Semi-Supervised Social Bot Detection with Initial Residual Relation Attention Networks

Ming Zhou, Wenzheng Feng, Yifan Zhu, Dan Zhang, Yuxiao Dong, and Jie Tang(✉)

Tsinghua University, Beijing, China
zhou-m19@mails.tsinghua.edu.cn, {zhuyifan,yuxiaod, jietang}@tsinghua.edu.cn, zd18@tsinghua.org.cn

Abstract. Social bot detection is a challenging task and receives extensive attention in social security. Previous researches for this task often assume the labeled samples are abundant, which neglects the fact that labels of social bots are usually hard to derive from the real world. Meanwhile, graph neural networks (GNNs) have recently been applied to bot detection. Whereas most GNNs are based on the homophily assumption, where nodes of the same type are more likely to connect to each other. So methods relying on these two assumptions will degrade while encountering graphs with heterophily or lack of labeled data. To solve these challenges above, we analyze human-bot networks and propose SIRAN, which combines *relation attention* with *initial residual connection* to reduce and prevent the noise aggregated from neighbors to improve the capability of distinguishing different kinds of nodes on social graphs with heterophily. Then we use a consistency loss to boost the detection performance of the model for limited annotated data. Extensive experiments on two publicly available and independent social bot detection datasets illustrate SIRAN achieves state-of-the-art performance. Finally, further studies demonstrate the effectiveness of our model as well. We have deployed SIRAN online: https://botdetection.aminer.cn/robotmain.

Keywords: social networks · social bot detection · heterophily-aware attention · semi-supervised learning

1 Introduction

Social bots are social media accounts controlled by automated programs. As social media has become a primary source of information for people around the world, malicious bots have posed a great threat to social security by spreading false information and inciting public opinion warfare on social media platforms [9]. For example, social bots are used to spread misinformation during the COVID-19 pandemic [18,31] and mislead about the reality of the Russia-Ukraine war [19,29,33]. In recent years, bots are also believed to have a significant impact on the outcomes of national events. For example, during the 2010 midterm election in the United States, social bots were used to attack candidates and spread

G. De Francisci Morales et al. (Eds.): ECML PKDD 2023, LNAI 14174, pp. 207–224, 2023.
https://doi.org/10.1007/978-3-031-43427-3_13

fake news to disrupt the election. Similar activities also appeared in the 2016 US presidential election, the 2018 US midterm election [40], and the 2017 French presidential election [15].

In view of the above risks, effective and reliable social bot detection methods are urgently needed to detect social bots' activities in advance and further protect social security. Traditional social bot detection methods generally adopt feature engineering. Specifically, they rely on statistical methods and expert knowledge to construct specific features based on user profile information and tweet content. However, these approaches suffer from limited scalability. Because of the excellent performance achieved by deep learning, based on it, more and more social bot detection tools have been proposed. For example, long short-term memory (LSTM) is adopted to extract the temporal features of user social activities [28] and model both tweet content and metadata to detect bots [23]. Recently, graph neural networks (GNNs) are used to leverage the relationship information of social networks and achieve leading performance [2].

However, there are still two challenges. Challenge 1 is that **traditional GNNs-based social bot detection methods cannot effectively deal with heterophilous graphs**. Through our research, we have found that there is strong heterophily in social graphs, where different kinds of nodes are more likely to establish connections with each other, which is known as "opposites attract". Most GNNs that are under the implicit homophily assumption will degrade when they encounter social graphs with low homophily [26]. Challenge 2 is that **labeled data usually cannot meet the training needs of traditional supervised bot detection models**. As social bots evolve [9], more and more labeled data are needed, so supervised GNNs-based algorithms also suffer from the high cost of data annotation, which leads to the lack of labeled data.

In this work, we propose **S**emi-supervised **I**nitial residual **R**elation **A**ttetion **N**etworks (**SIRAN**). For challenge 1, to capture heterophilous information in social graphs, SIRAN adopts heterophily-aware relation attention and initial residual connection. Specifically, for each node, it can aggregate neighbor information and reduce the impact of heterophilous noise. For challenge 2, SIRAN leverages a confidence-aware consistency loss [13,14] to train the model to achieve high accuracy using only a small amount of annotated data. Finally, the consistency loss adopted by SIRAN can also be used to generalize other social bot detection models to reduce the dependence on labeled data for semi-supervised learning.

The main contributions are summarized as follows:

- We analyze social human-bot data and find significant differences in the distribution of social relationships between humans and bots as well as strong heterophily in social human-bot graphs. These findings give guidance to further study on social bot detection.
- We propose SIRAN, a semi-supervised bot detection framework, which adopts relation attention and initial residual connection to reduce heterophilous noise and thus enhance node representations.

– Extensive experiments on two public and independent datasets demonstrate that SIRAN consistently outperforms state-of-the-art baselines. Further ablation and robustness studies are presented to prove the effectiveness of SIRAN's every component and its robustness.

2 Related Work

Social Bot Detection. Social bot detection methods can be divided into three categories: crowdsourcing, machine learning, and graph-based approaches [3]. Earlier works adopt crowdsourcing and traditional machine learning based on feature engineering. For example, Chu et al. [7] study a set of large-scale social accounts' information, including tweeting behavior, tweet content, and account properties to detect social bots. Because of the quick evolution of social bots, feature engineering and crowdsourcing cannot effectively detect bots due to poor scalability and high costs. Recently, graph neural networks achieve state-of-the-art performance in the bot detection field. Ali Alhosseini et al. [2] propose a detection model based on graph convolutional networks (GCN). Feng et al. [10] propose a model based on heterogeneous information. However, such supervised graph-based methods require a large amount of labeled data for training, which cannot be satisfied because of the high cost of data annotation.

Graph Neural Networks. Graph neural networks (GNNs) have received extensive attention in recent years by exploiting relational information to gain performance improvements on many graph-based tasks, such as fraud detection and anomaly detection in social networks. Recent studies [37,42] have classified the existing GNNs into the following categories: recurrent graph neural networks [16,24], convolutional graph neural networks [5,36], graph autoencoders [4,32], and spatial-temporal graph neural networks [17,39]. Most of them follow the homophilous assumption that nodes of the same or similar category are more likely to establish links to each other. However, this assumption is broken in heterophilous networks, which degrades the performance of GNNs, such as human-bot networks and e-commerce networks [41].

Graph Neural Networks with Heterophily. Recently, heterophilous graph learning is becoming an important research direction of GNNs, because heterophily is widespread in the real world, and graph learning with heterophily is still an open and challenging problem. One of the current effective methods is the inter-layer combination method, which adopts layer-wise operations to improve the representation ability of GNNs under low homophily [41]. JKNet adopts this mechanism firstly, which uses jump connections and an adaptive aggregation technique to gain stronger representation learning capability [38]. H2GCN concatenates the node representations of each layer with those of all previous layers together [43]. However, such heterophilous methods do not consider the influence of different types of edges on representation learning and the problem of insufficient labeled data, which makes them unable to meet the needs of social bot detection.

3 Problem Definition and Preliminaries

3.1 Problem Definition

We represent a social network as a directed graph $\mathcal{G} = (\mathcal{V}, \mathcal{E})$, where $\mathcal{V} = \{v_1, v_2, ..., v_{|\mathcal{V}|}\}$ is the set of nodes (i.e., the set of social user accounts) and $\mathcal{E} \in |\mathcal{V}| \times |\mathcal{V}|$ represents the set of edges, which indicates relationships between nodes. The neighbor set of node v is represented as $\mathcal{N}(v) = \{u : (v, u) \in \mathcal{E}\}$. We use $\mathbf{A}$ to represent the adjacency matrix and $\mathbf{D}$ to represent the diagonal degree matrix. Let $\mathbf{X} \in \mathbb{R}^{|\mathcal{V}| \times d}$ represent node features, where the i-th row $\mathbf{X}_i \in \mathbb{R}^F$ is the feature vector of node i with F denoting its dimension.

The goal of social bot detection is to detect whether a given social account is a social bot, which can be viewed as a node binary classification problem. More formally, let $\mathbf{Y} \in \{0, 1\}^{|\mathcal{V}|}$ denote the nodes' label vector, where $\mathbf{Y}_i \in \{0, 1\}$ represents the ground truth of node i. If it is a social bot, then $\mathbf{Y}_i = 1$, otherwise $\mathbf{Y}_i = 0$. Then our goal is to learn a function:

$$f : (\mathcal{G}, \mathbf{X}) \longrightarrow \mathbf{Y}. \tag{1}$$

3.2 Homophily and Heterophily

Homophily. In a homophilous graph, nodes with similar features or the same class labels are tend to be linked together. For instance, a study usually cites papers from the same or similar research area [8].

Heterophily. In a heterophilous graph, nodes with dissimilar features and different class labels are tend to be linked together. For example, bots are more likely to follow humans rather than other bots in social networks.

Measure of Heterophily and Homophily. The homophility of a graph can be measured by the edge homophily ratio [43]: $\mathcal{H}_{edge} = \frac{|\{(v,u):(v,u)\in\mathcal{E}\wedge y_v = y_u\}|}{|\mathcal{E}|}$, where $\mathcal{H}_{edge}$ is the proportion of edges connecting nodes of the same category and $\mathcal{H}_{edge} \in [0, 1]$. $\mathcal{H}_{edge} \to 0$ means the graph has strong heterophily.

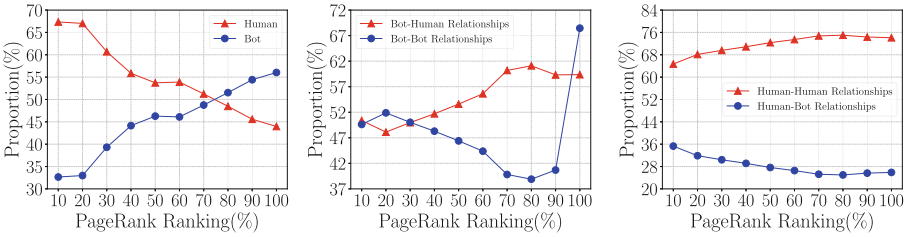
4 Human-Bot Network Analysis

In this section, we give a detailed analysis of the human-bot graph (HBG) built by TwiBot-20 [12] as a typical social bot scenario. Particularly, we focus on social relationship analysis (RA).

Social Influence Analysis. Influence plays an important role in people's social activities, because users may change their social activities due to the influence of friends [20,35]. In the social network built by TwiBot-20, We carry out the following analysis to further explore the distribution of human and bot influence.

We use the PageRank algorithm [27] to quantify the social influence of humans and bots. Based on social accounts' pagerank value, we can rank all nodes in HBG and analyze the proportion of humans and bots under different

influence rankings (represented by pagerank rankings). The results of the analysis are shown in Fig. 1a. It is obvious that there is a significant difference between humans' and bots' distribution under different influence rankings. Specifically, the proportion of the humans is higher than the bots' in the top 10%–70% and the proportion of bots is higher than humans' in the top 80%–100%.



(a) Influence Analysis (b) BRA (c) HRA

Fig. 1. Social relationship analysis. Y-axis: the proportion of humans and bots; X-axis: social accounts' influence ranking (i.e., pagerank). (a) Humans' and bots' influence comparison; (b) Bots' relationship analysis (BRA); (c) Humans' relationship analysis (HRA).

Overall Relationship Analysis of Humans and Bots. In social networks, bots spread misinformation through interactions between users, so social interaction information is very important for detection models to distinguish between humans and bots. We summarize the interaction information between humans and bots to obtain Table 1. And from it, we find out that bots like to follow humans more than to follow bots themselves (i.e., heterophily). However, humans prefer to follow humans that have the same category as themselves (i.e., homophily). Besides, from the "Total" column in Table 1, it can be observed that bots tend to construct more interactive relationships than humans. So the heterophily of HBG is mainly contributed by bots. Furthermore, through quantitative analysis, the edge homophily ratio $\mathcal{H}_{edge} = 0.5316$ (see Sect. 3.2), which confirms the above observation, i.e., there is strong heterophily in HBG.

Table 1. Humans' and bots' interactive statistics

Category	Bot[b]	Human[b]	Total
Bot[a]	4182(40.64%)	**6109(59.36%)**	**10291(62.09%)**
Human[a]	1627(25.90%)	**4655(74.10%)**	6282(37.91%)

[a]The category of the source node.
[b]The category of the destination node.

Relationship Analysis of Humans and Bots w.r.t. Pagerank. In order to further explore the differences between humans' and bots' relational distribution, we conduct a detailed analysis of their relational distribution with respect to social influence. Let the influence ranking (represented by pagerank ranking) be the horizontal axis, and the vertical axis be the proportion of different social relationships, we can get a visual description of the analysis results, which is as follows:

- From Fig. 1b, we can observe that bots at both ends of the influence ranking distribution are more likely to follow bots, and bots in the middle of the influence distribution are more likely to follow humans.
- For humans, it is learned from Fig. 1c that humans always like to follow humans and humans' relational distribution is more stable than bots' under different influence rankings.

According to the relationship analysis above, there are significant differences in the distribution of social relationships between humans and bots, which inspires us to introduce relationship information into our model to improve the ability to detect bots.

5 SIRAN

5.1 Overview of SIRAN

In order to solve the problem of model performance degradation caused by heterophilous noise and lack of annotated data, we propose **S**emi-supervised **I**nitial residual **R**elation **A**ttetion **N**etworks (**SIRAN**), which is illustrated in Fig. 2.

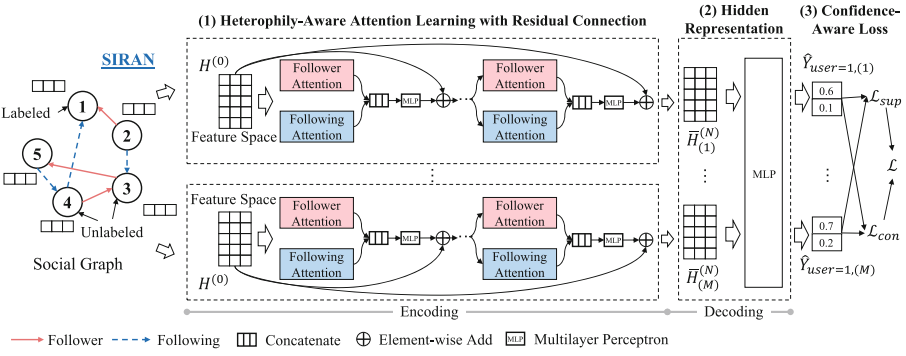


Fig. 2. Illustration of SIRAN. (1) SIRAN adopts relation attention and initial residual connection to get node representations, and it also uses dropout to get diversity. (2) Through N SIRAN layers, an enhanced feature matrix $\overline{\mathbf{H}}_{(m)}^{(N)}$ can be obtained, where $m \in [1, M]$ indicates the m-th channel. (3) At last, the enhanced feature matrix is processed by MLP to conduct semi-supervised training, which leverages the confidence-aware consistency loss $\mathcal{L}_{con}$ and the supervised loss $\mathcal{L}_{sup}$.

5.2 Heterophily-Aware Attention Mechanism

In order to make full use of social relationship information, we use a heterophily-aware relation attention mechanism to fuse relationship information into node representations to improve the ability to distinguish between bots and humans.

Although the social network could be considered as a bidirectional or unidirectional graph with two self-reciprocal kinds of relationships, following and follower, the semantic impact of such two relations may be largely different. Thus the proposed model regards the social network as a heterogeneous graph with two different types of relations. For node i, we can learn two different node representations based on following and follower relationships. Here we adopt a multi-head attention mechanism in the model to obtain node representations. Specifically, given the n-th layer node features $\mathbf{H}^{(n)} = \left\{\mathbf{h}_1^{(n)}, \mathbf{h}_2^{(n)}, \ldots, \mathbf{h}_{|\mathcal{V}|}^{(n)}\right\}$. Query, key, and value from node j to node i for the h-th attention head in the n-th layer with r_f relationship (where $f = 1$ is the following relationship and $f = 2$ is the follower relationship.) can be calculated respectively:

$$\begin{aligned}
\mathbf{q}_{h,i}^{(n),r_f} &= \mathbf{W}_{h,q}^{(n),r_f}\mathbf{h}_i^{(n)} + b_{h,q}^{(n),r_f}, \\
\mathbf{k}_{h,j}^{(n),r_f} &= \mathbf{W}_{h,k}^{(n),r_f}\mathbf{h}_j^{(n)} + b_{h,k}^{(n),r_f}, \\
\mathbf{v}_{h,j}^{(n),r_f} &= \mathbf{W}_{h,v}^{(n),r_f}\mathbf{h}_j^{(n)} + b_{h,v}^{(n),r_f},
\end{aligned} \tag{2}$$

where $\mathbf{W}$ and b are learnable parameters. Then the attention weight calculated by Eq. (3) describes the degree of concern from node i to node j, which models the heterophily in social graphs.

$$\alpha_{h,ij}^{(n),r_f} = \frac{\left\langle \mathbf{q}_{h,i}^{(n),r_f}, \mathbf{k}_{h,j}^{(n),r_f} \right\rangle}{\sum_{u\in\mathcal{N}(i)} \left\langle \mathbf{q}_{h,i}^{(n),r_f}, \mathbf{k}_{h,u}^{(n),r_f} \right\rangle}, \tag{3}$$

where $\langle \mathbf{q}, \mathbf{k} \rangle = exp\left(\frac{\mathbf{q}^T\mathbf{k}}{\sqrt{d}}\right)$ and d is the hidden dimension of each head. $\mathcal{N}(i)$ denotes the set of neighbors of node i. After having q, k, and v, we can aggregate the information from node j to node i to get the n-th layer's node feature matrix:

$$\mathbf{z}_i^{(n),r_f} = \frac{1}{H}\sum_{h=0}^{H-1}\left(\sum_{j\in\mathcal{N}(i)} \alpha_{h,ij}^{(n),r_f}\mathbf{v}_{h,j}^{(n),r_f}\right), \tag{4}$$

where H is the number of attention heads. $\mathbf{z}_i^{(n),r_f}$ indicates the node i's representation in the n-th layer with relationship r_f. Then, for fusing relationship information, we concatenate $\mathbf{z}_i^{(n),r_1}$ and $\mathbf{z}_i^{(n),r_2}$ and put it into MLP to obtain $\mathbf{z}_i^{(n)} = \text{MLP}\left(\left[\mathbf{z}_i^{(n),r_1} \,||\, \mathbf{z}_i^{(n),r_2}\right], \mathbf{\Theta}\right)$, where $\mathbf{\Theta}$ is the hyperparameter in MLP. Finally, by iterating all nodes in the graph, $\mathbf{Z}^{(n)} = \left\{\mathbf{z}_i^{(n)} : 1 \leqslant i \leqslant |\mathcal{V}|\right\}$ can be obtained. The overall flow is shown in Fig. 3.

Based on the above, the n-th ($n \geq 1$) layer of relation attention module can be defined as

$$\mathbf{H}^{(n+1)} = \sigma\left(\widetilde{\mathbf{P}}\mathbf{Z}^{(n)}\mathbf{W}^{(n)}\right), \tag{5}$$

where σ is the activation function, here we use the ReLU operation. $\widetilde{\mathbf{P}} = \widetilde{\mathbf{D}}^{-\frac{1}{2}}\widetilde{\mathbf{A}}\widetilde{\mathbf{D}}^{-\frac{1}{2}} = (\mathbf{D}+\mathbf{I})^{-\frac{1}{2}}(\mathbf{A}+\mathbf{I})(\mathbf{D}+\mathbf{I})^{-\frac{1}{2}}$ is the convolved signal matrix with the *renormalization trick* [22]. $\mathbf{W}^{(n)}$ is the weight matrix.

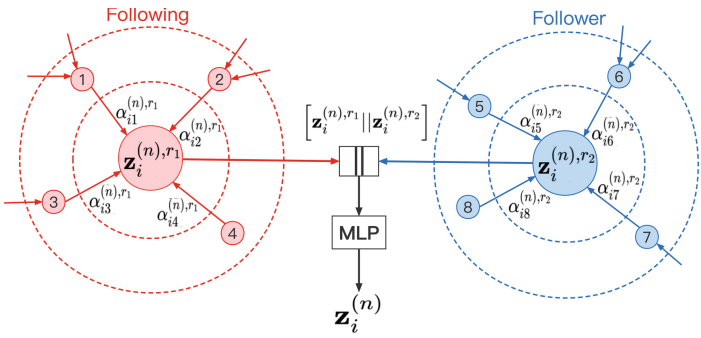


Fig. 3. Illustration of heterophily-aware relation attention. (1) We aggregate respectively following neighbors' and follower neighbors' features into node i with the attention mechanism. $\mathbf{z}_i^{(n),r_1}$ (following relationship) and $\mathbf{z}_i^{(n),r_2}$ (follower relationship) can be obtained, where n denotes the n-th layer. (2) Then, concatenate $\mathbf{z}_i^{(n),r_1}$ and $\mathbf{z}_i^{(n),r_2}$ to get $\left[\mathbf{z}_i^{(n),r_1} \,||\, \mathbf{z}_i^{(n),r_2}\right]$. (3) Finally, put $\left[\mathbf{z}_i^{(n),r_1} \,||\, \mathbf{z}_i^{(n),r_2}\right]$ into MLP to fuse following and follower information.

5.3 Initial Residual Connection

For the heterophilous graphs, with GNNs going deeper, more information from different kinds of nodes is aggregated into node representations. That makes models difficult to distinguish different kinds of nodes. So, reducing the noise from different kinds of neighbors is the key point to improving the representation learning ability. Initial residual connection is an effective method to reduce the noise aggregated from neighbors. Meanwhile, it can enhance node representations by ensuring that the initial node features, which contain important prior information, act on node representations at each layer. Through importing the initial residual connection, the n-th layer of SIRAN can be defined as

$$\mathbf{H}^{(n+1)} = \sigma\left(\left((1-\alpha_n)\widetilde{\mathbf{P}}\mathbf{Z}^{(n)} + \alpha_n\mathbf{H}^{(0)}\right)\mathbf{W}^{(n)}\right), \tag{6}$$

where $\alpha_n \in [0,1]$ is a hyperparameter. $\mathbf{H}^{(0)}$ is the initial representation, which may not be equal to the input feature matrix $\mathbf{X}$, e.g., if the dimension of $\mathbf{X}$ is large, MLP can be used to reduce its dimension and get a low-dimensional $\mathbf{H}^{(0)}$.

5.4 Training and Optimization

SIRAN adopts both **confidence-aware consistency loss** and **supervised loss** to optimize model parameters in the training stage.

Supervised Loss. The supervised loss is defined as the average cross entropy of labeled nodes sampled over M channels:

$$\mathcal{L}_{sup} = -\frac{1}{|\mathbf{L}|\, M} \sum_{i \in L} \sum_{m=1}^{M} \mathbf{Y}_i \log \left(\hat{\mathbf{Y}}_{i,(m)} \right), \tag{7}$$

where $\mathbf{L}$ represents the set of labeled training data. $\mathbf{Y}_i$ is the ground-truth label of node i in the labeled data set. $\widehat{\mathbf{Y}}_{i,(m)} \in \mathbb{R}^{\mathbb{C}}, m \in [1, M]$ denotes the probability vector of the predicted category of node i on the m-th channel.

Confidence-Aware Consistency Loss. Due to the long-standing challenge of lack of annotated data, we employ the confidence-aware consistency loss for semi-supervised learning to solve it. Specifically, according to the procedure in Fig. 2, the enhanced features are obtained through M-channel representation learning with dropout. Then we feed them into MLP to get outputs: $\widehat{\mathbf{Y}}_{i,(m)} = \text{MLP}\left(\mathbf{h}_{i,(m)}, \mathbf{\Theta}\right)$, where $\mathbf{\Theta}$ is the hyperparameter in MLP, and $\mathbf{h}_{i,(m)}$ is the enhanced feature matrix of node i on the m-th channel.

For semi-supervised learning, we sample unlabeled data and denote it as $\mathbf{U}$. The center of the distribution of $\mathbf{U}$ can be obtained by averaging the predicted probabilities over M channels, i.e., $\overline{\mathbf{Y}}_i = \frac{1}{M} \sum_{m=1}^{M} \widehat{\mathbf{Y}}_{i,(m)}$. The confidence-aware consistency loss is defined as:

$$\mathcal{L}_{con} = \frac{1}{|\mathbf{U}|\, M} \sum_{i \in \mathbf{U}} \delta\left(\overline{\mathbf{Y}}_i\right) \sum_{m=1}^{M} D\left(\overline{\mathbf{Y}}_i, \widehat{\mathbf{Y}}_{i,(m)}\right), \tag{8}$$

where $\mathbf{U}$ is the set of unlabeled data. $\delta\left(\overline{\mathbf{Y}}_i\right) = \begin{cases} 1, & max\left(\overline{\mathbf{Y}}_i\right) \geqslant \gamma \\ 0, & otherwise \end{cases}, \gamma \in [0, 1]$ is the confidence threshold. $D\left(\overline{\mathbf{Y}}_i, \widehat{\mathbf{Y}}_{i,(m)}\right)$ is a distance function that measures the distribution discrepancy between $\overline{\mathbf{Y}}_i$ and $\widehat{\mathbf{Y}}_{i,(m)}$. Here the distance function mainly includes $L2$ norm and KL divergence.

From Eq. (8), $\mathcal{L}_{con}$ only considers **highly confident** unlabeled data selected by the threshold γ in the training stage, it is the reason of being called confidence-aware consistency loss. It can filter out the training noise brought by heterophily to improve the prediction performance.

Finally, by combining $\mathcal{L}_{sup}$ with $\mathcal{L}_{con}$, we can obtain the final loss defined as $\mathcal{L} = \mathcal{L}_{sup} + \lambda(t)\, \mathcal{L}_{con}$, where $\lambda(t)$ is a decay function which decreases in the range of $[0, \lambda_{max}]$, and usually $\lambda_{max} = 2$. The whole training procedure is shown in Algorithm 1.

6 Experiments

6.1 Experimental Setup

Table 2. Dataset statistics.

Dataset	Nodes	Human	Bot	Edges	Classes	$\mathcal{H}_{edge}$
TwiBot-20	229,580	5,237	6,589	33,716,171	2	0.5316
TwiBot-22	162,865	81,433	81,432	151,841	2	0.4963

Datasets. To verify the performance of the model in the heterophilous graphs, We evaluate our model on two public and independent datasets, namely TwiBot-20 [12] and TwiBot-22 [11], whose statistics are shown in Table 2. For TwiBot-20, we follow the same data setup as in [12]. For TwiBot-22, we randomly sample 81, 432 bots as negative examples and 81, 433 humans as positive examples to ensure that their proportions are relatively balanced, resulting in a total of 162, 865 social accounts. To ensure a fair comparative experiment, we randomly split the sampled dataset 7 : 2 : 1 to obtain training, validation, and test sets, respectively.

Comparing Baselines. We compare SIRAN and its three variants with 9 baselines, including 4 general GNNs, 4 non-homophilous methods, and 1 bot detection method with heterogeneity. All the experiments use the same input features, including users' (1) attributes: username, location, verified, registration time, description, tweet count, listed count, follower count, following count; (2) tweet content; (3) social relationships: list of following and follower friends. The full list of baseline methods is: **Four general GNNs:** GAT [36], GCN [22], JKNet (GCNJK) [38], R-GCN [30]. **Four non-homophilous methods:** MixHop [1], LINKX [25], H2GCN [43], GPR-GNN [6]. **One heterogeneous bot detection method:** Feng et al. [10]. **Three variants of SIRAN:** (1) SIRAN+PLR (Add the previous layer's residual connection), (2) SIRANJK (Combine the jumping knowledge network with SIRAN), (3) SIRAN-CONCAT (Concatenation is used instead of matrix addition to combine the representation of each layer with the initial node features).

Implementation Details. Due to the interactive characteristics of social networks, we build the human-bot networks as directed graphs. For experimental optimization, the AdamW optimizer [21] is used with weight decay 3×10^{-5}. Learning rate is 10^{-2} and 10^{-3} on TwiBot-20 and TwiBot-22, respectively. To avoid overfitting, early stopping and Dropout [34] are used for model training. We use grid search to adjust hyperparameters of SIRAN on the validation set, and use the best configuration for prediction. Specifically, on TwiBot-20 and

Algorithm 1: SIRAN

input : Social human-bot dataset S
output: Optimized model parameters $\mathbf{\Theta}$

Initialize $\mathbf{\Theta}$;
Preproccess S *to build graph* $\mathcal{G}(\mathcal{V}, \mathcal{E})$, *relation set* $r_f \subseteq \mathcal{E}, f = 1\,(following)\,or 2\,(follower)$, *feature matrix* $\mathbf{X} \in \mathbb{R}^{|\mathcal{V}| \times d}$, *labeled node set* $\mathbf{L}$, *unlabeled node set* $\mathbf{U}$, *ground-truth label set* $\mathbf{Y}_L \in \mathbb{R}^{|\mathbf{L}| \times C}$;
Function `Pipeline(`$\mathcal{G}$, $\mathbf{X}$`):`
  for *Node* i *in* $\mathcal{G}$ **do**
    for $h : 0 \rightarrow H - 1$ **do**
      $\alpha_{h,ij}^{(n),r_f} \leftarrow$ Eq. (3)
    $\mathbf{z}_i^{(n),r_f}, \mathbf{z}_i^{(n)} \leftarrow$ Eq. (4)
    $\mathbf{Z}^{(n)} = \left\{ \mathbf{z}_i^{(n)} : 1 \leqslant i \leqslant |\mathcal{V}| \right\}$
  $\mathbf{H}^{(n+1)} \leftarrow$ Eq. (6)
  `// Through N layers SIRAN`
  $\mathbf{H}^{(N)} = \left\{ \mathbf{H}^{(n)} : 0 \leqslant n \leqslant N - 1 \right\}$
  return $\mathbf{H}^{(N)}$

while $\mathbf{\Theta}$ *does not converge OR* $t : 0 \rightarrow T$ **do**
  for $m : 1 \rightarrow M$ **do**
    `// Parallel M-channel processing`
    $\mathbf{H}_{(m)}^{(N)} =$ Pipeline($\mathcal{G}$, $\mathbf{X}$)
  $\mathcal{L}_{sup}, \mathcal{L}_{con}, \mathcal{L} \leftarrow$ Eq. (7 - 8)
  $\mathbf{\Theta} \leftarrow BackPropagate\,(Loss)$
Return $\mathbf{\Theta}$

TwiBot-22, the model adopts three hidden layers with 128 hidden size and 8 attention heads over three encoding channels, and the distance function and confident threshold are set to L2 norm and 0.7, respectively. The initial residual weight α is 0.5 and 0.9 on TwiBot-20 and TwiBot-22, respectively. The model configurations of other baselines follow previous works [10,25,30]. Accuracy, F1-score, and ROC-AUC are used to evaluate our model and baselines.

6.2 Overall Results

Table 3 shows the experimental results comparing SIRAN and its three variants with 9 baselines on the test set. We run each experiment 5 times with random weight initializations and report the mean values with standard deviation.

Table 3. Overall performance comparison. **Bold** and underline represent the best and runner-up performance, respectively.

Datasets	TwiBot-20			TwiBot-22		
Method	**Accuracy**	**F1-Score**	**ROC-AUC**	**Accuracy**	**F1-Score**	**ROC-AUC**
GAT	72.89 ± 0.88	79.13 ± 0.33	71.61 ± 1.27	59.66 ± 1.64	67.68 ± 0.15	59.51 ± 1.69
GCN	74.17 ± 0.44	79.91 ± 0.17	72.53 ± 0.57	62.16 ± 0.46	67.99 ± 0.61	62.13 ± 0.51
GCNJK	75.09 ± 0.61	80.63 ± 0.56	73.44 ± 0.46	62.96 ± 0.64	69.06 ± 0.32	62.91 ± 0.65
R-GCN	79.91 ± 0.42	83.68 ± 0.26	79.37 ± 0.69	63.63 ± 1.38	69.03 ± 0.51	63.56 ± 1.36
LINKX	76.35 ± 0.95	81.60 ± 0.23	74.26 ± 1.94	61.24 ± 0.87	67.60 ± 0.48	61.18 ± 0.93
MixHop	79.62 ± 0.46	83.35 ± 0.55	78.62 ± 0.70	62.41 ± 0.25	68.70 ± 1.44	62.40 ± 0.26
GPR-GNN	78.82 ± 0.49	82.89 ± 0.43	78.19 ± 0.52	61.63 ± 0.17	67.67 ± 0.19	61.48 ± 0.38
H2GCN	79.53 ± 0.29	83.10 ± 0.41	78.78 ± 0.31	61.65 ± 0.11	67.61 ± 0.18	61.61 ± 0.10
Feng et al.	76.13 ± 2.40	81.64 ± 0.96	73.55 ± 3.07	56.59 ± 0.16	67.49 ± 0.45	56.42 ± 0.15
SIRAN+PLR	80.68 ± 0.40	84.03 ± 0.29	79.98 ± 0.47	62.34 ± 1.01	69.23 ± 1.38	62.30 ± 0.99
SIRANJK	80.60 ± 0.47	83.94 ± 0.33	79.81 ± 0.57	62.95 ± 0.72	70.05 ± 0.97	62.91 ± 0.71
SIRAN-CONCAT	80.56 ± 0.61	83.85 ± 0.33	79.65 ± 0.49	65.31 ± 3.40	71.14 ± 2.49	65.26 ± 3.40
SIRAN(Ours)	**81.11 ± 0.51**	**84.25 ± 0.32**	**80.11 ± 0.54**	**65.67 ± 2.85**	**71.95 ± 2.04**	**65.59 ± 2.81**

Analysis and Discussion. From the test results, our model achieves state-of-the-art bot detection performance, and two of its variants also achieve the top-2 performance on the two datasets respectively.

Among the general GNNs, R-GCN is the most competitive approach. Based on GCN, it adds consideration of the influence of different types of edges on node representations, and its performance is improved. This illustrates the importance of edge category information for node representations. In contrast, our model not only adopts relation attention, but also uses initial residual to augment node representations to further improve the detection performance.

For non-homophilous models, they mainly consider reducing the impact of heterophilous noise to improve performance, e.g., MixHop mixes the representations of neighbors at different distances to reduce heterophilous noise. However, while using initial residual to reduce heterophilous noise, our model also employs relation attention and a consistency loss to further improve the detection performance in the absence of labels, which will be discussed in detail in Sect. 6.4.

For the social bot-oriented GNNs (i.e., [10]), our proposed SIRAN shows a significant improvement by 6.5%, which indicates the effectiveness of reducing heterophilous noise and label dependence in social networks with heterophily.

All the above observations prove that our model can effectively detect social bots by employing relation attention and reducing heterophilous noise.

6.3 Ablation Study

We conduct ablation studies on TwiBot-20 and TwiBot-22 to demonstrate the effectiveness of each component of SIRAN.

- SIRAN-withoutIR: Remove initial residual connection.
- SIRAN-PLR: Replace the initial residual with the previous layer residual.

– SIRAN-GCN: Replace transformer with GCN to verify the effectiveness of the attention mechanism.
– SIRAN-GAT: Replace transformer with GAT in the relation attention module to compare the performance difference between transformer's and GAT's attention mechanisms.
– SIRAN-withoutSR: Remove the separation of relationships (follower and following) from the attention module.
– SIRAN-withoutCL: Remove the confidence-aware consistency loss.

Table 4. F1-score of ablation experiments.

Ablation Settings	TwiBot-20	TwiBot-22
SIRAN(full model)	**84.25 ± 0.32**	**71.95 ± 2.04**
SIRAN-withoutIR	83.46 ± 0.46	67.09 ± 1.58
SIRAN-PLR	83.08 ± 0.28	67.81 ± 0.05
SIRAN-GCN	83.72 ± 0.49	70.73 ± 2.63
SIRAN-GAT	84.14 ± 0.26	71.76 ± 2.53
SIRAN-withoutSR	84.20 ± 0.21	68.34 ± 1.66
SIRAN-withoutCL	84.15 ± 0.72	68.81 ± 0.78

Table 4 shows the experimental results of the ablation study on the test set, from which we can get the following observations:

(1) Removing or replacing any component makes SIRAN degrade with comparing to the full model, so it can be concluded that each component of the model makes a contribution to the effectiveness of SIRAN.

(2) From SIRAN-withoutIR, we observe that removing the initial residual connection makes the performance drop more heavily than that of the relation attention and consistency loss. Besides, from SIRAN-PLR, replacing the initial residual with the previous layer residual also degrades the performance of the model, which is attributed to the heterophilous noise from the previous layer. So the initial residual plays a more important role.

(3) From SIRAN-GCN, we find that removing the attention module with GCN makes the model's performance degrade, and from SIRAN-GAT, for a different attention mechanism, the performance improvement of GAT's attention is less than that of the Transformer's attention.

(4) From SIRAN-withoutSR and SIRAN-withoutCL, we observe that the performance degradation of the model on TwiBot-22 (which has stronger heterophily) is larger than that on TwiBot-20, which further illustrates their important role in reducing the heterophilous noise.

6.4 Robustness Study

Due to the evolution of bots and the high cost of data annotation, the challenge of lack of annotated data would degrade the performance of most bot detection

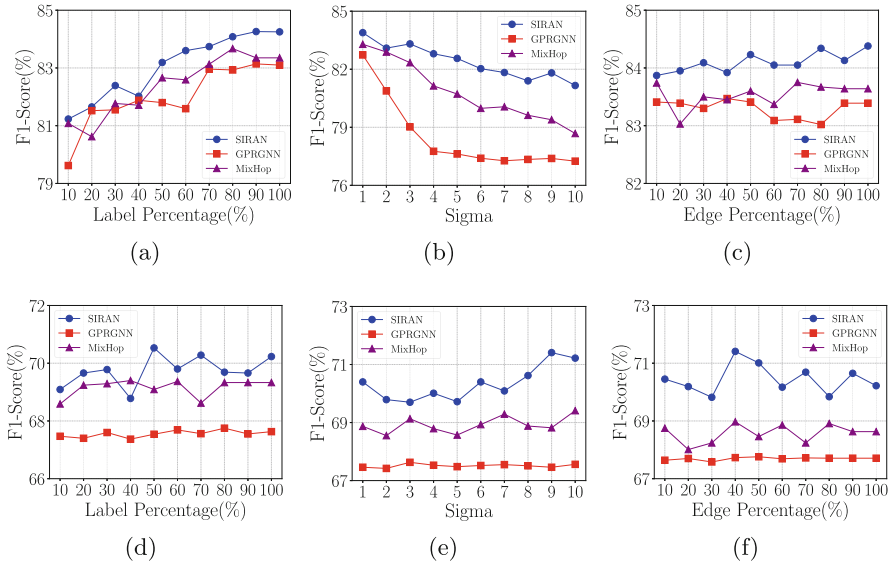


Fig. 4. Robustness experiments. (a) Label robustness, (b) feature robustness, and (c) edge robustness experiments are on TwiBot-20. (d) Label robustness, (e) feature robustness, and (f) edge robustness experiments are on TwiBot-22.

methods that rely on the quality and quantity of annotated data. In the following, we conduct robustness experiments on TwiBot-20 and TwiBot-22 to verify the robustness of our model.

Label Robustness Experiments. First, some labels in the training set are randomly masked, and the experimental results on the test set are shown in Fig. 4a and d. **Feature robustness experiments.** Second, we randomly add White Gaussian Noise (WGN, $X \sim \mathcal{N}(\mu, \sigma^2)$) to node features. The power of WGN is σ^2, the experimental results are illustrated in Fig. 4b and 4e. **Edge robustness experiments.** Third, Some edges are randomly removed to demonstrate our model's robustness of relationships, and the results are shown in Fig. 4c and 4f.

Analysis and Discussion. Based on the results, we have the following discussions: (1) From the above robustness experiments, our model outperforms the baselines under all testing settings. (2) For the label robustness experiments, on TwiBot-20 with fewer labels, the performance of our model shows a continuous growth trend as the number of labels increases, while, on TwiBot-22 with more labels, that of our model increases first, and reaches the highest value with adopting 50% of labels and then levels off. The reason could be that the semi-supervised loss function used by our model can improve the performance in the absence of labels to achieve better performance with only fewer labels. (3) For

the feature robustness experiments, our model is more robust to noise in features than the baselines. This is because the relation attention mechanism adaptively adjusts the weight of feature aggregation for the interference of noise so that the model has a stronger feature representation ability. (4) For the edge robustness experiments, on TwiBot-22 with stronger heterophily, as the number of edges increases, the performance of our model first decreases slightly, then grows from the point where 30% of the edges are adopted, and finally levels off. This is because more edges bring both more information and noise, the relation attention and the initial residual can adaptively reduce and prevent the heterophilous noise to make our model more robust, which is also the reason why our model is more stable than the baselines on TwiBot-20.

7 Conclusion and Future Work

In this work, we study the problem of social bot detection, which has two challenges: heterophily in social graphs and lack of labeled data. To solve the challenges mentioned above, we propose SIRAN, which combines *relation attention* with *initial residual connection* to reduce and prevent the heterophilous noise. Then we adopt a confidence-aware consistency loss to improve the model's generalization on unlabeled data for semi-supervised learning. Extensive experiments show that SIRAN consistently outperforms state-of-the-art baselines on two real-world datasets. Up to now, we have deployed SIRAN online to detect Twitter bots.

Limitations and Future Work. A limitation of SIRAN is that cross-platform bot detection is not supported, which is still a common challenge for social bot detection due to differences in data distribution between multiple platforms and limited data collection and annotation capabilities. We leave it for future work.

Acknowledgments. This work was supported by Natural Science Foundation of China (NSFC) 61836013, 61825602, 62276148, and China Postdoctoral Science Foundation (2022 M711814).

Ethics Statement. This work aims to propose a semi-supervised bot detection framework with heterophily, which will protect people from being disturbed by bots and ensure social security. We have deployed SIRAN online and it has been widely used. To the best of our knowledge, SIRAN currently ranks No.1 in Baidu (https://www.baidu.com/) and Google (https://www.google.com/) searches. However, as we know that *"A coin has two sides"*, bot creators can also use SIRAN to improve their bots' performance. To alleviate this problem, we have strengthened the authentication management of APIs.

References

1. Abu-El-Haija, S., et al.: MixHop: higher-order graph convolutional architectures via sparsified neighborhood mixing. In: International Conference on Machine Learning, pp. 21–29. PMLR (2019)
2. Ali Alhosseini, S., Bin Tareaf, R., Najafi, P., Meinel, C.: Detect me if you can: spam bot detection using inductive representation learning. In: Companion Proceedings of The 2019 World Wide Web Conference, pp. 148–153 (2019)
3. Alothali, E., Zaki, N., Mohamed, E.A., Alashwal, H.: Detecting social bots on twitter: a literature review. In: 2018 International Conference on Innovations in Information Technology (IIT), pp. 175–180. IEEE (2018)
4. Bojchevski, A., Shchur, O., Zügner, D., Günnemann, S.: NetGAN: generating graphs via random walks. In: International Conference on Machine Learning, pp. 610–619. PMLR (2018)
5. Chen, J., Ma, T., Xiao, C.: FastGCN: fast learning with graph convolutional networks via importance sampling. arXiv preprint arXiv:1801.10247 (2018)
6. Chien, E., Peng, J., Li, P., Milenkovic, O.: Adaptive universal generalized pagerank graph neural network. arXiv preprint arXiv:2006.07988 (2020)
7. Chu, Z., Gianvecchio, S., Wang, H., Jajodia, S.: Detecting automation of twitter accounts: are you a human, bot, or cyborg? IEEE Trans. Dependable Secure Comput. **9**(6), 811–824 (2012)
8. Ciotti, V., Bonaventura, M., Nicosia, V., Panzarasa, P., Latora, V.: Homophily and missing links in citation networks. EPJ Data Sci. **5**, 1–14 (2016)
9. Cresci, S.: A decade of social bot detection. Commun. ACM **63**(10), 72–83 (2020)
10. Feng, S., Tan, Z., Li, R., Luo, M.: Heterogeneity-aware twitter bot detection with relational graph transformers. arXiv preprint arXiv:2109.02927 (2021)
11. Feng, S., et al.: TwiBot-22: towards graph-based twitter bot detection. arXiv preprint arXiv:2206.04564 (2022)
12. Feng, S., Wan, H., Wang, N., Li, J., Luo, M.: TwiBot-20: a comprehensive twitter bot detection benchmark. In: Proceedings of the 30th ACM International Conference on Information & Knowledge Management, pp. 4485–4494 (2021)
13. Feng, W., et al.: Grand+: scalable graph random neural networks. arXiv preprint arXiv:2203.06389 (2022)
14. Feng, W., et al.: Graph random neural networks for semi-supervised learning on graphs. In: NeurIPS2020, pp. 22092–22103 (2020)
15. Ferrara, E.: Disinformation and social bot operations in the run up to the 2017 French presidential election. arXiv preprint arXiv:1707.00086 (2017)
16. Gallicchio, C., Micheli, A.: Graph echo state networks. In: The 2010 International Joint Conference on Neural Networks (IJCNN), pp. 1–8. IEEE (2010)
17. Guo, S., Lin, Y., Feng, N., Song, C., Wan, H.: Attention based spatial-temporal graph convolutional networks for traffic flow forecasting. In: Proceedings of the AAAI Conference on Artificial Intelligence, vol. 33, pp. 922–929 (2019)
18. Himelein-Wachowiak, M., et al.: Bots and misinformation spread on social media: implications for COVID-19. J. Med. Internet Res. **23**(5), e26933 (2021). https://doi.org/10.2196/26933. www.jmir.org/2021/5/e26933
19. Jarynowski, A.: Conflicts driven pandemic and war issues in social media via multilayer approach of German twitter (2022)
20. Kelman, H.C.: Compliance, identification, and internalization three processes of attitude change. J. Conflict Resolut. **2**(1), 51–60 (1958)

21. Kingma, D.P., Ba, J.: Adam: a method for stochastic optimization. arXiv preprint arXiv:1412.6980 (2014)
22. Kipf, T.N., Welling, M.: Semi-supervised classification with graph convolutional networks. arXiv preprint arXiv:1609.02907 (2016)
23. Kudugunta, S., Ferrara, E.: Deep neural networks for bot detection. Inf. Sci. **467**, 312–322 (2018)
24. Li, Y., Tarlow, D., Brockschmidt, M., Zemel, R.: Gated graph sequence neural networks. arXiv preprint arXiv:1511.05493 (2015)
25. Lim, D., et al.: Large scale learning on non-homophilous graphs: new benchmarks and strong simple methods. In: Advances in Neural Information Processing Systems 34 (2021)
26. McPherson, M., Smith-Lovin, L., Cook, J.M.: Birds of a feather: homophily in social networks. Ann. Rev. Sociol. **27**(1), 415–444 (2001)
27. Page, L., Brin, S., Motwani, R., Winograd, T.: The pagerank citation ranking: bringing order to the web. Tech. Rep, Stanford InfoLab (1999)
28. Ping, H., Qin, S.: A social bots detection model based on deep learning algorithm. In: 2018 IEEE 18th International Conference on Communication Technology (ICCT), pp. 1435–1439. IEEE (2018)
29. Purtill, J.: Twitter bot network amplifying Russian disinformation about Ukraine war, researcher says (2022). www.abc.net.au/news/science/2022-03-30/ukraine-war-twitter-bot-network-amplifies-russian-disinformation/100944970. Accessed 05 Feb 2023
30. Schlichtkrull, M., Kipf, T.N., Bloem, P., van den Berg, R., Titov, I., Welling, M.: Modeling relational data with graph convolutional networks. In: Gangemi, A., et al. (eds.) ESWC 2018. LNCS, vol. 10843, pp. 593–607. Springer, Cham (2018). https://doi.org/10.1007/978-3-319-93417-4_38
31. Shi, W., Liu, D., Yang, J., Zhang, J., Wen, S., Su, J.: Social bots' sentiment engagement in health emergencies: a topic-based analysis of the COVID-19 pandemic discussions on twitter. Int. J. Environ. Res. Public Health **17**(22), 8701 (2020)
32. Simonovsky, M., Komodakis, N.: GraphVAE: towards generation of small graphs using variational autoencoders. In: Kůrková, V., Manolopoulos, Y., Hammer, B., Iliadis, L., Maglogiannis, I. (eds.) ICANN 2018. LNCS, vol. 11139, pp. 412–422. Springer, Cham (2018). https://doi.org/10.1007/978-3-030-01418-6_41
33. Smart, B., Watt, J., Benedetti, S., Mitchell, L., Roughan, M.: # istandwithputin versus# istandwithukraine: the interaction of bots and humans in discussion of the Russia/Ukraine war. arXiv preprint arXiv:2208.07038 (2022)
34. Srivastava, N., Hinton, G., Krizhevsky, A., Sutskever, I., Salakhutdinov, R.: Dropout: a simple way to prevent neural networks from overfitting. J. Mach. Learn. Res. **15**(1), 1929–1958 (2014)
35. Tang, J., Sun, J., Wang, C., Yang, Z.: Social influence analysis in large-scale networks. In: Proceedings of the 15th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, pp. 807–816 (2009)
36. Veličković, P., Cucurull, G., Casanova, A., Romero, A., Lio, P., Bengio, Y.: Graph attention networks. arXiv preprint arXiv:1710.10903 (2017)
37. Wu, Z., Pan, S., Chen, F., Long, G., Zhang, C., Philip, S.Y.: A comprehensive survey on graph neural networks. IEEE Trans. Neural Netw. Learn. Syst. **32**(1), 4–24 (2020)
38. Xu, K., Li, C., Tian, Y., Sonobe, T., Kawarabayashi, K.i., Jegelka, S.: Representation learning on graphs with jumping knowledge networks. In: International Conference on Machine Learning, pp. 5453–5462. PMLR (2018)

39. Yan, S., Xiong, Y., Lin, D.: Spatial temporal graph convolutional networks for skeleton-based action recognition. In: Thirty-second AAAI Conference on Artificial Intelligence (2018)
40. Yang, K.C., Hui, P.M., Menczer, F.: Bot electioneering volume: visualizing social bot activity during elections. In: Companion Proceedings of The 2019 World Wide Web Conference, pp. 214–217 (2019)
41. Zheng, X., Liu, Y., Pan, S., Zhang, M., Jin, D., Yu, P.S.: Graph neural networks for graphs with heterophily: a survey. arXiv preprint arXiv:2202.07082 (2022)
42. Zhou, J., et al.: Graph neural networks: a review of methods and applications. AI Open **1**, 57–81 (2020)
43. Zhu, J., Yan, Y., Zhao, L., Heimann, M., Akoglu, L., Koutra, D.: Beyond homophily in graph neural networks: current limitations and effective designs. Adv. Neural. Inf. Process. Syst. **33**, 7793–7804 (2020)