Interpretable and Robust Behavior Abstraction via Environment-Disentangled Heterogeneous Graph

Zhibin Ni[1], **Hai Wan**[1,2], **Xibin Zhao**[1*]

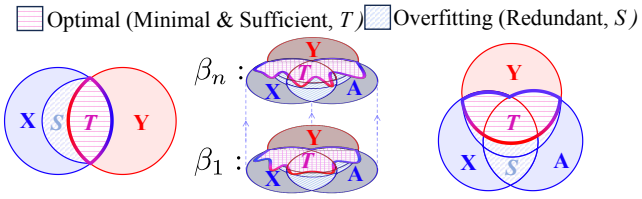
[1] BNRist, KLISS, and School of Software, Tsinghua University
[2] Hunan Sanyou Environmental Technology Co., Ltd., Changsha, Hunan, China
{nzb22}@mails.tsinghua.edu.cn, {wanhai,zxb}@tsinghua.edu.cn

Abstract

To identify the root causes of attacks, behavior abstraction (BA) converts audit logs into multiple behavior graphs and finds similar ones, which has proven effective in bridging the semantic gap and reducing manual workload. Existing works fail to achieve both interpretability and generalization, while also exhibiting limited robustness when facing adversarial attacks. In this paper, we give the first attempt at interpretable and robust behavior abstraction and propose a novel method called ***E**nvironment-**D**isentangled **H**eterogeneous **G**raph **N**eural **N**etwork (**EDHGNN**)*. Motivated by Information Bottleneck (IB) principle, we propose a Heterogeneous Subgraph Disentanglement (HSD) module to disentangle label-relevant and environmental subgraphs through single optimization. We also introduce an Adapted Graph-Level Attention (AGLA) module to extract minimal sufficient representations from label-relevant subgraphs, a Label-Guided Graph Reconstructor (LGGR) to maximize environmental information coverage via reconstruction, and a Relevance Discriminator (RD) to enhance disentanglement quality. Additionally, we construct a new dataset contains ground-truth explanations and 4,160 behavior graphs. Extensive experiments demonstrate that EDHGNN outperforms the state-of-the-art methods in terms of interpretability and robustness against adversarial attacks.

Introduction

Large enterprise systems face increasingly sophisticated global attacks. When reacting to an attack, cyber analysts face the challenge of uncovering the root causes and extent of damages from complex and massive audit logs, which poses *semantic gap* issues and increases their manual workload (Inam et al. 2023). Behavior abstraction (BA) has proven to be an effective solution for the aforementioned issues by abstracting audit logs into multiple behavior graphs and identifying similar ones. With the support of BA, similar behavior graphs are classified into human-understandable behaviors, allowing analysts to examine only a few representative behavior graphs related to threats rather than the entire set of logs (Zengy et al. 2022), thereby significantly reducing the manual workload.



(a) Disentangled IB. (b) Traditional Graph IB. **(c) EDHGNN.**
Y: Predict Target; X: Input Node Features; A: Graph Structure

Figure 1: A comparison of EDHGNN with existing Disentangled IB and Traditional Graph IB.

Existing BA task works fall into two categories: manual-based (Zong et al. 2015; Hossain et al. 2017; Milajerdi et al. 2019a; Hassan, Bates, and Marino 2020; Hossain, Sheikhi, and Sekar 2020; Zhang et al. 2022) and learning-based (Zeng et al. 2021). Manual-based methods suffer from non-generalizable expert-defined patterns and rules, limiting practical value. Recently, learning-based WatSon addressed this by leveraging knowledge graphs to abstract behaviors without predefined patterns. However, two critical issues remain. First, existing works fail to combine interpretability and generalization. Manual-based methods provide interpretations through explicit pattern matching but lack generalization. Conversely, learning-based methods completely overlook interpretability. Second, existing works are vulnerable to noises and adversarial attacks due to susceptibility to perturbation-induced distortions (Zügner, Akbarnejad, and Günnemann 2018), limited generalization across attack strategies (Mbow, Sakurai, and Koide 2022), and vulnerability to evasion tactics (Goyal et al. 2023). To the best of our knowledge, there is no existing solution that is robust against potential noises and adversarial attacks.

Based on the above discussion, BA task fundamentally constitutes a graph classification task. However, directly applying existing methods is insufficient despite their interpretability and robustness. Previous works provide interpretability and robustness by learning minimal and sufficient information using Information Bottleneck (IB) principle (Yu et al. 2021; Sun et al. 2022; Miao, Liu, and Li 2022). Given input G and label Y, graph-based IB obtains label-relevant

*Corresponding author.

subgraph G_T by optimizing:

$$\mathcal{L}_{\mathrm{IB}}\left[p\left(G_T|G\right);\beta\right] = -I\left(G_T;Y\right) + \beta I\left(G;G_T\right), \quad (1)$$

where β controls the trade-off between terms. However, graph-based IB in Eq.(1) causes two issues in Fig. 1: (1) Information amount in label-relevant subgraphs is difficult to guarantee, as boundaries between label-relevant G_T and environmental information G_S are unclear. (2) Multiple β adjustments from β_1 to β_n are required for desired compression levels. These issues can be solved by explicitly modeling environmental information (Pan et al. 2021). While such approaches succeed in other domains (Gao et al. 2021; Jaiswal et al. 2020), they remain unexplored for graph tasks. Additionally, behavior graphs are inherently heterogeneous, yet prior works ignore this nature.

To address these issues, we propose a feasible method termed ***E**nvironment-**D**isentangled **H**eterogeneous **G**raph **N**eural **N**etwork (**EDHGNN**)*, which can provide label-relevant subgraphs for inherent interpretation and learn robust and generalizable graph-level representations for BA task. **First**, inspired by disentangled information bottleneck, Heterogeneous Subgraph Disentanglement (HSD) disentangles label-relevant and environmental subgraphs via stochastic attention mechanisms. **Second**, Adapted Graph-Level Attention (AGLA) module generates minimal sufficient representations through adaptive node importance weighting. **Third**, Label-Guided Graph Reconstructor (LGGR) maximizes environmental information coverage by reconstructing original graphs from environmental subgraphs and labels. **Finally**, Relevance Discriminator (RD) enforces effective disentanglement between label-relevant and environmental information. **Additionally**, to further validate the performance of EDHGNN, we collect a new dataset called Heterogeneous Behavior Graph with Explanations (HBGE). The HBGE dataset is the first to include ground-truth explanations labeled by experts. It consists of 4,160 behavior graphs that represent a wide range of system activities. Briefly, our contributions are as follows:

- To the best of our knowledge, we give first attempt to the interpretable and robust BA task which expands the applications of those traditional ones.
- We propose EDHGNN for interpretable and robust heterogeneous graph representation learning by explicitly disentangling label-relevant and environmental heterogeneous subgraphs with supervision. By extending disentangled information bottleneck, EDHGNN achieves maximum compression without multiple optimizations.
- We construct a new dataset containing ground-truth explanations and 4,160 behavior graphs for benchmarking. Extensive experiments demonstrate that EDHGNN signifcantly outperforms state-of-the-art methods.

Related Work

Behavior Abstraction

Abstracting logs into behavior graphs enhances OS-level activity comprehension and attack investigation. Existing methods divide into: 1) manual-based methods (Zong et al. 2015; Zhang et al. 2022; Milajerdi et al. 2019b,a; Hossain et al. 2017; Hossain, Sheikhi, and Sekar 2020) mine graph patterns as templates or match audit events against knowledge base; 2) learning-based method (Zeng et al. 2021) utilizes TransE and IDF (Ramos et al. 2003) weighted average pooling to represent behaviors as vectors, enabling similar behavior clustering. In contrast, we pioneer interpretable behavior abstraction, demonstrating greater real-world applicability. Moreover, we first propose robust behavior abstraction solutions against adversarial attacks.

Graph Information Bottleneck

Many works have extended the IB (Tishby, Pereira, and Bialek 1999) principle to graph-based tasks. In this paper, we focus on applying IB to graph-level classification tasks. GIB (Yu et al. 2021) addresses key subgraph recognition. GSAT (Miao, Liu, and Li 2022) proposes IB-based attention for interpretable graph learning. VGIB (Yu, Cao, and He 2022) decomposes subgraph recognition into graph perturbation and subgraph selection. PGIB (Seo, Kim, and Park 2024) integrates prototype learning into the IB framework. In contrast, we are the first to explicitly learn both label-relevant and environmental information via disentangled information bottleneck, enabling maximum compression and better environmental coverage. Additionally, we leverage heterogeneous networks to enhance model performance.

Preliminaries

Problem Definition

In this paper, we focus on the interpretable and robust behavior abstraction task. Following convention, given the raw audit logs, we generate a behavior graph dataset $\mathcal{D}$ which consists of a number of behavior graphs $\mathcal{G} = \{G_1, \cdots, G_N\}$ and a set of distinct labels $\mathcal{Y} = \{Y_1, \cdots, Y_N\}$. A behavior graph can be defined as $G = \{\mathcal{V}, \mathcal{E}, \mathcal{R}\}$, where $\mathcal{V}$ is the set of nodes, $\mathcal{E}$ is the set of edges and $\mathcal{R}$ is the set of edge types. We aim to build a model $f(\cdot) : \mathcal{G} \rightarrow \mathcal{Y}$ that produces label-relevant subgraph G_T for interpretation and learn robust graph-level representation for prediction.

Objective of EDHGNN

By extending the disentangled information bottleneck, we formally define the objective of EDHGNN as follows:

$$\begin{aligned}&\mathcal{L}_{\mathrm{EDHGNN}}\left[p\left(G_S|G\right), p\left(G_T|G\right)\right] \\ &\quad = -I\left(G_T;Y\right) - I\left(G;G_S,Y\right) + I\left(G_S;G_T\right).\end{aligned} \quad (2)$$

We encourage (G_S, Y) to represent overall information of G by maximizing $I\left(G;G_S,Y\right)$, ensuring G_S covers environmental information. We encourage accurate Y decoding from G_T by maximizing $I\left(G_T;Y\right)$, ensuring G_T covers label-relevant information. Hence, information in G_S and G_T are both lower bounded. Forcing G_S to be disentangled from G_T by minimizing $I\left(G_S;G_T\right)$ eliminates overlapping information and tightens both bounds, leaving exact information relevant (*resp*, irrelevant) to Y in G_T (*resp*, G_S).

We now implement the objective by deriving variational approximations to $I\left(G_T;Y\right)$ and $I(G;G_S,Y)$. By

introducing variational probabilistic mappings $c\left(y|G_T\right)$ and $r(G|G_S, y)$, tractable variational lower bounds can be formulated as:

$$\begin{aligned} I\left(G_T;Y\right) &= \mathbb{E}_{p(G_T)}\left[\mathbb{E}_{p(y|G_T)}\left[\log c\left(y|G_T\right)\right]\right] \\ &+ D_{\mathrm{KL}}\left[p\left(y|G_T\right)||c\left(y|G_T\right)\right] + H(Y) \\ &\geq \mathbb{E}_{p(G_T)}\left[\mathbb{E}_{p(y|G_T)}\left[\log c\left(y|G_T\right)\right]\right] + H(Y) \\ &\geq \mathbb{E}_{p(G_T,y)}\left[\log c\left(y|G_T\right)\right], \end{aligned} \tag{3}$$

$$\begin{aligned} I(G;G_S,Y) &= \mathbb{E}_{p(G_S,y)}[\mathbb{E}_{p(G|G_S,y)}[\log r(G|G_S,y)]] \\ &+ D_{\mathrm{KL}}(p(G|G_S,y)||r(G|G_S,y))] + H(G) \\ &\geq \mathbb{E}_{p(G,G_S,y)}[\log r(G|G_S,y)], \end{aligned} \tag{4}$$

By using the Markov Chain $Y \leftrightarrow G \leftrightarrow G_T$ and $Y \leftrightarrow G \leftrightarrow G_S$, we can rewrite $p\left(G_T,y\right)$ and $p(G,G_S,y)$ as:

$$\begin{aligned} p\left(G_T,y\right) &= \sum_{G\in\mathcal{G}} p_{data}\left(G\right) p_{data}\left(y|G\right) q\left(G_T|G\right) \\ &= \mathbb{E}_{p_{data}(G)} p_{data}\left(y|G\right) q\left(G_T|G\right), \end{aligned} \tag{5}$$

$$p(G,G_S,y) = p_{\text{data}}(G) p_{\text{data}}(y|G) q(G_S|G), \tag{6}$$

where $q(G_S|G)$ represents the variational probabilistic mapping extracting environment subgraphs G_S from behavior graph G, $p_{\text{data}}(\cdot)$ denotes the data distribution and $q\left(G_T|G\right)$ is the variational probabilistic mapping for extracting the behavior-related subgraph G_T from the behavior graph G.

According to Eq.(3) and Eq.(5), maximizing $I\left(G_T;Y\right)$ is equivalent to:

$$\min_{c,q} \mathbb{E}_{p_{data}(G)}\left[\mathbb{E}_{p_{data}(y|G)}\mathbb{E}_{q(G_T|G)}\left[-\log c\left(y|G_T\right)\right]\right]. \tag{7}$$

The objective $\mathbb{E}_{p_{\text{data}}(y|G)}\mathbb{E}_{q(G_T|G)}\left[-\log c(y|G_T)\right]$ is equivalent to the cross-entropy loss:

$$\begin{aligned} \mathcal{L}_{\text{clf}}(\omega_t(\phi_t\left(G\right)), y) &= -\log \omega_t\left(\phi_t\left(G\right)\right)_y \\ &= \mathbb{E}_{p_{\text{data}}(y|G)}\mathbb{E}_{q(G_T|G)}\left[-\log c(y|G_T)\right], \end{aligned} \tag{8}$$

where $\omega\left(\phi_t\left(G\right)\right)_y$ denotes the y-th element of the $|\mathcal{Y}|$-dimensional probability distribution vector $\omega\left(\phi_t\left(G\right)\right)$. Therefore, we prove that maximizing $I(G_T;Y)$ can be achieved using cross-entropy loss.

From Eq.(4) and Eq.(6), maximizing $I(G;G_S,Y)$ is equivalent to:

$$\min_{r,q} \mathbb{E}_{p_{\text{data}}(G)}\left[\mathbb{E}_{p_{\text{data}}(y|G)}\mathbb{E}_{q(G_S|G)}[-\log r(G|G_S,y)]\right]. \tag{9}$$

The reconstruction loss can be used to model the objective function $\mathbb{E}_{p_{\text{data}}(y|G)}\mathbb{E}_{q(G_S|G)}\left[-\log r(G|G_S,y)\right]$, as follows:

$$\begin{aligned} \mathcal{L}_{\text{recon}}(R_s(\phi_s(G),y),G) &= \frac{1}{|\mathcal{V}|^2}\sum_{u,v}(\mathbf{A}_{uv} - \hat{\mathbf{A}}_{uv})^2 \\ &= \mathbb{E}_{p_{\text{data}}(y|G)}\mathbb{E}_{q(G_S|G)}\left[-\log r(G|G_S,y)\right], \end{aligned} \tag{10}$$

where $|\mathcal{V}|$ denotes the number of nodes in G. Therefore, we prove that maximizing $I(G_T;Y)$ can be achieved by using the aforementioned reconstruction loss.

Finally, we address the implementation of minimizing $I(G_S;G_T)$. Since directly minimizing $I(G_S;G_T) = D_{\mathrm{KL}}[p(G_S,G_T) \parallel p(G_S)p(G_T)]$ is intractable due to complex mixture distributions, we leverage adversarial training inspired by GANs (Goodfellow et al. 2014) to reduce the distributional gap between $p(G_S,G_T)$ and $p(G_S)p(G_T)$, effectively minimizing $I(G_S;G_T)$.

Specifically, EDHGNN samples x uniformly from the dataset and draws from $p(G_S,G_T|x)$ (equivalent to sampling from joint distribution $p(G_S,G_T)$), then randomly shuffles samples along the batch axis to simulate sampling from the marginal product $p(G_S)p(G_T)$.

Finally, we employ the density-ratio trick (Kim and Mnih 2018) by introducing a discriminator d that estimates the probability of its input coming from $p(G_S,G_T)$ rather than $p(G_S)p(G_T)$, and utilizes adversarial training for the discriminator:

$$\begin{aligned} \min_p \max_d V(p,d) &= \mathbb{E}_{p(G_S)p(G_T)}[\log d(G_S,G_T)] \\ &+ \mathbb{E}_{p(G_S,G_T)}[\log(1-d(G_S,G_T))]. \end{aligned} \tag{11}$$

When the Nash Equilibrium is achieved, it holds that $p(G_S,G_T) = p(G_S)p(G_T)$, thereby minimizing $I(G_S;G_T)$.

Methodology

Overview

The proposed framework is illustrated in Fig.2. EDHGNN consists of four modules: Heterogeneous Subgraph Disentanglement (HSD), Adapted Graph-Level Attention (AGLA), Label-Guided Graph Reconstructor (LGGR) and Relevance Discriminator (RD). **First**, HSD disentangles label-relevant and environmental heterogeneous subgraphs via stochastic attention. **Second**, AGLA extracts minimal sufficient graph-level representations for prediction. **Third**, LGGR models irrelevant environmental information through label-guided reconstruction. **Finally**, RD enforces better disentanglement.

Behavior Graph Generation

The first step involves identifying behavior graphs from audit logs. Following prior works (Zeng et al. 2021), we employ adapted forward depth-first search (DFS) to obtain behavior graphs and use TransE for node feature initialization.

Graph Reduction. Since audit logs' low-level nature generates redundant events that do not impact BA tasks (Gao et al. 2018; Inam et al. 2022), we apply existing graph reduction algorithms including LogGC (Lee, Zhang, and Xu 2013), CPR (Xu et al. 2016) and NodeMerge (Tang et al. 2018) to eliminate redundancy.

Heterogeneous Subgraph Disentanglement

Behavior graphs are affected by environmental factors causing large intra-class variations. Mainstream solutions extend

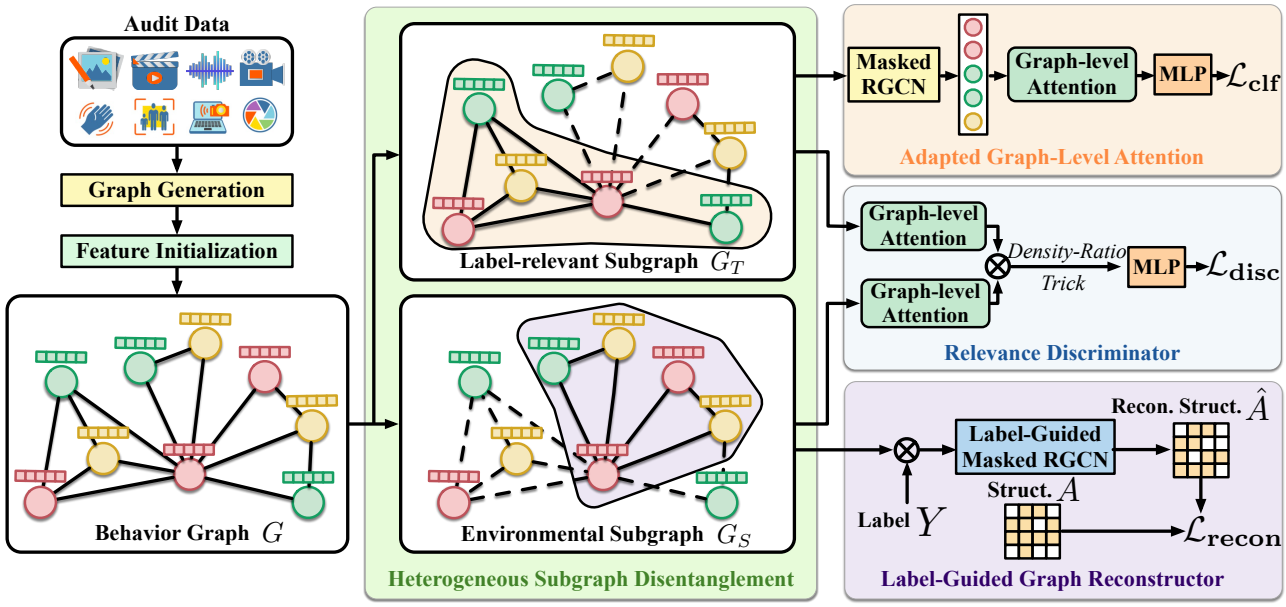


Figure 2: The main framework of the proposed EDHGNN for interpretable and robust behavior abstraction task.

IB principle to extract predictive subgraphs. However, previous IB-based methods fail to explicitly model environmental information, making them inefficient for IB-based deep neural network optimization. Additionally, behavior graphs are naturally heterogeneous graphs, yet no existing solution recognizes label-relevant and environmental heterogeneous subgraphs.

To address this, we propose Heterogeneous Subgraph Disentanglement (HSD) based on disentangled information bottleneck and graph stochastic attention. To extract heterogeneous graph $G_T \in \mathbb{G}_{\text{sub}_t}(G)$, HSD learns heterogeneous subgraph extractor $g_{\phi_{t_1}}$ with parameter ϕ_{t_1}. $g_{\phi_{t_1}}$ blocks environmental information in data G via stochastic attention generated by heterogeneous graph neural networks while preserving label-relevant information in G_T for accurate predictions. The extractor $g_{\phi_{t_1}}$ encodes input graph G via RGCN (Schlichtkrull et al. 2018) into node features $\{z_v^{t_1} | v \in \mathcal{V}\}$. We leverage RGCN for its simplicity and effectiveness in heterogeneous graph learning.

For each edge $(u, v) \in \mathcal{E}$, $g_{\phi_{t_1}}$ uses an MLP layer with sigmoid function to map concatenation $(z_u^{t_1}, z_v^{t_1})$ into $p_{uv}^{t_1} \in [0, 1]$. During training, we sample stochastic attention from Bernoulli distributions $\alpha_{uv}^{t_1} \sim \text{Bern}(p_{uv}^{t_1})$. To ensure gradient computability w.r.t. $p_{uv}^{t_1}$, we employ gumbel-softmax reparameterization (Jang, Gu, and Poole 2017). The extracted heterogeneous graph G_T has attention-selected subgraph $A_T = \alpha^{t_1} \odot A$, where α^{t_1} contains entries $\alpha_{uv}^{t_1}$ for $(u, v) \in \mathcal{E}$ and zeros otherwise. A is the adjacency matrix and $\odot$ denotes entry-wise product. $\alpha_{uv}^{t_1}$ represents edge (u, v) importance in the label-relevant subgraph. The distribution $\mathbb{P}_\phi(G_T|G) = \prod_{u,v \in \mathcal{E}} \mathbb{P}(\alpha_{uv}^{t_1} | p_{uv}^{t_1})$ characterizes G_T given G, where $p_{uv}^{t_1}$ depends on G. This makes attention $\alpha_{uv}^{t_1}$ conditionally independent across edges given input graph G.

For environmental subgraph G_S, we use another extractor g_{ϕ_s} with identical architecture but unshared parameters ϕ_s and ϕ_{t_1}. Similarly, we encode the input graph through RGCN to obtain α_{uv}^s representing edge (u, v) importance in the environmental subgraph.

Adapted Graph-level Attention

Inspired by SiamHAN (Cui et al. 2021), we propose adapted graph-level attention module $w_{\phi_{t_2}}$ to generate graph-level representations. Since α^{t_1} serves as soft edge masks, we propose a masked RGCN model utilizing G_T to obtain node features $\{z_v^{t_2} | v \in \mathcal{V}\}$. We rewrite RGCN's message passing process incorporating α^{t_1}:

$$z_v^{(t_2,l+1)} = \sigma\Big(\sum_{r \in \mathcal{R}} \sum_{u \in \mathcal{N}_v^r} \alpha_{u,v}^{t_1} \cdot \Big(\frac{1}{c_{v,r}} W_r^{(t_2,l)} z_u^{(t_2 l)} + W_0^{(t_2,l)} z_v^{(t_2,l)}\Big)\Big), \tag{12}$$

where $\mathcal{N}_v^r$ denotes neighbor indices of node v under relation $r \in \mathcal{R}$, and $c_{v,r}$ is a normalization constant.

We weight node feature importance for prediction. Node v's importance is:

$$g_v = q^T \cdot tanh(W_g \cdot z_v^{t_2} + b_g), \tag{13}$$

$$\gamma_v = softmax_v(g_v) = \frac{exp(g_v)}{\sum_{v \in \mathcal{V}}(g_v)}, \tag{14}$$

where W_g, b_g, and q are weight matrix, bias vector, and attention parameter matrix. The graph-level representation is:

$$h^t = \sum_{v \in \mathcal{V}} \gamma_v \cdot z_v^{t_2}, \tag{15}$$

We feed h^t into MLP and use cross-entropy loss $\mathcal{L}_{\text{clf}}$ to maximize label-relevant information.

Label-Guided Graph Reconstructor

We use environmental subgraph and label to reconstruct the input graph, maximizing environmental information coverage. However, previous graph reconstruction algorithms have not incorporated label Y, making reconstruction with label information a novel problem. Inspired by RGCN, we propose a label-guided graph reconstructor (R_S) that efficiently incorporates Y into reconstruction. A naive approach assigns separate parameters per label class, but this scales poorly with increasing classes, causing parameter explosion and overfitting on rare labels, identical to RGCN's edge-type scaling problem. Therefore, we adopt the same approach as RGCN. For a given environmental graph G_S, the label Y is treated as the type of every edge in the graph. Consequently, according to the message passing mechanism of RGCN, combined with the α^s, the message passing process on the environmental graph is as follows:

$$z_v^{(s,l+1)} = \sigma\Big(\sum_{Y\in\mathcal{Y}}\sum_{u\in\mathcal{N}_v^r}\alpha_{u,v}^s\cdot\Big(\frac{1}{c_{v,r}}W_Y^{(s,l)}z_u^{(s,l)} + W_0^{(s,l)}z_v^{(s,l)}\Big)\Big), \tag{16}$$

where $\mathbf{W}_Y^{(s)}$ represents the parameters of the RGCN layer corresponding to the label Y.

After obtaining the node embeddings $\mathbf{Z}^S$ (denoted as Z in the following equation) for graph reconstruction, we calculate the reconstructed adjacency matrix $\hat{\mathbf{A}}$ as follows:

$$\hat{\mathbf{A}} = \sigma\left(\mathbf{Z}\mathbf{Z}^\top\right). \tag{17}$$

We proceed to use squared reconstruction loss to build the objective $\mathcal{L}_{\text{recon}}$ that needs to be optimized:

$$\mathcal{L}_{\text{recon}} = \frac{1}{M_n^2}\sum_{u,v}\left(\mathbf{A}_{uv} - \hat{\mathbf{A}}_{uv}\right)^2, \tag{18}$$

Where M_n denotes the number of nodes in G_S.

Relevance Discriminator

Inspired by GAN, we use the the density-ratio-trick and propose a relevance discriminator d to reduce the overlapping information between the label-relevant subgraph and the environmental subgraph. Specifically, $\{z_v^s|v\in\mathcal{V}\}$ and $\{z_v^{t_1}|v\in\mathcal{V}\}$ are connected to graph-level attention layers to obtain graph embeddings h_t and h_s. Then, the relevance discriminator $d:\mathbb{R}^{2K}\to\mathbb{R}$, composed of an MLP and sigmoid function, where K represents the dimensionality of the graph-level embeddings, takes the concatenated vector (h_s, h_t) as input and estimates the probability that this input comes from the joint distribution $p(G_S, G_T)$.

Training Procedure

EDHGNN employs adversarial training where relevance discriminator d is jointly trained with the main architecture. Main architecture parameters are updated using cross-entropy loss, graph reconstruction loss, and discriminator loss. Discriminator d estimates the probability that input originates from the joint distribution between G_S and G_T. The training process is outlined in Alg. 1.

Algorithm 1: Training Procedure for EDHGNN.

Input: Training set $\mathcal{D} = \{(G_1, Y_1), \ldots, (G_N, Y_N)\}$, heterogeneous subgraph extractor $g_{\phi_{t_1}}, g_{\phi_s}$, AGLA module $w_{\phi_{t_2}}$ and LGGR module R_s, RD module d, parameter $\theta = \{\phi_{t_1}, \phi_{t_2}, \phi_s\}$, parameter of discriminator γ, parameter θ and γ optimizer g_θ and g_γ, parameter α and β.

Output: The robust EDHGNN model $f(\cdot)$ and label-relevant subgraph G_T.

while *not converge* **do**

Randomly select batch $\{(G_i, y_i)\}_{i\in\mathcal{B}}$;

Obtain $\{G_{T_i}, G_{S_i}\}_{i\in\mathcal{B}}$ from $p(G_S, G_T)$;

Calculate loss $\mathcal{L}_\theta$ with respect to parameter θ;

$$\mathcal{L}_\theta = \frac{1}{|\mathcal{B}|}\sum_{i\in\mathcal{B}}(\mathcal{L}_{\text{clf}} + \alpha\mathcal{L}_{recon} - \beta\log d(G_{T_i}, G_{S_i})). \tag{19}$$

Update parameter $\theta \leftarrow g_\theta\nabla_\theta\mathcal{L}_\theta$;

Randomly shuffle batch indices $\mathcal{B}$;

Obtain samples $\left\{G_{T_{\pi(i)}}, G_{S_{\pi(i)}}\right\}_{i\in\mathcal{B}}$ from product of marginals $p(G_S)\,p(G_T)$;

Calculate loss $\mathcal{L}_\gamma$ with respect to parameter γ;

$$\mathcal{L}_\gamma = \frac{1}{|\mathcal{B}|}\sum_{i\in\mathcal{B}} -\log\left(1 - d\left(G_{T_i}, G_{S_i}\right)\right) - \log d\left(G_{T_{\pi(i)}}, G_{S_{\pi(i)}}\right). \tag{20}$$

Update parameter $\gamma \leftarrow g_\gamma\nabla_\gamma\mathcal{L}_\gamma$;

end

Experiment Setup

Datasets

We construct a new behavior dataset named HBGE that features ground-truth explanations and sufficient quantity and diversity, addressing the limitations of existing open-source datasets like DARPA Trace (DARPA 2014) and StreamSpot (Manzoor, Milajerdi, and Akoglu 2016) which lack ground-truth explanations and contain only 221 and 600 graphs respectively. Our HBGE dataset comprises 4,160 graphs constructed from 4 real-world attack scenarios: **Apache-1** (CVE-2017-15715 vulnerability), **Apache-2** (Apache SSI remote command execution), **IM-1** (ImageMagick arbitrary file read vulnerability CVE-2022-44268) and **IM-2** (ImageMagick command injection vulnerability CVE-2016–3714). We use Linux Audit as the audit log source and invite domain experts to generate ground truth explanations for each behavior graph.

Baselines

We compare EDHGNN with two categories of baselines. First, we compare prediction performance with existing prediction baselines, including **WatSon** (Zeng et al.

2021),**TopkPool** (Gao and Ji 2019),**SagPool** (Lee, Lee, and Kang 2019),**SortPool** (Zhang et al. 2018),**GIB** (Yu et al. 2021),**GSAT** (Miao, Liu, and Li 2022), **PGIB** (Seo, Kim, and Park 2024). Second, we compare interpretability with existing interpretability baselines. In addition to the **GIB**, **GSAT** and **PGIB**, we also include the following post-hoc interpretation baselines **PGE** (Luo et al. 2020), **GME** (Schlichtkrull, Cao, and Titov 2021).

Adversarial Attack Settings

We compare baselines and the proposed EDHGNN under two adversarial attack settings.

- **Non-targeted Adversarial Attack:** We produce synthetic datasets by attacking *graph structures* and *node features*, respectively. **(1) Attack graph structures.** We randomly remove 10%, 20%, and 30% of the edges from each graph in each dataset. **(2) Attack node features.** We add random Gaussian noise $\lambda \cdot r \cdot \epsilon$ to each dimension of the node features, where r is the reference amplitude of original features, and $\epsilon \sim N(\mathbf{0}, \mathbf{I})$. $\lambda \in \{1.0, 2.0, 3.0\}$ controls the attacking degree.
- **Targeted Adversarial Attack:** We adapt mimicry attacks (Goyal et al. 2023), a strong targeted adversarial strategy making certain graph classes mimic substructures of other classes. We evaluate both *evasion* and *poisoning* attacks. **(1) Evasion attack:** Models train on clean datasets and attack a given proportion of testing behavior graphs. **(2) Poisoning attack:** We attack a proportion of the entire dataset before training and testing. Following default settings in (Goyal et al. 2023), we explore two configurations: (1) misclassifying class 0 as class 1, and (2) misclassifying class 2 as class 3. We set the portion as 50%.

Metrics

For prediction performance, we report accuracy for all datasets. For interpretation evaluation, we report explanation ROC AUC following prior works (Miao, Liu, and Li 2022; Li et al. 2022).

Implementation Details

The proposed EDHGNN model is implemented by PyTorch 2.1 framework on Ubuntu 22.04, and all the evaluations are conducted on NVIDIA GeForce RTX 4090 card. For a fair comparison, we tune the hyper-parameters of all models using grid-search: learning rate $lr \in \{3.0e-5, 1.0e-4, 3.0e-4, 1.0e-3, 3.0e-3, 1.0e-2\}$, batch size $b \in \{18, 32, 64, 128, 256\}$, hidden size $hid \in \{20, 64, 96, 128, 192, 256, 328\}$. In addition, we perform a grid search for the important hyper-parameters of the EDHGNN model, specified as follows: the weight of graph reconstruction loss $\alpha \in \{0, 0.3, 1, 3, 5, 30\}$, the weight of discriminator loss $\beta \in \{0, 0.3, 1, 3, 5, 30\}$. We set the maximum epoch number as 1000 with the early stopping strategy. We report the results of all models in runs with 20 random seeds to minimize the impact of random noise.

	Apache-1	Apache-2	IM-1	IM-2
PGE	58.86	53.25	44.32	49.03
GME	68.54	53.29	51.14	50.19
GIB	<u>86.44</u>	53.23	59.57	84.16
GSAT	78.25	52.68	68.62	62.61
PGIB	82.09	<u>59.12</u>	<u>84.04</u>	<u>89.21</u>
EDHGNN	**89.09**	**61.03**	**89.52**	**91.36**

Table 1: ROC AUC (%). The best results are shown in **bold type** and the runner-ups are <u>underlined</u>.

Results And Analysis

Interpretability Results

In this section, we conduct quantitative experiments measuring explanation accuracy for label-relevant subgraphs. Results are summarized in Table 1.

Results. Post-hoc methods exhibit worse interpretability performance with significantly larger variance across datasets compared to inherently interpretable models. Existing inherently interpretable methods fail to explicitly eliminate biased factors affecting prediction and interpretation, yielding consistently inferior performance versus EDHGNN across all scenarios. EDHGNN significantly outperforms all baselines by $3.05\% \uparrow$ on average and up to $5.48\% \uparrow$, while providing more stable interpretation with smaller variance.

Adversarial Adversarial Attack

We evaluate EDHGNN's prediction performance on behavior abstraction tasks and robustness against both non-targeted and targeted adversarial attacks. We train baselines and EDHGNN on clean datasets, then evaluate on perturbed test sets following standard settings. Results for non-targeted and targeted attacks are presented in Table 2.

Results. EDHGNN consistently outperforms state-of-the-art baselines across all datasets and attack scenarios. Against non-targeted attacks, WatSon and graph pooling baselines exhibit weak robustness, while robust and generalized baselines underperform due to insufficient separation of label-relevant and environmental information and inability to handle heterogeneity challenges. For targeted adversarial attacks, including both evasion and poisoning scenarios, EDHGNN demonstrates exceptional resilience with the lowest average accuracy decrease across all datasets, confirming its robustness against the most challenging targeted adversarial scenarios.

Ablation Study

To verify each module's effectiveness, we conduct ablation study on Apache-1 with four variants:

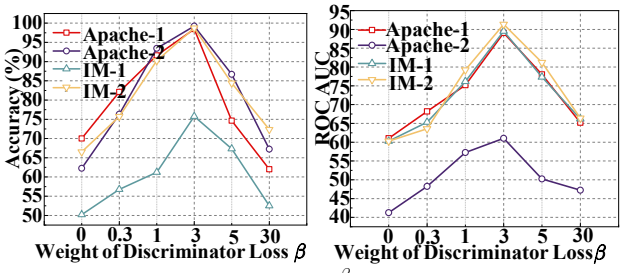
- **w/o AGLA:** Replaces AGLA with readout function.
- **w/o LGGR:** Removes LGGR module.
- **w/o RD:** Removes RD module.
- **w/o HSD:** Removes HSD, disabling AGLA, LGGR, and RD (equivalent to GSAT).

Dataset	Method	Clean	Targeted Attack						Non-targeted Attack							
			Evasion			Poisoning			Feature Attack				Structure Attack			
			$0 \to 1$	$2 \to 3$	Avg↓	$0 \to 1$	$2 \to 3$	Avg↓	$\lambda = 1$	$\lambda = 2$	$\lambda = 3$	Avg↓	10%	20%	30%	Avg↓
Apache-1	WatSon	73.54	56.39	53.18	18.76↓	53.42	52.63	20.52↓	67.59	57.18	50.12	15.24↓	65.42	53.93	50.78	16.83↓
	TopkPool	84.56	69.21	66.56	16.68↓	66.59	60.43	21.05↓	76.21	66.49	57.92	17.69↓	66.59	59.71	56.74	23.55↓
	SagPool	83.27	69.91	65.22	15.70↓	69.12	62.73	17.34↓	76.91	68.87	59.67	14.79↓	69.02	57.62	52.76	23.47↓
	SortPool	68.19	59.65	51.11	12.81↓	59.48	50.39	13.25↓	57.72	51.11	49.06	14.56↓	52.48	51.39	50.68	16.67↓
	GIB	91.50	70.95	72.63	19.71↓	71.47	71.03	20.25↓	86.95	77.63	67.98	14.21↓	71.47	61.03	58.03	27.99↓
	GSAT	91.73	78.36	77.79	11.16↓	74.06	71.93	18.73↓	84.36	78.79	70.25	13.93↓	84.06	75.63	69.57	15.31↓
	PGIB	95.84	81.29	78.32	16.04↓	87.55	82.99	10.57↓	91.29	87.32	69.03	13.29↓	87.55	84.92	68.62	15.48↓
	Ours	**98.45**	**90.83**	**88.56**	**8.76↓**	**89.56**	**87.82**	**9.76↓**	**93.38**	**89.53**	**85.79**	**8.88↓**	**91.63**	**88.99**	**84.78**	**9.98↓**
IM-1	WatSon	40.18	30.59	29.98	9.89↓	30.55	27.15	11.33↓	33.59	31.98	30.23	8.25↓	32.55	30.15	29.02	9.61↓
	TopkPool	42.39	34.14	26.25	12.20↓	31.26	25.43	14.04↓	32.14	31.25	29.75	11.34↓	31.26	30.43	30.35	11.71↓
	SagPool	41.34	30.64	28.35	11.85↓	26.21	26.28	15.10↓	33.64	31.35	30.36	9.56↓	32.21	30.28	29.45	10.69↓
	SortPool	42.36	32.64	30.11	10.98↓	29.71	29.43	12.79↓	35.64	32.11	30.99	9.45↓	34.71	30.43	29.97	10.66↓
	GIB	73.46	60.61	54.08	16.11↓	58.01	47.04	20.93↓	67.61	61.08	58.44	11.08↓	62.01	59.04	57.85	13.83↓
	GSAT	65.31	59.83	55.81	7.49↓	54.92	53.76	10.97↓	62.83	57.81	54.95	6.78↓	64.92	60.76	52.27	5.99↓
	PGIB	68.46	60.69	53.34	11.44↓	57.31	52.25	13.68↓	64.98	61.78	58.69	6.64↓	62.16	57.71	52.38	11.04↓
	Ours	**75.75**	**68.98**	**68.18**	**7.17↓**	**67.16**	**66.36**	**8.99↓**	**70.85**	**69.97**	**66.72**	**6.57↓**	**71.24**	**70.76**	**67.43**	**5.94↓**

Table 2: Accuracy (%) of behavior abstraction task on real-world datasets under targeted and non-targeted adversarial attacks. The best results are in **bold**, runner-ups are underlined.

Datasets	Apache-1		IM-1	
Metric	Accuracy	ROC AUC	Accuracy	ROC AUC
Ours	**98.45**	**89.09**	**75.75**	**89.52**
w/o AGLA	95.51	86.48	74.28	84.92
w/o LGGR	93.22	84.86	71.45	82.73
w/o RD	92.74	80.10	69.67	78.84
w/o HSD	91.73	78.25	68.46	62.62

Table 3: The ablation study results. The best results are shown in **bold type** and the runner-ups are underlined.



(a) Effect on Accuracy (b) Effect on ROC AUC

Figure 3: Hyper-parameter sensitivity of the weight of discriminator loss β.

Results. Results on Apache-1 and IM-1 are conclude in Table 3. Results show that w/o HSD performs worst due to inefficient environmental modeling and heterogeneity handling. Removing RD causes significant decline, confirming the necessity of disentanglement. LGGR removal shows spurious correlations between environmental information and labels. Excluding AGLA degrades performance as basic readout fails to accommodate heterogeneity and node significance. EDHGNN achieves optimal prediction and interpretability by leveraging all modules synergistically.

Hyperparameter Analysis

In this section, we investigate the effect of the weight of discriminator loss β, which is important hyper-parameter of our EDHGNN. Results are reported in Fig. 3.

Results. When β is assigned a very low value, the model performance is significantly reduced. This is because such a choice causes the model to disregard the separation of G_T and G_S, relying solely on the AGLA module and LGGR module to focus on modeling the two subgraphs separately. When β is increased to 3, both the model's interpretability and accuracy reach the optimal level, indicating that setting β to 3 is the best choice. When β rises to 5 and 30, both the interpretability and accuracy of the model decline. This indicates that overly high values of β have adverse effects. The underlying reason may be that, at such high weights, the model struggles to learn both G_T and G_S, thereby rendering ineffective disentanglement.

Conclusion

In this paper, we propose a novel model named EDHGNN which is the first work on interpretable and robust behavior abstraction task. We introduce a HSD module to explicitly disentangle the label-relevant and environmental heterogeneous subgraph. Then a AGLA module is proposed to maximize the label-relevant information. We also propose a LGGR module to fully exclude the environmental information. Besides, a RD module is proposed to enable better disentanglement. We construct a new dataset named HBGE which contains ground-truth explanations and 4,160 behavior graphs. Results show that EDHGNN achieves high interpretability and robustness on the HBGE dataset.

Acknowledgments

This work was partially supported by the Guangdong S&T Programme (No. 2024B0101030002), the NSFC (No. 6212780016), the Ministry of Industry and Information Technology of China, the National Key Research and Development Program of China (No. 2023YFB3307500).

References

Cui, T.; Gou, G.; Xiong, G.; Li, Z.; Cui, M.; and Liu, C. 2021. SiamHAN: IPv6 Address Correlation Attacks on TLS Encrypted Traffic via Siamese Heterogeneous Graph Attention Network. In *USENIX Security Symposium*, 4329–4346.

DARPA. 2014. Transparent Computing Engagement 3 Data Release.

Gao, G.; Huang, H.; Fu, C.; Li, Z.; and He, R. 2021. Information bottleneck disentanglement for identity swapping. In *CVPR*, 3404–3413.

Gao, H.; and Ji, S. 2019. Graph u-nets. In *ICML*, 2083–2092.

Gao, P.; Xiao, X.; Li, D.; Li, Z.; Jee, K.; Wu, Z.; Kim, C. H.; Kulkarni, S. R.; and Mittal, P. 2018. SAQL: A stream-based query system for Real-Time abnormal system behavior detection. In *USENIX Security Symposium*, 639–656.

Goodfellow, I.; Pouget-Abadie, J.; Mirza, M.; Xu, B.; Warde-Farley, D.; Ozair, S.; Courville, A.; and Bengio, Y. 2014. Generative adversarial nets. *NIPS*.

Goyal, A.; Han, X.; Wang, G.; and Bates, A. 2023. Sometimes, you aren't what you do: Mimicry attacks against provenance graph host intrusion detection systems. In *NDSS*.

Hassan, W. U.; Bates, A.; and Marino, D. 2020. Tactical provenance analysis for endpoint detection and response systems. In *IEEE Symposium on Security and Privacy*, 1172–1189.

Hossain, M. N.; Milajerdi, S. M.; Wang, J.; Eshete, B.; Gjomemo, R.; Sekar, R.; Stoller, S.; and Venkatakrishnan, V. 2017. SLEUTH: Real-time attack scenario reconstruction from COTS audit data. In *USENIX Security Symposium*, 487–504.

Hossain, M. N.; Sheikhi, S.; and Sekar, R. 2020. Combating dependence explosion in forensic analysis using alternative tag propagation semantics. In *IEEE Symposium on Security and Privacy*, 1139–1155.

Inam, M. A.; Chen, Y.; Goyal, A.; Liu, J.; Mink, J.; Michael, N.; Gaur, S.; Bates, A.; and Hassan, W. U. 2023. Sok: History is a vast early warning system: Auditing the provenance of system intrusions. In *IEEE Symposium on Security and Privacy*, 2620–2638.

Inam, M. A.; Goyal, A.; Liu, J.; Mink, J.; Michael, N.; Gaur, S.; Bates, A.; and Hassan, W. U. 2022. FAuST: Striking a Bargain between Forensic Auditing's Security and Throughput. In *Proceedings of the Computer Security Applications Conference*, 813–826.

Jaiswal, A.; Moyer, D.; Ver Steeg, G.; AbdAlmageed, W.; and Natarajan, P. 2020. Invariant representations through adversarial forgetting. In *AAAI*.

Jang, E.; Gu, S.; and Poole, B. 2017. Categorical reparameterization with gumbel-softmax. In *ICLR*.

Kim, H.; and Mnih, A. 2018. Disentangling by factorising. In *ICML*, 2649–2658.

Lee, J.; Lee, I.; and Kang, J. 2019. Self-attention graph pooling. In *International conference on machine learning*, 3734–3743. PMLR.

Lee, K. H.; Zhang, X.; and Xu, D. 2013. Loggc: garbage collecting audit log. In *Proceedings of the ACM SIGSAC Conference On Computer & Communications Security*, 1005–1016.

Li, H.; Zhang, Z.; Wang, X.; and Zhu, W. 2022. Learning invariant graph representations for out-of-distribution generalization. *NIPS*.

Luo, D.; Cheng, W.; Xu, D.; Yu, W.; Zong, B.; Chen, H.; and Zhang, X. 2020. Parameterized explainer for graph neural network. *NIPS*.

Manzoor, E.; Milajerdi, S. M.; and Akoglu, L. 2016. Fast memory-efficient anomaly detection in streaming heterogeneous graphs. In *KDD*, 1035–1044.

Mbow, M.; Sakurai, K.; and Koide, H. 2022. Advances in adversarial attacks and defenses in intrusion detection system: A survey. In *International Conference on Science of Cyber Security*, 196–212.

Miao, S.; Liu, M.; and Li, P. 2022. Interpretable and generalizable graph learning via stochastic attention mechanism. In *ICML*.

Milajerdi, S. M.; Eshete, B.; Gjomemo, R.; and Venkatakrishnan, V. 2019a. Poirot: Aligning attack behavior with kernel audit records for cyber threat hunting. In *Proceedings of the ACM SIGSAC Conference On Computer and Communications Security*, 1795–1812.

Milajerdi, S. M.; Gjomemo, R.; Eshete, B.; Sekar, R.; and Venkatakrishnan, V. 2019b. Holmes: real-time apt detection through correlation of suspicious information flows. In *IEEE Symposium on Security and Privacy*, 1137–1152.

Pan, Z.; Niu, L.; Zhang, J.; and Zhang, L. 2021. Disentangled information bottleneck. In *AAAI*.

Ramos, J.; et al. 2003. Using tf-idf to determine word relevance in document queries. In *ICML*, 29–48.

Schlichtkrull, M.; Kipf, T. N.; Bloem, P.; Van Den Berg, R.; Titov, I.; and Welling, M. 2018. Modeling relational data with graph convolutional networks. In *ESWC*.

Schlichtkrull, M. S.; Cao, N. D.; and Titov, I. 2021. Interpreting Graph Neural Networks for NLP With Differentiable Edge Masking. In *ICLR*.

Seo, S.; Kim, S.; and Park, C. 2024. Interpretable Prototype-based Graph Information Bottleneck. *NIPS*.

Sun, Q.; Li, J.; Peng, H.; Wu, J.; Fu, X.; Ji, C.; and Philip, S. Y. 2022. Graph Structure Learning with Variational Information Bottleneck. In *AAAI*.

Tang, Y.; Li, D.; Li, Z.; Zhang, M.; Jee, K.; Xiao, X.; Wu, Z.; Rhee, J.; Xu, F.; and Li, Q. 2018. Nodemerge: Template based efficient data reduction for big-data causality analysis. In *Proceedings of the ACM SIGSAC Conference on Computer and Communications Security*, 1324–1337.

Tishby, N.; Pereira, F. C.; and Bialek, W. 1999. The information bottleneck method. *Allerton*.

Xu, Z.; Wu, Z.; Li, Z.; Jee, K.; Rhee, J.; Xiao, X.; Xu, F.; Wang, H.; and Jiang, G. 2016. High fidelity data reduction for big data security dependency analyses. In *Proceedings of the ACM SIGSAC Conference on Computer and Communications Security*, 504–516.

Yu, J.; Cao, J.; and He, R. 2022. Improving subgraph recognition with variational graph information bottleneck. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 19396–19405.

Yu, J.; Xu, T.; Rong, Y.; Bian, Y.; Huang, J.; and He, R. 2021. Graph information bottleneck for subgraph recognition. *ICLR*.

Zeng, J.; Chua, Z. L.; Chen, Y.; Ji, K.; Liang, Z.; and Mao, J. 2021. WATSON: Abstracting Behaviors from Audit Logs via Aggregation of Contextual Semantics. In *NDSS*.

Zengy, J.; Wang, X.; Liu, J.; Chen, Y.; Liang, Z.; Chua, T.-S.; and Chua, Z. L. 2022. Shadewatcher: Recommendation-guided cyber threat analysis using system audit records. In *IEEE Symposium on Security and Privacy*, 489–506.

Zhang, H.; Cai, L.; Zhao, L.; Yu, A.; Ma, J.; and Meng, D. 2022. LogMiner: A System Audit Log Reduction Strategy Based on Behavior Pattern Mining. In *MILCOM IEEE Military Communications Conference*, 292–297.

Zhang, M.; Cui, Z.; Neumann, M.; and Chen, Y. 2018. An end-to-end deep learning architecture for graph classification. In *AAAI*.

Zong, B.; Xiao, X.; Li, Z.; Wu, Z.; Qian, Z.; Yan, X.; Singh, A. K.; and Jiang, G. 2015. Behavior query discovery in system-generated temporal graphs. *Proceedings of the VLDB Endowment*, 9(4): 240–251.

Zügner, D.; Akbarnejad, A.; and Günnemann, S. 2018. Adversarial attacks on neural networks for graph data. In *KDD*, 2847–2856.