Leveraging Homophily-Augmented Energy Propagation for Bot Detection on Graphs

Bradley Ashmore and Lingwei Chen(✉)

Wright State University, Dayton, OH 45435, USA
{ashmore.3,lingwei.chen}@wright.edu

Abstract. As the developers of malware continuously evolve their attacks and infection methods, so to must bot detection methods advance. Graph Neural Networks (GNNs) have emerged as a promising detection method. However, in most cases communications graphs reflecting bot-infected networks are plagued with class imbalance and a high level of heterophily. Graph oversampling techniques employed to tackle class imbalance on graphs have drawbacks, such as introducing noisy topological structures or exacerbating heterophily within the graph. Out-of-distribution detection (ODD) is considered as an alternative solution to address data imbalance issues, but when applied to graphs, it assumes that the underlying graph structure does not interfere with the learning of data distributions. In this paper, we present the first application of ODD methods for bot detection in a network. We propose a new energy-based ODD model, which surpasses existing ODD methods, including those tailored for ODD on graph data, and effectively mitigates performance degradation caused by graph heterophily. We substantiate our claims through extensive experiments on the TON IoT dataset, which comprises real captured bot data. The experimental results demonstrate that our model achieves state-of-the-art performance in bot detection on graphs with high graph heterophily and extreme class imbalance.

Keywords: Bot Detection · Out-of-distribution Detection · Class Imbalance · Graph Heterophily

1 Introduction

Malicious bots are compromised victim computers or IoT devices, infiltrated by automated malware programs that perform predefined assignments, which can infect a system, steal data, or commit other malicious activities [5,28]. As these bots generally reside in a network by communicating in peer-to-peer (P2P) structures [37], graph neural networks (GNNs) have emerged as one of the most effective bot detection methods [22,36,39], due to their graph learning capabilities [7], where GNN models [17] are devised to perform neighborhood aggregation

Supplementary Information The online version contains supplementary material available at https://doi.org/10.1007/978-981-97-5572-1_5.

M. Onizuka et al. (Eds.): DASFAA 2024, LNCS 14855, pp. 68–83, 2024.
https://doi.org/10.1007/978-981-97-5572-1_5

and label propagation through graph structure. For example, Zhao et al. [39] customized GNNs to detect botnet nodes within Internet communication graphs by identifying their topological patterns. Lo et al. [22] built a graph isomorphism network (GIN) with grouped reversible residual connections to handle performance degradation on botnet communication graphs.

Communication graphs often exhibit a natural imbalance among labeled nodes due to the inherent rarity of bots. We calculate the imbalance ratios (as defined in Sect. 2) over TON IoT networks [1,26], a dataset widely used for bot detection, where in most cases the imbalance ratios are below 0.03. This indicates a significant disparity where malicious bots are greatly outnumbered by benign devices. When the GNN model is trained on such class-imbalanced graphs, it is difficult to accurately identify bots [2,14]. Unfortunately, though class imbalance is prevalent in communication graphs for bot detection, existing graph-based bot detection methods have rarely explored this issue. To mitigate the impact of class imbalance on GNN models, oversampling has recently been generalized to imbalanced graphs [2,38]. This line of research synthesizes nodes in the embedding space and then generates edges to connect them with existing nodes, or removes selected edges to refine the node embedding to exclusively synthesizes nodes that balance class distribution. Either way may significantly disrupt graph structural semantics, resulting in noisy or incomplete neighborhoods for message passing. For instance, GraphSMOTE [38] generates edges using weighted inner products of node embeddings, which may not represent real-world relations between devices only enabled by TCP; HOVER [2] removes edges that satisfy heuristic criteria to prevent node oversmoothing, which may in turn conceal bots from their exploitation of other devices.

Out-of-distribution detection (ODD) [8] has been introduced to empower models with an introspective capability to recognize samples that deviate from the training data used for model preparation, and reject data for which they exhibit low confidence [18]. This becomes particularly advantageous in security-critical applications where anomalous or malicious samples are often limited in representation [4]. This feasibility inspires us to cast bot detection as an ODD problem, which aims at discriminating bots (out-of-distribution) from benign devices (in-distribution). While a surge of ODD methods have been developed [3,12,30], their effectiveness hinges on the assumption of independently sampled inputs which disables them from generalizing with graph data. Such a premise propels ODD exploration extended to graph data [24,32,34]. Addressing distribution shifts in an inter-dependence graph is more challenging, which often requires graph-specific technical innovations. For example, Li et al. [16] introduced a generative framework that incorporates node features, labels, and graph structures to derive a posterior distribution for ODD. Stadler et al. [29] explored uncertainty quantification by proposing Graph Posterior Network (GPN) to perform Bayesian posterior updates for predicting out-of-distribution nodes.

In this paper, we investigate how to leverage ODD for bot detection on graphs. More specifically, our model proceeds by learning node embeddings that encode both node features and graph structure for representation and prediction; these node embeddings are then fed to energy-based supervised classification

loss that is regulated by semi-supervised energy propagation for optimization. Recent work GNNSafe [32] utilizing energy function to handle ODD focuses on energy formulation and transformation across different nodes, which, however, is encumbered with the same drawbacks that plague GNNs - over-smoothing and reliance on the presence of homophily [40]. As discussed in Section 3.1, GNNSafe surprisingly underperforms on graphs when imbalance exists between in-distribution and out-of-distribution nodes, due to its extracted energies being degraded by the GNN's message passing and energy propagation. In communication graphs for bot detection, class imbalance is often accompanied by a high degree of heterophily [2], where bots tend to massively connect with benign devices. To address this issue, we propose a simple yet effective method to mitigate over-smoothing on node embeddings caused by heterophily. Our method treats node neighborhood as structural features of nodes, embedding both structural and semantic features separately, which are then integrated to learn the final node embeddings. These embeddings enable subsequent homophily-augmented energy propagation to amplify the extracted energies for effective out-of-distribution (i.e., bot) detection. In summary, our major contributions are listed as follows:

- We cast bot detection on graphs as an ODD problem to discriminate bots (out-of-distribution) from benign devices (in-distribution).
- We design a new energy-based ODD framework, which can better deal with graph data with extreme low imbalance ratio and high heterophily level.
- Extensive experiments are conducted on TON IoT networks, demonstrating our model can achieve state-of-the-art bot detection performance on graphs.

2 Preliminaries and Problem Statement

Notations. We represent a communication graph for bot detection as $\mathcal{G} = (\mathcal{V}, \mathcal{E}, \mathbf{X})$, where $\mathcal{V}(n = |\mathcal{V}|)$ is the set of devices, $\mathcal{E}$ is the set of edges indicating communication between devices, and $\mathbf{X} \in \mathbb{R}^{n \times d}$ is the feature matrix. Edges E can be organized as an adjacency matrix $\mathbf{A} \in \mathbb{R}^{n \times n}$ and $\mathbf{A}_{ij} = \{0, 1\}$ [15]. Each labeled node has a ground truth $y \in \mathcal{Y} = \{benign, bot\}$. Bot detection learns a model $f_{\mathbf{W}} : (\mathbf{A}, \mathbf{X}) \rightarrow \mathbf{y}$ in a semi-supervised manner. The node embeddings $\mathbf{Z} = f_{\mathbf{W}}(\mathbf{A}, \mathbf{X})$ are used to formulate loss function for model optimization.

Class Imbalance. We define imbalance ratio to quantify the nature of class imbalance in communication graphs as follows: given a graph $\mathcal{G}$ where $\#m$ is the number of malicious bots and $\#b$ is the number of benign devices, the imbalance ratio is represented as $r_i = \frac{\#m}{\#b}$.

Homophily and Heterophily. This paper focuses on homophily in class labels, where a graph with high homophily suggests that connected nodes share the same label with a high probability across an entire graph. Homophily ratio can thus be defined: given a graph $\mathcal{G}$, the homophily ratio is measured as a ratio

of homophilic edges to all edges $r_h = \frac{1}{|\mathcal{E}|}\sum_{(v_i,v_j)\in\mathcal{E}} 1(y_{v_i} = y_{v_j})$. Heterophily is the opposite of homophily to describe the status of heterophilic edges that connect nodes belonging to different labels. Graphs with high homophily have $r_h \to 1$, while graphs with high heterophily exhibit low homophily with $r_h \to 0$.

Energy-based Out-of-Distribution Detection. In this paper, we explore out-of-distribution detection to identify bots using energy-based models (EBMs) [20]. An EBM defines an energy function that maps any input $\mathbf{x}$ to a single scalar value $E(\mathbf{x}) : \mathbb{R}^D \to \mathbb{R}$. This non-probabilistic scalar is called *energy*, which can be expressed as the negative of the log partition function:

$$E(\mathbf{x}) = -\log \sum_{y'} e^{-E(\mathbf{x},y')} \tag{1}$$

A collection of energy values can be converted to a probability density $p(\mathbf{x})$ through Gibbs distribution as:

$$p(y|\mathbf{x}) = \frac{e^{-E(\mathbf{x},y)}}{\sum_{y'} e^{-E(\mathbf{x},y')}} = \frac{e^{-E(\mathbf{x},y)}}{e^{-E(\mathbf{x})}} \tag{2}$$

When connecting the EBM with a discriminative neural classifier $f(\mathbf{x})$ that maps the input $\mathbf{x}$ to k-dimensional logits (k is the number of classes), we can define an energy for a given input $\mathbf{x}$ with a given class label y as $E(\mathbf{x}, y) = -f_y(\mathbf{x})$. In this respect, the energy function can be further derived as:

$$E(\mathbf{x}; f) = -\log \sum_{i}^{k} e^{f_i(\mathbf{x})} \tag{3}$$

Due to the intrinsic characteristics of the energy $E(\mathbf{x}; f)$ that the extracted energy values for in-distribution samples tend to be lower than those of out-of-distribution samples, $E(\mathbf{x}; f)$ can be effectively used to facilitate ODD [20,32] with a pre-specified threshold τ:

$$g(\mathbf{x}; \tau, f) = \begin{cases} 0, & \text{if } E(\mathbf{x}; f) \leq \tau, \\ 1, & \text{if } E(\mathbf{x}; f) > \tau \end{cases} \tag{4}$$

In this paper, we designate positive (1) as the prediction of out-of-distribution (bots), while negative (0) signifies the declaration for in-distribution (benign).

3 Proposed Model

In this section, we present our proposed model to leverage ODD for bot detection. We first analyze our motivation to better understand the impacts of graph structure on in-distribution learning before diving into technical details. Figure 1 depicts the overview of our complete model.

3.1 Impacts of Graph Structure on In-Distribution Learning

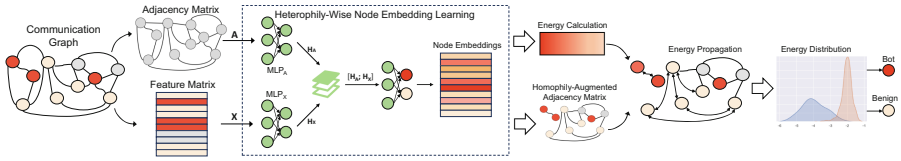


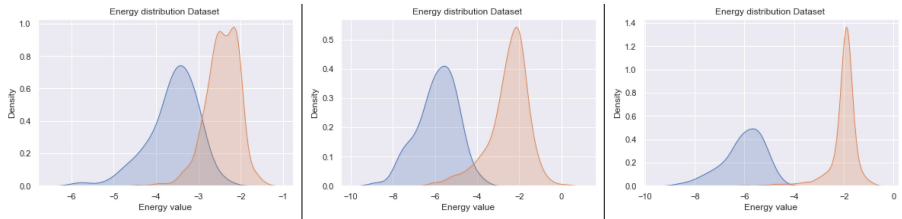
Fig. 1. Overview of our proposed model.

When extending energy-based ODD to graph data, the reliance on GNN backbones may result in these models inheriting heterophiliy, oversmoothing, and imbalance problems that are known to degrade GNN performance [33]. To demonstrate the impacts less-than-favorable graph conditions can have on graph-based ODD, we replicate GNNSafe [32], which is a recent state-of-the-art ODD model for graphs, with out-of-distribution exposure and belief propagation to work on imbalanced versions of the Cora dataset [35] by varying the number of in-distribution classes between six and three, and inversely adjusting the number of out-of-distribution classes between one and three. A more detailed description of the data configurations can be found in Table 1. Each adjustment to the dataset impacts which edges are homophilic, which edges are heterophilic, and the number of homophilic edges present in the in-distribution classes. This is significantly relevant due to the presence of the graph structure nullifies the independent and identically distribution (i.i.d.) assumption that regular machine learning-based ODD models (defined in Sect. 2) rely on, making the learning process sensitive to topological features of the graph [24].

Table 1. Data configurations on Cora dataset for graph structure analysis.

In-dist classes	In-dist Nodes	OOD Classes	OOD Nodes
6	2,140	1	568
5	1,722	2	986
4	904	3	1,804

We report the learned energy distributions in Fig. 2, with in-distribution nodes highlighted in blue and out-of-distribution nodes in orange. Notably, there is an observable trend of increased overlap between in-distribution and out-of-distribution energy values as the class imbalances between them grow, which implies a greater number of misclassifications. This phenomenon can be attributed to the fact that while shifting a single Cora class does not notably affect the overall number of heterophilic edges throughout the entire graph,



2,140 In-distribution Nodes | 1,722 In-distribution Nodes | 904 In-distribution Nodes

Fig. 2. Impacts of varying the number of in-distribution classes on energy distribution.

it does influence specific edges, potentially leading to the formation of heterophilic neighborhoods around individual nodes. Nodes with heterophilic neighborhoods tend to contribute to their over-smoothed embeddings and predicted logits; this results in less differentiable energy values that impact on subsequent energy propagation, ultimately leading to the observed overlap. Figure 2 provides insight into the graph-based ODD detection capability and its potential improvement by mitigating the impacts of heterophilic graph structure.

3.2 Heterophily-Wise Node Embedding Learning

To generalize GNNs to heterophilic graphs, the prevalent methods attend to refine neighborhood aggregation to adaptively exploit homophilic and heterophilic information [21,23,27], which improve the learning performance. However, these approaches not only complicate the models with extra computational cost, but also unstable generalization with high variance across diverse graphs [19]. In this paper, we take a different direction to formulate a learning-effective yet cost-efficient node embedding method to mitigate the impact of heterophily. Considering that heterophily impacts on node embeddings through neighborhood aggregation mechanism [25], we treat graph structure encoded as adjacency matrix $\mathbf{A}$ as structural features of nodes indicating inter-dependence with other nodes and their label distribution, instead of using it as low-pass filter to perform neighborhood aggregations. More specifically, we extract structural information from adjacency matrix $\mathbf{A}$ and semantic information from feature matrix $\mathbf{X}$, and then integrate them to learn the final node embeddings, which encode both node features and graph structure without excessive smoothing or compromise.

Formally, given a communication graph $\mathcal{G} = (\mathcal{V}, \mathcal{E}, \mathbf{X})$, we train two separate multi-layer perceptions (MLPs) on feature matrix $\mathbf{X}$ and adjacency matrix $\mathbf{A}$ to capture higher-level representations $\mathbf{H_X}$ and $\mathbf{H_A}$. Both representations are then concatenated as $[\mathbf{H_X}; \mathbf{H_A}]$ and fed to a fully-connected layer to fine-tune the presentation importance, deriving the final node embeddings $\mathbf{H}$:

$$\mathbf{H} = \sigma([\mathrm{MLP}_{\mathbf{X}}(\mathbf{X}), \mathrm{MLP}_{\mathbf{A}}(\mathbf{A})]\mathbf{W}), \quad \mathbf{H} \in \mathbb{R}^{n \times d} \tag{5}$$

The learned heterophily-wise node embeddings $\mathbf{H}$ will further proceed with two operations: (1) facilitating energy calculation; and (2) improving energy propagation. These two operations are detailed in Sect. 3.3 and Section 3.4, respectively.

3.3 Energy Calculation

To map the node embeddings $\mathbf{H}$ to the logit matrix $\mathbf{Z}$, we train another MLP to perform $\mathbf{Z} = \text{MLP}_{\mathbf{Z}}(\mathbf{H}), \mathbf{Z} \in \mathbb{R}^{n \times k}$, where $k = 2$ in our application scenario since we only have two classes to predict in bot detection. In this way, given an input node $\mathbf{x} \in \mathbf{X}$ from the graph $\mathcal{G}$ and its logit $\mathbf{z} \in \mathbf{Z}$, the energy function can be calculated as follows:

$$E(\mathbf{x}, \mathcal{G}_{\mathbf{x}}; f) = -\log \sum_{i}^{k} e^{\mathbf{z}_{[i]}} \tag{6}$$

Since $\mathbf{z}$ is derived from node embeddings $\mathbf{h} \in \mathbf{H}$ that encodes the graph structure, the energy function $E(\mathbf{x}, \mathcal{G}_{\mathbf{x}}; f)$ indicates the energy information of the node itself and other nodes that share potential dependencies within the graph structure.

3.4 Homophily-Augmented Energy Propagation

Inspired by label propagation, our model further elaborates a non-parametric energy propagation to aggregate energy values from neighboring nodes and takes advantage of unlabeled nodes across graph structure to enhance the generalization ability. A general edge propagation can be implemented as follows:

$$\mathbf{E}^{(l)} = \alpha \mathbf{E}^{(l-1)} + (1 - \alpha)\mathbf{D}^{-1}\mathbf{A}\mathbf{E}^{(l-1)} \tag{7}$$

where $E_{\mathbf{x}}^{(l)} = E^{(l)}(\mathbf{x}, \mathcal{G}_{\mathbf{x}}; f) \in \mathbf{E}^{(l)}$ at step l, and α is a balance parameter in the range of $[0, 1]$ to adjust the relative weight on the energy of the node itself and its neighboring nodes. Such a formulation implicitly relies on a high degree of graph homophily, where connected nodes tend to be sampled from similar distributions, such that the propagation will push energy towards the majority of neighbored nodes. Unfortunately, the given assumption does not hold under heterophilic conditions, leading to the failure of the propagation in Eq. (7). Based on this observation, we assert the following lemma:

Lemma 1. *Nodes with a heterophilic neighborhood, identified by a homophily ratio $r_h < 0.5$, will not trend towards the mean energy level of the distribution within which the node is contained.*

Proof of Lemma 1. For ODD to succeed, two distributions must be considered, one containing in-distribution data, $\mathcal{D}_{\text{in}}$, and another containing out-of-distribution data, $\mathcal{D}_{\text{out}}$. Each distribution has an expected value denoted as $\mathbb{E}_{\text{in}}$ and $\mathbb{E}_{\text{out}}$ respectively, representing the average energy value of in-distribution or out-of-distribution nodes, which can be specified as:

$$\mathbb{E}_{\text{type}} = \frac{\sum_{i}^{|\mathcal{V}_{\text{type}}|} E_i}{|\mathcal{V}_{\text{type}}|}, \quad \text{type} \in \{\text{in}, \text{out}\} \tag{8}$$

The expected energy values change during the training process. The effectiveness of ODD depends on a sufficiently large distance ψ between these two expected values, ensuring significant separation to minimize overlap in the tails of the learned distributions:

$$\psi < ||\mathbb{E}_{\text{in}} - \mathbb{E}_{\text{out}}|| \tag{9}$$

A node's neighborhood can be naturally divided into homophilic edges $\mathcal{E}_{\text{homo}}$ and heterophlic edges $\mathcal{E}_{\text{heter}}$. During a given training epoch l, $E_i^{(l)}$ is expected to approach a weighted average of the expected value of its neighbors. For an in-distribution node, this approximation can be expressed as:

$$E_i^{(l)} \approx \frac{|\mathcal{E}_{\text{homo},i}|\mathbb{E}_{\text{in}} + |\mathcal{E}_{\text{heter},i}|\mathbb{E}_{\text{out}}}{\sum_j \mathbf{A}_{ij}} \tag{10}$$

In homophilic neighborhoods, $|\mathcal{E}_{\text{homo},i}| > |\mathcal{E}_{\text{heter},i}|$ holds true and $E_i^{(l)}$ approaches the current $\mathbb{E}_{\text{in}}$, maintaining the desired relationship with ψ. In heterophilic neighborhoods where $|\mathcal{E}_{\text{homo},i}| < |\mathcal{E}_{\text{heter},i}|$, $E_i^{(l)}$ approaches the current $\mathbb{E}_{\text{out}}$. Adjustments to E_i impact the expected value of $\mathcal{D}_{in}$, potentially reducing the distance between expected values and violating the desired relationship with ψ.

To counter the negative impact of heterophily on energy propagation, we propose to empower our model to identify homophilic and heterophilic edges. This capability facilitates adjusting the energy propagation accordingly, ensuring a sufficiently large distance between $\mathbb{E}_{\text{in}}$ and $\mathbb{E}_{\text{out}}$.

Identification of Homophilic and Heterophilic Edges. We assess the likelihoods of edges being homophilic by examining the similarity of the connected node embeddings $\mathbf{H}$, as learned in Sect. 3.2. This is achieved through the application of two similarity measures:

- *Cosine Similarity.* Cosine similarity is used to evaluate the angle between two vectors, or in this case, node embeddings, which is calculated as:

$$\cos(\mathbf{h}_i, \mathbf{h}_j) = \frac{\mathbf{h}_i \cdot \mathbf{h}_j}{||\mathbf{h}_i||||\mathbf{h}_i||} \tag{11}$$

  To predict homophilic and heterophilic edges, we consider a cosine similarity > 0 as homophilic and ≤ 0 as heterophilic.
- *Euclidean Distance.* We first calculate Euclidean distance between the connecting node embeddings to obtain distances for all edges:

$$d_{i,j} = ||\mathbf{h}_i - \mathbf{h}_j||_2, \quad \forall(v_i, v_j) \in \mathcal{E} \tag{12}$$

  Subsequently, each distance $d_{i,j}$ is normalized as follows:

$$\text{norm}(d_{i,j}) = 1 + \frac{d_{\min} - d_{i,j}}{d_{\max} - d_{\min}} \tag{13}$$

  where $d_{\min}$ and $d_{\max}$ are the minimal and maximal distances among all edges. The normalized output falls within the range $[0, 1]$, where we consider $\text{norm}(d_{i,j}) \leq 0.5$ as heterophilic and $\text{norm}(d_{i,j}) > 0.5$ as homophilic.

Both similarity metrics can serve as either a binary or a fractional belief assessment of homophily. Here we use a binary assessment to improve energy propagation. Their effectiveness in edge identification will be evaluated in Sect. 4.2.

Use of Homophily-Augmented Adjacency Matrix. To incorporate the identification of heterophilic and homophilic edges into downstream tasks, we create an altered adjacency matrix $\hat{\mathbf{A}}$, which is defined as the edges believed to be homophilic in the graph, with belief determined by one of the two similarity measures described in Sect. 3.4. $\hat{\mathbf{A}}$ replaces the original adjacency matrix in the energy propagation, explicitly discouraging the propagation of energy across heterophilic edges. The energy propagation can be accordingly updated as:

$$\mathbf{E}^{(l)} = \alpha\mathbf{E}^{(l-1)} + (1-\alpha)\hat{\mathbf{D}}^{-1}\hat{\mathbf{A}}\mathbf{E}^{(l-1)} \tag{14}$$

The original normalization term $\mathbf{D}^{-1}$ is also replaced in our improved belief scheme with $\hat{\mathbf{D}}^{-1}$, which is the inverse diagonal degree matrix of $\hat{\mathbf{A}}$. This replacement is required to account for instances where a high-degree node contains a very small number of homophilic edges, ensuring energy being propagated as a proportion of homophilic edges and not restricted by a high degree. After L-step propagation, the energy values will be fed to the loss function for optimization, or used to make final out-of-distribution (bot) prediction using Eq. (4).

3.5 Loss Function

Our model utilizes both a supervised loss and a regularization loss for optimization. The supervised loss $\mathcal{L}_{\text{sup}}$ is a modified negative log-likelihood of the labeled training data designed to accommodate energy values:

$$\mathcal{L}_{\text{sup}} = \mathbb{E}_{(\mathbf{x},\mathcal{G}_{\mathbf{x}},y)\sim\mathcal{D}_{\text{in}}}(-\log p(y|\mathbf{x},\mathcal{G}_x)) = \sum_{i\in\mathcal{V}_{\text{in}}}(-\mathbf{z}_{i,[y_i]} + \log\sum_{j}^{k} e^{\mathbf{z}_{i,[j]}}) \tag{15}$$

where $\mathcal{D}_{\text{in}}$ is the in-distribution data where the labeled nodes (i.e., benign nodes) of the training data are sampled. Inspired by the bounding constraints for absolute energy [20], we further add a regularization loss $\mathcal{L}_{\text{reg}}$ to constrain the energy gap, bounding the energy for in-distribution data [32], and also expose a fraction of out-of-distribution data (i.e., bot nodes) to training that gives the model the opportunity to learn how to discriminate against known out-of-distribution nodes and assists in fine-tuning the model:

$$\begin{aligned}\mathcal{L}_{\text{reg}} = &\frac{1}{|\mathcal{V}_{\text{in}}|}\sum_{i\in\mathcal{V}_{\text{in}}}(\text{ReLU}(E(\mathbf{x}_i,\mathcal{G}_{\mathbf{x}_i};f) - t_{\text{in}}))^2 \\ &+ \frac{1}{|\mathcal{V}_{\text{out}}|}\sum_{j\in\mathcal{V}_{\text{out}}}(\text{ReLU}(t_{\text{out}} - E(\mathbf{x}_j,\mathcal{G}_{\mathbf{x}_j};f)))^2\end{aligned} \tag{16}$$

$\mathcal{L}_{\text{reg}}$ will push energy values in $[t_{\text{in}}, t_{\text{out}}]$ to be lower for in-distribution and higher for out-od-distribution nodes. The final loss function for optimizing our model is $\mathcal{L} = \mathcal{L}_{\text{sup}} + \lambda\mathcal{L}_{\text{reg}}$, where λ is a balance parameter.

Table 2. Statistics of bot detection datasets.

File	1	2	3	4	5	6	7	8	9	10
#Bots	19	35	32	38	41	39	49	57	57	56
#Benigns	1,534	1,563	1,604	1,623	1,638	1,654	1,688	2,304	2,043	2,048
Imb. Ratio	0.012	0.022	0.020	0.023	0.024	0.023	0.028	0.027	0.027	0.027
#Edges	8,264	9,296	9,492	9,644	9,716	9,812	10,016	11,816	11,880	11,928
Homo. Ratio	0.300	0.264	0.651	0.270	0.269	0.660	0.272	0.668	0.668	0.668

4 Experimental Results and Analysis

4.1 Experimental Setup

Datasets. We perform our experiments using the TON IoT dataset which is an assortment of recorded Industry 4.0/IoT and IIoT network traffic [1,26]. Each network is recorded in an individual capture file that is translated into an undirected communication graph, where protocol used, packet size, and amount of data transmitted and received are used as node features. To represent unique graph constructs, we select capture files with high heterophily ($r_h < 0.4$), high homophily ($r_h > 0.6$), and extreme class imbalance ($r_i < 0.03$) respectively. Table 2 details the data statistics. We separate nodes in each graph into an 80-10-10 split for training, testing, and validation.

Baselines. We select 11 different models as our baselines, which include three traditional GNNs (GCN [13], GAT [31], and GraphSAGE [9])), three graph-based oversampling models (SMOTE [6], GraphSMOTE [38], and HOVER [2]), and five ODD models (MSP [10], MSP with outlier exposure (OE) [11], and and three variants of GNNSafe [32]).

Implementation Details. We implement our model as described in Sect. 3 using the following hyperpameter settings: hidden size in $\text{MLP}_{\mathbf{A}} = \{128, 256\}$; hidden size in $\text{MLP}_{\mathbf{X}} = \{24, 48, 64, 128\}$; output size is $\{1x, 2x, 4x, 8x, 64x\}$ per number of classes x. In addition to these adjustable parameters, we use the following fixed configurations: layers in $\text{MLP}_{\mathbf{A}} = \{2\}$ and $\text{MLP}_{\mathbf{X}} = \{2\}$; propagation layer number $L = 2$; balance parameter $\alpha = 0.5$; we select the values $t_{in} = -5$ and $t_{out} = -1$ used in the regularization term of our loss function.

4.2 Effectiveness of Edge Prediction

We initiate our experimentation by assessing our model's ability to differentiate between homophilic and heterophilic edges. We perform this experiment by training our model on capture files 1, 3, 5, and 6 from TON IoT, due to their inclusion of both high and low levels of heterophily. During every training epoch, we generate and store improved node embeddings, and feed them into our edge

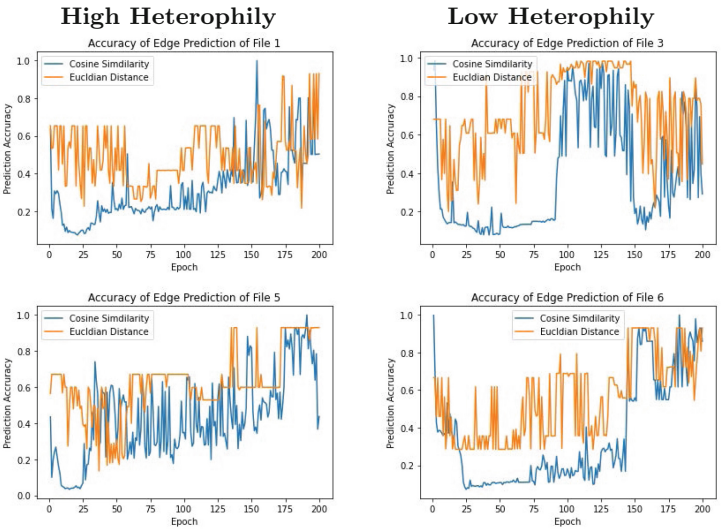


Fig. 3. Effectiveness of edge identification by comparing cosine similarity and Euclidean distance across different training epochs.

prediction module. The module assesses if an edge is homophilic or heterophilic according to cosine similarities and normalized Euclidean distances. We compare the edge prediction outputs with the true edge type as determined by node labels, and calculate the accuracy of both predictions at every epoch. Our model learns only from a loss function quantifying the quality of node classifications.

The edge prediction performance is recorded in Fig. 3. From this figure, we can see that our model's edge identification approach is capable of making highly accurate assessments of edge predictions, exceeding 80% accuracy during the training process. Both cosine similarity and Euclidean distance similarity approaches display generally stable behaviour. However, Euclidean distance outperforms cosine similarity almost universally. As such, we conclude that we can identify homophilic edges in heavily heterophilic graphs using heterophily-wise embeddings learned from node features and graph structure.

4.3 Comparison with Baselines for Bot Detection

We compare our model against multiple baselines, including basic GNN models: GCN [13], GAT [31], and GraphSage [9], oversampling approaches: SMOTE (applied on node embeddings learned by GCN) [6], GraphSMOTE [38], and HOVER [2], and ODD methods: MSP [10], MSP with outlier exposure [11], and three variants of GNNSafe [32]. We report the performance of each model and our proposed model as measured by F-1 (%) in Table 3.

Our model shows substantial performance over existing baselines with limited exceptions. Traditional GNN models fall inline with the expected behavior under heterophilic and extremely class-imbalanced conditions, which cannot gen-

Table 3. Comparison of our model with different baselines in terms of F-1 (%).

Model	File 1	File 2	File 3	File 4	File 5	File 6	File 7	File 8	File 9	File 10
GCN	00.75	06.09	03.95	04.85	08.51	04.62	01.46	07.37	09.95	01.19
GAT	00.00	01.09	03.75	04.75	04.68	08.51	08.84	07.37	01.01	08.82
GraphSage	03.13	01.54	05.41	08.70	02.22	05.41	02.58	01.69	01.80	07.92
SMOTE	02.28	02.13	01.23	01.57	07.75	01.10	08.04	02.89	01.24	02.81
GraphSMOTE	66.74	55.10	49.50	64.21	54.28	58.10	57.00	56.00	56.00	49.33
HOVER	60.45	33.33	**80.00**	74.16	66.67	66.67	60.00	69.21	71.78	66.67
MSP	07.27	06.00	06.67	06.15	05.46	05.71	04.00	04.00	05.00	04.29
OE	06.67	05.46	06.67	05.72	05.46	05.33	03.81	03.81	04.44	04.44
GNN-ODD w/o Reg	75.00	66.67	60.00	61.54	57.14	50.00	53.33	35.29	31.58	42.11
GNNSafe	85.71	66.67	60.00	66.67	61.54	54.55	57.14	35.30	35.29	42.11
GNNSafe++	02.14	01.60	01.31	01.47	01.53	01.39	01.73	01.70	02.90	01.65
Our model	**88.89**	**72.22**	71.43	**80.00**	**88.89**	**85.71**	**75.00**	**83.33**	**73.68**	**70.00**

erate enough discriminative information to separate bots in the learned embedding space. Oversampling approaches provide a good improvement over the traditional GNN models but all have their flaws: SMOTE does not synthesize structural information, resulting in oversampled nodes being disjoint from the rest of the graph with very limited contribution to the learning process; GraphSMOTE addresses the structural limitations of SMOTE by creating synthetic edges, which, however, does not consider the impact of heterophily present in the graph, limiting its classification performance; HOVER periodically shows very good performance but performance is inconsistent, and it requires multiple training cycles, modifies the semantics of the graph structure, and is sensitive to hyperparameter selection. ODD methods provide far more completion for our model. An interesting point is the negative impact that belief propagation has when graph heterophily is high. A comparison of GNNSafe and GNNSafe++ quantifies this observation, where in all data cases, the addition of belief propagation to GNNSafe hurts the detection performance. The cause of this is the nature of bot graphs. The out-of-distribution nodes, or bots, have a high local heterophily ratio regardless of the heterophily ratio of the entire graph. Naively, applying belief propagation to heterophilic graphs will pull bot samples toward the mean of non-bot samples. Bot nodes are then masked by benign nodes after energy values are calculated and propagated.

Our model outperforms all models in the tradition and ODD categories, with only one exception that our model performs second to HOVER. A direct comparison of our model to the various flavors of GNNSafe demonstrates the benefit of our heterophily-wise node embedding learning and homophily-augmented energy propagation scheme. The addition of edge prediction to energy propagation reduces the influence heterophily has on ODD.

4.4 Case Study: ODD for Bot Detection

We dig deeper into the performance of our model and the performance of a selection of baselines with a case study. Specifically, in this section, we look at

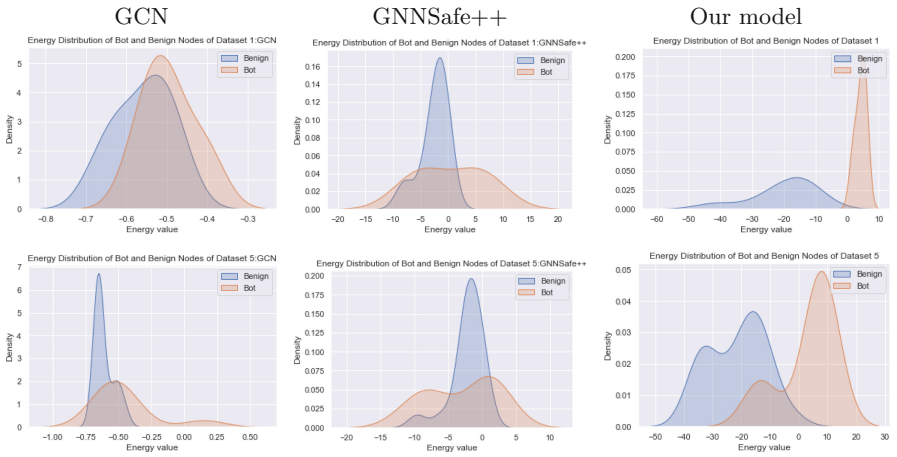


Fig. 4. Energy distributions learned by GCN, GNNSafe++, and our model. The upper row contains the distributions learned on capture file 3, while the lower row contains the distributions learned on capture file 5.

the learned energy distributions from GCN, GNNSafe++, and our model. We select two capture files for use in this case study: File 3 has low heterophily, File 5 has high heterophily, and both files has extremely low imbalance ratio.

GCN struggles to learn separated distributions for bot and benign samples. The impacts of heterophily can be seen by comparing the GCN results of each capture file. Capture file three, the top left of Fig. 4, contains mostly homophilic edges. This allows the GCN to constrain bot samples to energy levels less than the most extreme values of benign samples. In contrast, capture file five, the bottom left of Fig. 4, contains mostly heterophilic edge which causes the benign samples to attract bot samples. GNNSafe++ fails to separate or even constrain bot samples. GNNSafe++ does utilize OE and regularization during training which causes both learned distributions to be centered at approximately the same group about the same energy value. However, regularization alone is not capable of restraining the bot distribution. The bot distribution overlaps the entire learned benign distribution, creating a situation when no useful division between distributions can be found.

Our model is capable of separating the densest portions of the learned distributions. Moreover, our model impacts the extremes of both distributions. The energy values of lower tail of benign samples is pushed to extremely low energy values, extending below the -20 and -30 values that competing models reach on files three and five, respectively. The upper tail of the bot distribution displays similar behavior extending into positive energy values in both instances. This behavior allows our model to clearly separate many bots from benign samples.

4.5 Ablation Study

We conclude our experimentation with an ablation study to verify that all aspects of our model contribute in a manner that are both beneficial and nec-

Table 4. Ablation study to evaluate different model components in terms of F-1 (%).

Model	File 1	File 2	File 3	File 4	File 5	File 6	File 7	File 8	File 9	File 10
No Independent Embedding	02.14	01.15	01.24	14.24	01.60	01.38	02.88	01.58	01.59	01.58
No Propagation	75.00	**72.73**	60.00	66.67	61.53	61.53	57.14	15.38	22.48	15.38
No Edge Prediction	40.00	25.00	66.67	40.00	77.78	79.99	57.14	73.68	43.90	15.85
Our model	**88.89**	72.22	**71.43**	**80.00**	**88.89**	**85.71**	**75.00**	**83.33**	**73.68**	**70.00**

essary to the learning and classification processes. We evaluate the necessity of three elements to our model: (1) **Independent node and structural embedding**: we remove the MLPs from our model and replace them with a GCN. We refer to this experiment as "No independent embedding" in Table 4; (2) **Energy propagation**: we remove energy propagation entirely, which is denoted as "No Propagation" in Table 4, where we make a determination if a data sample is in or out of distribution immediately following the energy calculation; (3) **Edge prediction**: we repeat the experiment without using the homophily-augmented $\hat{\mathbf{A}}$ adjacency matrix, but use the original adjacency matrix $\mathbf{A}$, which is denoted as "No Edge Prediction" in Table 4.

The experiment results in Table 4 allows us to draw several conclusions. First, the fact that our model outperforms the reduced models on almost all capture files shows that all elements of our model contribute to the overall performance. File 2 in the "No Propagation" case is the lone instance where our complete model does not exceed the performance of a reduced model. As the difference between the complete and reduced model is negligible, we do not consider this undermines the utility of energy propagation. Moreover, as the "No Edge Prediction" case shows dramatically reduced performance, we assess that the anomaly speaks more to the benefit of independent representation embedding than it implies frivolous use of energy propagation. Our model is outperformed by "No Propagation" implying that the edge classification scheme is lacking in this instance. We consider this a motivation to explore improved edge prediction schemes in the future. A column-wise comparison of results highlights the individual contribution of each element of our model. The addition of edge prediction making arguable the biggest impact, improving performance by up to 48% on extremely heterophilic graphs and a more modest by still respectable improvement of 10% to 30% on the homophilic graphs (Files 6, 8, 9, and 10).

5 Conclusion

In this paper, we generalize the utility of ODD to the bot detection problem on graphs. To address over-smoothing and graph heterophily introduced by class imbalance, we propose a new energy-based ODD model to better deal with graph data. Our model formulates a simple yet effective node embedding method to encode node features and graph structure without the need for neighborhood aggregation. These node embeddings not only enhance energy calculation, but also enable homophily-augmented energy propagation, which

significantly separate energy distributions between bots and benign samples, and thus improve the detection performance. Our experiments validate our proposed model and demonstrate that independent node embedding can overcome extreme heterophily, and the addition of edge prediction to existing energy propagation approaches advances the existing state-of-the-art in graph ODD and bot detection.

Acknowledgments. This work is partially supported by the NSF under grant CNS-2245968. The authors would also like to thank the reviewers for their valuable feedback.

References

1. Alsaedi, A., Moustafa, N., Tari, Z., Mahmood, A., Anwar, A.: Ton_iot telemetry dataset: a new generation dataset of iot and iiot for data-driven intrusion detection systems. IEEE Access **8**, 165130–165150 (2020)
2. Ashmore, B., Chen, L.: Hover: Homophilic oversampling via edge removal for class-imbalanced bot detection on graphs. In: CIKM, pp. 3728–3732 (2023)
3. Bitterwolf, J., Meinke, A., Augustin, M., Hein, M.: Breaking down out-of-distribution detection. In: ICML, pp. 2041–2074 (2022)
4. Cen, J., Yun, P., Cai, J., Wang, M.Y., Liu, M.: Deep metric learning for open world semantic segmentation. In: ICCV, pp. 15333–15342 (2021)
5. Chatterjee, M., Namin, A.S., Datta, P.: Evidence fusion for malicious bot detection in iot. In: IEEE International Conference on Big Data, pp. 4545–4548 (2018)
6. Chawla, N.V., Bowyer, K.W., Hall, L.O., Kegelmeyer, W.P.: Smote: synthetic minority over-sampling technique. JAIR **16**, 321–357 (2002)
7. Chen, L., Li, X., Wu, D.: Enhancing robustness of graph convolutional networks via dropping graph connections. In: ECML PKDD, pp. 412–428 (2021)
8. Cui, P., Wang, J.: Out-of-distribution (ood) detection based on deep learning: a review. Electronics **11**(21), 3500 (2022)
9. Hamilton, W., Ying, Z., Leskovec, J.: Inductive representation learning on large graphs. In: Advances in Neural Information Processing Systems, vol. 30 (2017)
10. Hendrycks, D., Gimpel, K.: A baseline for detecting misclassified and out-of-distribution examples in neural networks. In: ICLR (2017)
11. Hendrycks, D., Mazeika, M., Dietterich, T.: Deep anomaly detection with outlier exposure. In: ICLR (2019)
12. Hsu, Y.C., Shen, Y., Jin, H., Kira, Z.: Generalized odin: detecting out-of-distribution image without learning from out-of-distribution data. In: CVPR, pp. 10951–10960 (2020)
13. Kipf, T.N., Welling, M.: Semi-supervised classification with graph convolutional networks. In: ICLR (2017)
14. Li, Q., Chen, L., Cai, Y., Wu, D.: Hierarchical graph neural network for patient treatment preference prediction with external knowledge. In: PAKDD, pp. 204–215 (2023)
15. Li, Q., Li, X., Chen, L., Wu, D.: Distilling knowledge on text graph for social media attribute inference. In: SIGIR, pp. 2024–2028 (2022)
16. Li, Z., Wu, Q., Nie, F., Yan, J.: Graphde: a generative framework for debiased learning and out-of-distribution detection on graphs. Adv. Neural. Inf. Process. Syst. **35**, 30277–30290 (2022)

17. Li, Z., Liu, F., Yang, W., Peng, S., Zhou, J.: A survey of convolutional neural networks: analysis, applications, and prospects. TNNLS **33**(12), 6999–7019 (2021)
18. Liang, S., Li, Y., Srikant, R.: Enhancing the reliability of out-of-distribution image detection in neural networks. In: ICLR (2018)
19. Lim, D., Hohne, F., Li, X.: Large scale learning on non-homophilous graphs: new benchmarks and strong simple methods. NeurIPS **34**, 20887–20902 (2021)
20. Liu, W., Wang, X., Owens, J., Li, Y.: Energy-based out-of-distribution detection. Adv. Neural. Inf. Process. Syst. **33**, 21464–21475 (2020)
21. Liu, Y., Zheng, Y., Zhang, D., Lee, V.C., Pan, S.: Beyond smoothing: Unsupervised graph representation learning with edge heterophily discriminating. In: AAAI, pp. 4516–4524 (2023)
22. Lo, W.W., Kulatilleke, G., Sarhan, M., Layeghy, S., Portmann, M.: Xg-bot: an explainable deep graph neural network for botnet detection and forensics. Internet Things **22**, 100747 (2023)
23. Luan, S., Hua, C., Lu, Q., Zhu, J., Zhao, M.: Revisiting heterophily for graph neural networks. NeurIPS **35**, 1362–1375 (2022)
24. Ma, J., Deng, J., Mei, Q.: Subgroup generalization and fairness of graph neural networks. NeurIPS **34**, 1048–1061 (2021)
25. Ma, Y., Liu, X., Shah, N., Tang, J.: Is homophily a necessity for graph neural networks? In: ICLR (2022)
26. Moustafa, N.: A new distributed architecture for evaluating ai-based security systems at the edge: Network ton_iot datasets. Sustainable Cities and Society (2021)
27. Pei, H., Wei, B., Chang, K.C.C., Lei, Y., Yang, B.: Geom-gcn: geometric graph convolutional networks. In: ICLR (2020)
28. Rahal, B.M., Santos, A., Nogueira, M.: A distributed architecture for ddos prediction and bot detection. IEEE Access **8**, 159756–159772 (2020)
29. Stadler, M., Charpentier, B.: Graph posterior network: Bayesian predictive uncertainty for node classification. NeurIPS **34**, 18033–18048 (2021)
30. Sun, Y., Guo, C., Li, Y.: React: Out-of-distribution detection with rectified activations. Adv. Neural. Inf. Process. Syst. **34**, 144–157 (2021)
31. Veličković, P., Cucurull, G., Casanova, A., Romero, A., Lio, P., Bengio, Y.: Graph attention networks. In: ICLR (2018)
32. Wu, Q., Chen, Y., Yang, C., Yan, J.: Energy-based out-of-distribution detection for graph neural networks. In: ICLR (2023)
33. Yan, Y., Hashemi, M., Swersky, K., Yang, Y., Koutra, D.: Two sides of the same coin: Heterophily and oversmoothing in graph convolutional neural networks. In: ICDM, pp. 1287–1292 (2022)
34. Yang, N., Zeng, K., Wu, Q., Jia, X., Yan, J.: Learning substructure invariance for out-of-distribution molecular representations. NeurIPS **35**, 12964–12978 (2022)
35. Yang, Z., Cohen, W., Salakhudinov, R.: Revisiting semi-supervised learning with graph embeddings. In: ICML, pp. 40–48 (2016)
36. Zhang, B., Li, J., Chen, C., Lee, K., Lee, I.: A practical botnet traffic detection system using gnn. In: CSS, pp. 66–78 (2022)
37. Zhang, J., Perdisci, R., Lee, W., Luo, X., Sarfraz, U.: Building a scalable system for stealthy p2p-botnet detection. TIFS **9**(1), 27–38 (2013)
38. Zhao, T., Zhang, X., Wang, S.: Graphsmote: imbalanced node classification on graphs with graph neural networks. In: WSDM, pp. 833–841 (2021)
39. Zhou, J., Xu, Z., Rush, A.M., Yu, M.: Automating botnet detection with graph neural networks. In: MLSys (2020)
40. Zhu, J., et al.: Graph neural networks with heterophily. In: AAAI, pp. 11168–11176 (2021)