Group-based Fraud Detection Network on e-Commerce Platforms

Jianke Yu\textsuperscript{1,2}, Hanchen Wang\textsuperscript{1,3}, Xiaoyang Wang\textsuperscript{4}, Zhao Li\textsuperscript{5,6}, Lu Qin\textsuperscript{3}, Wenjie Zhang\textsuperscript{4}, Jian Liao\textsuperscript{2}, Ying Zhang\textsuperscript{1,3}

\textsuperscript{1} Zhejiang Gongshang University, \textsuperscript{2} Alibaba Group, \textsuperscript{3} University of Technology Sydney, \textsuperscript{4} University of New South Wales, \textsuperscript{5} Zhejiang University, \textsuperscript{6} Hangzhou Link2Do Technology

KDD 2023
Background

Attributed Bipartite Graph

An attributed bipartite graph is a type of graph which consists of two sets of vertices that are linked by edges. The vertices have additional attributes, making this graph particularly useful for representing information in the field of e-commerce.

![Attributed Bipartite Graph Diagram]

- Lisa \((u_1)\)
- Sam \((u_2)\)
- Bob \((u_3)\)
- Andy \((u_4)\)
- Kane \((u_5)\)
- Eric \((u_6)\)
- Kate \((u_7)\)

- Bottle \((v_1)\)
- Lamp \((v_2)\)
- Headphones \((v_3)\)
- Basketball \((v_4)\)
- Camera \((v_5)\)
- Microphone \((v_6)\)
Background

Group-based Frauds on Attributed Bipartite Graphs

Group-based fraud is becoming increasingly rampant:

“Ride Item’s Coattails” attack (edge classification)

Sockpuppet-based Targeted Attack on Reviewing Systems

(STARS attack) (vertex classification)

![Attributed Bipartite Graph Diagram]

<table>
<thead>
<tr>
<th>Rating</th>
<th>Score</th>
<th>Rating</th>
<th>Score</th>
</tr>
</thead>
<tbody>
<tr>
<td>(u₁, p₁)</td>
<td>1</td>
<td>(u₅, p₅)</td>
<td>0.5</td>
</tr>
<tr>
<td>(u₁, p₂)</td>
<td>-1</td>
<td>(u₆, p₂)</td>
<td>0</td>
</tr>
<tr>
<td>(u₁, p₃)</td>
<td>0.5</td>
<td>(u₆, p₃)</td>
<td>1</td>
</tr>
<tr>
<td>(u₃, p₅)</td>
<td>0.5</td>
<td>(u₆, p₆)</td>
<td>1</td>
</tr>
<tr>
<td>(u₂, p₆)</td>
<td>0.5</td>
<td>(u₇, p₁)</td>
<td>1</td>
</tr>
<tr>
<td>(u₃, p₁)</td>
<td>0</td>
<td>(u₇, p₃)</td>
<td>1</td>
</tr>
<tr>
<td>(u₂, p₃)</td>
<td>1</td>
<td>(u₇, p₄)</td>
<td>1</td>
</tr>
<tr>
<td>(u₂, p₅)</td>
<td>1</td>
<td>(u₇, p₆)</td>
<td>0.5</td>
</tr>
<tr>
<td>(u₃, p₃)</td>
<td>0</td>
<td>(u₈, p₁)</td>
<td>1</td>
</tr>
<tr>
<td>(u₃, p₄)</td>
<td>-0.5</td>
<td>(u₈, p₂)</td>
<td>0</td>
</tr>
<tr>
<td>(u₃, p₆)</td>
<td>0.5</td>
<td>(u₈, p₄)</td>
<td>0.5</td>
</tr>
<tr>
<td>(u₄, p₂)</td>
<td>-1</td>
<td>(u₈, p₅)</td>
<td>1</td>
</tr>
<tr>
<td>(u₄, p₆)</td>
<td>1</td>
<td>(u₉, p₁)</td>
<td>1</td>
</tr>
<tr>
<td>(u₅, p₁)</td>
<td>-1</td>
<td>(u₉, p₃)</td>
<td>0.5</td>
</tr>
<tr>
<td>(u₅, p₂)</td>
<td>-1</td>
<td>(u₉, p₅)</td>
<td>0.5</td>
</tr>
<tr>
<td>(u₅, p₄)</td>
<td>0</td>
<td>(u₉, p₆)</td>
<td>0.5</td>
</tr>
</tbody>
</table>

Image source: STARS: Defending against Sockpuppet-Based Targeted Attacks on Reviewing Systems
Background

SOTA method for “Ride Item’s Coattails” attack

RICD ((α, k1, k2)-biclique): fraud detection method for “Ride Item’s Coattails” attack. Can only utilize structural information.

Tianchi competition winner’s algorithm: classification method. Can only use attribute information.
Background

SOTA method for STARS attack

Unable to make good use of label information.

Algorithm RTV

Input: Rating graph $G = (\mathcal{U} \cup \mathcal{P}, \mathcal{R}, sc)$, weights $\alpha_1, \alpha_2, \beta_1, \beta_2, \gamma_1, \gamma_2, \gamma_3, \gamma_4$, threshold $\epsilon$

Output: $\text{fair}(u) \ \forall u \in \mathcal{U}$, $\text{good}(p) \ \forall p \in \mathcal{P}$, $\text{rel}(u, p) \ \forall (u, p) \in \mathcal{R}$

1. for each $u \in \mathcal{U}$, $\text{fair}_0(u) \leftarrow \text{norm}(u)$
2. for each $p \in \mathcal{P}$, $\text{good}_0(p) \leftarrow \text{norm}(p)$
3. for each $(u, p) \in \mathcal{R}$, $\text{rel}_0(u, p) \leftarrow \text{norm}(u, p)$
4. $\mu_f \leftarrow \frac{\sum_{u \in \mathcal{U}} \text{fair}_0(u)}{|\mathcal{U}|}$, $\mu_g \leftarrow \frac{\sum_{p \in \mathcal{P}} \text{good}_0(p)}{|\mathcal{P}|}$
5. $t \leftarrow 1$
6. for each $u \in \mathcal{U}$, $\text{fair}_t(u) \leftarrow$ value computed as specified in Section 4.1, with $\text{rel}(u, p) = \text{rel}_{t-1}(u, p)$
7. for each $p \in \mathcal{P}$, $\text{good}_t(p) \leftarrow$ value computed as specified in Section 4.1, with $\text{rel}(u, p) = \text{rel}_{t-1}(u, p)$
8. for each $(u, p) \in \mathcal{R}$, $\text{rel}_t(u, p) \leftarrow$ value computed as specified in Section 4.1, with $\text{fair}(u) = \text{fair}_t(u)$
9. $\Delta \leftarrow \max \left( \sum_{u \in \mathcal{U}} |\text{fair}_t(u) - \text{fair}_{t-1}(u)|, \sum_{p \in \mathcal{P}} |\text{good}_t(p) - \text{good}_{t-1}(p)|, \sum_{(u, p) \in \mathcal{R}} |\text{rel}_t(u, p) - \text{rel}_{t-1}(u, p)| \right)$
10. if $\Delta > \epsilon$ or $t = 1$ then $t \leftarrow t + 1$ and go to Line 6
11. return $\text{fair}_t(u) \ \forall u \in \mathcal{U}$, $\text{good}_t(p) \ \forall p \in \mathcal{P}$, $\text{rel}_t(u, p) \ \forall (u, p) \in \mathcal{R}$
Background

Existing methods

Classification Methods:
• Imbalanced labeled vertices, community information.

Cohesive Subgraph Mining Methods:
• Attribute and label information, suffer from NP-completeness.

Fraud Detection Methods:
• Global topological and attribute information, label information, manual parameter setting.
Overview

Group-based Fraud Detection method: GFDN
Overview

Group-based Fraud Detection method: GFDN

Attributed Bipartite Graph

$(\alpha, \beta)$—core

GNNLayer

$W(\mathcal{U}, s)$

$W(\mathcal{V}, s)$

Encoder

Decoder

$t$—distribution

$X(\mathcal{U}, e)$

$X(\mathcal{U}, d)$

Group-Based Fraud Detection

$Y_{\mathcal{U}}$

MLP

$C_{\mathcal{U}}$
Overview

Group-based Fraud Detection method: GFDN
Overview

Group-based Fraud Detection method: GFDN
GFDN

Structural Feature Initialization

(\(\alpha, \beta\))-core:

Given a bipartite graph \(G\) and integers \(\alpha, \beta \in \mathbb{Z}^+\), (\(\alpha, \beta\))-core of \(G\) is denoted as \(G'\) which consists of two vertex sets \(U' \subseteq U\) and \(V' \subseteq V\). The (\(\alpha, \beta\))-core \(G'\) is a maximal bipartite subgraph induced by \(U' \cup V'\) from \(G\) in which all the vertices in \(U'\) have degrees at least \(\alpha\) and all the vertices in \(V'\) have degrees at least \(\beta\).
GFDN

Structural Feature Initialization

GFDN will generate structural features for vertices based on their existence in different \((\alpha, \beta)\)-core.

\[
\hat{X}_{(U,s)} = X_{(U,s)} \odot (I_U W_{(U,s)}), \quad \hat{X}_{(V,s)} = X_{(V,s)} \odot (I_V W_{(V,s)})
\]

Structural Features  Element-wise Product  All-ones Vector  Weight Matrix
GFDN

Fraudster Community Detection

BDCN - Autoencoder:
Autoencoder in Bipartite Deep Clustering Network (BDCN) can:
1. preserving both structural and attribute information from the input features.
2. Generate high-quality community representation for customer vertices.

It can achieve self-supervised fraud community detection using a loss function measures with Student's t-distribution kernel.
GFDN

Fraudster Community Detection

BDCN - GNN:
GNN in BDCN can aggregate on attribute bipartite graph and preserve the attribute information and structural information of the graph well. The output of each encoding layer will be used.
"Ride Item’s Coattails" Attack:
In "Ride Item’s Coattails" attack, not all edges related to fraudsters necessarily have attack implications. GFDN will perform multi-task training on this issue, predicting both fraudsters and fraudulent attack.

STARS Attack:
STARS attack detection aims to detect fraudsters, in which case GFDN only needs to perform the vertex classification task.
The final loss function will be composed of the loss functions of the aforementioned training objectives, including reconstruction of autoencoder, community prediction, fraudster prediction, and fraudulent attack prediction. The sum of the weights of all parts of them is 1.

\[ L = \omega_{ae} L_{ae} + \omega_{c} L_{c} + \omega_{l} L_{l} + \omega_{e} L_{e} \]
Experiments

Experimental Setup

• Dataset
  • 4 real-life datasets.

• Compared methods
  • 5 learning-based methods.
  • 2 pattern-based methods.
  • 4 fraud detection methods.
  • A naïve model and four ablated GFDNs

• Parameter settings
  • The number of GNN layer: 4.
  • The number of community: 32.
  • Hidden dimension: 128.
  • The selected GNN is GraphSAGE.

• Implementation
  • Structure information extraction: C++
  • Other Parts of the Model: Python + Pytorch Geometric.

| Dataset | |E| |U| |V| % Fraudulent | % Legitimate |
|---------|----------|----------|----------|----------|----------------|----------------|
| TB      | 3,085,653 | 996,090  | 381,611  | 0.62%    | 3.53%          |
| TC      | 1,050,000  | 532,345  | 239,840  | 2.86%    | 11.43%         |

Table 1: Datasets for “Ride Item’s Coattails” Attack Detection

| Dataset | |E| |U| |V| % Fraudulent | % Legitimate |
|---------|----------|----------|----------|----------|----------------|----------------|
| Alpha   | 24,186    | 3,286    | 3,754    | 3.10%    | 4.20%          |
| OTC     | 35,592    | 4,814    | 5,858    | 3.70%    | 2.80%          |

Table 2: Datasets for STARS Attack Detection
# Experiments

## Effectiveness Evaluation Results for “Ride Item’s Coattails” Detection

<table>
<thead>
<tr>
<th></th>
<th>TB Data</th>
<th></th>
<th></th>
<th>TC Data</th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>F1</td>
<td>Acc</td>
<td>AUC</td>
<td>Pre</td>
<td>Recall</td>
<td>F1</td>
</tr>
<tr>
<td>LPA</td>
<td>0.2737</td>
<td>0.4627</td>
<td>0.5517</td>
<td>0.1715</td>
<td>0.6785</td>
<td>0.2056</td>
</tr>
<tr>
<td>SBGNN</td>
<td>0.4789</td>
<td>0.8228</td>
<td>0.7947</td>
<td>0.4279</td>
<td>0.5438</td>
<td>0.3676</td>
</tr>
<tr>
<td>BiGI</td>
<td>0.5359</td>
<td>0.8540</td>
<td>0.8491</td>
<td>0.5097</td>
<td>0.5649</td>
<td>0.4039</td>
</tr>
<tr>
<td>SIHG</td>
<td>0.6449</td>
<td>0.8709</td>
<td>0.8692</td>
<td>0.5470</td>
<td>0.7853</td>
<td>0.5947</td>
</tr>
<tr>
<td>Tianchi</td>
<td>0.6446</td>
<td>0.8752</td>
<td>0.9342</td>
<td>0.5606</td>
<td>0.7581</td>
<td>0.5364</td>
</tr>
<tr>
<td>RICD</td>
<td>0.6518</td>
<td>0.8405</td>
<td>0.9063</td>
<td>0.4834</td>
<td><strong>1.0000</strong></td>
<td>0.4784</td>
</tr>
<tr>
<td>(α, β)-core</td>
<td>0.8081</td>
<td>0.9449</td>
<td>0.8757</td>
<td>0.8417</td>
<td>0.7770</td>
<td>0.6348</td>
</tr>
<tr>
<td>FRAUDAR</td>
<td>0.2580</td>
<td>0.1481</td>
<td>0.4963</td>
<td>0.1483</td>
<td>0.9927</td>
<td>0.2020</td>
</tr>
<tr>
<td>CF1</td>
<td>0.2407</td>
<td>0.7698</td>
<td>0.5532</td>
<td>0.2371</td>
<td>0.2445</td>
<td>0.1620</td>
</tr>
<tr>
<td>CF2</td>
<td>0.4675</td>
<td>0.7603</td>
<td>0.7376</td>
<td>0.3497</td>
<td>0.7052</td>
<td>0.3588</td>
</tr>
<tr>
<td>Naive</td>
<td>0.8109</td>
<td>0.9473</td>
<td>0.9844</td>
<td>0.8736</td>
<td>0.7565</td>
<td>0.6397</td>
</tr>
<tr>
<td>GFDN-S</td>
<td>0.6867</td>
<td>0.9202</td>
<td>0.9653</td>
<td>0.8284</td>
<td>0.5864</td>
<td>0.6122</td>
</tr>
<tr>
<td>GFDN-F</td>
<td>0.9212</td>
<td>0.9754</td>
<td>0.9886</td>
<td>0.8821</td>
<td>0.9639</td>
<td>0.6401</td>
</tr>
<tr>
<td>GFDN-L</td>
<td>0.9398</td>
<td>0.9813</td>
<td>0.9964</td>
<td>0.9050</td>
<td>0.9775</td>
<td>0.7015</td>
</tr>
<tr>
<td>GFDN-C</td>
<td>0.9423</td>
<td>0.9821</td>
<td>0.9967</td>
<td>0.9086</td>
<td>0.9785</td>
<td>0.7048</td>
</tr>
<tr>
<td>GFDN</td>
<td><strong>0.9522</strong></td>
<td><strong>0.9853</strong></td>
<td><strong>0.9974</strong></td>
<td><strong>0.9254</strong></td>
<td>0.9806</td>
<td><strong>0.7226</strong></td>
</tr>
</tbody>
</table>
Experiments

Comparison with Pattern-based Algorithms

![Comparison with Pattern-based Algorithms Chart]

- AUC
- F1

(2,1)-core (2,5)-core RICD (2,1)-split (2,5)-split GFDN
Experiments

Query Time Evaluation of “Ride Item’s Coattails” Detection

<table>
<thead>
<tr>
<th>Algorithm</th>
<th>TC (Time(s))</th>
<th>TB (Time(s))</th>
</tr>
</thead>
<tbody>
<tr>
<td>LPA</td>
<td></td>
<td></td>
</tr>
<tr>
<td>SBGNN</td>
<td></td>
<td></td>
</tr>
<tr>
<td>BiGI</td>
<td></td>
<td></td>
</tr>
<tr>
<td>SHIG</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Tianchi</td>
<td></td>
<td></td>
</tr>
<tr>
<td>RICD</td>
<td></td>
<td></td>
</tr>
<tr>
<td>(α, β)-core</td>
<td></td>
<td></td>
</tr>
<tr>
<td>FRAUDAR</td>
<td></td>
<td></td>
</tr>
<tr>
<td>CF1</td>
<td></td>
<td></td>
</tr>
<tr>
<td>CF2</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Naive</td>
<td></td>
<td></td>
</tr>
<tr>
<td>GFDN</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Graph showing the query time evaluation for different algorithms.
## Experiments

### Effectiveness Evaluation Results for STARS Detection

<table>
<thead>
<tr>
<th></th>
<th>Alpha</th>
<th>OTC</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>F1</td>
<td>Acc</td>
</tr>
<tr>
<td>FRAUDAR</td>
<td>0.3800</td>
<td>0.2626</td>
</tr>
<tr>
<td>RTV-SUP</td>
<td>0.8652</td>
<td><strong>0.9452</strong></td>
</tr>
<tr>
<td>(α, β)-core</td>
<td>0.7857</td>
<td>0.8767</td>
</tr>
<tr>
<td>Naive</td>
<td>0.8089</td>
<td>0.9018</td>
</tr>
<tr>
<td>GFDN</td>
<td><strong>0.8919</strong></td>
<td><strong>0.9452</strong></td>
</tr>
</tbody>
</table>
Experiments

Effectiveness Evaluation Results for STARS Detection

![Graph showing Alpha and OTC times for different methods: FRAUDAR, RTV-SUP, (α, β)-core, Naive, GFDN.](image)

- **Alpha Time(s):**
  - FRAUDAR: $10^{-3}$
  - RTV-SUP: $10^{-1}$
  - (α, β)-core: $10^3$
  - Naive: $10^5$
  - GFDN: $10^5$

- **OTC Time(s):**
  - FRAUDAR: $10^{-3}$
  - RTV-SUP: $10^{-1}$
  - (α, β)-core: $10^3$
  - Naive: $10^5$
  - GFDN: $10^5$
Experiments

In-Depth Effectiveness Analysis of GFDN

(a) Heatmap of $C_U$  (b) Heatmap of $W(U,s)$, (c) Varying number of $\beta$
Experiments

Parameter Analysis Results in GFDN

(a) Varying $lr$

(b) Varying $K$

(c) Varying $\tau_E$
Thank you!

Jianke Yu
jiankey.zjgsu@gmail.com

Hanchen Wang
hanchen.wang@uts.edu.au