

Community-based Outlier Detection for Edge-attributed Graphs

Supriya Pandhre
Indian Institute of Technology, Hyderabad
Hyderabad, India
cs15mtech11016@iith.ac.in

Manish Gupta
Microsoft
Hyderabad, India
gmanish@microsoft.com

Vineeth N Balasubramanian
Indian Institute of Technology, Hyderabad
Hyderabad, India
vineethnb@iith.ac.in

Abstract—The study of networks has emerged in diverse disciplines as a means of analyzing complex relationship data. Beyond graph analysis tasks like graph query processing, link analysis, influence propagation, there has recently been some work in the area of outlier detection for information network data. Although various kinds of outliers have been studied for graph data, there is not much work on anomaly detection from edge-attributed graphs. In this paper, we introduce a method that detects novel outlier graph nodes by taking into account the node data and edge data simultaneously to detect anomalies. We model the problem as a community detection task, where outliers form a separate community. We propose a method that uses a probabilistic graph model (Hidden Markov Random Field) for joint modeling of nodes and edges in the network to compute *Holistic Community Outliers (HCO outliers)*. Thus, our model presents a natural setting for heterogeneous graphs that have multiple edges/relationships between two nodes. EM (Expectation Maximization) is used to learn model parameters, and infer hidden community labels. Experimental results on synthetic datasets and the DBLP dataset show the effectiveness of our approach for finding novel outliers from networks.

I. INTRODUCTION

Outlier (or anomaly) detection is a very broad field which has been studied in the context of a large number of research areas like statistics, data mining, sensor networks, environmental science, distributed systems, spatio-temporal mining, etc. Many algorithms have been proposed for outlier detection in high-dimensional data, uncertain data, stream data and time series data. By its inherent nature, network data provides very different challenges that need to be addressed in a special way. Network data is gigantic, contains nodes of different types, rich nodes with associated attribute data, noisy attribute data, noisy link data, and is dynamically evolving in multiple ways. Outlier detection on networks for various applications can be useful for: (1) identification of interesting entities or sub-graphs, (2) data de-noising (both with respect to the network nodes and edges), (3) understanding the anomalous temporal behavior of entities, and (4) identification of new trends or suspicious activities in both static and dynamic scenarios. Some examples of network outliers include users who spread rumors on Twitter, unusual but successful associations of persons with various organizations in an organizational network, sensor with anomalous readings in a sensor network, anomalous traffic between two components in a distributed network, snapshot with broken correlation between network properties across

time, etc.

Given a static graph, one can find node, edge or subgraph outliers. For dynamic graphs, the outliers could be time stamps where the properties of the snapshot are significantly different from the properties of other snapshots. In the dynamic scenario, one can also identify a node (or an edge, or a subgraph) which shows anomalous behavior across time as an outlier. Popular outlier detection methods for static graphs include the Minimum Description Length (MDL) method [10], frequent subgraph mining, egonet analysis [5], and community analysis [12], [14]. Popular methods for outlier detection for dynamic graphs include graph similarity based methods [20], temporal spectral analysis [19], temporal structural connectivity analysis [3], and temporal community analysis [15], [16].

While most work in the past on network outlier detection has focused on graphs with node data and links, there is very little work in the area of edge-attributed graphs. In some cases, node data and linkage could be normal even when edge data is abnormal, in the context of nodes on which the edge is incident. With the rich variety of interactions in various complex graphs across a variety of domains, it is critical to incorporate such edge attributes in finding novel outliers. If we could assign a latent community to every node and every edge, a HCO outlier node is one which is linked to many nodes of another community, or has incident edges belonging to another community.

The following real-world examples demonstrate the importance of HCO outliers.

Organizational Graph Example: A graph of people working in a company is called an organization graph. Edges represent frequent communication. Role of a person can be considered as the community label. Usually a chemist in a company would communicate frequently with other chemists. But a suspicious HCO outlier node would not just communicate frequently with members of other roles, but also the community of the communications would be different from the one expected for its role. Such a person could be an activist planning a malicious activity with other folks in the company, or a star performer working on a secret collaborative project.

Co-authorship Graph Example: A co-authorship graph contains authors as nodes. Two authors are connected if they have published a paper together. On such graphs, research areas can be considered as latent communities. Most authors

collaborate with other authors, and publish papers *in the same research area*. An HCOutlier would be an author who: (1) collaborates frequently with authors from other research areas, and (2) collaborates with other authors of same or different community on papers which belong to other research areas. Such authors perform inter-disciplinary research and are vital in cross-pollination of concepts across research areas.

Co-actorship Graph Example: A co-actorship graph contains actors as nodes. Two actors are connected if they have worked in a movie together. On such graphs, movie genres can be considered as latent communities. Most actors collaborate with other actors, and work in movies *in the same genre*. An HCOutlier would be an actor who: (1) collaborates frequently with actors from other genres, and (2) collaborates with other actors of same or different genre in movies which belong to other genres. HCOutlier actors are ones who have built a reputation for being multi-faceted actors with an eclectic range of movies.

Limitations of Past Approaches: In the past, various approaches have been proposed by looking only at node data and ignoring network structure. Clearly, HCOutliers cannot be detected using such a global approach because HCOutlier nodes belong to popular communities. There has been work on finding community outliers for both homogeneous [12] and heterogeneous graphs [14] by considering node data and link structure but ignoring edge data. Clearly, such an approach can identify some HCOutliers who are linked to nodes with a different community label. But they cannot identify a HCOutlier who could be connected to nodes of the same community label but has incident edges with a different community label.

Brief Overview of the Proposed Approach: We propose a probabilistic model for detection of Holistic Community Outliers. We model the problem as a community detection task where the outliers are modeled as a separate community. A Hidden Markov Random Field (HMRF) is used to perform joint community analysis for both nodes and edges considering node data, edge data and the linkage structure. Thus, the HMRF contains both nodes and edges from the graph as vertices, referred to as a node-vertex and an edge-vertex respectively. Community labels for outliers are sampled from a uniform distribution while normal vertices could be sampled from Gaussian or a multinomial distribution. EM is used for the inference, and to compute the hidden community label Z for every vertex.

We make the following contributions in this paper:

- We look at the community outlier detection problem from a holistic perspective by incorporating node data, edge data, and the linkage structure. Based on such a view, we propose the problem of detecting novel HCOutliers.
- We model the problem as a community analysis task over a HMRF and provide inference using an EM algorithm.
- Using several experiments on synthetic datasets and a DBLP dataset, we show the effectiveness of the proposed approach in identifying HCOutliers.

Section II gives a brief summary of related work. We describe our HMRF model in detail in Section III and present the

inference in Section IV. We discuss hyper-parameter settings and initialization details in Section V. We provide details of experiments on synthetic datasets and the DBLP dataset in Section VI and conclude in Section VII.

II. RELATED WORK

Outlier detection has a long history and [2], [11], [18] provide extensive overviews of popular methods; [6], [13] focus on network outlier detection methods. Our work is most related to two main sub-areas of network outlier detection: community-based outlier detection and analysis of edge-attributed graphs, each of which is discussed below.

A. Community-based Graph Outlier Detection

Community-based outlier detection methods perform community analysis on graphs. Nodes that do not belong to any community are labeled as outliers. Community-based outlier detection has been studied both for static networks and dynamic networks. The notion of community outliers has been studied for static bipartite graphs using random walks in [23], for static homogeneous networks using probabilistic models in [12], and for static heterogeneous graphs using non-negative matrix factorization in [14]. [22] summarizes different anomaly detection methods in dynamic graphs such as decomposition based methods [4], [24], distance based methods [1]. Community based outlier detection method, for dynamic graphs, is proposed for a two-snapshots setting in [16] and for a series of snapshots in [15].

B. Analysis of Edge-attributed Graphs

Edge content indicates the type of relationship between the two nodes. However very little work exists on analysis of edge-attributed graphs. Qi et al. [21] proposed a edge-induced matrix factorization based method for finding communities in social graph using edge content. [8] proposed a method that considers edge data of User-Likes-Pages Facebook graph for temporal analysis to find the page-like pattern. [9] proposed a method for mining coherent sub-graphs in multi-layer edges by exploiting edge content. [17] proposed a method for finding top- K interesting subgraphs matching a query template where interestingness is defined using edge weights. We propose a community-based outlier detection method for edge-attributed graphs which jointly exploits edge content along with node data and the linkage structure.

III. ANOMALY DETECTION USING NODE DATA, EDGE DATA, AND LINKAGE

In this section, we develop the HMRF model which we use to perform joint community analysis for both nodes and edges considering node data, edge data and the linkage structure. Consider an undirected graph $G = \{V, E\}$, where V is the set of vertices and E is the set of edges.

Node and Edge Data: Let S be the set of observed data.

- Let S_i represent the observed data of node $v_i \in V$ which has p attributes, $S_{i_1}, S_{i_2}, \dots, S_{i_p}$.

- Let S_{ij} represent the observed data of edge $e_{ij} \in E$, i.e., edge data between nodes v_i and v_j , and there are q number of attributes, $S_{ij_1}, S_{ij_2}, \dots, S_{ij_q}$.

HMRF Model: The HMRF contains a vertex representing each node and each edge from G . We refer to the HMRF vertices representing nodes and edges as node-vertices and edge-vertices respectively. We use the index i to refer to node-vertices in the HMRF, and the index ij to refer to edge-vertices in the HMRF. We use the index b to refer to all the vertices in the HMRF. Thus $b \in B = \{1, 2, \dots, i, \dots, |V|, (1, 2), (1, 3), \dots, (1, |V|), (2, 3), \dots, (|V| - 1, |V|)\}$. Note that $ij \in B$ if $i < j$. Thus, $|B| = |V| + |E|$.

- Let $X = \{X_b\}$ be a set of random variables. X_i generates node data S_i , and X_{ij} generates edge data S_{ij} .
- Let $Z = \{Z_b\}$ be the set of hidden random variables. Z_i indicates the community assignment of v_i . Suppose there are K communities, then $Z_i \in \{0, 1, 2, \dots, K\}$. If $Z_i = 0$, then v_i is an outlier. If $Z_i = k$, then v_i belongs to the community k . Similarly, Z_{ij} denotes the community assignment for edge e_{ij} .
- Let W denote the weights in the HMRF. W_{v_i, v_j} is set to the weight of the edge $e_{ij} \in E$ and represent edge strength. $W_{v_i, v_j, e_{ij}}$ are weights for a triangle clique consisting of v_i, v_j and e_{ij} . Thus, $|W| = 4|E|$ (3 faces of the triangle formed by two node-vertices and an edge-vertex, and the whole triangle itself).

$W_{v_i, e_{ij}}$ can be set in multiple ways. We set it to the inverse of number of neighbors of v_i in G . Thus, for a co-authorship graph, this weight can be set to the inverse of the number of co-authors of author v_i in the graph. For a Twitter users graph, this weight can be set to the inverse of the number of other users that the user v_i is connected to. In a directed network sense, the weight can also be set to the ratio of the number of tweets from user v_i to user v_j to number of total tweets from user v_i .

$W_{v_i, v_j, e_{ij}}$ should reflect some property of the triangle clique which cannot be captured using individual edges. For a co-authorship network, this could be set to ratio of number of papers where authors v_i and v_j are co-authors to the total number of papers on which either v_i or v_j are authors. Similarly, for a Twitter user network, this weight can be set to the ratio of tweets exchanged between users v_i and v_j to the total tweets posted by either v_i or v_j .

Z_b 's are influenced by their neighbors, i.e., if two nodes are linked, it is likely that both belong to the same community. This does not hold for the outliers. Let N_i denote the set of neighbors of node v_i . If $Z_i = 0$, then v_i is an outlier, and so the neighborhood set is empty. Also, each node-vertex is connected to the edge-vertex corresponding to the edges which are incident on the node in G .

$$N_i = \begin{cases} \{v_j | W_{v_i v_j} > 0, i \neq j, Z_j \neq 0\} \cup \\ \{e_{xy} | W_{v_i e_{xy}} > 0, x = i \text{ or } y = i, Z_{xy} \neq 0\} & Z_i \neq 0 \\ \phi & Z_i = 0 \end{cases} \quad (1)$$

Similarly, we also define neighbors for edge-vertices, N_{ij} , as follows. Note that for an edge-vertex, the neighbors are only the node-vertices on which the edge-vertex is incident except for the case when these node-vertices are outliers.

$$N_{ij} = \begin{cases} \{v_i | z_i \neq 0\} \cup \{v_j | z_j \neq 0\} & Z_{ij} \neq 0 \\ \phi & Z_{ij} = 0 \end{cases} \quad (2)$$

Data Likelihood: Let $\Theta = \{\theta_1, \theta_2, \dots, \theta_K\}$ be the set of parameters describing the normal communities. Thus, likelihood for node/edge data can be written as follows.

$$P(X_b = S_b | Z_b = k) = P(X_b = S_b | \theta_k) \quad (3)$$

We assume the outliers follow uniform distribution as we do not know beforehand which elements (node-vertex or edge-vertex) are anomalous or what they look like.

$$P(X_b = S_b | Z_b = k) = \rho_0 \quad (4)$$

where ρ_0 is a constant.

Figures 1 and 2 illustrate a sample graph and its corresponding HMRF model. The graph has two communities, blue and red. The node v_5 has connection with the member of blue community as if it is part of blue community but its node attributes are similar to red community. Hence, in the HMRF model, it is not connected to any other node, indicating v_5 as an outlier. The edge between v_4 and v_6 indicate the cross-community collaboration and thus detected as interesting nodes.

A. Cliques and Potentials in HMRF

As the community label assignment follows an HMRF, we can define $P(Z) = \frac{1}{H_1} \exp(-U(Z))$ where $U(Z)$ is the potential function, defined as sum over all possible cliques.

We consider two kinds of clique potentials: pairwise (v_i to v_j, v_i to e_{ij}) and triangular (between v_i, v_j and e_{ij}). Then, the potential function can be written as follows.

$$U(Z) = -\lambda_1 \sum_{\substack{W_{v_i v_j} > 0, \\ Z_i \neq 0, \\ Z_j \neq 0}} W_{v_i v_j} \delta(Z_i - Z_j) \\ -\lambda_2 \sum_{\substack{W_{v_i e_{ij}} > 0, \\ Z_i \neq 0, \\ Z_{ij} \neq 0}} W_{v_i e_{ij}} \delta(Z_i - Z_{ij}) \\ -\lambda_2 \sum_{\substack{W_{v_j e_{ij}} > 0, \\ Z_j \neq 0, \\ Z_{ij} \neq 0}} W_{v_j e_{ij}} \delta(Z_j - Z_{ij}) \\ -\lambda_3 \sum_{\substack{W_{v_i v_j e_{ij}} > 0, \\ Z_i \neq 0, \\ Z_j \neq 0, \\ Z_{ij} \neq 0}} W_{v_i v_j e_{ij}} \psi(Z_i, Z_j, Z_{ij}) \quad (5)$$

where $\lambda_1, \lambda_2, \lambda_3$ are constants, $W_{v_i v_j} > 0, W_{v_i e_{ij}} > 0, W_{v_j e_{ij}} > 0, W_{v_i v_j e_{ij}} > 0$, imply that there are links

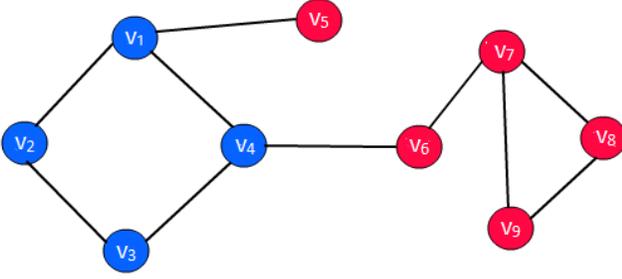


Fig. 1: Original Graph $G=(V,E)$

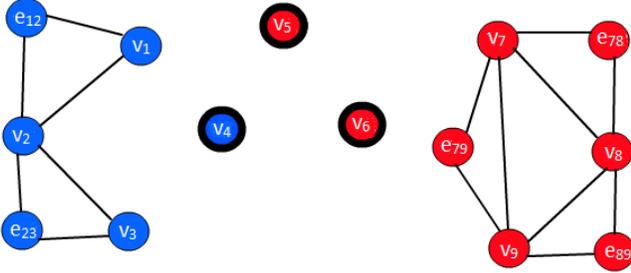


Fig. 2: HMRF model of G indicating anomalous node v_4, v_5 and v_6

connecting v_i-v_j , v_i-e_{ij} , v_j-e_{ij} and Z_i, Z_j, Z_{ij} are non-zero. The δ function is defined as $\delta(x) = 1$ if $x = 0$ and $\delta(x) = 0$ otherwise. Also, $\psi(Z_i, Z_j, Z_{ij}) = 1$ if $\delta(Z_i - Z_j) + \delta(Z_i - Z_{ij}) + \delta(Z_j - Z_{ij}) \leq 1$, and 0 otherwise. Note that we define ψ this way to capture the intuition that the clique potential should fire if at least two of the members in the clique share the same community label. Overall, the potential function suggests that, if members of the clique are normal objects, they are more likely to be in the same community when there exists a link connecting them in G , and the probability becomes higher if their link (weight) is stronger.

B. Probability of Generating the Data

Given the community label of a vertex (Z_i or Z_{ij}) in the HMRF, the corresponding data (X_i or X_{ij}) can be modeled as data generated from a distribution with parameters specific to the community. Community labels can be obtained using either Gaussian mixture model on the data if it is continuous, or as multinomial if the data is textual.

Continuous Data: If the data is numeric and 1-dimensional, we need to learn the mean (μ_k), and standard deviation (σ_k) for each community k represented using a Gaussian. Given the parameters and the community label, the log likelihood can be written as follows.

$$\ln P(X_i = S_i | Z_i = k) = -\frac{(S_i - \mu_k)^2}{2\sigma_k^2} - \ln \sigma_k - \ln \sqrt{2\pi} \quad (6)$$

In case of multi-dimensional data with p dimensions, the log likelihood can be written as follows.

$$\ln P(X_i = S_i | Z_i = k) = -\frac{(S_i - \mu_k)^T \Sigma_k^{-1} (S_i - \mu_k)}{2} - \frac{\ln \Sigma_k}{2} - \ln \sqrt{(2\pi)^p} \quad (7)$$

Discrete Data: If the node (or edge) data can be represented as a document, we assume that the attributes (or words) are independent of each other, given the community label.

$$P(X_i = S_i | \theta_k) = \prod_{c=1}^p P(X_{i_c} = S_{i_c} | \theta_k) \quad (8)$$

$$P(X_{ij} = S_{ij} | \theta_k) = \prod_{c=1}^q P(X_{ij_c} = S_{ij_c} | \theta_k) \quad (9)$$

For text data, given a vocabulary of T words, let d_{bl} be the number of times the word l appears in data related to vertex b . Then the parameters $\theta_k = \{\beta_{k1}, \dots, \beta_{kT}\}$ denote the probability of a particular word belonging to community k . Given the parameters, the data likelihood of an object belonging to the k^{th} community can be computed as follows.

$$\ln P(X_{i_c} = S_{i_c} | Z_i = k) = \sum_{l=1}^T d_{il} \ln \beta_{kl} \quad (10)$$

where β_{kl} is the probability of the word with index l belonging to community k .

IV. INFERRING Z AND ESTIMATING Θ

The model parameters Θ and the set of hidden labels Z are unknown as described in Section III. In this section, we describe methods to infer the hidden labels and also estimate the model parameters.

A. Inferring Hidden Labels

Assuming the model parameters Θ are known, we find the configuration Z that will maximize the posterior distribution as follows.

$$\hat{Z} = \arg \max_Z P(X = S | Z) P(Z) \quad (11)$$

To find such a configuration, we use the Iterated Conditional Modes (ICM) algorithm [7]. It is a greedy algorithm which guarantees convergence to a local optima by sequentially updating one Z_b at a time assuming other Z 's (i.e., Z_{B-b}) as constant. At each step, the algorithm updates Z_b given $X_b = S_b$ and the other labels such that the posterior $P(Z_b | X_b = S_b, Z_{B-b})$ is maximized. If $Z_b \neq 0$, applying Bayes rule,

$$P(Z_b | X_b = S_b, Z_{B-b}) \propto P(X_b = S_b | Z) P(Z) \quad (12)$$

In Eq. 12, $P(Z)$ can be computed using Eq. 5. But rather than considering potential for the entire graph, Eq. 12 needs potentials defined over only those cliques in which Z_b is involved. Thus, we can write the following equation for node-vertices:

$$\begin{aligned}
P(Z_i|X_i = S_i, Z_{B-\{i\}}) &\propto P(X_i = S_i|Z) \times \\
&\exp(\lambda_1 \sum_{v_j \in N_i} W_{v_i v_j} \delta(Z_i - Z_j)) \\
&+ \lambda_2 \sum_{e_{xy} \in N_i} W_{v_i e_{xy}} \delta(Z_i - Z_{xy}) \\
&+ \lambda_3 \sum_{\substack{v_j \in N_i, \\ e_{xy} \in N_i}} W_{v_i v_j e_{xy}} \psi(Z_i, Z_j, Z_{xy})
\end{aligned} \tag{13}$$

Also, the corresponding equation for the edge-vertices can be written as follows.

$$\begin{aligned}
P(Z_{ij}|X_{ij} = S_{ij}, Z_{B-\{(i,j)\}}) &\propto P(X_{ij} = S_{ij}|Z) \times \\
&\exp(\lambda_2 \sum_{v_{i'} \in N_{ij}} W_{v_{i'} e_{ij}} \delta(Z_{i'} - Z_{ij})) \\
&+ \lambda_3 W_{v_i v_j e_{ij}} \psi(Z_i, Z_j, Z_{ij})
\end{aligned} \tag{14}$$

Taking log of the posterior probability, we can transform the maximizing posterior problem to minimizing the conditional posterior energy function defined as shown in Eqs. 15 and 16 for node-vertices and edge-vertices respectively.

$$\begin{aligned}
U_i(k) &= -\ln P(X_i = S_i|Z_i = k) - \lambda_1 \sum_{v_j \in N_i} W_{v_i v_j} \delta(k - Z_j) \\
&- \lambda_2 \sum_{e_{xy} \in N_i} W_{v_i e_{xy}} \delta(k - Z_{xy}) \\
&- \lambda_3 \sum_{\substack{v_j \in N_i, \\ e_{xy} \in N_i}} W_{v_i v_j e_{xy}} \psi(k, Z_j, Z_{xy})
\end{aligned} \tag{15}$$

$$\begin{aligned}
U_{ij}(k) &= -\ln P(X_{ij} = S_{ij}|Z_{ij} = k) \\
&- \lambda_2 \sum_{v_{i'} \in N_{ij}} W_{v_{i'} e_{ij}} \delta(Z_{i'} - k) - \lambda_3 W_{v_i v_j e_{ij}} \psi(Z_i, Z_j, k)
\end{aligned} \tag{16}$$

If $Z_b = 0$, the vertex has no neighbors, and thus

$$P(Z_b|X_b = S_b, Z_{B-\{b\}}) \propto P(X_b = S_b|Z_b = 0)P(Z_b = 0) \tag{17}$$

Hence,

$$U_b(0) = -\ln(\rho_0 \pi_0) = a_0 \tag{18}$$

Finally, the label Z_b for vertex b in HMRF is set to $k \in \{0, 1, \dots, K\}$ such that $U_b(k)$ is minimized. The pre-defined hyper-parameters, $\lambda_1, \lambda_2, \lambda_3$, represent the importance of the respective components of the graph structure. a_0 is the outlierness threshold. Algorithm 1 first randomly initializes the labels for all vertices. At each step, labels are updated sequentially by minimizing $U_b(k)$.

Algorithm 1 Inferring Hidden Labels

Input: Node/Edge data S , weights W , set of model parameters θ , number of clusters K , hyper-parameters $(\lambda_1, \lambda_2, \lambda_3)$, threshold a_0 , initial assignment of labels $Z^{(1)}$

Output: Updated assignment of labels Z

- 1: Randomly set $Z^{(0)}$
 - 2: $t \leftarrow 1$
 - 3: **while** $Z^{(t)}$ is not close enough to $Z^{(t-1)}$ **do**
 - 4: $t \leftarrow t + 1$
 - 5: **for** $b = 1; b \leq |B|; i++$ **do**
 - 6: Update $Z_b^{(t)} = k$ which minimizes $U_b(k)$ using Eqs. 15, 16 and 18
 - 7: **return** $Z^{(t)}$
-

B. Estimating Parameters

In this subsection, we discuss a method for estimating unknown θ from the data. θ describes the model that generates node data and edge data. Hence we seek to maximize the data likelihood $P(X = S|\theta)$ to obtain $\hat{\theta}$. However, since both the hidden labels and the parameters are unknown and they are inter-dependent, we use expectation-maximization (EM) algorithm (as shown in Algorithm 2) to solve the problem. We direct the reader to [12] for details.

The algorithm starts with an initial estimate of hidden labels $Z^{(1)}$. Given the current configuration of hidden labels, we can estimate model parameters as follows. For univariate numeric data, $\mu_k^{(t+1)}$ and $\sigma_k^{(t+1)}$ are estimated as follows:

$$\mu_k^{(t+1)} = \frac{\sum_{b=1}^{|B|} P(Z_b = k|X_b = S_b, \Theta^{(t)}) S_b}{\sum_{b=1}^{|B|} P(Z_b = k|X_b = S_b, \Theta^{(t)})} \tag{19}$$

$$(\sigma_k^{(t+1)})^2 = \frac{\sum_{b=1}^{|B|} P(Z_b = k|X_b = S_b, \Theta^{(t)}) (S_b - \mu_k)^2}{\sum_{b=1}^{|B|} P(Z_b = k|X_b = S_b, \Theta^{(t)})} \tag{20}$$

For text data, given a vocabulary of T words, let d_{bl} be the number of times the word l appears in data related to vertex b . Then, given the current configuration of hidden labels, we can estimate parameters of the multinomial distribution β_{kl} for $k \in \{1, \dots, K\}$ and $l \in \{1, \dots, T\}$ as follows:

$$\beta_{kl}^{(t+1)} = \frac{\sum_{b=1}^{|B|} P(Z_b = k|X_b = S_b, \Theta^{(t)}) d_{bl}}{\sum_{l=1}^T \sum_{b=1}^{|B|} P(Z_b = k|X_b = S_b, \Theta^{(t)}) d_{bl}} \tag{21}$$

Given the updated parameters, the new configuration of hidden labels can be estimated using Algorithm 1 where $P(Z_b = k^*|X_b = S_b, \Theta^{(t)}) = 1$ if $k^* = \arg \min_k U_b(k)$, and 0 otherwise.

In summary, the Holistic Community Outlier detection algorithm shown in Algorithm 2 works as follows. It begins with some initial label assignment of the vertices in the HMRF. In the M-step, the model parameters are estimated using the EM algorithm to maximize the data likelihood, based on the current label assignment. In the E-step, Algorithm 1 is run to re-assign the labels to the objects by minimizing $U_b^{(k)}$ for each HMRF vertex sequentially. The E-step and M-step are

repeated until convergence is achieved, and the outliers are the nodes that have 0 as the final estimated labels.

Algorithm 2 Holistic Community Outlier Detection

Input: Node/Edge data S , weights W , set of model parameters θ , number of clusters K , hyper-parameters $(\lambda_1, \lambda_2, \lambda_3)$, threshold a_0 , initial assignment of labels $Z^{(1)}$

Output: Set of Holistic Community Outliers

- 1: Randomly set $Z^{(0)}$
 - 2: $t \leftarrow 1$
 - 3: **while** $Z^{(t)}$ is not close enough to $Z^{(t-1)}$ **do**
 - 4: **M-step:** Given $Z^{(t)}$, update parameters $\Theta^{(t+1)}$ according to Eqs. 19, 20 or 21.
 - 5: **E-step:** Given $\Theta^{(t+1)}$, update the hidden labels as $Z^{(t+1)}$ using Algorithm 1.
 - 6: $t \leftarrow t + 1$
 - 7: **return** the indices of outliers: $\{i : Z_b^{(t)} = 0, b \in B\}$
-

V. DISCUSSIONS

In this section, we discuss how to set the hyperparameters and how to initialize the hidden labels.

Setting Hyperparameters: The HCOutlier detection algorithm has the following hyperparameters: threshold a_0 , clique importance variables, $\lambda_1, \lambda_2, \lambda_3$ and number of communities K .

$\lambda_1, \lambda_2, \lambda_3$ indicate the importance of link between two nodes, the importance of link between edge-vertex and a node-vertex, and the importance of the triangle between the edge-vertex and the node-vertices of which the edge is incident respectively. If λ s are set to low values, the algorithm will consider only node information for finding the outliers. If their values are set too high, all connected nodes will have the same label, so an upper bound can be set for $\lambda_1 + \lambda_2 + \lambda_3$, such that a value above this bound will result into empty communities. High λ_1 will give more importance to the linkage in the graph. High λ_2 will give high importance to consistency between the node data and edge data in the graph. High λ_3 will give high importance to consistency between the edge data and both the incident nodes in the graph.

The threshold a_0 can be replaced by another parameter (percentage of outliers r) as follows. In Algorithm 1, first calculate $\hat{Z}_i = \arg \min_k U_i(k) (k \neq 0)$ for each $i \in \{1, \dots, |V|\}$ and then sort $U_i(\hat{Z}_i)$ in non-descending order. Finally, select top r percentage as outliers.

K represents the number of normal communities. For small value of K , algorithm will find the global outliers, and for large value of K many local outliers will be detected because of many communities. An appropriate K can be set using a variety of methods like Akaike Information Criterion (AIC), Bayesian Information Criterion (BIC), Minimum Description Length (MDL), etc.

Initialization of labels: Instead of initializing Z randomly, we initialize Z values by clustering the nodes without considering

outliers. To overcome local optima issues, we run the algorithm multiple times with different initialization, and choose the one with the largest data likelihood value.

VI. EXPERIMENTS

Evaluation of outlier detection algorithms is difficult in general due to lack of ground truth. Hence, we perform experiments on multiple synthetic datasets. We evaluate outlier detection accuracy of the proposed algorithm based on outliers injected in synthetic datasets. We evaluate the results on real datasets using case studies. We perform comprehensive analysis of objects to justify the top few outliers returned by the proposed algorithm. The code and the data sets are available at [XYZ] (link not shared now for anonymity purposes). All experiments are performed on a machine with Intel core i3-2330M processor running at 2.20GHz and 4GB RAM.

We compare our algorithm with Community Outlier Detection Algorithm (CODA) [12]. CODA is also an algorithm for community-based outlier detection which performs community detection using node data and linkage; but ignores edge-data completely.

A. Synthetic Datasets

We synthetically generate three graphs: Graph A ($|V|=1000$), Graph B ($|V|=10000$) and Graph C ($|V|=100000$). The node data is generated using 5 different Gaussian distributions and similar procedure is followed to generate edge data as well. Then for each of these graphs, different percentage of outliers are injected by randomly changing the data associated with 1%, 5% and 10% of the nodes.

We generate a variety of synthetic datasets by varying the number of nodes in the graph, percentage of outliers injected in the graph and percentage of outliers to be extracted by the algorithm. Table I shows the synthetic dataset results for the baseline (CODA) as well as the proposed algorithm (HCODA). We set the number of clusters (K) to 5 for these experiments (based on the generation process). We measure the performance of the two algorithms from two perspectives: firstly, accuracy of extraction of the injected outliers (OD Acc.) and secondly, how well it assigns the community label to all the nodes of the graph (CA Acc.). Note that each cell of the table corresponds to average over five runs of the algorithm. As the table shows, the proposed algorithm (HCODA) is 6.2% more accurate than the baseline on average in terms of outlier detection accuracy. Similarly, HCODA is 2.2% better than CODA on average. The performance is consistent across various graph sizes as well as across different degrees of outlierness in the graph.

Hyperparameter Sensitivity: To understand the impact of various hyperparameters (λ 's), we performed a few experiments. The results are shown in Figures 3, 4 and 5. Figure 3 shows the sensitivity of the outlier detection accuracy with respect to variation of λ_3 . We plot the curves for Graph A and B for the case of 10% injected outliers and 10% outliers to be extracted. Recall that λ_3 is the weight for the triangular clique. As can

TABLE I: Synthetic Dataset Results on Graphs A, B and C (K=5; OD Acc.=Outlier Detection Accuracy, CA Acc.=Community Assignment Accuracy; CODA is Baseline, HCODA is the proposed algorithm)

$ V $	% injected	% extracted	CODA [12] OD Acc.	HCODA OD Acc.	CODA [12] CA Acc.	HCODA CA Acc.	Average runtime of HCODA (ms)
1000	1	1	1.000	1.000	0.714	0.713	898.6
1000	5	1	1.000	1.000	0.725	0.735	930
1000	5	5	0.760	0.800	0.739	0.756	899
1000	10	1	1.000	1.000	0.683	0.699	917.2
1000	10	5	0.960	0.980	0.720	0.738	913
1000	10	10	0.760	0.840	0.729	0.762	926
10000	1	1	0.690	0.730	0.770	0.784	9846.6
10000	5	1	0.970	1.000	0.740	0.755	10259
10000	5	5	0.746	0.780	0.758	0.777	10478.8
10000	10	1	1.000	1.000	0.705	0.718	10026.8
10000	10	5	0.958	1.580	0.741	0.757	9840.4
10000	10	10	0.795	0.814	0.759	0.777	9626.8
100000	1	1	0.722	0.735	0.773	0.787	120587
100000	5	1	0.995	0.996	0.747	0.762	113171.5
100000	5	5	0.751	0.789	0.764	0.783	113749.2
100000	10	1	0.997	0.999	0.698	0.714	104812.4
100000	10	5	0.946	0.974	0.733	0.751	103004
100000	10	10	0.784	0.808	0.750	0.770	100314

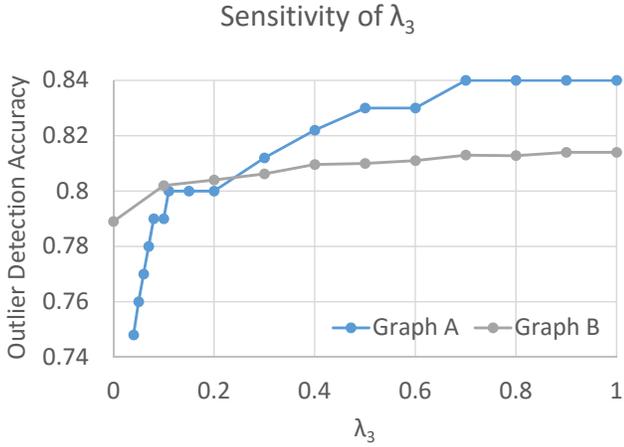


Fig. 3: Sensitivity of λ_3

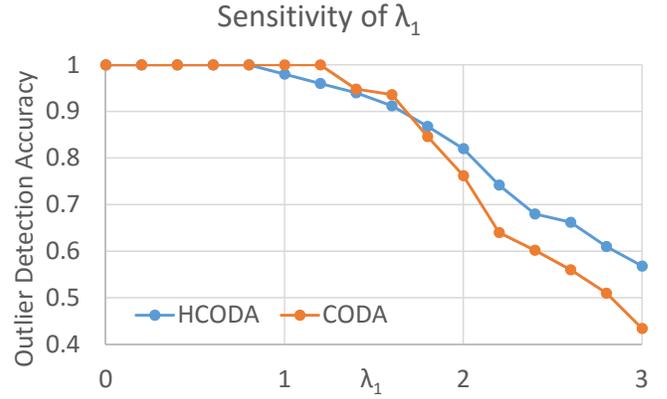


Fig. 4: Sensitivity of λ_1 (Graph A)

be seen from the figure, higher values of λ_3 are preferable but in general any value greater than 0.5 is reasonable.

Figures 4 and 5 show the sensitivity of the outlier detection accuracy with respect to variation of λ_1 for both the proposed algorithm (HCODA) and the baseline (CODA). Figures 4 is the plot for Graph A while Figures 5 is the one for Graph B. Both the plots are for the case of 10% injected outliers and 1% outliers to be extracted. Recall that λ_1 is the weight for the linkage in the original graph. As can be seen from the figure, lower values of λ_1 are preferable both in the case of CODA as well as HCODA.

We noticed that the algorithm is not significantly sensitive to the parameter λ_2 and hence we do not show the corresponding plot.

B. Four Area Dataset

DBLP (<http://dblp.uni-trier.de/>) contains information about computer science journals and proceedings. Four Area is a subset of DBLP for the four areas of data mining (DM), databases (DB), information retrieval (IR) and machine learning (ML). It

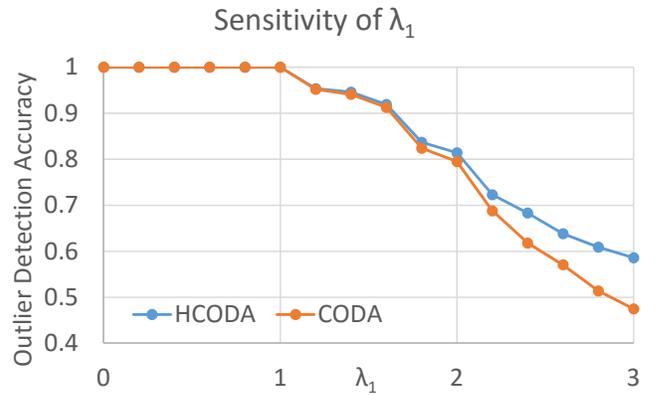


Fig. 5: Sensitivity of λ_1 (Graph B)

consists of papers from 20 conferences (five per area): KDD, PAKDD, ICDM, PKDD, SDM, ICDE, VLDB, SIGMOD, PODS, EDBT, SIGIR, WWW, ECIR, WSDM, IJCAI, AAAI, ICML, ECML, CVPR, CIKM. Thus, we consider the authors who have published papers in these conferences, and work with the co-authorship network (edge weight = co-authorship frequency). The edge data consisted of a vector of size four representing the count of papers co-authored in a particular research area.

The graph contains a total of 42844 author nodes. Each author is associated with a vector of size 20 containing the count of papers published by an author in the twenty conferences. There is an edge between two authors if they co-authored a paper together and the count of such co-authored papers is the weight $W_{v_i v_j}$ between two nodes.

The graph contains 118618 co-authorship edges. The weight of the link between the author-vertex and the co-authorship-edge-vertex in the HMRF (i.e., $W_{v_i e_{ij}}$) is set to the inverse of the number of co-authors of author v_i in the graph. $W_{v_i, v_j, e_{ij}}$ is set to ratio of number of papers where authors v_i and v_j are co-authors to the total number of papers on which either v_i or v_j are authors. We set the number of communities to $K=4$ (i.e., four normal communities and an outlier community). Also, we set the percentage of outlier parameter r as 1%. The average execution time of the algorithm is 30.32 seconds.

Case Studies: Result of algorithm show that it is able to identify interesting nodes(authors) and edges(co-authorship). We validate the result produced by algorithm by manually visiting the authors' homepages. For example, work by Sameer K. Antani is mainly in Clinical data standards and electronic medical records, but he published a paper with authors working in information retrieval i.e. node data of Sameer K. Antani is different from the fellow co-authors but the relationship is similar to what two authors from information retrieval should have and hence the algorithm has identified this node as interesting one. Similarly, Ivo Krka mainly focuses on software engineering and system modeling, however, he has published a paper on data mining and hence identified by the algorithm.

The proposed algorithm also identified authors based on the edge information. We discuss a few cases now. Anindya Datta usually publishes in information systems, and Debra E. VanderMeer usually publishes work related to design science research, but their co-authored work is published on data management and very large databases topics. Sabyasachi Saha has co-authored papers with Sandip Sen on topics related to artificial intelligence and he has also published work on information and knowledge management with Pang-Ning Tan. The algorithm was able to identify him because of his work in multiple research areas.

VII. CONCLUSION

In this paper, we introduced a method (HCODA) that detects novel Holistic Community Outlier graph nodes by taking into account the node data and edge data simultaneously to detect

anomalies. We modeled the problem as a community detection task using a Hidden Random Markov Model, where outliers form a separate community. We used ICM and EM to infer the hidden community labels and model parameters iteratively. Experimental results show that the proposed algorithm consistently outperforms the baseline CODA method on synthetic data, and also identifies meaningful community outliers from the Four Area network data.

REFERENCES

- [1] James Abello, Tina Eliassi-Rad, and Nishchal Devanur. Detecting novel discrepancies in communication networks. In *ICDM 2010, The 10th IEEE International Conference on Data Mining, Sydney, Australia, 14-17 December 2010*, pages 8–17, 2010.
- [2] Charu C. Aggarwal. *Outlier Analysis*. Springer, 2013.
- [3] Charu C. Aggarwal, Yuchen Zhao, and Philip S. Yu. Outlier Detection in Graph Streams. In *Proc. of the 27th Intl. Conf. on Data Engineering (ICDE)*, pages 399–409. IEEE Computer Society, 2011.
- [4] Leman Akoglu and Christos Faloutsos. Event detection in time series of mobile communication graphs. In *Army Science Conference*, pages 77–79, 2010.
- [5] Leman Akoglu, Mary McGlohon, and Christos Faloutsos. Oddball: Spotting Anomalies in Weighted Graphs. In *Pacific-Asia Conference on Knowledge Discovery and Data Mining*, pages 410–421. Springer, 2010.
- [6] Leman Akoglu, Hanghang Tong, and Danai Koutra. Graph based Anomaly Detection and Description: A Survey. *Data Mining and Knowledge Discovery*, 29(3):626–688, 2015.
- [7] Julian Besag. On the Statistical Analysis of Dirty Pictures. *Journal of the Royal Statistical Society. Series B (Methodological)*, pages 259–302, 1986.
- [8] Alex Beutel, Wanhong Xu, Venkatesan Guruswami, Christopher Palow, and Christos Faloutsos. Copycatch: Stopping Group Attacks by Spotting Lockstep Behavior in Social Networks. In *Proceedings of the 22nd international conference on World Wide Web*, pages 119–130. ACM, 2013.
- [9] Brigitte Boden, Stephan Günnemann, Holger Hoffmann, and Thomas Seidl. Mining Coherent Subgraphs in Multi-layer Graphs with Edge Labels. In *Proceedings of the 18th ACM SIGKDD international conference on Knowledge discovery and data mining*, pages 1258–1266. ACM, 2012.
- [10] Deepayan Chakrabarti. Autopart: Parameter-free Graph Partitioning and Outlier Detection. In *European Conference on Principles of Data Mining and Knowledge Discovery*, pages 112–124. Springer, 2004.
- [11] Varun Chandola, Arindam Banerjee, and Vipin Kumar. Anomaly detection: A Survey. *ACM computing surveys (CSUR)*, 41(3):15, 2009.
- [12] Jing Gao, Feng Liang, Wei Fan, Chi Wang, Yizhou Sun, and Jiawei Han. On Community Outliers and their Efficient Detection in Information Networks. In *Proceedings of the 16th ACM SIGKDD international conference on Knowledge discovery and data mining*, pages 813–822. ACM, 2010.
- [13] Manish Gupta, Jing Gao, Charu Aggarwal, and Jiawei Han. Outlier Detection for Temporal Data. *Synthesis Lectures on Data Mining and Knowledge Discovery*, 5(1):1–129, 2014.
- [14] Manish Gupta, Jing Gao, and Jiawei Han. Community Distribution Outlier Detection in Heterogeneous Information Networks. In *Joint European Conference on Machine Learning and Knowledge Discovery in Databases*, pages 557–573. Springer, 2013.
- [15] Manish Gupta, Jing Gao, Yizhou Sun, and Jiawei Han. Community Trend Outlier Detection using Soft Temporal Pattern Mining. In *Proc. of the European Conf. on Machine Learning and Knowledge Discovery in Databases (ECML PKDD)*, pages 692–708, 2012.
- [16] Manish Gupta, Jing Gao, Yizhou Sun, and Jiawei Han. Integrating Community Matching and Outlier Detection for Mining Evolutionary Community Outliers. In *Proc. of the 18th ACM SIGKDD Intl. Conf. on Knowledge Discovery and Data Mining (KDD)*, pages 859–867, 2012.
- [17] Manish Gupta, Jing Gao, Xifeng Yan, Hasan Cam, and Jiawei Han. Top-k interesting subgraph discovery in information networks. In *2014 IEEE 30th International Conference on Data Engineering*, pages 820–831. IEEE, 2014.

- [18] Victoria J Hodge and Jim Austin. A Survey of Outlier Detection Methodologies. *Artificial intelligence review*, 22(2):85–126, 2004.
- [19] Tsuyoshi Idé and Hisashi Kashima. Eigenspace-based Anomaly Detection in Computer Systems. In *Proc. of the 10th ACM SIGKDD Intl. Conf. on Knowledge Discovery and Data Mining (KDD)*, pages 440–449, 2004.
- [20] Brandon Pincombe. Anomaly Detection in Time Series of Graphs using ARMA Processes. *ASOR Bulletin*, 24(4):2–10, 2005.
- [21] Guo-Jun Qi, Charu C Aggarwal, and Thomas Huang. Community Detection with Edge Content in Social Media Networks. In *2012 IEEE 28th International Conference on Data Engineering*, pages 534–545. IEEE, 2012.
- [22] Stephen Ranshous, Shitian Shen, Danai Koutra, Steven Harenberg, Christos Faloutsos, and Nagiza F. Samatova. Anomaly detection in dynamic detworks: A survey. *WIREs Comp Stat*, 7(3):223–247, 2015.
- [23] Jimeng Sun, Huiming Qu, Deepayan Chakrabarti, and Christos Faloutsos. Neighborhood Formation and Anomaly Detection in Bipartite Graphs. In *Fifth IEEE International Conference on Data Mining (ICDM'05)*, pages 418–425. IEEE, 2005.
- [24] Jimeng Sun, Dacheng Tao, and Christos Faloutsos. Beyond streams and graphs: dynamic tensor analysis. In *Proceedings of the 12th ACM SIGKDD international conference on Knowledge discovery and data mining*, pages 374–383. ACM, 2006.