RICC: Robust Collective Classification of Sybil Accounts

Dongwon Shin*
School of Computing, KAIST
dongwon.shin@kaist.ac.kr

Suyoung Lee*
School of Computing, KAIST
suyoung.lee@kaist.ac.kr

Sooel Son
School of Computing, KAIST
sl.son@kaist.ac.kr

ABSTRACT

A Sybil attack is a critical threat that undermines the trust and integrity of web services by creating and exploiting a large number of fake (i.e., Sybil) accounts. To mitigate this threat, previous studies have proposed leveraging collective classification to detect Sybil accounts. Recently, researchers have demonstrated that state-of-the-art adversarial attacks are able to bypass existing collective classification methods, posing a new security threat. To this end, we propose RICC, the first robust collective classification framework, designed to identify adversarial Sybil accounts created by adversarial attacks. RICC leverages the novel observation that these adversarial attacks are highly tailored to a target collective classification model to optimize the attack budget. Owing to this adversarial strategy, the classification results for adversarial Sybil accounts often significantly change when deploying a new training set different from the original training set used for assigning prior reputation scores to user accounts. Leveraging this observation, RICC achieves robustness in collective classification by stabilizing classification results across different training sets randomly sampled in each round. RICC achieves false negative rates of 0.01, 0.11, 0.00, and 0.01 in detecting adversarial Sybil accounts for the Enron, Facebook, Twitter_S, and Twitter_L datasets, respectively. It also attains respective AUCs of 0.99, 1.00, 0.89, and 0.74 for these datasets, achieving high performance on the original task of detecting Sybil accounts. RICC significantly outperforms all existing Sybil detection methods, demonstrating superior robustness and efficacy in the collective classification of Sybil accounts.

CCS CONCEPTS

• **Security and privacy**; • **Computing methodologies** → **Machine learning**;

KEYWORDS

Sybil attack, robustness, adversarial attack, collective classification

ACM Reference Format:
Dongwon Shin, Suyoung Lee, and Sooel Son. 2023. RICC: Robust Collective Classification of Sybil Accounts. In *Proceedings of the ACM Web Conference 2023 (WWW '23), April 30–May 04, 2023, Austin, TX, USA.* ACM, New York, NY, USA, 11 pages. https://doi.org/10.1145/3543507.3583475

*Both authors contributed equally to the paper.

WWW '23, April 30–May 04, 2023, Austin, TX, USA

ACM ISBN 978-1-4503-9416-1/23/04…$15.00
https://doi.org/10.1145/3543507.3583475

1 INTRODUCTION

Recent websites provide security- and privacy-sensitive services spanning banking, web payment, medical, social network, and governmental services. Owing to the sensitivity and importance of such web services, they seek to establish a strong identity for each user account. Unfortunately, popular web services have suffered from Sybil attacks that abuse user accounts [10, 32]. A Sybil attack refers to an attack that creates a large number of fake accounts, thus abusing a target service by exploiting generated fake accounts, each of which is called a *Sybil* account.

The adversary abusing Sybil accounts imposes a critical threat to establishing trust and integrity in web services. Amazon and Alibaba have suffered from fake reviews on products and services by Sybil accounts, which have greatly undermined trust in and the reputation of their services [21, 23].

Previous studies [2, 4, 5, 8, 11, 13, 14, 19, 20, 26–28, 32–34, 36, 38] have suggested diverse ways of detecting Sybil accounts. One prevalent trend in these studies is to model user accounts and their relationships into a graph and then conduct semi-supervised node classification on this graph. Specifically, Wang *et al.* [28] have proposed a representative approach of leveraging collective classification. Given a graph and a subset of the labeled nodes on this graph, collective classification involves classifying the remaining unlabeled nodes into benign or Sybil nodes. They demonstrated the superior performance of their approach in detecting Sybil accounts. Due to its high performance and practicality, collective classification has become a prevalent trend in identifying Sybil accounts in many existing web services, including Twitter and Sina Weibo [11, 26–28].

Unfortunately, recent studies [25, 31] have demonstrated that state-of-the-art collective classification methods are vulnerable to two proposed adversarial attacks; these attacks cause state-of-the-art classifiers to misclassify target Sybil accounts as benign ones. However, no prior studies have proposed mitigation of these adversarial attacks through robust collective classification.

Contributions. We propose the Robust Inhibitor of differences in Collective Classification (RICC), the first collective classification framework designed for the robust identification of Sybil accounts. A common approach in collective classification is to assign a prior reputation score to each node in a given graph, of which the nodes and edges represent user accounts and their relationships, respectively. Given the prior reputation scores for a limited number of benign and Sybil nodes (which constitute the training set), previous classification methods [13, 26, 28] iteratively propagate the prior reputation scores to their neighbor nodes and compute posterior reputation scores to classify the remaining nodes as either benign or Sybil. They leverage the assumption that Sybil nodes tend to group themselves according to the adversary's strategy of creating low-cost edges among Sybil nodes.

Due to this adversarial strategy, we discover that when target Sybil nodes are misclassified due to adversarial attacks [25, 31], the

posterior reputation scores of these target Sybil nodes are highly dependent on the initial training set of prior reputation scores. Based on this observation, we propose a robust collective classification method that renders the posterior reputation scores after classification stable across different training sets of prior reputation scores.

Specifically, RICC conducts random sampling-based collective classification; it iteratively conducts collective classification over a given number of rounds to compute an initial training set optimized to conduct robust classification. For each round, RICC randomly samples a subset of the prior reputation scores and conducts collective classification. Therefore, RICC computes two sets of posterior reputation scores: one computed from the training set of prior scores and the other randomly sampled from this training set. Over multiple rounds, RICC performs optimization in the direction of minimizing the differences between the two sets chosen in each round by adjusting the original prior scores. These changed prior scores then become the original prior scores for the next round. Thus, RICC is designed to find the optimal training set of prior scores that contributes to a classifier performing robust collective classification, thereby providing stable posterior reputation scores for adversarial Sybil nodes. RICC not only excels at identifying adversarial nodes but also improves Sybil detection performance without additional labeling effort or data augmentation. These advantages make RICC backward-compatible with existing Sybil detection systems using collective classification.

We evaluate the efficacy of RICC in identifying adversarial Sybil nodes that bypass state-of-the-art collective classification methods. For the task of detecting adversarial Sybil nodes, RICC achieves false negative rates of 0.01, 0.11, 0.00, and 0.01 for the Enron, Facebook, Twitter_S, and Twitter_L datasets, respectively, significantly outperforming the performance of existing Sybil detection methods [27, 28]. RICC also achieves areas under the curve of 0.99, 1.00, 0.89, and 0.74 for the Enron, Facebook, Twitter_S, and Twitter_L datasets, respectively, surpassing the performance of existing Sybil detection methods.

In summary, we present the first robust collective classification method that identifies Sybil nodes even given the presence of a powerful adversary conducting adversarial attacks. The evaluation results demonstrate the superior performance of RICC in identifying adversarial Sybil nodes, thus advancing state-of-the-art robust collective classification. To facilitate follow-on research, we release our code at https://github.com/WSP-LAB/RICC.

2 BACKGROUND

A Sybil attack has been a notorious threat that undermines trust in web services. The adversary creates a large number of fake (i.e., Sybil) accounts and abuses these Sybil accounts by leaving spurious comments or conducting fraudulent transactions. Previous research has proposed various methods of identifying Sybil accounts to prevent the adversary from abusing web services.

Semi-supervised node classification. One notable and prevalent method of identifying Sybil accounts is to leverage semi-supervised node classification. Consider a graph $G = (V, E)$, where V is a set of nodes, E is a set of edges, and each node belongs to one of the classes in a label set $L = \{positive, negative\}$. Given a set of labeled nodes in G, a semi-supervised node classification problem aims to predict the labels of the remaining nodes. We refer to the set of labeled nodes as the training set T.

Algorithm 1: Collective Classification Algorithm.

```
Input  : A graph (G).
         A training set (T).
         A matrix of edge weights (W).
Output : A set of labeled nodes (𝕃).
1 function RunCollectiveClassification(G, T, W)
2     q ← AssignPriorScore(G, T)
3     Initialize p_0 ← q and t ← 1.
4     while t ≤ T_max do
5         p_t ← ComputePosteriorScore(G, q, W, p_{t−1})
6         t ← t + 1
7     𝕃 ← PredictNodeLabel(G, p_t)
8     return 𝕃
```

Collective classification. For this semi-supervised node classification problem, the dominant trend in previous research has been to conduct collective classification [2, 5, 6, 11, 13, 14, 18, 22, 24, 26–28, 33]. Algorithm 1 summarizes a common algorithm of this collective classification. Given a graph G, a training set T, and a matrix of edge weights $\mathbf{W}$, the `AssignPriorScore` function initializes the prior reputation score q_u for each node $u \in V$ (Line 2). It then iteratively propagates the prior reputation score of each node to its neighbor nodes and updates the posterior reputation score p_u via invoking the `ComputePosteriorScore` function in Line 5. Based on the final posterior reputation scores, the `PredictNodeLabel` function labels each node in G (Line 7).

Note that collective classification methods may use a different `AssignPriorScore` function, a different `ComputePosteriorScore` function, or a different weight matrix $\mathbf{W}$. For instance, loopy belief propagation (LBP)-based methods [6, 11, 13, 15, 22, 24, 28] leverage the following `AssignPriorScore` function:

$$q_u = \begin{cases} \theta & u \in L_P \\ -\theta & u \in L_N \\ 0 & otherwise \end{cases}, \tag{1}$$

where L_P is a set of known positive (i.e., Sybil) nodes, L_N is a set of known negative (i.e., benign) nodes, and θ ($\theta > 0$) is a positive prior reputation score for a Sybil node.

SybilSCAR [28] utilizes the following `ComputePosteriorScore` function for iteratively propagating the posterior reputation scores:

$$\mathbf{p}_t = \mathbf{q} + 2\mathbf{W}\mathbf{p}_{t-1}, \tag{2}$$

where $\mathbf{p}_t$ refers to a set of posterior reputation scores at each propagation step t, and $\mathbf{q}$ indicates a set of prior reputation scores.

Adversarial attacks against collective classification. Previous studies have proposed adversarial attacks against collective classification [25, 31]. In this attack scenario, an attacker attempts to induce a target classifier to misclassify target nodes by manipulating the underlying graph structure. To achieve this goal, Wang *et al.* [25] framed the problem of manipulating the graph structure as an optimization problem. Specifically, they designed their objective function to change the classifier's decision on the target nodes while achieving a minimum manipulation cost. Unfortunately, no existing collective classification method has successfully mitigated these adversarial threats. Therefore, in this paper, we propose RICC, the first collective classification method robust to state-of-the-art adversarial attacks [25, 31].

3 THREAT MODEL

We describe an attack scenario in which an adversary modifies the graph structure to bypass the detection of target Sybil nodes. We then introduce prior knowledge of an adversary regarding the detection method.

3.1 Attack Scenario

Analogous to prior studies [25, 31], we assume two parties in the attack scenario: a victim classifier and an adversary. Given a graph $G = (V, E)$, where each $u \in V$ corresponds to either a benign (i.e., negative) or Sybil (i.e., positive) node, the adversary selects a set of target Sybil nodes and aims to deceive the victim classifier such that the classifier incorrectly classifies these target nodes as benign nodes. For each target node that causes misclassification, we refer to it as an *adversarial Sybil node* or *adversarial node*, interchangeably, in the paper. The adversary may add new edges or delete existing edges [25]; otherwise, the adversary may create new positive nodes and connect them to the existing nodes in the graph [31].

For example, consider a graph having fraudulent social network accounts. The nodes and edges of the graph correspond to user accounts and follower-followee relationships, respectively. In this case, the adversary is able to select target accounts and make those accounts follow benign accounts to bypass detection. Given this manipulated graph G', the victim classifier seeks to identify all Sybil nodes, including the target nodes.

Note that the attacker is able to manipulate the structure of the underlying graph. These modifications require a non-trivial cost, such as creating new accounts or following other accounts. Therefore, the objective of the adversary is to evade detection at a minimum cost. The adversary is able to conduct stronger attacks if she can afford a larger attack budget. We control the adversary's attack budget with the number of manipulated nodes and edges.

3.2 Adversarial Knowledge

To evaluate the robustness of RICC to the fullest extent, we assume a strong adversary who knows that a victim classifier performs collective classification to detect Sybil nodes. We further assume that the adversary knows the details below about the victim classifier. Our goal is to mitigate threats that this strong adversary poses.

Parameters (θ and W). The adversary has prior knowledge regarding parameters used for training the victim classifier. Specifically, the adversary knows the exact parameters (i.e., the prior reputation score θ and the weight matrix $\mathbf{W}$).

Training Set (T). Recall from §2 that a victim classifier requires a set of labeled nodes for training. The attack becomes more powerful when the adversary knows which nodes are used for training.

Graph (V and E). Due to a lack of information, the adversary may not know the complete graph structure (i.e., V and E). For example, for private accounts, the adversary may not know the followers and is thus unable to construct the complete graph. We assume that our strong adversary knows the complete graph structure.

4 DESIGN

We present the first collective classification method that aims to identify target Sybil nodes manipulated by adversarial attacks [25, 31] while accurately spotting other Sybil nodes. We note that adversarial attacks commonly add an edge between target nodes and other nodes with considerably different features and labels. Recent studies have reported that this strategy is the most effective way of achieving the adversarial goal [16, 29, 31].

Based on this observation, we discover that the benign nodes in the training set T correspond to the nodes that have notably different features compared to the target nodes, and successful adversarial attacks [25, 31] necessarily connect their target nodes to those benign nodes. Recall from §2 that those nodes labeled benign are assigned with the highest negative prior score $-\theta$; consequently, those benign nodes are more likely to have high negative posterior scores, effectively propagating their negative reputation scores to the connected nodes. Given the adversary's imperative of saving on cost by minimizing the number of edges to manipulate, the adversary would certainly add edges between them.

We propose a novel random sampling-based collective classification method that exploits this observation. The adversarial attacks above are highly tailored to the training set T. When we run the same collective classification algorithm using a different training set sampled from T, the prior scores of non-sampled benign nodes in the original training set T become 0 instead of the highest negative score of $-\theta$ because these benign nodes are not included in this new sampled training set. Recall that these benign nodes in T are heavily connected to the adversarial nodes. Accordingly, these benign nodes deliver fewer negative scores to the adversarial nodes and contribute to correctly identifying the adversarial nodes as positive (i.e., Sybil nodes). That is, using a randomly sampled training set, we are able to compute more reliable posterior scores that render the adversarial attack abusing these benign nodes less effective.

Given a training set T, our goal here is to guide RICC such that it emits posterior reputation scores close to those computed from a randomly sampled training set, thus robustly performing collective classification of adversarial Sybil nodes; RICC seeks out a set of prior scores for which the posterior scores of collective classification do not significantly change when using a different training set.

4.1 Random Sampling-Based Collective Classification

Figure 1 illustrates the overall idea of our defensive design. As shown in the figure, we propose to gradually change the prior scores for each node in the graph such that posterior scores computed based on T exhibit small differences compared to those measured using other randomly sampled training sets. Specifically, we iteratively sample a new training set, run collective classification campaigns, and incorporate the posterior score differences into the prior scores to be assigned in the next iteration.

In Figure 1, RICC misclassifies the target node D as benign by using the posterior scores $\mathbf{p}_{i-1}$ computed with the original training set T at the iteration $i-1$. However, when computing the posterior scores $\mathbf{p}'_{i-1}$ with a randomly sampled training set T'_{i-1}, RICC correctly classifies the target node as Sybil. After adding the posterior score differences $\mathbf{d}_{i-1}$ to the prior scores $\mathbf{q}_i$ at the iteration i, RICC becomes able to identify the target node as an adversarial Sybil node using the posterior scores $\mathbf{p}_i$. Note that the gap between the two posterior scores $\mathbf{p}$ and $\mathbf{p}'$ of a target node became narrower over iterations, which accords with our design goal.

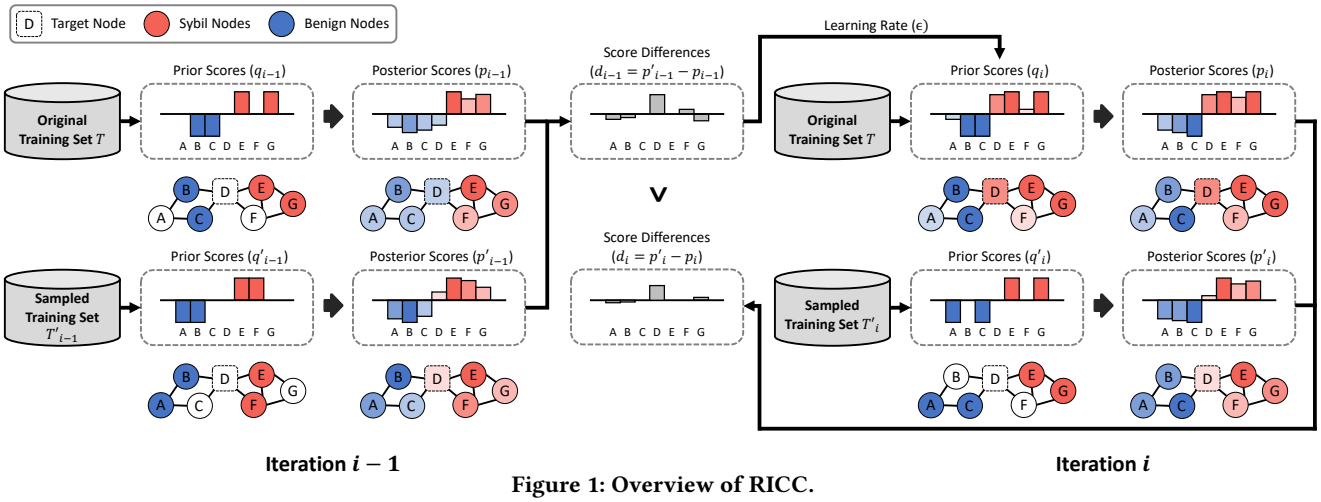


Figure 1: Overview of RICC.

Algorithm 2: Random Sampling-Based Collective Classification Algorithm.

```
Input  : A graph (G).
         A training set (T).
         A matrix of edge weights (W).
Output : A set of labeled nodes (𝕃).
1  function RunRandomSampleCC(G, T, W)
2      Initialize d_0 ← 0, q_0 ← {θ, −θ, 0}, and i ← 1.
3      while i ≤ I do
4          q_i, p_i ← RunCollectiveClassification(G, T, q_{i−1}, d_{i−1})
5          𝕃_i ← PredictNodeLabel(G, p_i)
6          T'_i ← RandomSampleTrainingSet(G, 𝕃_i)
7          q'_i, p'_i ← RunCollectiveClassification(G, T'_i, q_0, 0)
8          d_i ← ComputeScoreDiff(p_i, p'_i)
9          i ← i + 1
10     q, p ← RunCollectiveClassification(G, T, q_I, 0)
11     𝕃 ← PredictNodeLabel(G, p)
12     return 𝕃
13 function RunCollectiveClassification(G, T, q, d)
14     q ← AssignPriorScore(G, T, q, d)
15     Initialize p_0 ← q and t ← 1.
16     while t ≤ T_max do
17         p_t ← ComputePosteriorScore(G, q, W, p_{t−1})
18         t ← t + 1
19     return q, p_t
```

Algorithm 2 describes the overall process of RICC. RICC takes in the following four configurable parameters:

- ϵ The learning rate. This parameter controls the step size when incorporating the posterior score differences into the subsequent prior scores.
- N The number of sampled nodes. RICC randomly samples N benign nodes and N Sybil nodes and compares the posterior scores computed with the sampled training set T' and the original training set T.
- ρ The buffer ratio. RICC resorts to this parameter to provide a buffer zone that can adjust the subtle effect of slightly fluctuating posterior scores.
- I The maximum number of iterations. In each iteration, RICC gradually moves the posterior scores computed from T toward those calculated with other training sets T'.

The `RunRandomSampleCC` function takes a graph G, a training set T, a weight matrix $\mathbf{W}$, and the four configurable parameters from users. At each iteration i, it starts by running a collective classification algorithm with the training set T to compute the posterior scores $\mathbf{p}_i$ (Line 4). Based on these posterior scores $\mathbf{p}_i$, RICC labels each node in the graph and randomly samples the same number of benign and Sybil nodes from the labeled set $\mathbb{L}_i$ (Lines 5–6). Using these nodes as another training set T'_i, RICC computes a different set of posterior scores $\mathbf{p}'_i$ (Line 7). The `ComputeScoreDiff` function then calculates the difference between the two posterior scores: $\mathbf{p}_i$ and $\mathbf{p}'_i$ (Line 8). Note that these differences are consolidated into the prior scores $\mathbf{q}_{i+1}$ in the next iteration (Line 14 invoked by Line 4). After I times of iterations, RICC runs the collective classification algorithm using the final prior scores $\mathbf{q}_I$ to detect the Sybil nodes (Lines 10–11).

Assigning prior scores. When assigning prior scores using the original training set T, the `AssignPriorScore` function accumulates the posterior score differences over iterations, as shown in the following equation:

$$q_{u,i} = \begin{cases} \theta & u \in L_P \\ -\theta & u \in L_N \\ q_{u,i-1} + \epsilon \cdot d_{u,i-1} & otherwise \end{cases}, \tag{3}$$

where $q_{u,i}$ and $d_{u,i}$ correspond to the prior score and posterior score difference of the node u at the iteration i, respectively. Note that the `AssignPriorScore` function assigns a fixed prior score (i.e., θ or $-\theta$) to the labeled nodes in T. For a randomly sampled training set T', RICC assigns prior scores using Equation 1.

Computing posterior scores. The `ComputePosteriorScore` function iteratively propagates the posterior score of each node to its neighbor nodes. Interestingly, after several exploratory experiments, we observed that when RICC slightly adjusts the prior score of each target node based on the computed differences, the impact of such adjustments is aggregated across all target nodes and directly added to the posterior scores of benign nodes in the original training set T. Considering that these nodes are highly interconnected, this causes the overall posterior score of those benign nodes and target nodes to fluctuate over iterations.

$$\mathbf{p}_t = \rho(\mathbf{q}_i + 2\mathbf{W}\mathbf{p}_{t-1}) \tag{4}$$

Therefore, as shown in Equation 4, we designed the `ComputePosteriorScore` function to scale down the impact of posterior scores aggregated from connected neighbor nodes by multiplying these

scores by a buffer ratio ρ when aggregating the posterior scores. We empirically chose the buffer ratio to be 0.8. After I times of iterations, RICC computes the final posterior scores without the buffer ratio using the optimized prior scores $\mathbf{q}_I$.

Advantages. We emphasize that existing collective classification algorithms assign a fixed prior score (i.e., θ or $-\theta$) to the labeled nodes and a prior score of zero to the unlabeled nodes. Since the adversary is aware that collective classification algorithms assign prior scores in this manner, the adversary exploits this initial set of prior scores to adapt their attack. Therefore, instead of assigning prior scores of zero to unlabeled nodes, we propose to gradually adjust the prior scores of unlabeled nodes using a randomly sampled training set such that the computed posterior scores become robust to those adversarial attacks.

Although RICC requires a new training set at each iteration, it leverages the prediction made in the previous iteration when sampling a new training set. Therefore, it does not require any additional labeling effort for robust Sybil detection. Furthermore, unless the adversary compromises the graph so aggressively that even using a randomly sampled training set yields a high negative posterior score for target nodes, RICC can still perform robust detection (see §5.3). Finally, RICC is compatible with any collective classification algorithms and is thus readily applicable to existing collective classification services.

5 EVALUATION

We describe our evaluation settings (§5.1) and then evaluate the efficacy of RICC in classifying adversarial Sybil nodes (§5.2). We also measure varying performance across different attack budgets and attack strategies (§5.3 and 5.4). In the appendix, we further analyze the performance of RICC with different hyperparameters that RICC operators may choose (§A.2).

5.1 Experimental Setup

We conducted experiments on a machine running 64-bit Ubuntu 20.04 LTS with two Intel Xeon Gold 6258R (2.7 GHz) CPUs (112 cores), four GeForce RTX 3090 GPUs, and 640 GB of main memory. To implement RICC, we revised SybilSCAR [28], a state-of-the-art collective classification framework, which has been evaluated against adversarial attacks [25, 31]. We implemented RICC with 1K LoCs in Python on top of SybilSCAR.

5.1.1 Datasets. We used four datasets that previous studies [11, 25, 27, 28, 31, 32] have used to demonstrate performance in identifying Sybil nodes: Enron, Facebook, Twitter_S, and Twitter_L. These datasets cover diverse scenarios, each of which differs in terms of composition method (i.e., synthesized or real-world datasets), graph size, and average node degree. The Enron and Facebook datasets contain synthesized graphs, which have the same number of benign and Sybil nodes. The Enron dataset contains a total of 67K+ nodes and 371K+ edges, whereas the Facebook dataset has 8K+ nodes and 176K+ edges. Note that despite the smaller graph size, the average node degree of the Facebook graph (i.e., 44) is four times greater than that of the Enron graph (i.e., 11). The Twitter datasets correspond to real-world datasets. Twitter_S consists of 8K+ nodes and 54K+ edges, with 7K+ benign and 809 Sybil nodes. Twitter_L comprises 21M+ nodes and 265M+ edges, including 10M benign and 100K Sybil nodes.

Table 1: Sybil node detection performance of three detection methods: RICC, SybilSCAR, and JWP. We marked values in bold for outperforming methods.

Attack	Dataset	FNR (↓)			AUC (↑)		
		RICC	SybilSCAR	JWP	RICC	SybilSCAR	JWP
ENM [25]	Enron	$\mathbf{0.01_{\pm.00}}$	1.00	1.00	$\mathbf{0.9912_{\pm.0002}}$	0.9884	0.9875
	Facebook	$\mathbf{0.11_{\pm.02}}$	0.95	0.97	$\mathbf{0.9995_{\pm.0001}}$	0.9372	0.9551
	Twitter_S	$\mathbf{0.00_{\pm.00}}$	1.00	0.99	$\mathbf{0.8911_{\pm.0022}}$	0.7117	0.6921
	Twitter_L	$\mathbf{0.01_{\pm.02}}$	1.00	1.00	$\mathbf{0.7388_{\pm.0001}}$	0.7371	0.7375
NNI [31]	Enron	$\mathbf{0.00_{\pm.00}}$	1.00	1.00	$\mathbf{0.9904_{\pm.0001}}$	0.9871	0.9878
	Facebook	$\mathbf{0.00_{\pm.00}}$	0.93	0.39	$\mathbf{0.9999_{\pm.0607}}$	0.9533	0.9841
	Twitter_S	$\mathbf{0.00_{\pm.00}}$	0.88	0.88	$\mathbf{0.6771_{\pm.0016}}$	0.6632	0.6686
	Twitter_L	$\mathbf{0.00_{\pm.00}}$	0.90	0.99	$\mathbf{0.7391_{\pm.0001}}$	0.7368	0.7369

5.1.2 Hyperparameters. For Enron, Facebook, and Twitter_S, we set $|T| = 100$, $N = 100$, $\epsilon = 0.005$, and $I = 3{,}000$. For the Twitter_L dataset, we used $|T| = 3{,}000$, $N = 3{,}000$, $\epsilon = 0.15$, and $I = 100$, due to its large size. We further study the effect of using different hyperparameters in §A.2.

5.1.3 Adversarial Attacks. To evaluate the efficacy of Sybil detection methods, we conducted two state-of-the-art adversarial attacks: optimization-based existing node manipulation (ENM) [25] and new node insertion (NNI) [31]. These adversarial attacks assume a strong white-box adversary who has full knowledge of a target detection method. The goal of the adversary is to cause the misclassification of a given set of target nodes. We demonstrate the robustness of RICC against this strong adversary using 100 target nodes, which is the same number of target nodes used in previous studies of adversarial attacks [25, 31]. Since the implementation of NNI is publicly unavailable, we implemented this attack in 1,200 LoCs of C++ and validated its performance by comparing the metrics to those reported in the paper [31].

5.1.4 Metrics. Recall that our goal is to correctly identify target adversarial Sybil nodes as well as other Sybil nodes. To evaluate whether Sybil detection methods achieve this goal, we use the false negative rates (FNRs) of target nodes and the area under the curve (AUC), which have been widely adopted by prior studies [11, 25, 27, 28, 31, 39]. Since RICC samples a new training set T' in a random fashion, we run all experiments five times and report median values along with standard deviation values.

FNR. We measure a ratio of target Sybil nodes incorrectly identified as benign nodes among all target nodes, which indicates the extent to which adversarial attacks achieve their goal. Note that the FNR becomes lower as a Sybil detection method performs better at mitigating the adversarial attack threats.

AUC. We compute true positive and false positive rates for all nodes under varying thresholds. The AUC explains how well Sybil detection methods identify all Sybil nodes, including target adversarial nodes. The more accurately a Sybil detection method classifies Sybil and benign nodes, the greater the AUC it exhibits.

5.2 Identifying Sybil Nodes

To evaluate the ability to detect Sybil nodes under the presence of adversarial attacks, we conducted adversarial attacks on a target graph and compared the performance of RICC against two collective classification-based Sybil detection approaches: SybilSCAR [28] and JWP [27]. SybilSCAR and JWP are state-of-the-art Sybil detection

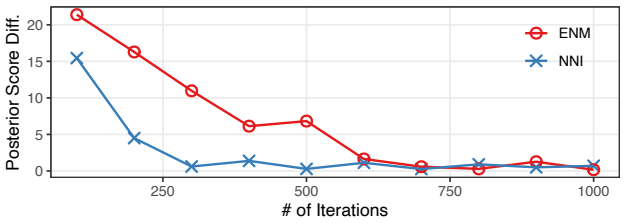
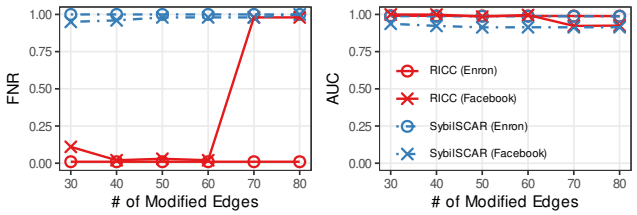


Figure 2: Sum of posterior score differences of the adversarial nodes on the Enron graph.



(a) FNR (↓) **(b) AUC (↑)**

Figure 3: Sybil detection performance against ENM attacks while varying the number of modified edges.

methods, which have been used as target methods in adversarial attack studies [25, 31].

Table 1 presents the performance of three detection methods in identifying Sybil nodes over the four datasets. The third to fifth columns represent FNRs when deploying RICC, SybilSCAR, and JWP, respectively. The sixth to eighth columns show the AUCs of collective classification for RICC, SybilSCAR, and JWP, respectively. The upper and lower halves show the detection performance of each method after conducting ENM and NNI, respectively.

Note that the adversaries have multiple sets of attack strategies. In this table, we only show the detection results after conducting the attacks with the strongest strategy reported in [25] and later elaborate on the effect of choosing a different attack strategy in §5.4. As shown in the fourth and fifth columns, the conducted attacks were strong enough to break all the existing detection methods, attaining FNRs of over 0.9 in most cases. However, JWP applied to the Facebook graph exhibited a relatively low FNR against NNI. This attack operates based on the assumption that the target detection methods employ SybilSCAR and resorts to the transferability of the attack when targeting other detection methods. Therefore, we believe that the attack was less successful in this case.

In contrast to SybilSCAR and JWP, the third column demonstrates that RICC is capable of mitigating these threats. We emphasize that FNRs decreased to 0 in most cases when RICC was deployed, detecting all the target nodes. Furthermore, the sixth to eighth columns report that RICC shows better performance in detecting other Sybil nodes as well. Note that the proposed detection method enables RICC to rectify the misclassification of target nodes, which also helps it correctly recognize other nodes connected to the target nodes, thus improving the AUC metric.

Figure 2 describes the sum of posterior score differences $\mathbf{d}_i$ for the target nodes on the Enron graph, measured at every 100 iterations. The figure demonstrates that the differences indeed converge to 0, denoting that RICC computes robust posterior scores for target nodes. Based on these observations, we confirm that RICC successfully mitigates state-of-the-art adversarial attacks by correctly classifying target Sybil nodes and other Sybil nodes. In §A.3, we further compared RICC with ProGNN [17], a seminal defense framework against GNN adversarial attacks.

5.3 Effect of the Attack Budget

The attack strength could vary depending on the attack budget that the adversary can leverage. For example, if the adversary has an unlimited attack budget to apply to connecting each target node to all benign nodes, collective classification algorithms would suffer from finding those target nodes, exhibiting a high FNR. We thus evaluate the extent to which Sybil detection methods can withstand adversarial attacks by gradually increasing the attack budget. The ENM and NNI adversaries have different sets of attack budgets. The ENM adversary is able to modify only a limited number of edges for each target node, whereas the NNI adversary inserts a limited number of new nodes and edges.

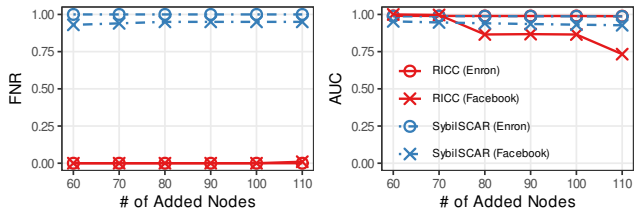
5.3.1 Number of Modified Edges. The ENM attack adds new edges between existing nodes or deletes existing edges. In Table 1, we set the ENM attack to manipulate at most 30 edges for each target node, which is a greater attack budget compared to that Wang *et al.* used [25], to conduct sufficiently strong attacks. In other words, the adversary can perform edge manipulation 3,000 times while conducting an attack against 100 target nodes. We now show whether RICC can still mitigate ENM attacks even when the adversary attempts to modify more edges for each target node.

Figure 3 summarizes the results for RICC and SybilSCAR on the Enron and Facebook graphs. The Twitter and JWP results were analogous to those for SybilSCAR on the Enron graph. We thus omit them in this figure and refer the reader to Appendix A.4 for the additional results. As Figure 3 demonstrates, RICC significantly outperformed the state-of-the-art detection method in identifying Sybil accounts, regardless of the number of modified edges. In the case of the Enron graph, RICC consistently yielded FNRs close to 0 and AUC metrics close to 1 across different numbers of modified edges. This result implies that RICC is resistant to ENM attacks even when the adversary manipulates up to 80 edges per target node. In contrast, SybilSCAR was susceptible to ENM attacks such that it yielded an FNR of 1 even when the adversary manipulated only 30 edges for each target node.

For the Facebook graph, RICC became non-functional when the adversary manipulated more than 7,000 edges. We attribute this result to the graph size. Since the Facebook graph is much smaller than the Enron graph, the Facebook graph was more vulnerable to such massive edge manipulation. However, considering that an ENM attack seeks to minimize the attack cost, we believe that the adversary would avoid such massive edge manipulation.

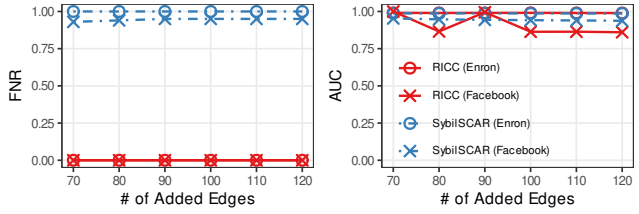
5.3.2 Number of Added Nodes and Edges. An NNI attack inserts new nodes and connects them to existing nodes. Therefore, the number of added nodes and the number of edges connected to those nodes play a role in attack budgets. In Table 1, we set the maximum number of added nodes and edges to 60 and 70, respectively. Note that this combination of budgets was reported as the most powerful attack in the original paper [31]. With these attack budgets, the adversary is able to add at most 60 new nodes and 4,200 new edges.

Figures 4 and 5 show that the Sybil detection performance changes against NNI attacks while we vary the number of added nodes and



(a) FNR (↓) (b) AUC (↑)

Figure 4: Sybil detection performance against NNI attacks while varying the number of added nodes.



(a) FNR (↓) (b) AUC (↑)

Figure 5: Sybil detection performance against NNI attacks while varying the number of added edges.

the number of edges connected to those nodes, respectively. When evaluating the effect of one attack budget, we kept the other budget fixed. Note from the figures that RICC consistently rendered superior detection performance over SybilSCAR in spotting target nodes, thus successfully mitigating the adversarial threats. Notably, the FNR of RICC was still stable when the attacker inserted 110 nodes or 120 edges into the graph, whereas SybilSCAR was already defeated when the adversary added only 60 nodes or 70 edges.

However, in the case of the Facebook graph, the AUC score of RICC started to plummet when the adversary appended more than 80 nodes or 100 edges. We analyzed the reason for this AUC drop and found that the false positive rates surge when the adversary inserts too many nodes. After several iterations, RICC correctly classified the inserted nodes as Sybil nodes. Since these nodes are connected to many benign nodes to bypass detection, the positive scores of the inserted nodes are aggregated and propagated to the connected benign nodes. Therefore, when the attacker aggressively adds many nodes, the benign nodes connected to those nodes gain a large positive score. Consequently, RICC misclassifies such benign nodes as positive nodes, which increases the false positive rates and drops the AUC score.

From our observations, we confirm that detection performance and attack budgets have a trade-off relationship. That is, the adversary is able to undermine the detection performance at the expense of a huge attack budget. However, we stress that investing a huge attack budget into adversarial attacks is contrary to the adversary's goal of minimizing attack costs. In Appendix A.1, we further evaluate the effect of the number of target nodes.

5.4 Effect of the Attack Strategy

The adversary's objective is to alter a detector's prediction on each target node while spending a minimum attack cost. To accomplish this objective, the adversary should consider multiple factors, which highly affect attack performance, to devise an attack strategy. As the previous study [25] considered two factors, namely target node selection methods and edge manipulation costs, we evaluate the robustness of Sybil detection methods while varying these two factors in our evaluation.

Table 2: Effect of target node selection methods on Sybil node detection performance.

Method	Attack	Dataset	FNR (↓)			AUC (↑)		
			RICC	SybilSCAR	JWP	RICC	SybilSCAR	JWP
CC	ENM [25]	Enron	$\mathbf{0.00}_{\pm.00}$	1.00	0.98	$\mathbf{0.9910}_{\pm.0003}$	0.9826	0.9762
		Facebook	$\mathbf{0.88}_{\pm.43}$	0.96	0.96	$0.9307_{\pm.0371}$	0.9206	**0.9719**
		Twitter_S	$\mathbf{0.00}_{\pm.00}$	0.87	0.87	$0.7552_{\pm.0011}$	**0.7568**	0.5984
		Twitter_L	$\mathbf{0.00}_{\pm.00}$	0.93	1.00	$\mathbf{0.7387}_{\pm.0001}$	0.7378	**0.7387**
	NNI [31]	Enron	$\mathbf{0.00}_{\pm.00}$	0.99	1.00	$\mathbf{0.9905}_{\pm.0003}$	0.9858	0.9842
		Facebook	$\mathbf{0.00}_{\pm.00}$	0.88	0.50	$\mathbf{0.9999}_{\pm.0000}$	0.9731	0.9839
		Twitter_S	$\mathbf{0.00}_{\pm.00}$	0.29	0.66	$\mathbf{0.7676}_{\pm.0013}$	0.7422	0.5804
		Twitter_L	$\mathbf{0.00}_{\pm.00}$	0.91	0.99	$\mathbf{0.7388}_{\pm.0000}$	0.7374	0.7378
Random	ENM [25]	Enron	$\mathbf{0.00}_{\pm.00}$	0.97	1.00	$\mathbf{0.9907}_{\pm.0002}$	0.9876	0.9825
		Facebook	$\mathbf{0.56}_{\pm.27}$	0.74	0.78	$\mathbf{0.9904}_{\pm.0578}$	0.9818	0.9779
		Twitter_S	$\mathbf{0.02}_{\pm.00}$	0.79	0.87	$\mathbf{0.7776}_{\pm.0020}$	0.7487	0.6232
		Twitter_L	$\mathbf{0.01}_{\pm.02}$	0.91	0.99	$0.7379_{\pm.0001}$	0.7363	**0.7380**
	NNI [31]	Enron	$\mathbf{0.00}_{\pm.00}$	0.96	1.00	$\mathbf{0.9907}_{\pm.0002}$	0.9869	0.9868
		Facebook	$\mathbf{0.00}_{\pm.00}$	0.51	0.55	$\mathbf{0.9999}_{\pm.0498}$	0.9818	0.9832
		Twitter_S	$\mathbf{0.06}_{\pm.01}$	0.57	0.77	$\mathbf{0.7834}_{\pm.0013}$	0.7183	0.5905
		Twitter_L	$\mathbf{0.00}_{\pm.00}$	0.70	0.93	$\mathbf{0.7390}_{\pm.0000}$	0.7370	0.7370

5.4.1 Target Node Selection Methods. The adversary is able to deploy three different strategies when selecting target nodes: Random, Connected components (CC), and Close. Random refers to the strategy in which the adversary randomly samples target nodes from the Sybil nodes. CC refers to the strategy whereby the adversary picks a random target node and conducts a breadth-first search from the node to collect target nodes. This strategy enables the target nodes to become densely connected. That is, once the adversary succeeds in evading the detection of a single target node, this effect would be rapidly propagated to the other target nodes. Lastly, for the Close strategy, the adversary selects Sybil nodes closely located to benign nodes as the target nodes. Since Wang *et al.* [25] reported this strategy as the most powerful strategy, we adopted this strategy for the adversarial attacks in Table 1 (see §5.2).

Table 2 describes the Sybil detection performance against the attacks with two target node selection methods: CC and Random. We note from the fifth and the sixth columns that these strategies contributed to achieving FNRs lower than 0.9 in many cases, implying that CC and Random are indeed the weaker strategies in implementing the adversarial attacks compared to the Close strategy. The table also confirms that RICC consistently outperforms SybilSCAR and JWP, regardless of the target node detection methods.

However, the detection performance of RICC plunged when the ENM adversary selected target nodes using CC on the Facebook graph. After manual inspection, we noticed that the Facebook dataset is a small graph that has a considerably large number of compromised nodes. Since only 50% of the nodes in the Facebook graph are benign, each benign node has a higher chance of being included in T'. Therefore, RICC often samples yet misclassified target nodes (i.e., false negative nodes) as benign nodes in T' for the next iteration. Note that these nodes are assigned with a high negative prior score and are closely connected to the other target nodes. RICC thus rapidly propagates the negative scores to the other target nodes. Consequently, the overall posterior scores of the target nodes move toward the negative end, and the FNR increases.

Table 3: Effect of cost types on Sybil detection performance.

Type	Attack	Dataset	FNR (↓)			AUC (↑)		
			RICC	SybilSCAR	JWP	RICC	SybilSCAR	JWP
Uniform	ENM [25]	Enron	$\mathbf{0.01}_{\pm.00}$	1.00	1.00	$\mathbf{0.9913}_{\pm.0001}$	0.9884	0.9875
		Facebook	$\mathbf{0.10}_{\pm.04}$	0.95	0.97	$\mathbf{0.9993}_{\pm.0001}$	0.9372	0.9558
		Twitter_S	$\mathbf{0.00}_{\pm.00}$	1.00	0.99	$\mathbf{0.8899}_{\pm.0026}$	0.7118	0.6932
		Twitter_L	$\mathbf{0.03}_{\pm.00}$	1.00	0.99	$\mathbf{0.7387}_{\pm.0001}$	0.7371	0.7379
	NNI [31]	Enron	$\mathbf{0.00}_{\pm.00}$	1.00	1.00	$\mathbf{0.9907}_{\pm.0006}$	0.9872	0.9873
		Facebook	$\mathbf{0.00}_{\pm.00}$	0.92	0.74	$\mathbf{0.9999}_{\pm.0632}$	0.9586	0.9762
		Twitter_S	$\mathbf{0.00}_{\pm.00}$	0.88	0.97	$\mathbf{0.6839}_{\pm.0023}$	0.6631	0.6508
		Twitter_L	$\mathbf{0.00}_{\pm.00}$	0.85	0.97	$\mathbf{0.7390}_{\pm.0000}$	0.7365	0.7373
Cat.	ENM [25]	Enron	$\mathbf{0.01}_{\pm.00}$	1.00	1.00	$\mathbf{0.9908}_{\pm.0002}$	0.9884	0.9875
		Facebook	$\mathbf{0.03}_{\pm.01}$	0.95	0.95	$\mathbf{0.9990}_{\pm.0530}$	0.9376	0.9559
		Twitter_S	$\mathbf{0.00}_{\pm.00}$	1.00	0.99	$\mathbf{0.8905}_{\pm.0019}$	0.7117	0.6931
		Twitter_L	$\mathbf{0.03}_{\pm.01}$	1.00	0.99	$\mathbf{0.7388}_{\pm.0001}$	0.7371	0.7376
	NNI [31]	Enron	$\mathbf{0.00}_{\pm.00}$	0.98	1.00	$\mathbf{0.9905}_{\pm.0005}$	0.9874	0.9874
		Facebook	$\mathbf{0.00}_{\pm.00}$	0.90	0.80	$\mathbf{0.9999}_{\pm.0438}$	0.9542	0.9751
		Twitter_S	$\mathbf{0.00}_{\pm.00}$	0.88	0.96	$\mathbf{0.6760}_{\pm.0021}$	0.6632	0.6503
		Twitter_L	$\mathbf{0.00}_{\pm.00}$	0.75	0.96	$\mathbf{0.7390}_{\pm.0001}$	0.7369	0.7375

We further evaluated whether RICC is capable of mitigating this case by using a two times smaller buffer ratio and a two times greater learning rate. The former mitigates the rapid propagation of negative scores (recall Equation 4), and the latter enables faster detection of target nodes. We confirmed that RICC successfully achieved an FNR of 0.00 (±.02) and an AUC score of 1.0000 (±.0317) under this setting. Hence, we conclude that RICC can mitigate the adversarial threats against all three target node selection methods.

5.4.2 Edge Manipulation Costs. The same edge manipulation could require different attack costs depending on an adversary or a target application. To this end, Wang *et al.* [25] simulated three attack scenarios: Equal, Uniform, and Categorical. For Table 1, we implemented the ENM and NNI attacks assuming that every edge modification demands an equal cost.

We additionally evaluated each detection method against the Uniform and Categorical scenarios as in the prior study [25]. In the Uniform scenario, the modification costs for each edge are uniformly distributed. However, considering that only Sybil nodes are under the adversary's control, connecting a benign node to a Sybil node would be more difficult than adding edges between Sybil nodes. The Categorical strategy addresses this manipulation cost discrepancy by assigning a higher cost for the former case. Similar to the results of Equal, Table 3 displays that RICC is robust against all these cost assignment strategies, while SybilSCAR and JWP are susceptible to all of them.

6 RELATED WORK

Collective classification. Collective classification is a prevailing semi-supervised node classification method used to identify fake accounts and reviews in web services. SybilGuard [34] and SybilLimit [33] find Sybil nodes by conducting random walks under the assumption that random walks from given benign nodes are likely to stay in a benign region.

LBP-based methods [6, 11, 13, 15, 22, 24, 28] have been proposed to address the limitation above. SybilSCAR [28] deploys a novel method called LinLBP. LinLBP is able to guarantee convergence even against a graph with loops as well as exhibits high performance in identifying Sybil nodes. GANG [26] considers Sybil nodes in a directed graph. GANG propagates the prior scores in the training set using LBP and conducts optimization to ensure convergence. All these LBP-based methods assumed that there exists only a small number of edges between a Sybil region and a benign region. Unfortunately, previous research has shown that real-world networks do not follow this assumption [1, 3]. To this end, various methods [2, 11, 27] have proposed to assign different weights to each edge. SybilFuse [11] assigns different weights by training local classifiers. JWP [27] learns edge weights by solving an optimization problem that aims to increase the edge weight of homogeneous nodes and decrease the edge weight of heterogeneous nodes.

Attacking collective classification. Wang *et al.* [25] proposed an adversarial attack that targets LinLBP. They first selected target nodes to attack among a given set of Sybil nodes. They then changed the connection of target nodes to evade the detection of target nodes. Xu *et al.* [31] also proposed a similar approach. They added a new node without changing the structure of the existing graph and then added a new edge between the new node and the existing node. Both of these attacks find the most efficient way in saving the attack budget and making the attacks successful by solving a defined optimization problem.

Defense against graph adversarial attacks. There has been a vast volume of defensive research [7, 9, 12, 17, 29, 30, 35, 37, 39] on adversarial attacks against graph neural networks (GNNs). Among these studies, graph purification addresses an attack scenario in which the adversary manipulates a graph to bypass the detection of target nodes, analogous to our threat model. Wu *et al.* [29] have observed that many adversarial attacks tend to add edges between dissimilar nodes. To mitigate such attacks, they removed edges between nodes with low Jaccard Similarity. Jin *et al.* [17] have found that adversarial attacks significantly change the intrinsic properties of a clean graph. They thus iteratively reconstructed a manipulated graph such that the reconstructed graph still preserves the properties of a clean graph and trains a robust GNN model.

7 CONCLUSION

We present RICC, the first collective classification framework to perform robust detection of Sybil accounts. RICC exploits the novel observation that the posterior reputation scores of Sybil accounts created by adversarial attacks are highly volatile with respect to an initial set of prior scores. RICC therefore learns a robust way of performing collective classification in which posterior scores remain steady across different sets of prior scores randomly sampled from the original training set. Our experimental results demonstrate that RICC outperforms all existing Sybil detection methods in identifying Sybil nodes, including adversarial nodes, thus establishing RICC as the first practical tool for performing robust identification of Sybil accounts.

ACKNOWLEDGMENTS

We thank the anonymous reviewers and Binghui Wang for their constructive feedback and research artifacts, respectively. This work was partly supported by (1) Institute of Information & communications Technology Planning & evaluation (IITP) grant funded by the Korea government (MSIT) (No.2020-0-00153) and (2) Korea Internet & Security Agency (KISA) grant funded by PIPC (No.1781000003).

REFERENCES

[1] Leyla Bilge, Thorsten Strufe, Davide Balzarotti, and Engin Kirda. 2009. All Your Contacts Are Belong to Us: Automated Identity Theft Attacks on Social Networks. In *Proceedings of the International Conference on World Wide Web*. 551–560.

[2] Yazan Boshmaf, Dionysios Logothetis, Georgos Siganos, Jorge Lería, Jose Lorenzo, Matei Ripeanu, and Konstantin Beznosov. 2015. Íntegro: Leveraging Victim Prediction for Robust Fake Account Detection in OSNs. In *Proceedings of the Network and Distributed System Security Symposium*.

[3] Yazan Boshmaf, Ildar Muslukhov, Konstantin Beznosov, and Matei Ripeanu. 2011. The Socialbot Network: When Bots Socialize for Fame and Money. In *Proceedings of the Annual Computer Security Applications Conference*. 93–102.

[4] Adam Breuer, Roee Eilat, and Udi Weinsberg. 2020. Friend or Faux: Graph-Based Early Detection of Fake Accounts on Social Networks. In *Proceedings of the Web Conference*. 1287–1297.

[5] Qiang Cao, Michael Sirivianos, Xiaowei Yang, and Tiago Pregueiro. 2012. Aiding the Detection of Fake Accounts in Large Scale Social Online Services. In *Proceedings of the USENIX Conference on Networked Systems Design and Implementation*. 197–210.

[6] Duen Horng "Polo" Chau, Carey Nachenberg, Jeffrey Wilhelm, Adam Wright, and Christos Faloutsos. 2011. Polonium: Tera-Scale Graph Mining and Inference for Malware Detection. In *Proceedings of the SIAM International Conference on Data Mining*. 131–142.

[7] Hanjun Dai, Hui Li, Tian Tian, Xin Huang, Lin Wang, Jun Zhu, and Le Song. 2018. Adversarial Attack on Graph Structured Data. In *Proceedings of the International Conference on Machine Learning*. 1115–1124.

[8] George Danezis and Prateek Mittal. 2009. SybilInfer: Detecting Sybil Nodes using Social Networks. In *Proceedings of the Network and Distributed System Security Symposium*.

[9] Zhijie Deng, Yinpeng Dong, and Jun Zhu. 2019. Batch Virtual Adversarial Training for Graph Convolutional Networks. *CoRR* abs/1902.09192 (2019).

[10] Hongyu Gao, Jun Hu, Christo Wilson, Zhichun Li, Yan Chen, and Ben Y. Zhao. 2010. Detecting and Characterizing Social Spam Campaigns. In *Proceedings of the ACM SIGCOMM Conference*. 35–47.

[11] Peng Gao, Binghui Wang, Neil Zhenqiang Gong, Sanjeev R. Kulkarni, Kurt Thomas, and Prateek Mittal. 2018. SybilFuse: Combining Local Attributes with Global Structure to Perform Robust Sybil Detection. In *Proceedings of the IEEE Conference on Communications and Network Security*.

[12] Simon Geisler, Tobias Schmidt, Hakan Şirin, Daniel Zügner, Aleksandar Bojchevski, and Stephan Günnemann. 2021. Robustness of Graph Neural Networks at Scale. In *Proceedings of the Advances in Neural Information Processing Systems*. 7637–7649.

[13] Neil Zhenqiang Gong, Mario Frank, and Prateek Mittal. 2014. SybilBelief: A Semi-Supervised Learning Approach for Structure-Based Sybil Detection. *IEEE Transactions on Information Forensics and Security* 9, 6 (2014), 976–987.

[14] Jinyuan Jia, Binghui Wang, and Neil Zhenqiang Gong. 2017. Random Walk Based Fake Account Detection in Online Social Networks. In *Proceedings of the IEEE/IFIP International Conference on Dependable Systems and Networks*. 273–284.

[15] Jinyuan Jia, Binghui Wang, Le Zhang, and Neil Zhenqiang Gong. 2017. AttriInfer: Inferring User Attributes in Online Social Networks Using Markov Random Fields. In *Proceedings of the International Conference on World Wide Web*. 1561–1569.

[16] Wei Jin, Yaxin Li, Han Xu, Yiqi Wang, Shuiwang Ji, Charu Aggarwal, and Jiliang Tang. 2020. Adversarial Attacks and Defenses on Graphs: A Review, A Tool and Empirical Studies. *ACM SIGKDD Explanations Newsletter* 22, 2 (2020), 19–34.

[17] Wei Jin, Yao Ma, Xiaorui Liu, Xianfeng Tang, Suhang Wang, and Jiliang Tang. 2020. Graph Structure Learning for Robust Graph Neural Networks. In *Proceedings of the ACM SIGKDD Conference on Knowledge Discovery and Data Mining*. 66–74.

[18] Zhou Li, Sumayah Alrwais, Yinglian Xie, Fang Yu, and XiaoFeng Wang. 2013. Finding the Linchpins of the Dark Web: a Study on Topologically Dedicated Hosts on Malicious Web Infrastructures. In *Proceedings of the IEEE Symposium on Security and Privacy*. 112–126.

[19] Xiao Liang, Zheng Yang, Binghui Wang, Shaofeng Hu, Zijie Yang, Dong Yuan, Neil Zhenqiang Gong, Qi Li, and Fang He. 2021. Unveiling Fake Accounts at the Time of Registration: An Unsupervised Approach. In *Proceedings of the ACM SIGKDD Conference on Knowledge Discovery and Data Mining*. 3240–3250.

[20] Yushan Liu, Shouling Ji, and Prateek Mittal. 2016. SmartWalk: Enhancing Social Network Security via Adaptive Random Walks. In *Proceedings of the ACM Conference on Computer and Communications Security*. 492–503.

[21] Annie Palmer. 2022. Amazon sues two companies that allegedly help fill the site with fake reviews. https://www.cnbc.com/2022/02/22/amazon-sues-alleged-fake-reviews-brokers-appsally-rebatest.html.

[22] Shashank Pandit, Duen Horng Chau, Samuel Wang, and Christos Faloutsos. 2007. NetProbe: A Fast and Scalable System for Fraud Detection in Online Auction Networks. In *Proceedings of the International Conference on World Wide Web*. 201–210.

[23] Feliz Solomon. 2016. Alibaba Takes On Fake Reviews in Its Latest Push For More Credibility. https://fortune.com/2016/12/20/alibaba-jack-ma-china-fake-reviews/.

[24] Acar Tamersoy, Kevin Roundy, and Duen Horng Chau. 2014. Guilt by Association: Large Scale Malware Detection by Mining File-relation Graphs. In *Proceedings of the ACM SIGKDD Conference on Knowledge Discovery and Data Mining*. 1524–1533.

[25] Binghui Wang and Neil Zhenqiang Gong. 2019. Attacking Graph-based Classification via Manipulating the Graph Structure. In *Proceedings of the ACM Conference on Computer and Communications Security*. 2023–2040.

[26] Binghui Wang, Neil Zhenqiang Gong, and Hao Fu. 2017. GANG: Detecting Fraudulent Users in Online Social Networks via Guilt-by-Association on Directed Graphs. In *Proceedings of the IEEE International Conference on Data Mining*. 465–474.

[27] Binghui Wang, Jinyuan Jia, and Neil Zhenqiang Gong. 2019. Graph-based Security and Privacy Analytics via Collective Classification with Joint Weight Learning and Propagation. In *Proceedings of the Network and Distributed System Security Symposium*.

[28] Binghui Wang, Le Zhang, and Neil Zhenqiang Gong. 2017. SybilSCAR: Sybil Detection in Online Social Networks via Local Rule based Propagation. In *Proceedings of the IEEE Conference on Computer Communications*.

[29] Huijun Wu, Chen Wang, Yuriy Tyshetskiy, Andrew Docherty, Kai Lu, and Liming Zhu. 2019. Adversarial Examples for Graph Data: Deep Insights into Attack and Defense. In *Proceedings of the International Joint Conference on Artificial Intelligence*. 4816–4823.

[30] Kaidi Xu, Hongge Chen, Sijia Liu, Pin-Yu Chen, Tsui-Wei Weng, Mingyi Hong, and Xue Lin. 2019. Topology Attack and Defense for Graph Neural Networks: An Optimization Perspective. *CoRR* abs/1906.04214 (2019).

[31] Xuening Xu, Xiaojiang Du, and Qiang Zeng. 2020. Attacking Graph-Based Classification without Changing Existing Connections. In *Proceedings of the Annual Computer Security Applications Conference*. 951–962.

[32] Chao Yang, Robert Harkreader, Jialong Zhang, Seungwon Shin, and Guofei Gu. 2012. Analyzing Spammers' Social Networks for Fun and Profit: A Case Study of Cyber Criminal Ecosystem on Twitter. In *Proceedings of the International Conference on World Wide Web*. 71–80.

[33] Haifeng Yu, Phillip B. Gibbons, Michael Kaminsky, and Feng Xiao. 2008. SybilLimit: A Near-Optimal Social Network Defense against Sybil Attacks. In *Proceedings of the IEEE Symposium on Security and Privacy*. 3–17.

[34] Haifeng Yu, Michael Kaminsky, Phillip B. Gibbons, and Abraham Flaxman. 2006. SybilGuard: Defending against Sybil Attacks via Social Networks. In *Proceedings of the ACM SIGCOMM Conference*. 267–278.

[35] Ao Zhang and Jinwen Ma. 2020. DefenseVGAE: Defending against Adversarial Attacks on Graph Data via a Variational Graph Autoencoder. *CoRR* abs/2006.08900 (2020).

[36] Xiaoying Zhang, Hong Xie, Pei Yi, and John C.S. Lui. 2022. Enhancing Sybil Detection via Social-Activity Networks: A Random Walk Approach. *IEEE Transactions on Dependable and Secure Computing* (2022).

[37] Xiang Zhang and Marinka Zitnik. 2020. GNNGuard: Defending Graph Neural Networks against Adversarial Attacks. *Proceedings of the Advances in Neural Information Processing Systems* 33 (2020), 9263–9275.

[38] Haizhong Zheng, Minhui Xue, Hao Lu, Shuang Hao, Haojin Zhu, Xiaohui Liang, and Keith Ross. 2018. Smoke Screener or Straight Shooter: Detecting Elite Sybil Attacks in User-Review Social Networks. In *Proceedings of the Network and Distributed System Security Symposium*.

[39] Dingyuan Zhu, Ziwei Zhang, Peng Cui, and Wenwu Zhu. 2019. Robust Graph Convolutional Networks Against Adversarial Attacks. In *Proceedings of the ACM SIGKDD Conference on Knowledge Discovery and Data Mining*. 1399–1407.

A APPENDIX

A.1 Effect of the Number of Target Nodes

Both ENM and NNI adversaries select a set of target nodes among Sybil nodes and aim to bypass the detection of these target nodes. In this regard, the adversary is able to perform adversarial attacks on a larger number of target nodes. We thus evaluate whether Sybil detection methods can still identify Sybil nodes even when the adversary selects more target nodes. In Table 1, we used 100 target nodes, following the experimental setups of prior adversarial attacks [25, 31].

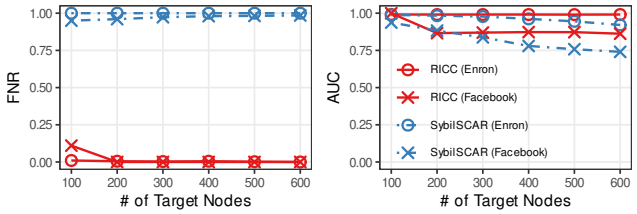
Recall from §5.3 that the attack budget of the ENM adversary is defined as the number of manipulated edges per target node. That is, if the adversary conducts ENM attacks on more target nodes, the adversary is able to modify more edges, thereby enhancing the attack strength. Therefore, as shown in Figure 6, the AUC measured using SybilSCAR gradually degraded as the adversary manipulated more target nodes. In terms of the detection of target nodes, exploiting 100 target nodes is already powerful enough to break SybilSCAR. In contrast, RICC significantly outperformed SybilSCAR in all cases, achieving lower FNRs and higher AUCs.

Figure 7 presents the attack results against NNI attacks. Recall that the NNI adversary's attack budget is limited by the number of edges connected to inserted nodes. Therefore, if the NNI adversary increases the number of target nodes, the adversary needs to connect each new node to a limited number of target nodes. In other words, the attack becomes weaker, as the adversary exploits more target nodes. Note from the figure that the FNRs computed against SybilSCAR decrease accordingly. RICC consistently exhibited superior performance over SybilSCAR in all cases.

A.2 Effect of the Hyperparameters

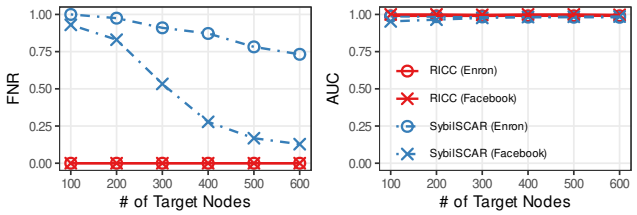
We evaluate the effect of three hyperparameters that RICC uses: a learning rate ϵ, a sampling size N, and a buffer ratio ρ.

Learning rate ϵ. Figure 8 describes the FNR changes of RICC in identifying adversarial nodes on the Enron graph across different learning rates. Since the FNRs measured on the other datasets also exhibited a similar pattern, we omit those results in the figure. A learning rate decides the step size used to reflect the posterior score differences into the prior scores. In other words, if RICC uses a larger learning rate, the FNR converges faster, and thus RICC users stop the iteration early. Note in the figure that RICC was capable of finding all target nodes before 200 iterations when we used 0.016 as the learning rate, whereas RICC with a learning rate of 0.002 was still unable to identify all target nodes even after 1,000 iterations. Thus, when running RICC on the Twitter_L dataset, a large graph with 21M+ nodes and 265M+ edges, we decreased the number of iterations (i.e., 3,000→100) and increased the learning rate (i.e., 0.005→0.15) with the same ratio for faster detection (see §5.1.2). The sixth and tenth rows of Table 1 demonstrate that RICC exhibited outstanding detection performance on this large graph.



(a) FNR (↓) (b) AUC (↑)

Figure 6: Sybil detection performance against ENM attacks while varying the number of target nodes.



(a) FNR (↓) (b) AUC (↑)

Figure 7: Sybil detection performance against NNI attacks while varying the number of target nodes.

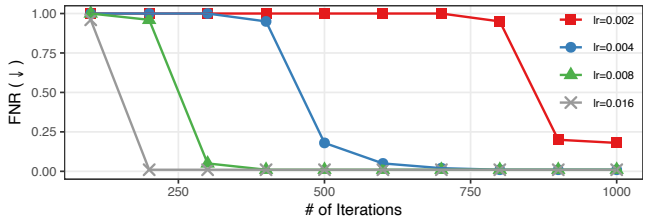
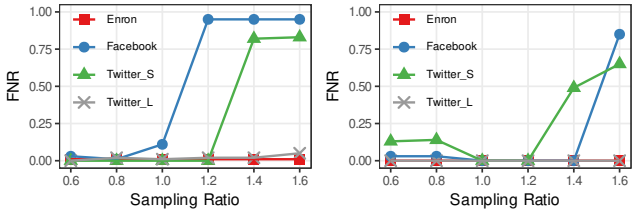


Figure 8: FNRs of ENM attacks at every 100 RICC iterations. We measured the FNRs after deploying RICC with four learning rate values on the Enron graph.



(a) FNRs against ENM attacks. (b) FNRs against NNI attacks.

Figure 9: FNRs measured against ENM and NNI attacks while varying the sampling ratio.

Sampling size N. At each iteration, we set a randomly sampled training set to have the same size as the original training set (recall §5.1.2). We now observe the target detection performance change while varying the sampling size N used for sampling a new training set T' at each iteration. Increasing the sampling size implies that many nodes are labeled, and therefore many nodes are assigned with the largest positive or negative prior scores. Considering that target nodes are densely connected to benign nodes rather than Sybil nodes, the posterior scores of those target nodes would become close to the negative end. As a result, even a randomly sampled training set would suffer from computing robust posterior scores. RICC thus obtained limited robustness in Sybil detection. Figure 9 illustrates the results. The sampling ratio in the figures refers to the ratio of the sampling size to the original training set size. The figure demonstrates that the FNR indeed increases as RICC samples more nodes. In particular, Facebook and Twitter_S were more sensitive to these changes due to their small graph sizes.

Buffer ratio ρ. Recall from §4.1 that aggregating scores from neighbor nodes without a buffer ratio results in fluctuating posterior

Table 4: Sybil node detection performance of two detection methods: RICC and ProGNN. We marked values in bold when a detection method showed superior performance.

Attack	Dataset	FNR (↓)		AUC (↑)	
		RICC	ProGNN	RICC	ProGNN
ENM [25]	Facebook	$\mathbf{0.11_{\pm.02}}$	0.55	$\mathbf{0.9995_{\pm.0001}}$	0.9984
	Twitter_S	$\mathbf{0.00_{\pm.00}}$	**0.00**	$0.8911_{\pm.0022}$	**0.9763**
NNI [31]	Facebook	$\mathbf{0.00_{\pm.00}}$	0.02	$\mathbf{0.9999_{\pm.0607}}$	0.9995
	Twitter_S	$\mathbf{0.00_{\pm.00}}$	0.04	$0.6771_{\pm.0016}$	**0.8954**

scores; we thus introduced ρ to mitigate such fluctuations. For example, without this buffer ratio, RICC achieves high FNRs of 0.31, 0.95, 0.85, and 0.03 for the Enron, Facebook, Twitter_S, and Twitter_L graphs manipulated by the ENM adversary, respectively. However, when RICC scales down the impact of scores aggregated from neighbor nodes using a ρ value of 0.8, RICC exhibited robust detection performance. Furthermore, this detection performance was stable across different buffer ratio values. For instance, when we set ρ as 0.6, which is smaller than the one we used in §5, RICC still showed outstanding FNRs of 0.01, 0.01, 0.00, and 0.00 for the same graphs, respectively.

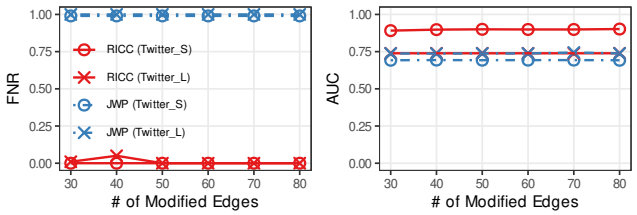
A.3 Comparison to Other Approaches

We compare RICC against ProGNN [17], a seminal defense framework against GNN adversarial attacks. Note that GNNs are not scalable to large graphs [25], requiring high-performing GPUs. Due to a lack of such computation resources, we were unable to evaluate ProGNN against the Enron and Twitter_L graphs.

Table 4 summarizes the comparison results. In general, ProGNN exhibited target node detection performance comparable to that of RICC. However, ProGNN was relatively weak against the Facebook graph manipulated by ENM. Since the clean Facebook graph already has a high node degree, it does not meet the underlying assumption of ProGNN that clean graphs are sparsely connected; therefore, ProGNN missed many target nodes in this case. Recall from §5.3 that the NNI adversary adds 1.4 times more edges compared to the ENM adversary. Accordingly, ProGNN was capable of removing densely connected edges and conducting robust classification against the Facebook graph attacked by the NNI adversary.

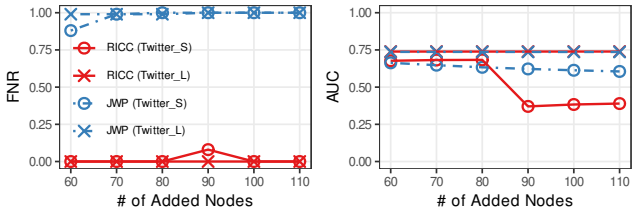
When it comes to the AUC scores measured on the Twitter_S graph, ProGNN significantly outperformed RICC. We found that this high AUC of ProGNN stems from using a GNN model. ProGNN remediates the manipulated graph before training a GNN model for robust Sybil detection (see §6); however, we observed that even GNN models directly trained using the manipulated graph also yields a surpassing AUC score compared to RICC, which is a collective classification-based tool. We believe that GNN models are able to perform more precise optimization as they are equipped with an enormous number of parameters. For example, the GNN model employed by ProGNN for Twitter_S has 130K+ parameters. In contrast, RICC conducts optimization on only 8K+ parameters (i.e., the number of prior scores).

We note that ProGNN is impractical for detecting Sybil accounts on a large graph. For instance, to deploy ProGNN on the Enron graph, ProGNN needs to perform optimization on the adjacency matrix that has 4.5B+ parameters in addition to optimization on the GNN model with 1M+ parameters. That is, for a large graph,



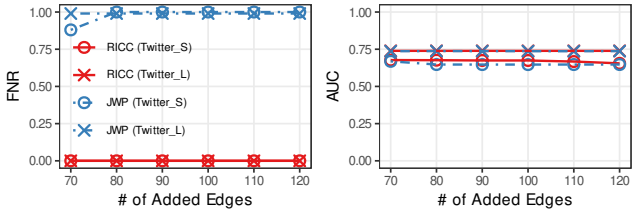
(a) FNR (↓) (b) AUC (↑)

Figure 10: Sybil detection performance against ENM attacks while varying the number of modified edges.



(a) FNR (↓) (b) AUC (↑)

Figure 11: Sybil detection performance against NNI attacks while varying the number of added nodes.



(a) FNR (↓) (b) AUC (↑)

Figure 12: Sybil detection performance against NNI attacks while varying the number of added edges.

ProGNN demands a high-performing resource that supports such massive computation. We further confirmed that ProGNN takes a much longer time compared to RICC for Sybil detection. In particular, ProGNN took 4.9 hours on our machine to detect Sybil nodes in the Twitter_S graph, which only has 8K+ nodes. On the other hand, RICC took only 0.4 hours for the same task. Moreover, we demonstrated the scalability of RICC on the large graphs (i.e., Enron and Twitter_L) in Table 1.

A.4 Effect of the Attack Budget on Twitter

Figures 10–12 summarize the effect of the attack budgets on Twitter_S and Twitter_L. The figure shows that JWP exhibited high FNRs, demonstrating its susceptibility against state-of-the-art adversarial attacks [25, 31] in all cases, whereas RICC performed robust Sybil detection in general. Interestingly, we observed that the results of Twitter_L and Twitter_S are analogous to those of Enron and Facebook, respectively. For instance, RICC was robust against the Twitter_L graph, regardless of the attack budgets; when we inserted many nodes in the Twitter_S graph through NNI attacks, the AUC score of RICC decreased. Recall from Figure 4b that RICC also exhibited a similar result against the Facebook graph. In summary, RICC achieved more robust detection performance on large graphs than those of JWP and SybilSCAR.