Principal Attribute Values)**.** *For each community C_i, we use P_i to denote its corresponding set of principal attribute values. And $a \in P_i$ if and only if:*

(1) a is the mode for some f

(2) its frequency $p_{C_i}(a) \geq \theta$, where $\theta \in (0.5, 1]$ is a high threshold, indicating that the principal value should appear in almost every node in C_i.

It is easily seen that $|P_i| \leq D$, and there may be no principal value for some attributes.

Example 2. *Consider again the nodes in Figure 1. Most of nodes in the left community have the common attribute values "Paris" and "Reading", and the set of principal attribute values for the left community C_1 is $P_1 = \{Paris, Reading\}$. Similarly, $P_2 = \{PhD, London\}$.*

Most nodes are normal ones; that is, each belongs to one of the communities and possesses the corresponding principal attribute values. We focus on the rest of few nodes which behave abnormally in contrast to normal nodes. There are three types of anomalies in total: global anomalies (GA), locally structural anomalies (LSA), and local attribute anomalies (LAA). Here the words "local" and "global" indicate whether the scope is in the community or the whole graph.

Definition 7 (GA)**.** *GA nodes do not belong to any community and deviate from most nodes having communities.*

Definition 8 (LSA)**.** *LSA nodes are sparsely connected to community C_i but possess its principal attribute values P_i.*

Definition 9 (LAA)**.** *LAA nodes belong to some community structurally C_i but disagree with the corresponding principal attribute values P_i.*

Now we can state our problem of anomaly detection in attributed graphs. Given the graph G, we try to detect three types of anomalous nodes GA, LSA, and LAA, respectively, while extracting information about the communities $\mathcal{C}$ and their corresponding sets P_i.

Let $nei(v) = \{u|(u, v) \in E\}$ be the neighbors of v. For any $S \subseteq V$, we use $E(S)$ to denote the set of edges that have both endpoints in S, and $\delta(S)$ to denote the set of edges that have exactly one endpoint in S. Abusing notations slightly, we may use $\delta(v)$ to denote the set of edges incident to v.

Algorithm 1: MixedAD

input : Attributed graph G
output: GA, LSA, LAA

1 $Unvisited \leftarrow V, Groups \leftarrow \emptyset$;
2 **while** *($Core_i \leftarrow$ InitiatingCore($G, Unvisited$))* $\neq Null$ **do**
3 $\quad Group_i, PG_i \leftarrow$ ExpandingCore($G, Core_i$) ;
4 $\quad Unvisited \leftarrow Unvisited \backslash Group_i$;
5 $\quad Groups \leftarrow Groups \cup \{Group_i\}$;
6 $\mathcal{C}, \mathcal{P} \leftarrow$ MergingGroups($Groups, PGs$);
7 $\mathcal{C} \leftarrow$ PropagatingLabels($\mathcal{C}$);
8 **return** *AnomalyDetection($\mathcal{C}, \mathcal{P}$);*

The Proposed Methodology

Overview

We first provide an overview of MixedAD, summarized in Algorithm 1. As all types of anomalies are defined *w.r.t.* communities, we need to identify them. MixedAD first finds groups, defined as the set of nodes which belong to the same community, and then merge them into communities (Line 6). Note that two groups may be from the same community. MixedAD finds such group by initiating a core (*i.e.*, a small group with high confidence that the nodes are from the same community) from unvisited nodes (Line 2), expanding it by iteratively adding the neighbor nodes, and stopping until we have low confidence in them (Line 3). During the expansion, we maintain the set of principal attribute values for the current group PG_i and add a new node while considering both the attribute and structure. After merging the groups (Line 6), there may still be normal nodes not assigned to any community. Notice that for any node v, if most of its neighbors belong to some community C_i, v is also in C_i. So we propagate the label of each assigned node (Line 7). At last, when we obtain the communities, the anomalies can be detected by referring to their definitions (Line 8). The reason why we first find groups instead of communities is we want to be more cautious about the principal attribute values, and communities can be found in the later phase where we can utilize these principal attribute values.

Initiating Core

Since the size of a community can be small, we focus on the cores with just three nodes, while, more importantly, we want to select such three nodes that they belong to the same community with high probability. The basic idea is to first select a node (called the "pivot") with most of its neighbors from the same community, and then its two neighbors.

We prefer the pivot u with most of its neighbor nodes in $nei(u)$ which own the same $L(u)$. However, the local information does not tell the label $L(v)$ for each $v \in nei(u)$. A simple idea is to calculate the similarity score (by using some well-known functions) between the attribute vec-

Algorithm 2: Initiating Core

input : Attributed graph G, Unvisited nodes $Unvisited$
output: $Core$

$Core \leftarrow \emptyset$;
while $Core = \emptyset$ **do**
  Try to find from the $Unvisited$ the pivot node u with the maximum uniformity;
  if *there are two edges* $(u, v), (u, w)$ *such that* v *and* w *are in* $Unvisited$ **then**
    choose two such edges with the maximum similarity score between $EP_{(u,v)}$ and $EP_{(u,w)}$;
    $Core \leftarrow \{u, v, w\}$;
  else
    $u \leftarrow Unvisited \backslash \{u\}$;
    $Core \leftarrow \emptyset$;
return $Core$

Algorithm 3: Expanding Core

input : $Core$
output: $Group, PG$

$Group \leftarrow Core, Nei \leftarrow \emptyset$;
foreach $v \in Core$ **do**
  $Nei \leftarrow Nei \cup nei(v)$;
while *true* **do**
  remove from Nei the nodes with their $AN_{PG}(v)$ greater than a threshold α *w.r.t.* statistics;
  choose the node v which decreases the conductance of $Group$ most;
  if $Cond(Group \cup \{v\}) \geq Cond(Group)$ **then**
    break;
  else
    $Group \leftarrow Group \cup \{v\}$;
    Update the statistics;
return $Group, PG$

tors $(g_{f_1}(u), \ldots, g_{f_D}(u))$ and $(g_{f_1}(v), \ldots, g_{f_D}(v))$ for each $v \in nei(u)$. If two nodes bear a high similarity, it is likely they are from the same community. We can then check if most of the pivot u's neighbors have a high similarity with u. Unfortunately, we argue that the similarity score can be high if u and v share many non-principal attribute values, and they can belong to different communities. The consistency in principal attribute values actually matters. But these values are unknown beforehand. We design the technique of Edge Pattern (EP) to tackle this issue.

Definition 10 (Edge Pattern)**.** *Given an edge* $(u, v) \in E$*, the edge pattern*

$$EP_{(u,v)} = \{g_{f_i}(u) \mid g_{f_i}(u) = g_{f_i}(v), i = 1, \ldots D\}, \quad (2)$$

which includes all the common values between u *and* v*.*

Claim 1. *For any community* C_i *and any two edges* e, e' *whose endpoints are normal nodes in* C_i*,* EP_e *is the same as* $EP_{e'}$ *with high probability.*

It can be easily proved by the definition of principal values. Based on the claim above, we only need to choose the pivot with its incident edges that follow a uniform edge pattern, since the endpoints are likely normal nodes from the same community. To reflect the degree of uniformity that EP_e (for $e \in \delta(u)$) show, we use the negative entropy. Formally, we define the uniformity of each node.

Definition 11 (Uniformity)**.** *Given a node* v*, let* $freq(EP_e)$ *denote the frequency of some edge pattern* w.r.t.*, all the edges incident to* u*.*

$$Uni(u) = \sum_{EP_e} freq(EP_e) \log(freq(EP_e)). \quad (3)$$

Note that if two edge patterns are similar (measured by Jaccard similarity index for example), we regard them as one entity when we count the frequencies.

The pivot with the maximum uniformity owns two neighbors from the same community with high probability. After finding the pivot, we simply select two incident edges with their edge patterns similar to each other and add the two endpoints to the core. The similarity can be measured by the Jaccard index. Algorithm 2 illustrates the main procedure.

Expanding Core

After initializing the core (*i.e.*, a small group with high confidence), we expand it by repeatedly adding the best neighbor node to the group, referred to as the $Group$, while maintaining statistics about the possible principal attribute values. To define what is the best node, we consider both the attribute and the structure. We filter all those neighbor nodes with anomalous attribute values and choose from the rest the node which best improves the stability of the current group structure. We next present the details of the two metrics.

The degree of attribute normality of some node v is measured by the principal values. If the node v has all of them, it is normal *w.r.t.* the attribute. However, as we only have the $Group$ during the expansion, principal values are uncertain. We adopt the technique of confidence bound to get the set of principal values with high probability $1 - \gamma$.

Specifically, we maintain the statistics for each value a

$$\widehat{N_f}(a) = \frac{|\{v \in Group \mid g_f(v) = a\}|}{|Group|}, \quad (4)$$

which is the frequency of the value a *w.r.t.* the current $Group$. And if the frequency $\widehat{N_f}(a)$ is smaller than the lower confidence bound

$$\theta - \sqrt{\ln(1/\gamma)/|Group|}, \quad (5)$$

the value a cannot be a principal value (the proof of which is deferred). Before $|Group|$ increases to $\lceil \ln(1/\gamma)/(\theta - 1/2)^2 \rceil$, for each attribute f, we choose as the principal value the one $a \in dom(f)$ such that $\widehat{N_f}(a)$ is the largest and greater than the lower bound. When $|Group|$ equals $\lceil \ln(1/\gamma)/(\theta - 1/2)^2 \rceil$, we claim that there is no principal value for the attribute f if there exist two values with their

frequencies $\widehat{N_f}(a)$ greater than the lower bound. If there is just one such value, we regard it as the principal one.

Definition 12 (Attribute Normality)**.** *Given some P, the attribute normality of v*

$$AN_P(v) = \sum_{i=1}^{D} \mathbf{1}_P(g_{f_i}(v))/|P|, \tag{6}$$

where $\mathbf{1}_P(a)$ is 1 if $a \in P$ and 0 otherwise.

And the node v is an attribute anomaly if its $AN_P(v)$ is lower than a high threshold $\alpha \in [0, 1]$. We simply ignore these nodes. If the attribute values of v are normal, we will choose the node which improves the stability of structure most. We use the graph conductance, which is a common measure for assessing the quality of a clustering (Kannan, Vempala, and Vetta 2004).

Definition 13 (Conductance)**.** *Given a set of nodes S on the graph, its conductance*

$$cond(S) = |\delta(S)|/(|E(S)| + |\delta(S)|). \tag{7}$$

The lower conductance means that there are more intra-edges and fewer inter-edges, which coincides with the definition of a densely connected community. Hence, the neighbor node which decreases the conductance most will be likely from the same community. We add such node until no node can bring about a decrease. It is observed that if we only use the conductance but do not consider the attribute, the larger group tends to absorb any small groups. We may get an intuition for such phenomenon if we notice $cond(V) = 0$, which is the minimum one can achieve.

It is worth noting that the weighted conductance (which uses the sum of weights of edges) is inapplicable here. The reason is the same as we explain above: the weights can be high if the endpoints from different communities share many non-principal attribute values. The main procedure is illustrated in Algorithm 3.

Merging Groups

As there may be groups from the same community, we should merge them into a single one. This can be easily done by referring to the set of principal values PG_i we maintain for each $Group_i$, since their PG_i should be similar.

Propagating Labels

A useful insight about finding the right label of each node is that for node v, if most of its neighbors own the same label l, the label $L(v)$ should also be l. Now that we obtain the labels of most nodes through the previous procedures, we can propagate these labels to find the labels of the rest of unassigned nodes. Note that after the propagation, the remaining nodes are structural anomalies (*i.e.*, either GA or LSA), because their neighbors belong to diverse communities.

Detecting Anomalies

Finally, we can detect the anomalies based on the information about the community and the principal attribute values. If the node has no label, it is either a GA node or an LSA node. However, the LSA nodes are different from the GA nodes for LSA nodes possess the set of principal values of some community. Therefore, we consider the attribute normality $AN_P(v)$ (Definition 12) for any $P \in \mathcal{P}$. It is a GA node if it does not possess any set of principal values P or an LSA node if it is normal *w.r.t.* some P. If it belongs to some C_i but has anomalous attribute values, it is an LAA.

Theoretical Analysis

Confidence Bound

We justify the confidence bound in equ. 5, which basically ensures that the set of principal values are found with high probability during the expansion. For ease of notation, let the group be $S = \{v_1, v_2, \ldots, v_k\}$, the nodes of which are all from some community C. For any value a, let X_i be 1 if v_i possesses it *w.r.t.* the corresponding attribute and 0 otherwise. Let $X = \sum_{i=1}^{k} X_i$, and X/k is the frequency *w.r.t.* the group S. Recall that the frequency *w.r.t.* the community is $p_C(a) = \frac{|\{v \in C | g_f(v) = a\}|}{|C|}$.

We first derive a lemma, stating that the frequency *w.r.t.* the group concentrates on its expectation with high probability.

Lemma 1. *For any value a,*

$$\Pr(|X/k - \mathbb{E}[X/k]| \geq \epsilon) \leq 2e^{-\epsilon^2 k}.$$

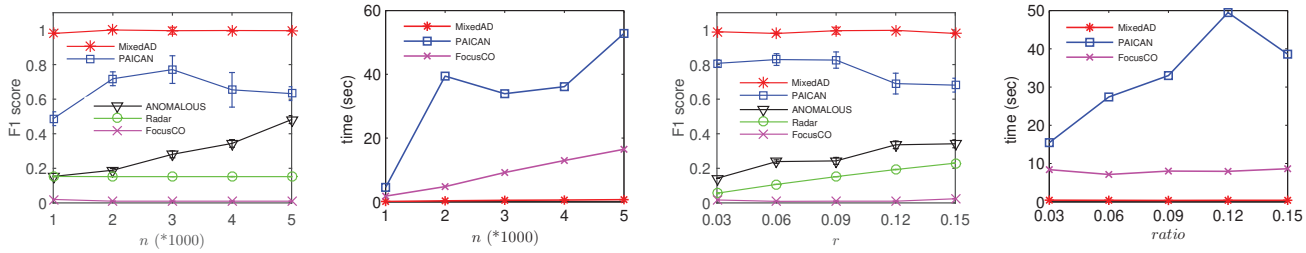
Proof. The proof mainly uses the Azuma-Hoeffding inequality. Specifically, the sequence $Z_i = \mathbb{E}[X|X_1, \ldots, X_i]$ forms a *Doob martingale*, where $Z_0 = \mathbb{E}[X]$ and $Z_k = X$. We claim that X satisfies the Lipschitz condition with bound 1. Formally, for any two sequences $\{X_1', ..., X_i', ..., X_k'\}$ and $\{X_1', ..., X_i'', ..., X_k'\}$ different in only one position, X's value changes at most 1. Using the Azuma-Hoeffding inequality, we have $\Pr(|X - \mathbb{E}[X]| \geq \epsilon) \leq 2e^{-\epsilon^2/k}$. Replacing ϵ by $k\epsilon'$, we obtain the inequality above. □

Theorem 1. *For some community C and any value a, if $X/k < \theta - \sqrt{\ln(1/\gamma)/k}$, a is not a principal value with probability at least $1 - \gamma$.*

Proof. Now we only need to derive the expectation value. For ease of notations, we denote the number of nodes who possess the value a by $N = |\{v \in C \mid g_f(v) = a\}|$. We first consider the case if $N \geq k$ and $|C| - N \geq k$.

$$\begin{aligned}
\mathbb{E}[X/k] &= \frac{\sum_{i=1}^{k} i\binom{N}{i}\binom{|C|-N}{k-i}}{k\binom{|C|}{k}} \\
&= \frac{\sum_{i=1}^{k} N\binom{N-1}{i-1}\binom{|C|-N}{k-i}}{k\binom{|C|}{k}} \\
&= \frac{N\binom{|C|-1}{k-1}}{k\binom{|C|}{k}} = \frac{N}{|C|}.
\end{aligned}$$

Using Lemma 1 and replacing ϵ by $\sqrt{\ln(1/\gamma)/k}$, we know that the difference between the frequency *w.r.t.* the



(a) F_1 of varying n (b) Time of varying n (c) F_1 of varying $ratio$ (d) Time of varying $ratio$

Figure 2: Results on varying n and $ratio$.

group and that *w.r.t.* the community $N/|C| - X/k \leq \sqrt{\ln(1/\gamma)/k}$ with probability at least $1-\gamma$. And we have

$$p_C(a) = N/|C| \leq X/k + \sqrt{\ln(1/\gamma)/k} < \theta.$$

If $k > N$ and $|C| - N \geq k$, the value a cannot be a principal one because $\theta > 1/2 > N/|C|$ (*i.e.*, the nodes who possess it are not the majority). If $N \geq k$ and $k > |C| - N$, the expectation $\mathbb{E}[X/k] \geq N/|C|$. And $p_C(a) = N/|C| \leq \mathbb{E}[X/k] \leq X/k + \sqrt{\ln(1/\gamma)/k} \leq \theta$. □

Theorem 2. *When $k \geq \lceil \ln(1/\gamma)/(\theta - 1/2)^2 \rceil$, if there are more than two values such that $X/k \geq \theta - \sqrt{\ln(1/\gamma)/k}$, the principal attribute value does not exist for attribute f.*

Proof. We have $\sqrt{\ln(1/\gamma)/k} \leq \theta - 1/2$. The proof mainly uses the contradiction. Suppose the principal value exists. Then for the rest of values $p_C(a) = N/|C| \leq 1 - \theta$. However, there are two values such that their $p_C(a) = N/|C| \geq X/k - \sqrt{\ln(1/\gamma)/k} \geq \theta - 2\sqrt{\ln(1/\gamma)/k} \geq 1 - \theta$, which contradicts the fact that there is only one value whose $p_C(a) \geq 1 - \theta$. □

Complexity Analysis

We analysis the time complexity of MixedAD. Among all the procedures, expanding the core is the most time-consuming part. There are mainly three phases for each expansion: finding the node from Nei with the largest decrease for the conductance, updating necessary values, and adding normal nodes to the Nei. Note that the decrease of the conductance incurred by each node can be calculated in $\mathcal{O}(1)$ if we maintain for each node in Nei its changes for $|E(Group)|$ and $|\delta(Group)|$ in case it will be added to the $Group$. Therefore, the first phase takes time in $\mathcal{O}(|Nei|)$. After choosing the node v which gives the largest decrease for $Cond(Group)$, we only need to update the changes of $|E(Group)|$ and $|\delta(Group)|$ for those nodes in $nei(v)$. Calculating these changes for any node w needs $\mathcal{O}(nei(w))$ time. The update needs $\mathcal{O}(nei_{max} nei(v))$ time, where $nei_{max} = \max_v |nei(v)|$. Finally, we filter out those nodes in $nei(v)$ with smaller $AN(v)$, which costs $\mathcal{O}(nei(v)D)$ time. There will be at most $\mathcal{O}(|V|)$ iterations. The total time cost will be $\mathcal{O}(\max(|V||Nei|, mnei_{max}, mD))$. It is obvious that the cost is $\mathcal{O}(mD)$ if D is large enough.

Table 2: Details of real datasets.

Datasets	n	m	D	MixedAD	PAICAN
PolBlogs	359	2233	44832	$2.58s$	$27.47s$
Amazon	29618	425174	4643	$4.66s$	$469.32s$

Experimental Study

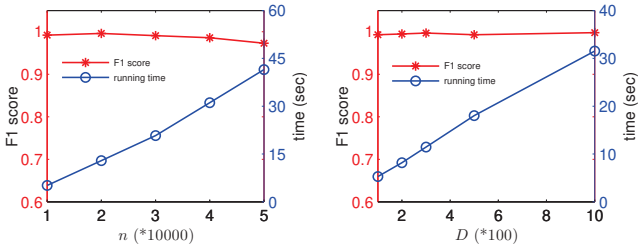
Experimental Settings

Synthetic datasets. We generate attributed graphs by first producing a plain graph without attributes, the LFR-benchmark (Lancichinetti, Fortunato, and Radicchi 2008) by convention, then associating the attribute values with each node, and adding the anomalies at last. The key parameters include those used in LFR-benchmark, *e.g.*, the number of nodes n and the average degree of nodes $\langle k \rangle$ (used to control the sparsity of edges), and also parameters for generating the attribute values. Due to limited space, we only show the results of varying n and the ratio of anomalies $ratio$. To test the scalability, we vary n from 10000 to 50000 and the number of attributes from 100 to 1000, where D actually lies in $[5827, 56082]$ in binary form. Besides, for the hyperparameters, we set $\alpha = 0.9$, $\theta = 0.8$, and $\gamma = 0.9$. For each setting of parameters we generate 10 random datasets and report the mean and standard deviation of the relevant metric.

Real datasets. We use two real datasets. PolBlogs (Perozzi et al. 2014) is a citation network with a collection of online blogs that discuss political issues and the attributes are the keywords in the texts of the blogs. Amazon (Bojchevski and Günnemann 2018) is a co-purchase network. The link between two products a and b indicates that people buy a and also b. The attribute values are binary product category indicators in one-hot encoding.

Compared algorithms. We compare four state-of-art algorithms with MixedAD. Note that PAICAN does not report GA. ANOMALOUS and Radar only finds LAA and GA, and FocusCo only finds LAA.

- PAICAN (Bojchevski and Günnemann 2018) mainly adopts a probabilistic generative model.
- ANOMALOUS (Peng et al. 2018) uses the CUR decomposition and residual analysis.
- Radar (Li et al. 2017) uses the residual analysis and the coherence with network information.



(a) varying n (b) varying D

Figure 3: Results of scalability

- FocusCO (Perozzi et al. 2014) is a semi-supervised approach. We implement it as in (Bojchevski and Günnemann 2018): we select some example nodes for each community and run the algorithm $|\mathcal{C}|$ times.

Metrics and implementation. To measure the effectiveness, we adopt the F_1 score in the synthetic experiments. The reason why we choose the F_1 score is that most algorithms cannot distinguish all the three types of anomalies. But it is worth noting that our algorithm is capable of correctly detecting their types, since most of the times the F_1 scores of our MixedAD are very close to 1. We show the running time to measure the efficiency. As Radar and ANOMALOUS often run several thousand times slower than our approach, we do not show their time costs.

We do case studies to test the effectiveness in real graphs due to the lack of ground truth and also a convention from previous work. We omit FocusCO since it is a semi-supervised approach and we cannot provide any prior correct examples in real scenarios. We also omit Radar and ANOMALOUS due to their tremendous time costs.

All algorithms are implemented by Python 3.6.6 and performed with Intel (R) Core (TM) i7 3.4GHz CPU and 8GB main memory.

Experimental Results on Synthetic Graphs

Impacts of n and $ratio$. The results of varying the number of nodes n from 1000 to 5000 are showed in Figure 2a and 2b, respectively. MixedAD achieves the highest F_1 scores close to 1, indicating that it correctly identifies all three types of anomalies. For running time, MixedAD runs the fastest among all the algorithms. FocusCO runs linearly in n, and PAICAN performs unstably due to the uncertain numbers of iterations for the convergence of its model. Similar results of varying the $ratio$ from 0.03 to 0.15 are reported in Figure 2c and 2d.

Scalability. To test the scalability of MixedAD, we vary the number of nodes n and dimensionality D, since we have shown that its time complexity is $\mathcal{O}(mD)$. Specifically, we vary n from 10000 to 50000, while fixing $D = 100$ and the other parameters to their default values, and vary the dimensionality D from 100 to 1000, while fixing $n = 10000$ and the other parameters to their default values. The results are showed in Figure 3a and 3b. For the F_1 score, MixedAD still performs well. The running time coincide with the time complexity linear in n and D. It is worth noting that our MixedAD does not require the attribute values in the binary form. The dimensionality (in the binary form) actually lies in $[5827, 56082]$ when $D \in [100, 1000]$.

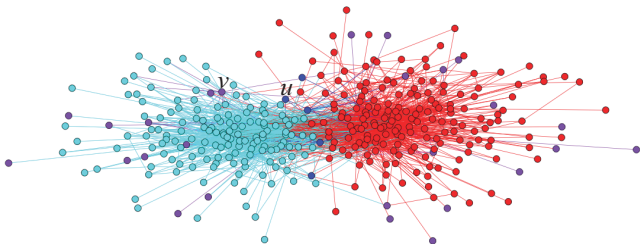
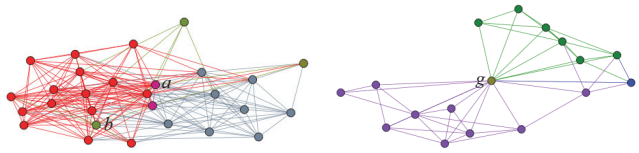


Figure 4: Anomalies on PolBlogs



(a) GA and LAA (b) LSA

Figure 5: Anomalies on Amazon

Experimental Results on Real-life Graphs

We first consider the PolBlogs dataset. The visualization of anomalies is showed in Figure 4. It can be observed that there are two communities (in red and blue) that MixedAD found, which coincides with the conservative and liberal parties in reality. Nodes in violet and cadet blue are instances of LSA and LAA, respectively. For example, node u is an LSA. The blogger u mentions words like "libertarians", "democracy" and so on. However, it connects to two opposite political factions. Another example is the node v in cadet blue, an LAA, and it is a well-connected liberal blogger who does not explicitly express the view of faction. PAICAN cannot find the anomalies stated above.

The second case is the Amazon co-purchase dataset, showed in Figure 5. Node a is a GA and its category includes "highchairs, baby products", but it is co-purchased with the products which are "Toys, Games, Pretend play, Dress up" and "Playsets, Vehicles". Another example is node b, which is an LAA because this kind of cookware is often co-purchased with a lot of toys. Moreover, node g is an LSA because it belongs to the category "Casual Clothing" with violet nodes, whereas it is often co-purchased with the products belong to the category "Speical Occasion Clothing". Likewise, PAICAN can not find these anomalies.

Conclusion

This paper studies the problem of anomaly detection in large attributed graphs. We identify three types of anomalies whose patterns are abnormal *w.r.t.* the attribute and structure in a mixed manner. We propose a scalable algorithm called MixedAD and provide theoretical analysis. We verify

its effectiveness and efficiency by conducting extensive experiments. The results demonstrate that our MixedAD often achieves F1 scores greater than other state-of-art algorithms by at least 25% and runs at least 10 times faster than them.

Acknowledgments

This research was supported by the National Natural Science Foundation under Grant 61702027, the Beijing Science and Technology Plan Project under Grant Z171100000117022, the Beijing Municipal Science & Technology Commission under Grant Z181100001918008.

References

Akoglu, L.; Tong, H.; Meeder, B.; and Faloutsos, C. 2012. PICS: parameter-free identification of cohesive subgroups in large attributed graphs. In *SDM*.

Akoglu, L.; McGlohon, M.; and Faloutsos, C. 2010. Oddball: Spotting anomalies in weighted graphs. In *PAKDD*.

Akoglu, L.; Tong, H.; and Koutra, D. 2015. Graph based anomaly detection and description: a survey. *Data Min. Knowl. Discov.* 29(3):626–688.

Bojchevski, A., and Günnemann, S. 2018. Bayesian robust attributed graph clustering: Joint learning of partial anomalies and group structure. In *AAAI*.

Bojchevski, A.; Matkovic, Y.; and Günnemann, S. 2017. Robust spectral clustering for noisy data: Modeling sparse corruptions improves latent embeddings. In *SIGKDD*.

Castillo, C.; Donato, D.; Gionis, A.; Murdock, V.; and Silvestri, F. 2007. Know your neighbors: web spam detection using the web topology. In *SIGIR*.

Catanese, S.; Meo, P. D.; Ferrara, E.; Fiumara, G.; and Provetti, A. 2011. Crawling facebook for social network analysis purposes. In *WIMS*.

Chakrabarti, D. 2004. Autopart: Parameter-free graph partitioning and outlier detection. In *PKDD*.

Chau, D. H.; Pandit, S.; and Faloutsos, C. 2006. Detecting fraudulent personalities in networks of online auctioneers. In *PKDD*.

Ding, Q.; Katenka, N.; Barford, P.; Kolaczyk, E. D.; and Crovella, M. 2012. Intrusion as (anti)social communication: characterization and detection. In *SIGKDD*.

Gao, J.; Liang, F.; Fan, W.; Wang, C.; Sun, Y.; and Han, J. 2010. On community outliers and their efficient detection in information networks. In *SIGKDD*.

Günnemann, S.; Färber, I.; Raubach, S.; and Seidl, T. 2013. Spectral subspace clustering for graphs with feature vectors. In *ICDM*.

Henderson, K.; Gallagher, B.; Li, L.; Akoglu, L.; Eliassi-Rad, T.; Tong, H.; and Faloutsos, C. 2011. It's who you know: graph mining using recursive structural features. In *SIGKDD*.

Hu, R.; Aggarwal, C. C.; Ma, S.; and Huai, J. 2016. An embedding approach to anomaly detection. In *ICDE*.

Huang, H.; Yoo, S.; Qin, H.; and Yu, D. 2011. A robust clustering algorithm based on aggregated heat kernel mapping. In *ICDM*.

Kannan, R.; Vempala, S.; and Vetta, A. 2004. On clusterings: Good, bad and spectral. *J. ACM* 51(3):497–515.

Lancichinetti, A.; Fortunato, S.; and Radicchi, F. 2008. Benchmark graphs for testing community detection algorithms. *Phys. Rev. E* 78:046110.

Ley, M. 2002. The DBLP computer science bibliography: Evolution, research issues, perspectives. In *SPIRE*.

Li, J.; Dani, H.; Hu, X.; and Liu, H. 2017. Radar: Residual analysis for anomaly detection in attributed networks. In *IJCAI*.

Moonesinghe, H. D. K., and Tan, P. 2008. Outrank: a graph-based outlier detection framework using random walk. *International Journal on Artificial Intelligence Tools* 17(1):19–36.

Müller, E.; Sánchez, P. I.; Mülle, Y.; and Böhm, K. 2013. Ranking outlier nodes in subspaces of attributed graphs. In *ICDEW*.

Peng, Z.; Luo, M.; Li, J.; Liu, H.; and Zheng, Q. 2018. ANOMALOUS: A joint modeling approach for anomaly detection on attributed networks. In *IJCAI*.

Perozzi, B.; Akoglu, L.; Sánchez, P. I.; and Müller, E. 2014. Focused clustering and outlier detection in large attributed graphs. In *SIGKDD*.

Shiokawa, H.; Fujiwara, Y.; and Onizuka, M. 2015. SCAN++: efficient algorithm for finding clusters, hubs and outliers on large-scale graphs. *PVLDB* 8(11):1178–1189.

Silva, A.; Jr., W. M.; and Zaki, M. J. 2012. Mining attribute-structure correlated patterns in large attributed graphs. *PVLDB* 5(5):466–477.

Sun, J.; Qu, H.; Chakrabarti, D.; and Faloutsos, C. 2005. Relevance search and anomaly detection in bipartite graphs. *SIGKDD Explorations* 7(2):48–55.

Tong, H., and Lin, C. 2011. Non-negative residual matrix factorization with application to graph anomaly detection. In *SDM*.

Tsourakakis, C. E.; Pachocki, J.; and Mitzenmacher, M. 2017. Scalable motif-aware graph clustering. In *WWW*.

Xu, X.; Yuruk, N.; Feng, Z.; and Schweiger, T. A. J. 2007. SCAN: a structural clustering algorithm for networks. In *SIGKDD*.

Xu, Z.; Ke, Y.; Wang, Y.; Cheng, H.; and Cheng, J. 2012. A model-based approach to attributed graph clustering. In *SIGMOD*.

Yang, C.; Harkreader, R. C.; Zhang, J.; Shin, S.; and Gu, G. 2012. Analyzing spammers' social networks for fun and profit: a case study of cyber criminal ecosystem on twitter. In *WWW*.

Yang, J.; McAuley, J. J.; and Leskovec, J. 2013. Community detection in networks with node attributes. In *ICDM*.