Dispelling the Fake: Social Bot Detection Based on Edge Confidence Evaluation

Boyu Qiao, Wei Zhou, Kun Li, Shilong Li, and Songlin Hu

***Abstract*— Social bot detection is essential for maintaining the safety and integrity of online social networks (OSNs). Graph neural networks (GNNs) have emerged as a promising solution. Mainstream GNN-based social bot detection methods learn rich user representations by recursively performing message passing along user–user interaction edges, where users are treated as nodes and their relationships as edges. However, these methods face challenges when detecting advanced bots interacting with genuine accounts. Interaction with real accounts results in the graph structure containing camouflaged and unreliable edges. These unreliable edges interfere with the differentiation between bot and human representations, and the iterative graph encoding process amplifies this unreliability. In this article, we propose a social Bot detection method based on Edge Confidence Evaluation (BECE). Our model incorporates an edge confidence evaluation module that assesses the reliability of the edges and identifies the unreliable edges. Specifically, we design features for edges based on the representation of user nodes and introduce parameterized Gaussian distributions to map the edge embeddings into a latent semantic space. We optimize these embeddings by minimizing Kullback–Leibler (KL) divergence from the standard distribution and evaluate their confidence based on edge representation. Experimental results on three real-world datasets demonstrate that BECE is effective and superior in social bot detection. Additionally, experimental results on six widely used GNN architectures demonstrate that our proposed edge confidence evaluation module can be used as a plug-in to improve detection performance.**

***Index Terms*— Edge confidence evaluation, graph neural network (GNN), parameterized Gaussian distribution, social bot detection.**

NOMENCLATURE

$\mathcal{G} = \{U, E_r\|_{r=1}^{R}, Y\}$	Social network topology.
U	Node set of $\mathcal{G}$.
$E_r\|_{r=1}^{R}$	Edge set of $\mathcal{G}$.
R	Number of relationship.
$Y = \{y_1, \ldots, y_n\}$	Ground truth of node.
$V = \{v_1, \ldots, v_n\}$	Value-type information.
$B = \{b_1, \ldots, b_n\}$	Boolean-type information.
$T = \{t_1, \ldots, t_n\}$	Text-type information.
n	Number of users.
$\mathbf{u}_i^v$, $\mathbf{u}_i^b$, and $\mathbf{u}_i^t$	Value, Boolean, and text feature of u_i.
$\hat{\mathbf{u}}_i$	Node representation.
$N_i = \{u_{i,j}^{r,(l)}\}_{j=1}^{J_i}$	Neighbor nodes of user u_i.
J_i	Number of one-hop neighbors.
$\mathbf{e}_{i,j}^{r,(l)}$	Edge representation between user u_i and u_j.
$\boldsymbol{\mu}_{i,j}^{r,(l)}$	Mean of edge feature.
$\boldsymbol{\sigma}_{i,j}^{r,(l)}$	Variance of edge feature.
$\tilde{\mathbf{e}}_{i,j}^{r,(l)}$	Edge embedding of reconstruction.
$p_{i,j}^{r,(l)}$	Edge confidence probability.
$f(s_{i,j}^{r,(l)} \mid p_{i,j}^{r,(l)})$	Bernoulli approximation.
$\mathcal{L}_{\text{edge}}^{(l)}$	Edge binary cross-entropy loss.
$\mathcal{L}_{\text{KL}}^{(l)}$	KL divergence.
$\mathbf{a}_i^{r,(l+1)}$	Polymerization neighbors representation.
$\hat{\mathbf{u}}_i^{r,(l+1)}$	Updated node representation.
$\mathcal{L}_{\text{con}}$	Node cross-entropy loss.
$\mathcal{L}$	Loss of BECE.

I. INTRODUCTION

ONLINE social networks (OSNs) offer an excellent platform for human communication. However, the existence of social bots poses risks to OSN security [1], [2]. Social bots are automated programs that engage in online social activities [3], [4]. These automated bots actively participate in user conversations, becoming a significant factor in shaping public opinion [5], interfering with the dissemination of disinformation [6], and restricting the right to freedom of speech [7]. For instance, social bots play a crucial role in the 2018 U.S. midterm presidential election [8] and the 2020 U.S. presidential election [9]. They spread misinformation about COVID-19 [10], [11] and financial markets [12], [13]. Consequently, extensive research on social bot detection is crucial to mitigate these threats.

Mainstream social bots detection methods can be divided into methods based on user information and topology [14]. User information-based methods rely on user profiles

Manuscript received 24 August 2023; revised 16 February 2024; accepted 23 April 2024. This work was supported by the National Key Research and Development Program of China under Grant 2022YFC3302102. *(Corresponding author: Wei Zhou.)*

Boyu Qiao, Kun Li, Shilong Li, and Songlin Hu are with the Institute of Information Engineering, Chinese Academy of Sciences, Beijing 100085, China, and also with the School of Cyber Security, University of Chinese Academy of Sciences, Beijing 100049, China (e-mail: qiaoboyu@iie.ac.cn; likun2@iie.ac.cn; lishilong@iie.ac.cn; husonglin@iie.ac.cn).

Wei Zhou is with the Institute of Information Engineering, Chinese Academy of Sciences, Beijing 100085, China (e-mail: zhouwei@iie.ac.cn).

Digital Object Identifier 10.1109/TNNLS.2024.3396192

[15], [16] and text [17]. Topology-based methods rely on relationships between users. User profile-based methods typically analyze differences in value-type features (e.g., the number of followers) and Boolean-type features (e.g., verification status) to detect bots [14]. For text, mainstream methods usually use bidirectional long short-term memory networks (Bi-LSTM) [17], [18], [19], pre-trained language models [20], and generative adversarial networks (GANs) [21] to extract latent features in the text for detection, such as inconsistencies in semantic information from bots' posts [22], [23]. Recently, topology-based methods have gained significant attention and achieved promising results [24], [25], [26]. For example, BotRGCN [24] and BotMoE [26] utilize relational graph convolutional neural networks (RGCN) [27] to aggregate intra- and inter-relational information among neighboring nodes. To explore the relationship heterogeneity and influence heterogeneity among users, relational graph transformer (RGT) [25] utilizes graph transformer [28] and semantic attention networks to perform information aggregation and improve bot detection performance. These approaches combine user information, neighbor information, and topology information to characterize user nodes [14] more thoroughly.

Although the topology-based methods mentioned above are effective in bot detection, a significant challenge that remains unaddressed is the existence of disguised and unreliable edges within the graph structure. Bots often establish connections with real nodes to avoid suspicion, resulting in the presence of these unreliable edge connections. As a result, when using graph neural network (GNN) for message aggregation, the representation of a human/bot node may aggregate information from bot/human nodes through unreliable edges. Moreover, this unreliable information is amplified during the encoding process of multiple layers of GNNs. For instance, in Fig. 1, *Human B* connects with two unreliable edges, one connection is *Bot A* following *Human B*, and another connection is *Bot B* commenting on *Human B*. When using GNN for message aggregation, the node *Human B* aggregates information from the bot nodes. This violates the homogeneity assumption of GNNs [29], which states that adjacent nodes in the graph should have similar features and labels. The aggregation of neighbor information through unreliable edges leads to a smoothing problem for node representations of differently labeled nodes [30], [31]. Consequently, the presence of unreliable edges hampers the performance of GNN-based bot detection methods.

In light of the aforementioned challenges, we propose a social Bot detection model based on Edge Confidence Evaluation (BECE). Our proposed model comprises three modules.

1) *User information representation module:* This module employs multihead attention mechanisms to extract and integrate users' profiles and text information.
2) *Edge confidence evaluation module:* This module constructs edge features based on node features and introduces a parameterized Gaussian distribution to reconstruct the edge representation in latent semantic space. Additionally, we predict the confidence of edges and remove unreliable edges through a Bernoulli distribution and confidence probability.

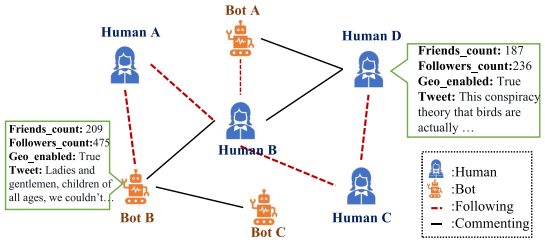

Fig. 1. Unreliable connections example in social networks.

3) *Node feature update module:* This module employs a multilayer graph encoder to aggregate information from neighboring nodes connected by reliable edges to obtain the final representation of the node and perform node classification detection.

We conduct experiments on three real-world datasets and thoroughly evaluate our proposed model against six widely used GNNs. The proposed model consistently outperforms the comparison methods in the social bot detection task. The main contributions of this article are summarized as follows.

1) We propose a method for social bot detection based on edge confidence evaluation. To the best of our knowledge, this is the first method to use edge confidence evaluation to identify and remove unreliable edges in social networks for bot detection purposes.
2) We propose an effective method to remove disguised and unreliable edges in a graph by designing an edge confidence evaluation module. This module can be applied as a plug-in to different GNN architectures and improve the bot detection performance.
3) We adopt parameterized Gaussian distribution to regularize the edge representations, which can enable the model to learn a fine spatial distribution of edge representations and generate discriminative and robust contextual representations of edges. This regularization method prevents the overfitting of the model and enhances its performance.
4) BECE employs six GNNs as the backbone network of graph encoders to conduct experiments on three real-world benchmark datasets, and the experimental results prove its effectiveness and superiority.

The remainder of this article is organized as follows. In Section II, we review the previous works in detail. In Section III, we define key terms and formally present the research problem. In Section IV, we introduce details of our proposed model. In Section V, we clarify the experimental setups and results. Finally, we summarize our study in Section VI.

II. Related Work

In this section, we introduce the most related work: social bot detection and unreliable edge identification.

A. Social Bot Detection

The social bot detection task aims to distinguish between humans and bots by analyzing the user's profile, text, and interaction information. There are four main methods commonly

employed to detect bots: profile-based method, text-based method, hybrid-based method, and topology-based method.

1) Approaches Based on Profile: Profiles typically contain two types of features: value-type and Boolean-type features. Profile features have been widely utilized in almost all social bot studies. Early research [16], [32], [33], [34] typically relies on manually crafted and derived features to identify bots. For instance, in the study [1], over 1000 features are designed, ranging from simple to complex, to train a random forest classifier to label OSN users as bots or benign. With the rapid development of deep learning technology, recent studies [24], [35] typically incorporate profile features as part of user representation and use methods like multilayer perception (MLP) to extract the latent features of profiles.

2) Approaches Based on Text: Previous studies on text detection typically focus on detecting the sequence of posting activities or the content of the posts. For posting activities, statistical analysis [36], [37], [38] and deoxyribonucleic acid (DNA) biosequence [39], [40], [41] methods have been commonly employed. For example, the work [39] proposes a bio-DNA-inspired method that encodes a user's posting, retweeting, commenting, and other activities as numerical DNA. On the constructed numerical DNA, they apply an analysis of the longest common subsequence to detect synchronous behavior among a group of users. For posting content, previous research has used deep learning methods [42] such as Bi-LSTM [17], [18], [19], CNN [43], GAN [21], and various pre-trained language models [20] to extract semantic inconsistency between texts to distinguish. However, with the development of deepfake technology, the humanoid nature of bots is improving, and without additional information, the performance of detection methods that only use text content or posting activity sequences is declining [44].

3) Approaches Based on Hybrid Features: The main manifestation of OSN user information is the mixture of various types of data. Recently, various techniques have been developed to identify bots by combining text and profile features, such as the methods proposed in [17], [19], [35], and [45]. For instance, the self-supervised approach to twitter account representation (SATAR) [35] leverages a deep neural network framework to merge multiple types of user information for bot detection. However, previous research that combines different features generally utilizes MLP or direct connections for classification, disregarding the modality distinctions between text and profiles. Therefore, we integrate multihead self-attention mechanisms to incorporate user information, extract interaction and dependency patterns between diverse information, and obtain more comprehensive user representations.

4) Approaches Based on Topology: Topology-based methods model the user information and connections as a graph. The central node gathers information from neighboring nodes to classify. Previous approaches utilize techniques such as homogeneous GNNs [graph convolutional network (GCN) [29], SAmple and aggreGatE (SAGE) [46], graph attention network (GAT) [47]] and heterogeneous GNNs (RGCN [27], relational GAT (RGAT) [48]) for feature extraction. However, these works all use spatial-based GNN structures [49] and fail to consider the presence of unreliable connections, which violates the homophily assumption [29]. Homophily assumes that adjacent nodes tend to share similar features and labels. To address this issue, study [50] proposes a two-stage training method that mitigates the influence of unreliable edges by adding homogeneous edges to the graph and leveraging the adaptive attention mechanism to extract high-pass information from heterogeneous edges. However, this two-stage training paradigm may lead to suboptimal performance and introduce noise. Another study [51] introduces high-similarity edges, homogeneous edges, and heterogeneous edges as three additional relationships to the graph. Although this edge-adding strategy improves performance, it fails to differentiate the features of the extracted homogeneous and heterogeneous nodes. In contrast to previous work, we propose BECE, which incorporates an edge confidence evaluation module to filter out unreliable edges. This approach aims to reduce the negative impact of heterogeneous nodes on the central node's representation. By implementing this removal strategy, our model can learn more discriminative node representations and ultimately improve classification performance.

B. Identify Unreliable Edges

Attempts to identify unreliable edges in graphs have been made in the fields of fraud detection and spam filtering. For instance, study [52] explores the use of attention mechanisms to reduce the weight of unreliable aggregation in graphs. To more thoroughly eliminate the interference caused by unreliable edges, studies [53], [54] employ reinforcement learning to delete a certain proportion of unreliable edges. However, this approach cannot accurately process each unreliable edge connected to individual nodes and remove them in a targeted manner. Therefore, it is crucial to design a mechanism that can precisely preserve reliable edges while removing unreliable ones for each node. In this context, most similar to our method is the study [55], which proposes a general graph augmentation scheme that determines the addition or removal of edges in the graph by reconstructing the values of the adjacency matrix. However, this approach encounters challenges in practical application. Due to the lack of ground truth, it is difficult to determine exactly which edges should be added to the graph, which may introduce noise information. To address these limitations, we develop a comprehensive bot detection model and introduce an edge confidence evaluation module. The core idea of this module is to carefully evaluate the edges connected to each node, and dynamically eliminate unreliable edges based on their confidence probability. This scheme can help the model to focus on the edge which is more critical to the detection task, and improve the performance.

III. Problem Definition

Social bots are bots controlled by computer programs, designed to simulate human social interaction. Bot detection is aimed at identifying whether automated programs control users on online platforms. For the convenience of the reader, the notations used in the article are summarized in Nomenclature.

Definition 1 (User Information Representation): There are three types of information for users, including profiles, texts, and topology. The profiles are composed of value-type features, represented as $V = \{v_1, \ldots, v_n\}$, and Boolean-type

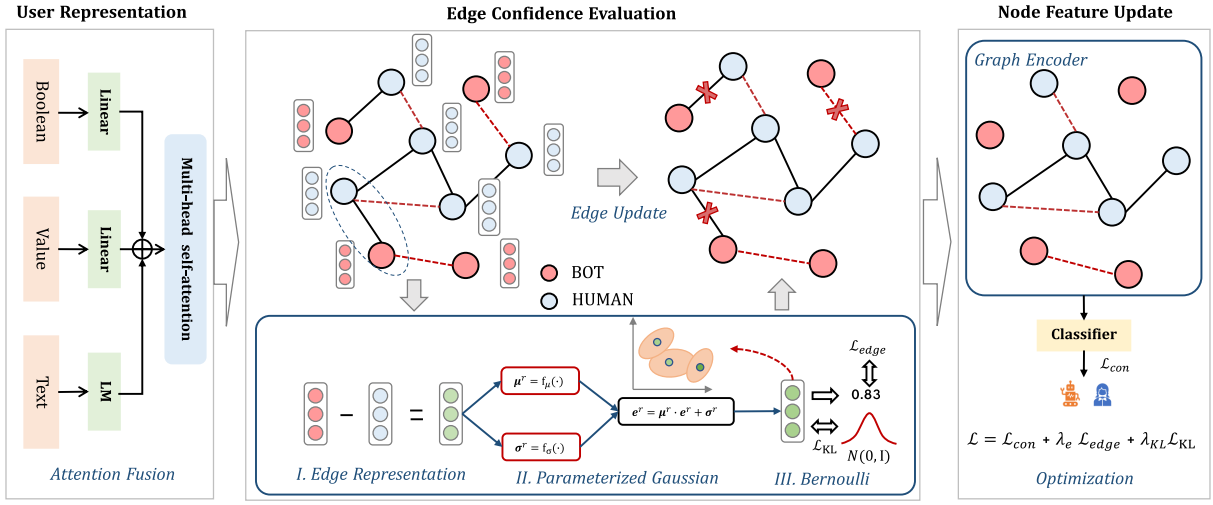

Fig. 2. Overall structure of BECE model, which consists of three modules: user information representation module, edge confidence evaluation module, and node feature update module.

features, represented as $B = \{b_1, \ldots, b_n\}$. The text information consists of multiple historical posts and is represented as $T = \{t_1, \ldots, t_n\}$, where $t_i = \{t_i^1, \ldots, t_i^m\}$, t_i^m represents the m-th text of the user u_i, and the n represents the number of users. The aforementioned three types of features constitute the representations of the user, denoted as $U = \{u_1, \ldots, u_n\}$. Topological information is composed of the relationships between users, such as following, commenting, and liking.

Definition 2 (Bot Detection on Graph): The GNN-based model be defined as $\mathcal{G} = \{U, E_r|_{r=1}^R, Y\}$ where we employ the user U as nodes, E_r as edges, and R is the number of relationships between users. $Y = \{y_1, \ldots, y_n\}$ represents the ground truth of user nodes. The neighbor of user u_i is denoted as $N_i^r = \{u_{i,j}^r\}_{j=1}^{J_i}$, where J_i denotes the number of one-hop neighbors, and N_i^r denotes neighbor nodes of user u_i under relationship r. The goal of this model is to learn a function $f : f(u_i) \to \hat{y}_i \in \{0, 1\}$, where $\hat{y}_i$ approximates the ground truth y_i with maximum prediction accuracy.

Definition 3 (Unreliable Edges): In social networks, three types of edge connections exist in different social relationships, including connections between humans, represented as (Human, Human), connections between bots, represented as (Bot, Bot), and connections between humans and bots, represented as (Human, Bot). As the presence of (Human, Bot) type edges violates the homogeneity assumption of GNN, we refer to this type of edge as an unreliable edge.

IV. Method

A. Model Overview

Fig. 2 illustrates the overall framework of BECE, which consists of three modules: user information representation module, edge confidence evaluation module, and node feature update module. These modules collaborate to enhance the overall performance of BECE. The user information representation module incorporates various data formats of users and utilizes multihead self-attention mechanisms to fuse these representations effectively (see Section IV-B). The edge confidence evaluation module employs node embeddings to construct edge features and leverages a parameterized Gaussian distribution to reconstruct edge representations. Then, the edge confidence is evaluated based on the reconstructed edge representation, and the unreliable edge is identified based on the confidence probability (see Section IV-C). The node feature update module utilizes the graph structure after removing unreliable edges for message passing and aggregation. It updates the node representations accordingly (see Section IV-D).

B. User Information Representation Module

1) User Information Embedding: OSN user information includes profiles and textual content. To obtain effective representations of users, this module utilizes MLP and pre-trained language models (Roberta [56] and LaBSE [57]) to represent the value, Boolean, and text information of the given input user u_i. These formulas are denoted as $\mathbf{u}_i^v$, $\mathbf{u}_i^b$ and $\mathbf{u}_i^t$

$$\begin{aligned} \mathbf{u}_i^v &= v_i W_v \\ \mathbf{u}_i^b &= b_i W_b \\ \mathbf{u}_i^t &= \left(\text{Ave-Pooling}\left\{\text{LM}\left(t_i^1, \ldots, t_i^m\right)\right\}\right) W_t \end{aligned} \tag{1}$$

where the text feature $\mathbf{u}_i^t$ is computed by taking the average pooling of the embeddings of each post. W_v, W_b, W_t are the trainable parameters of the value-type feature, Boolean-type feature, and text-type feature encoding, respectively.

2) Attention Fusion: Previous works usually employ MLP or concatenate to fuse the three types of features. However, considering the differences between the features, we suggest using multihead self-attention mechanisms to fuse the user's multiple types of features, thus achieving better semantic fusion. The fusion of the user's feature is represented as

$$\begin{aligned} \mathbf{U} &= \left\{\mathbf{u}_1, \mathbf{u}_2, \ldots, \mathbf{u}_n | \mathbf{u}_i = \left(\text{Concat}\left(\mathbf{u}_i^v, \mathbf{u}_i^b, \mathbf{u}_i^t\right)\right) W\right\} \\ \mathbf{Q}_j &= \mathbf{U} W_j^Q, \mathbf{K}_j = \mathbf{U} W_j^K, \mathbf{V}_j = \mathbf{U} W_j^V \\ A_j &= \text{Softmax}\left(\frac{\mathbf{Q}_j \mathbf{K}_j^T}{\sqrt{d_k}}\right) \\ \hat{\mathbf{U}} &= \text{Concat}(A_1 \mathbf{V}_1, A_2 \mathbf{V}_2, \ldots, A_h \mathbf{V}_h) W^O \end{aligned} \tag{2}$$

where $\mathbf{U}$ denotes the initial user embedding, and $\hat{\mathbf{U}} = \{\hat{\mathbf{u}}_1, \hat{\mathbf{u}}_2, \ldots, \hat{\mathbf{u}}_n\}$ is the user representation after the multihead self-attention mechanisms. $\mathbf{Q}_j$, $\mathbf{K}_j$, and $\mathbf{V}_j$ are the query, key, and value space of the jth head after linear transformations, there are a total of h groups of attention spaces. $W_j^Q \in \mathbb{R}^{d_{\text{model}} \times d_k}$, $W_j^K \in \mathbb{R}^{d_{\text{model}} \times d_k}$, $W_j^V \in \mathbb{R}^{d_{\text{model}} \times d_v}$, and $W^o \in \mathbb{R}^{hd_v \times d_{\text{model}}}$ are the weight matrix, and j indicates the jth attention head. A_j represents the attention weights for each group and $d_k = d_{\text{model}}/h$. The effectiveness of using multihead self-attention mechanisms for three types of feature fusion is demonstrated in the ablation study in Section V-C.

C. Edge Confidence Evaluation Module

Advanced bots not only mimic humans in profile and text but also establish camouflaged connections with real users to evade detection. These connections, represented as unreliable edges, violate the homogeneity assumption [29]. Consequently, after the aggregation process in GNN, the node representations of bots and humans become indistinguishable, adversely affecting the detector's performance. To address this issue, we introduce this module to remove the unreliable edges from the graph. In this module, we first design features for edges. Next, parameterized Gaussian distribution is used to reconstruct the edge features and predict their confidence. The predicted confidence probability and the Bernoulli distribution are then used to identify and remove unreliable edges.

1) Construct Edge Representation: In Section IV-B, we obtain the fused representation of the user, which serves as the node feature in the graph. Subsequently, we construct features for the edges. The construction of edge representation is a highly flexible process that allows for various feature combined functions. To evaluate the impact of different feature combinations on model performance, we conduct experiments and analysis in Section IV-C.

We propose designing edge representations based on node features. We use the L_1-distance to measure the feature deviation between the central node and its neighboring nodes for different relationship r. If the central node is a human and when the neighboring nodes exhibit high feature deviations from it, the neighbor nodes are more likely to be bots, indicating that the edge connection between the two nodes may be an unreliable edge. Conversely, suppose the feature deviation between two nodes is slight, the probability that the two nodes belong to the same type of node is higher, indicating that their edge connection is trustworthy. Specifically, in the lth layer of the graph, we obtain edge representations using the subtraction aggregation based on the feature vectors $\hat{\mathbf{u}}_i^{r,(l)}$ of user u_i and its neighboring nodes $N_i = \{u_{i,j}^{r,(l)}\}_{j=1}^{J_i}$

$$\mathbf{e}_{i,j}^{r,(l)} = \hat{\mathbf{u}}_i^{r,(l)} - \hat{\mathbf{u}}_{i,j}^{r,(l)}. \tag{3}$$

2) Parameterized Gaussian Reconstruction Edge Representation: Since the constructed edge representation inherently contains noise, to enhance the robustness of edge representation and mitigate the adverse effects of noisy samples, inspiration from the works of [58], we utilize a parameterized Gaussian distribution to model the features and noise of edges. We define the edge representation as a Gaussian distribution $\mathcal{N}(\mathbf{e}, \boldsymbol{\mu}, \boldsymbol{\sigma}^2 I)$. Specifically, we first use two feedforward layers to learn the mean μ and variance σ of the edge representations

$$\begin{aligned} \boldsymbol{\mu}_{i,j}^{r,(l)} &= f_{\theta_1}\left(\boldsymbol{e}_{i,j}^{r,(l)}\right) \\ \boldsymbol{\sigma}_{i,j}^{r,(l)} &= f_{\theta_2}\left(\boldsymbol{e}_{i,j}^{r,(l)}\right) \end{aligned} \tag{4}$$

where θ_1 and θ_2 denote the model parameters of the mean and variance networks, respectively. $\boldsymbol{\mu}_{i,j}^{r,(l)}$ (mean) represents the most probable feature embedding after edge reconstruction, and $\boldsymbol{\sigma}_{i,j}^{r,(l)}$ (variance) denotes the noisy of the edge features.

To make the model trainable, we employ the reparameterization technique [59] to sample a random distribution parameter ϵ from the standard distribution $\mathcal{N}(0, I)$, and generate target embeddings $\tilde{\mathbf{e}}_{i,j}^{r,(l)}$ with the same distribution using the following method:

$$\tilde{\mathbf{e}}_{i,j}^{r,(l)} = \boldsymbol{\mu}_{i,j}^{r,(l)} + \epsilon \boldsymbol{\sigma}_{i,j}^{r,(l)}, \quad \epsilon \sim \mathcal{N}(0, I). \tag{5}$$

Now, the representation of edge is not a deterministic point embedding anymore, but a stochastic embedding sampled from $\mathcal{N}(\tilde{\mathbf{e}}_{i,j}^{r,(l)}; \boldsymbol{\mu}_{i,j}^{r,(l)}, (\boldsymbol{\sigma}_{i,j}^{r,(l)})^2)$ in the latent space. The robustness of the model to noise is enhanced by random sampling around the mean and variance. The study of Section V-D demonstrates the effectiveness of edge representation reconstruction.

Subsequently, according to the edge representation, we utilize the Sigmoid activation function [60] to estimate the confidence of the connection between u_i and u_j

$$p_{i,j}^{r,(l)} = \text{Sigmoid}\left(\text{MLP}\left(\tilde{\mathbf{e}}_{i,j}^{r,(l)}\right)\right) \tag{6}$$

where $p_{i,j}^{r,(l)}$ represents the probability of edge confidence between u_i and u_j in the graph of relation r and lth layer.

3) Remove Unreliable Edges Based on Bernoulli Distribution: We consider the edges as binary events, that is, for an edge connecting nodes u_i and u_j, if the probability $p_{i,j}^{r,(l)}$ is low, it means that the unreliability between the two nodes is high. Therefore, we can consider removing it. The Bernoulli approximation is commonly used to handle binary events. Hence, we use it [61] to remove unreliable edges

$$f\left(s_{i,j}^{r,(l)} \mid p_{i,j}^{r,(l)}\right) = \begin{cases} p_{i,j}^{r,(l)}, & s_{i,j}^{r,(l)} = 1 \\ 1 - p_{i,j}^{r,(l)}, & s_{i,j}^{r,(l)} = 0 \end{cases} \tag{7}$$

where $s_{i,j}^{r,(l)} = 1$ denotes the preservation of the edge connecting between u_i and u_j, whereby the probability of retaining edge is $p_{i,j}^{r,(l)}$. Similarly, $s_{i,j}^{r,(l)} = 0$ means the edge is not kept. We optimize $p_{i,j}^{r,(l)}$ by edge labels and binary cross-entropy.

4) Optimization: We utilize binary cross-entropy to optimize the edge confidence evaluation module and Kullback–Leibler (KL) divergence [62] to optimize the edge representation of parametric Gaussian reconstruction.

First, we label the edge. Specifically, the connection between the bot and the human is an unreliable edge, that is, eage label $=$ 0. The connection between bot and bot, human and human is trustworthy and reliable edges, that is, edge label $=$ 1. Then, we employ binary cross-entropy to

optimize the edge confidence evaluation module

$$\mathcal{L}_{\text{edge}}^{(l)} = \mathbb{E}\left[\sum_{r=1}^{R}\left(\sum_{i,j:y_i=y_j} -\log p_{ij}^{r,(l)} - \sum_{i,j:y_i\neq y_j}\left(\log\left(1-p_{ij}^{r,(l)}\right)\right)\right)\right]. \quad (8)$$

The term $\sum_{i,j:y_i=y_j} -\log p_{ij}$ represents maximizing the probability of reliable edges, while $-\sum_{i,j:y_i\neq y_j}(\log(1-p_{ij}^{(l)}))$ represents minimizing the probability of unreliable edges. R indicates that there are R relationships among user nodes.

Second, we use KL divergence to optimize the edge representation. Define $\mathcal{L}_{\text{KL}}^{(l)}$ as the regularization term for layer l when reconstructing edge representations. To approximate the standard distribution $\mathcal{N}(\epsilon; 0, I)$ with a parameterized Gaussian distribution $\mathcal{N}(\tilde{\mathbf{e}}_{i,j}^{r,(l)}; \boldsymbol{\mu}_{i,j}^{r,(l)}, (\boldsymbol{\sigma}_{i,j}^{r,(l)})^2)$, we utilize KL divergence to measure the difference between the two distributions

$$\begin{aligned}\mathcal{L}_{\text{KL}}^{(l)} &= D_{\text{KL}}\left(\frac{1}{R}\sum_{r=1}^{R}\left(\mathcal{N}\left(\tilde{\mathbf{e}}_{i,j}^{r,(l)}; \boldsymbol{\mu}_{i,j}^{r,(l)}, \left(\boldsymbol{\sigma}_{i,j}^{r,(l)}\right)^2\right)\|\mathcal{N}(\epsilon; \mathbf{0}, I)\right)\right) \\ &= -\frac{1}{2R}\sum_{r=1}^{R}\left(1+\log\left(\boldsymbol{\sigma}_{i,j}^{r,(l)}\right)^2 - \left(\boldsymbol{\mu}_{i,j}^{r,(l)}\right)^2 - \left(\boldsymbol{\sigma}_{i,j}^{r,(l)}\right)^2\right)\end{aligned} \quad (9)$$

where $D_{\text{KL}}(\cdot||\cdot)$ denotes the KL divergence between two distributions.

D. Node Feature Update Module

1) Graph Encoder: In this module, we utilize GNNs to aggregate node features based on the graph after edge connection updates. This module is applicable to diverse GNN architectures, and the messaging passing paradigm can be defined as follows:

$$\begin{aligned}\mathbf{a}_i^{r,(l+1)} &= \text{AGGREGATE}_r^{(l+1)}\left(\left\{\hat{\mathbf{u}}_j^{r,(l)}, \forall j \in \mathcal{N}_r(i)\right\}, \hat{\mathbf{u}}_i^{r,(l)}\right) \\ \hat{\mathbf{u}}_i^{(l+1)} &= \text{UPDATE}^{(l+1)}\left(\hat{\mathbf{u}}_i^{r,(l)}, \mathbf{a}_i^{r,(l)}\right)|_{r=1}^{R}.\end{aligned} \quad (10)$$

$\mathcal{N}_r(i)$ denotes the neighboring nodes of u_i. $\mathbf{a}_i^{r,(l+1)}$ means to aggregate the neighbor nodes under the relationship r and lth layer. $\mathbf{u}_i^{(l+1)}$ indicates the node representation of layer $l+1$ is obtained by aggregating the central and neighbor nodes of layer l. In Section V-B, we conduct experiments on different GNNs and compare their performance.

2) Optimization: We employ the output layer and softmax function to acquire the anticipated values of the user, namely,

$$\hat{y}_i = \text{Softmax}(W_u \cdot \hat{\mathbf{u}}_i + b_u). \quad (11)$$

$\hat{y}_i$ denotes the predicted value of user type by the model, while W_u and b_u are the learnable parameters. Subsequently, we optimize our model utilizing cross-entropy loss

$$\mathcal{L}_{\text{con}} = -\sum_{i=1}^{n}\left[y_i \log(\hat{y}_i) + (1-y_i)\log(1-\hat{y}_i)\right] \quad (12)$$

where y_i represents the ground-truth label.

Algorithm 1 BECE

Input: User profile $\mathbf{u}_i^v, \mathbf{u}_i^b$, textual content $\mathbf{u}_i^t$, and topology relationship $\mathcal{N}$. Optimized model parameters θ.
Output: Predict user node types in the graph.

```
Initialize θ
Encode profile and text: u_i^v, u_i^b, u_i^t ← Equation (1)
User Information Representation: u_i ← Equation (2)
while l < L do
    for u_i and Neighborhood j ∈ N_i do
        Construct edge embeddings: e_{i,j}^{r,(l)} ← Equation (3)
        Parameterized Gaussian reconstruction edge representation: ẽ_{i,j}^{r,(l)} ← Equation (4), (5)
        Edge confidence probability p_{i,j}^{r,(l)} ← Equation (6)
        Bernoulli distribution removal unreliable edge: s_{i,j}^{r,(l)} ← Equation (7)
        Optimize: L_edge^{(l)}, L_KL^{(l)} ← Equation (8), (9)
    end for
    Update node features u_i^{(l+1)} ← Equation (10)
end while
Predict ŷ_i ← Equation (11)
Optimize: L_con ← Equation (12)
Loss: L ← Equation (13)
return θ
```

E. Training and Reasoning

The optimization objective of our model consists of three parts. Equation (12) minimizes the objective function $\mathcal{L}_{\text{con}}$ of the main task bot detection. Equation (8) minimizes the prediction loss $\mathcal{L}_{\text{edge}}$ of unreliable edges to remove them from the graph. Equation (9) minimizes the KL divergence $\mathcal{L}_{\text{KL}}$ of the edge representation Gaussian regularization to obtain a more stable edge representation

$$\mathcal{L} = \mathcal{L}_{\text{con}} + \lambda_e \mathcal{L}_{\text{edge}} + \lambda_{\text{KL}}\mathcal{L}_{\text{KL}} \quad (13)$$

where $\mathcal{L}_{\text{edge}} = (1/L)\sum_{l=1}^{L}\mathcal{L}_{\text{edge}}^{(l)}$, $\mathcal{L}_{\text{KL}} = (1/L)\sum_{l=1}^{L}\mathcal{L}_{\text{KL}}^{(l)}$, and L is the number layer of GNN. To effectively avoid cumulative edge removal errors, we re-perform edge removal in each layer of GNN according to the latest updated node representation and calculate the loss accordingly. To balance the GNN contribution to the total loss, we multiply the loss by $1/L$. λ_{edge} and λ_{KL} are the trade-off hyperparameters of the loss function, and we conduct experiments on the trade-off hyperparameters in Section V-E. Algorithm 1 outlines the training process.

V. Experiment

In this section, we assess the performance of BECE on three real-world datasets. First, we present the key findings of our approach using homogeneous and heterogeneous GNN architectures. Then, we conduct ablation experiments on the three modules of the BECE. Third, we conduct further study to investigate the impact of hyperparameter selection on model robustness. Finally, we conduct a case study on the edge confidence evaluation module.

A. Experiment Setup

1) Datasets: We utilize three real-world datasets, namely Cresci-15 [63], Twibot-20 [64], and multi-relational graph-based Twitter account detection benchmark (MGTAB) [65] to investigate the problem of unreliable edges in social topology and the GNN-based model detection. Table I illustrates the detailed information regarding these datasets. Specifically, **Cresci-15** [63] is a dataset created by the false account detection task for the social media platform Twitter, including the HUM dataset and the FAK dataset. **Twibot-20** [64] is a collection of user attention relationships that are obtained using the seed users and breadth-first search method. The accounts collected by Twibot-20 are primarily focused on business, entertainment, politics, and sports. **MGTAB** [65] is a large-scale standardized expert-annotated dataset for stance and bot detection. It contains three modes of data, namely profile, text, and relationship.[1]

In this article, we conduct experiments on Cresci-15, Twibot-20, and MGTAB. For Twibot-20 and Cresci-15, we follow the profile and text features used in previous work [22], [26]. For MGTAB, we follow the features adopted in [65]. We create a graph using the follower and friend relations as edges, and users as nodes. To split the datasets, we adopt the method used in previous literature [24], [65], where 70% of the nodes are allocated to the training set, 20% to the validation set, and 10% to the test set. To prevent the issue of edge label leakage during the training process, we implement a semi-supervised learning technique. Specifically, all edges within the graph are utilized for training, but only those connected to nodes within the training set are labeled. Meanwhile, edges connected to nodes within the validation or test set remain unlabeled.

2) Evaluation Metrics: Based on the previous work [26], we utilize the Accuracy (Acc) and F1-score as evaluation metrics for Cresci-15 and Twibot-20. Due to the unbalanced MGTAB, we employ the Accuracy (Acc) and macro-averaged-F1 (F1-macro) as evaluation metrics instead [65].

3) Implementation: We utilize the different graphs integrated by Pytorch Geometric [70] as the backbone of the third module of the BECE to implement. Table II lists the hyperparameters used in the experiment for easy reproduction. We experiment on clustered devices equipped with four RTX A6000 GPU, 48 GB of memory, 24 CPU cores, and 252 GB of CPU memory. Training BECE on Cresci-15, TwiBot-20, and MGTAB datasets requires approximately 5 min, 19 min, and 2 h, respectively.

4) Baselines: First, we compare BECE to several bot detection models utilizing different modalities. Specifically, Kudugunta and Ferrara [17] and Wei and Nguyen [18] employ the text content information, while Botometer [1] and Yang et al. [16] use the profile information, and SATAR [35] utilizes both profile and text features. Furthermore, we compare BECE with several GNN-based bot detection methods, including BotRGCN [24] and RGT [25]. BotRGCN [24] uses profiles and text as node features and employs RGCN as the backbone network for detection. RGT [25] is a heterogeneous GNN for bot detection that incorporates influence heterogeneity and relation heterogeneity into the graph structure. Furthermore, we investigate the effect of different GNN polymerizes on BECE by conducting comparative experiments with six different GNN structures, namely the homogeneous graph GAT [47], GCN [29], and the heterogeneous graph transformer (HGT) [71], RGT [25], RGCN [27], and simple heterogeneous graph network (S-HGN) [72]. Among them,

TABLE I

STATISTICS FOR CRESCI-15, TWIBOT-20, AND MGTAB, WHERE "EDGES" INDICATES THAT THE USER NODES IN THE GRAPH ARE CONNECTED BY THE FOLLOWER AND FRIEND RELATIONSHIP. (HUMAN, BOT) REPRESENTS THE NUMBER OF UNRELIABLE EDGES. (BOT, BOT) AND (HUMAN, HUMAN) REPRESENT THE NUMBER OF TRUSTWORTHY EDGES

Datasets	Cresci-15	Twibot-20	MGTAB
Users	5,301	11,826	10,199
Human	1,950	5,237	7,451
Bot	3,351	6,589	2,748
Training	3,708	8,278	7,139
Validation	958	2,040	2,365
Testing	535	1,183	1,020
Edges	14,220	15,434	720,695
(Human, Bot)	158	7,286	64,200
(Bot, Bot)	746	3,717	518
(Human, Human)	13,316	4,431	655,977

TABLE II

HYPERPARAMETER SETTING OF BECE

Hyperparameter	Cresci-15	Twibot-20	MGTAB
Dropout	0.3	0.3	0.3
Learning rate	1e-4	1e-4	1e-4
Batch size	1024	1024	1024
λ_e	1	3	0.1
λ_{KL}	1	0.01	0.1
Seed	10-14	10-14	10-14
Optimizer	Adam	Adam	Adam
Hidden size	128	128	128
Attention head number	4	4	4
Weight decay	5e-3	5e-3	5e-3
Numerical feature dim	1	6	10
Boolean feature dim	5	11	10
Text feature dim	768	768	768
Hidden layer dim	128	128	128
Epoch	100	100	200
Language model	Roberta [56]	Roberta	LaBSE [57]
GNN layer number	2	2	1

[1]Although previous works have proposed other social bot detection datasets, such as Cresci-17 [66], political-bots-2019 [67], and cresci-rtbust-2019 [68]. However, these datasets are unable to construct meaningful graph. Twibot-22 [69] is a dataset that can construct a graph. Since it is a dataset with weakly supervised labeling (only 1000 pieces of data are labeled by experts), the reliability of the labeling cannot be determined, so we do not choose it as the experimental dataset.

TABLE III

MAIN RESULTS. WE RUN EACH METHOD FIVE TIMES AND REPORT THE MEAN AND STANDARD DEVIATION IN THE TABLE. BOLD INDICATES OPTIMAL PERFORMANCE, UNDERLINE INDICATES SUBOPTIMAL, AND "-" INDICATES THAT THE METHOD IS NOT SCALABLE TO MGTAB. BECE (GNN) REPRESENTS USING DIFFERENT GNNS AS THE BACKBONE NETWORK. GNN (+) REPRESENTS THE PERFORMANCE IMPROVEMENT RATIO OF BECE (GNN) COMPARED TO THE GNN IN THE BASELINE MODEL. †REPRESENTS MODEL PERFORMANCE FROM [69]. ‡REPRESENTS OUR REPRODUCTION RESULT, AND THE NODE FEATURES OF THE REPRODUCED GRAPH STRUCTURE FOLLOW THOSE CONSTRUCTED BY BOTRGCN [24] AND RGT [25]

Methods	Cresci-15		Twibot-20		MGTAB	
	Acc(%)	F1-score(%)	Acc(%)	F1-score(%)	Acc(%)	F1-macro(%)
Bot Detection Based on Different Modes						
Wei et al. † [18]	96.18 ± 1.54	82.65 ± 2.47	70.23 ± 0.10	53.61 ± 0.10	-	-
Kudugunta et al. † [17]	75.33 ± 0.13	75.74 ± 0.16	59.59 ± 0.65	47.26 ± 1.35	-	-
Yang et al. † [16]	77.08 ± 0.21	77.91 ± 0.11	81.64 ± 0.46	84.89 ± 0.42	-	-
Botometer † [1]	57.92	66.90	53.09	55.13	-	-
SATAR † [35]	93.42 ± 0.48	95.05 ± 0.34	84.02 ± 0.85	86.07 ± 0.70	-	-
Graph-based Bot Detection Models						
GCN ‡ [29]	96.51 ± 0.26	96.22 ± 0.28	77.70 ± 0.85	82.76 ± 0.80	84.20 ± 1.72	78.56 ± 2.75
GAT ‡ [47]	96.27 ± 0.35	95.96 ± 0.38	83.31 ± 1.12	86.11 ± 1.39	87.24 ± 1.71	82.71 ± 3.03
RGT ‡ [25]	97.15 ± 0.32	97.78 ± 0.24	86.57 ± 0.41	88.01 ± 0.41	90.06 ± 0.92	87.06 ± 1.34
BotRGCN ‡ [24]	96.52 ± 0.71	97.30 ± 0.53	83.27 ± 0.57	85.26 ± 0.38	89.27 ± 1.27	86.01 ± 2.11
HGT ‡ [71]	96.10 ± 0.21	96.93 ± 0.31	85.78 ± 0.25	87.48 ± 0.27	89.78 ± 1.48	87.02 ± 2.03
S-HGN ‡ [72]	96.70 ± 0.50	97.20 ± 0.45	86.72 ± 0.28	89.38 ± 0.28	89.27 ± 1.82	86.57 ± 2.15
***BECE* Combines Different GNN Architectures**						
***BECE* (GCN [29])**	98.38 ± 0.46	98.72 ± 0.37	84.18 ± 0.70	87.24 ± 0.49	85.45 ± 1.60	80.46 ± 2.49
GCN (+)	+ 1.87	+ 2.50	+ 6.48	+ 4.48	+ 1.25	+ 1.90
***BECE* (GAT [47])**	98.23 ± 0.61	98.60 ± 0.49	86.14 ± 0.34	89.12 ± 0.23	90.16 ± 1.47	87.71 ± 1.91
GAT (+)	+ 1.96	+ 2.64	+ 2.83	+ 3.01	+ 2.92	+ 5.00
***BECE* (RGT [25])**	**100.00 ± 0.00**	**100.00 ± 0.00**	**87.73 ± 0.26**	**90.20 ± 0.18**	90.71 ± 1.64	88.26 ± 2.11
RGT (+)	+ 2.85	+ 2.22	+ 1.16	+ 2.19	+ 0.65	+1.20
***BECE* (RGCN [27])**	98.31 ± 0.62	98.67 ± 0.48	87.50 ± 0.38	90.01 ± 0.28	**91.08 ± 1.84**	**88.61 ± 2.45**
BotRGCN (+)	+ 1.79	+ 1.37	+ 4.23	+ 4.75	+ 1.81	+ 2.60
***BECE* (HGT [71])**	99.10 ± 0.44	99.29 ± 0.34	87.55 ± 0.24	89.92 ± 0.24	91.00 ± 1.41	88.49 ± 1.74
HGT (+)	+ 3.00	+ 2.36	+ 1.77	+ 2.44	+ 1.22	+ 1.47
***BECE* (S-HGN [72])**	98.64 ± 0.32	98.93 ± 0.26	87.61 ± 0.32	90.06 ± 0.25	90.86 ± 1.78	88.45 ± 2.29
S-HGN (+)	+ 1.94	+ 1.73	+ 0.89	+ 0.68	+ 1.59	+ 1.88

BotRGCN [24] uses RGCN as the backbone graph network, and RGT is a graph structure proposed by Feng et al. [25] for bot detection.

B. Main Results

In this section, we perform experiments on three real-world datasets to compare the performance of our method with 11 baseline algorithms. Table III shows the experimental overall results of BECE. For privacy considerations, the creators of the MGTAB dataset only provide the encoded user information, but not the original user profile and text information [65]. Consequently, we are unable to reproduce some of the previous methods on the MGTAB. Based on the analysis of the experimental results, we have the following observations.

1) *BECE Outperforms Baseline Methods on All Three Datasets:* On the Cresci-15, our proposed model achieves 100% performance on the RGT [25] graph structure. Moreover, compared to the previous state-of-the-art method RGT in the baseline method, the accuracy of BECE (the optimal GNN) on the three datasets has been improved by 2.85%, 1.16%, and 1.02%, respectively, and the performance of F1-score has been improved by 2.22%, 2.19%, and 1.55%, respectively.
2) *Using heterogeneous graph as the backbone network of BECE outperforms using homogeneous graph:* The heterogeneous graph employs an edge-heterogeneous structure, incorporating follower and friend edges across all three datasets. Conversely, in the homogeneous graph, both types of edges are treated as the same type. We apply both homogeneous and heterogeneous GNN structures to the third module of the BECE and compare the performance of different GNN architectures in Table III. Experimental results indicate that the performance using a homogeneous graph (e.g., GCN, GAT) is lower than the performance using a heterogeneous graph (e.g., RGT, S-HGN, HGT, RGCN). This confirms that multirelation graph structures are beneficial for bot detection tasks.
3) *Compared to using single GNN, the combination of GNN with BECE demonstrates significant performance*

TABLE IV
PERFORMANCE OF DIFFERENT COMBINATIONS OF USER FEATURES. "ONLY" INDICATES THAT ONLY ONE FEATURE IS USED

Albation	Twibot-20		MGTAB	
	Acc(%)	F1-score(%)	Acc(%)	F1-macro(%)
text-only	78.09±0.52	81.80±0.42	86.29±1.06	82.10±1.70
value-only	65.11±0.33	71.59±0.29	72.80±0.96	42.34±0.57
bool-only	81.67±0.06	85.51±0.05	73.25±1.19	46.67±4.18
value-bool	81.61±0.14	85.44±0.12	79.39±1.13	71.24±2.17
text-bool	83.40±2.32	86.14±1.57	86.55±1.03	82.54±1.82
text-value	78.58±0.47	81.90±0.28	88.92±1.55	86.05±2.17
text-value-bool	**85.70±0.19**	**87.69±0.06**	**88.96±1.54**	**86.07±2.09**

TABLE V
DETECTION PERFORMANCE OF DIFFERENT FEATURE FUSION METHODS

Albation	Twibot-20		MGTAB	
	Acc(%)	F1-score(%)	Acc(%)	F1-macro(%)
MLP (RGT)	87.13±0.17	89.66±0.10	89.38±0.58	86.58±0.74
Ave Pooling (RGT)	86.65±0.25	89.35±0.12	89.02±1.57	85.62±2.35
Attention (RGT)	**87.73±0.26**	**90.20±0.18**	**90.07±1.64**	**88.26±2.11**
MLP (RGCN)	87.11±0.12	89.67±0.11	90.26±1.72	87.51±2.59
Ave Pooling (RGCN)	87.22±0.24	89.72±0.18	89.55±1.28	86.55±2.23
Attention (RGCN)	**87.50±0.38**	**90.01±0.28**	**91.08±1.84**	**88.61±2.45**
MLP (S-HGN)	87.30±0.20	89.66±0.18	89.69±1.90	87.12±2.93
Ave Pooling (S-HGN)	86.65±0.43	89.40±0.24	89.12±1.59	86.08±2.62
Attention (S-HGN)	**87.61±0.32**	**90.06±0.25**	**90.86±1.78**	**88.45±2.29**

advantage: On Cresci-15, BECE outperforms the GNN baseline by 1.79%–3.00% in accuracy and 1.37%–2.64% in F1-score. On Twibot-20, it improves accuracy by 0.89%–6.48% and F1-score by 0.68%–4.75%. For MGTAB, accuracy gains are 0.65%–2.92% and F1-macro increases by 1.20%–5.00%. These performance enhancements can be attributed to our fusion processing of user profiles, text, and the effective removal of unreliable edges in the graph.

4) *Combining multiple types of features outperforms methods based on single feature type or simple feature connection:* Specifically, BECE outperforms methods that rely on a single type of feature (such as works [1], [16], [17], and [18]), highlighting the limitations of single-feature detection methods. Both SATAR and BECE leverage users' multitype features. Our experimental results on Cresci-15 and Twibot-20 show that BECE surpasses SATAR, with F1-scores improving by 4.95% and 4.13%, respectively. This indicates the effectiveness of our proposed multitype information fusion.

Since BECE has shown excellent performance on Cresci-15, we will mainly conduct experimental research on Twibot-20 and MGTAB in the following experiments.

C. User Information Representation Module Study

To validate the effectiveness of the user information representation module, we research its different components and arrive at the following conclusions.

1) Multitype Feature Outperforms Single-Type Feature Performance: Table IV shows our experimental results. In the experiment, we design different combinations of user information features for comparative analysis. The results indicate that fusing multiple types of features significantly improves the detection performance, surpassing methods that only rely on a single type of feature. Furthermore, we observe that specific types of features perform prominently on different datasets. Notably, Boolean-type features excel in Twibot-20 detection, while text-type features yield the best detection results on MGTAB.

2) Fusion Method Based on Attention Mechanism Outperforms MLP and Average Pooling: Table V shows our experimental results. Specifically, we conduct experiments on RGT, RGCN, and S-HGN graph structures, and the experimental results show that the fusion method based on the attention mechanism outperforms MLP and average pooling. This confirms the superiority of our proposed fusion approach for multiple types of information and our use of the attention fusion mechanism to learn a more comprehensive and integrated representation for the user.

D. Edge Confidence Evaluation Module Study

To assess the effectiveness of the edge confidence evaluation module, we conduct a series of studies. First, we conduct a removal experiment to verify the effect of this module. Next, we conduct experiments on the parametric Gaussian reconstructed edge representation of this module. We then compare the detection performance of BECE unreliable edge removal method and random edge removal method. Additionally, we evaluate the performance of different combinations of edge features. Finally, we compare the removal rate of homogeneous and heterogeneous edges. Based on our findings, we have reached the following conclusions.

1) Excluding the Edge Confidence Evaluation Module Significantly Decreases the Performance of BECE: Fig. 3 illustrates that the edge confidence evaluation module can be applied as a plug-in incorporated into various GNN architectures (such as GNN, GAT, RGT, RGCN, S-HGN, HGT) to enhance the overall performance of the model. In addition, by comparing the results with and without the edge confidence evaluation module (marked as **"W/O ECE"** in Fig. 3), we find that when this module is removed, the accuracy and F1-score of the model significantly decrease. This finding indicates that the module effectively mitigates noise between neighboring nodes, thereby improving the stability and accuracy of the model.

2) Parameterized Gaussian Distribution Can Effectively Improve the Detection Performance of the Model: To obtain a noise-compatible edge representation, we leverage a parameterized Gaussian distribution to reconstruct the edge representation. In Table VI, we assess the impact of removing this component ("**W/O Gaussian**") on the performance of the different BECE (GNN) model. After removal, the accuracy and F1-score on Twibot-20 decrease by a maximum of 1.11% and 0.99%, respectively. Similarly, the accuracy and F1-macro performance drop by a maximum of 1.29% and 1.10% on MGTAB. These results suggest that the parameterized

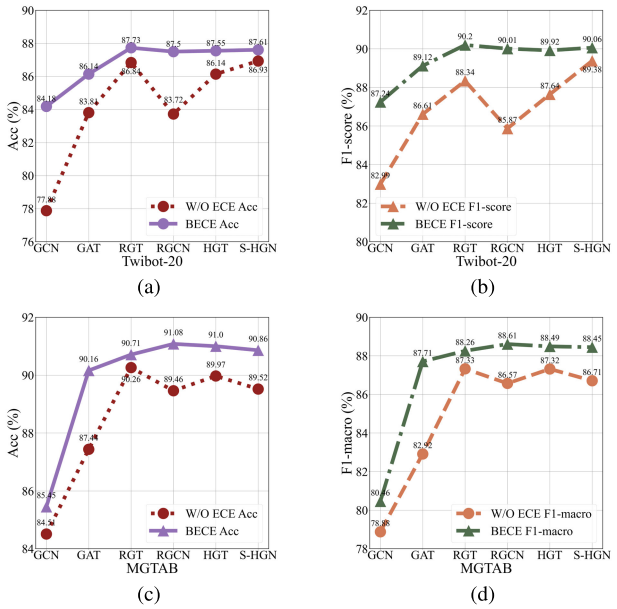

Fig. 3. Edge confidence evaluation module ablation study. "W/O ECE" means the model does not include an edge confidence evaluation module. (a) Twibot-20 Ablation (Acc). (b) Twibot-20 Ablation (F1-score). (c) MGTAB Ablation (Acc). (d) MGTAB Ablation (F1-macro).

TABLE VI

PARAMETERIZED GAUSSIAN RECONSTRUCTION EDGE FEATURE ABLATION. "W/O GAUSSIAN" MEANS REMOVING THE PARAMETERIZED GAUSSIAN COMPONENT

Albation	Twibot-20		MGTAB	
	Acc(%)	F1-score(%)	Acc(%)	F1-macro(%)
***BECE* (GCN)**	**84.18±0.70**	**87.24±0.49**	**85.45±1.60**	**80.46±2.49**
W/O Gaussian	83.49±0.32	86.41±0.34	84.16±1.86	79.71±1.98
***BECE* (GAT)**	**86.14±0.34**	**89.12±0.23**	**90.16±1.47**	**87.71±1.91**
W/O Gaussian	85.24±0.25	88.57±0.24	89.51±1.64	86.61±2.35
***BECE* (HGT)**	**87.55±0.24**	**89.92±0.24**	**91.00±1.41**	**88.49±1.74**
W/O Gaussian	86.44±0.20	88.98±0.26	90.16±1.41	87.58±1.57
***BECE* (RGCN)**	**87.50±0.38**	**90.01±0.28**	**91.08±1.84**	**88.61±2.45**
W/O Gaussian	86.63±0.25	89.02±0.21	90.24±1.88	87.82±2.50
***BECE* (RGT)**	**87.73±0.26**	**90.20±0.18**	**90.71±1.64**	**88.26±2.11**
W/O Gaussian	87.03±0.16	89.62±0.14	90.16±1.41	87.58±1.57
***BECE* (S-HGN)**	**87.61±0.32**	**90.06±0.25**	**90.86±1.78**	**88.45±2.29**
W/O Gaussian	86.95±0.49	89.48±0.23	90.02±1.64	87.66±2.17

Gaussian can effectively regulate noise edge representation, enabling the model to acquire highly robust and discriminative edge features.

3) Well-Designed Unreliable Edges Removal Strategy Outperforms Random Edge Removal: Specifically, we employ the form of randomly deleting a certain proportion of edges to compare with the unreliable edge removal method proposed by BECE. The results of our study, depicted in Fig. 4, clearly demonstrate that our proposed method outperforms random edge removal. This improvement can be attributed to the incorporation of well-designed edge features and the utilization of parametric Gaussian for feature regularization, which enhances the detection performance of BECE in identifying social bots.

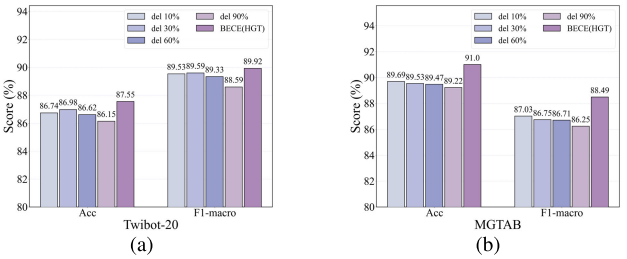

Fig. 4. Randomly remove edges of different proportions to compare with BECE. (a) Random edge removal in Twibot-20. (b) Random edge removal in MGTAB.

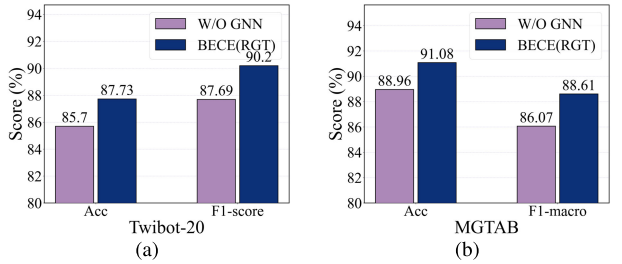

Fig. 5. Node feature update module study. The GNN module can bring significant performance improvement to BECE. (a) GNN removal in Twibot-20. (b) GNN removal in MGTAB.

TABLE VII

DIFFERENT CONSTRUCTION FORMS OF EDGE REPRESENTATION

Dataset	Twibot-20		MGTAB	
	Acc(%)	F1-score(%)	Acc(%)	F1-macro(%)
①(BECE)	**87.50 ± 0.38**	**90.01 ±0.28**	91.08 ± 1.84	88.61 ± 2.45
① + ②	87.45 ± 0.25	89.96 ± 0.18	**91.25 ± 1.50**	**88.94 ± 1.60**
① + ③	87.14 ± 0.31	89.66 ± 0.23	91.06 ± 1.45	88.58 ± 1.98
① + ② + ③	87.21 ±0.23	89.78 ± 0.22	90.88 ± 1.17	88.09 ± 1.91

TABLE VIII

REMOVAL RATES OF EDGES IN BECE

	Edges	Before	After	Removal rate(%)
Twibot-20	(Human, Bot)	7,286	3,545	51.35
	(Bot, Bot)	3,717	2,841	23.57
	(Human, Human)	4,431	3,894	12.12
MGTAB	(Human, Bot)	64,200	35,878	44.12
	(Bot, Bot)	518	334	35.52
	(Human, Human)	655,977	480,057	26.82

4) Subtraction Polymerization Edge Feature Exhibits Excellent Performance: We explore three different features to construct the edge feature: ① L_1-distance between the central node and neighbor nodes; ② the representation of central node and neighbor nodes; and ③ the average pooling feature of the subgraph with the L_1-distance to neighbor nodes. By combining these three features, we construct four types of edge representation. A detailed performance comparison of these four edge construction methods is presented in Table VII. While all four methods have a relatively small impact on the final model's performance, the L_1-distance feature (①) demonstrates the most distinctive characteristics. To enhance training efficiency and reduce the number of parameters required for model training, we choose the ① as the edge representation.

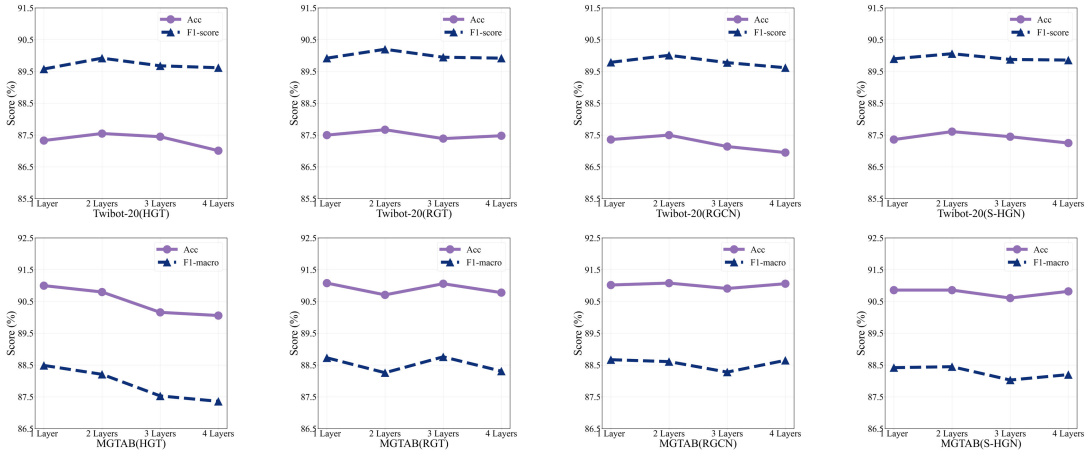

Fig. 6. Ablation of the number of layers of the model GNN. BECE is less affected by the number of GNN layers.

5) Heterogeneous Edges Removal Rate Than Homogeneous Edges: To assess the impact of removing unreliable edges, Table VIII compares the number of edges in the original graph to the remaining edges after removal, calculating the removal rate. The results show that BECE removes heterogeneous edges more effectively than homogeneous edges. However, the removal rate for heterogeneous edges is relatively low in both datasets due to high node pair representation similarity, which challenges the model's ability to discern heterogeneity. Additionally, there is a disparity in removal rates between MGTAB and Twibot-20. On MGTAB, the removal rate for heterogeneous edges is lower compared to Twibot-20, while the removal rate for homogeneous edges shows the opposite trend. This discrepancy may arise from an imbalance in the number of samples between homogeneous and heterogeneous edges in the MGTAB. Future improvements may include balancing the dataset or adopting appropriate sampling strategies to address the issue of edge imbalance.

E. Node Feature Update Module Study

To assess the impact of the node feature update module on the model performance, we first conduct experiments by removing this module. Then, for the purpose of selecting the most appropriate depth for GNN, we conduct an ablation study on the number of GNN layers.

1) GNN Module Can Bring Significant Performance Improvement: In Fig. 5, we present the experimental results. Our study suggests that using graph structure in Twibot-20 and MGTAB can improve the model's detection performance by 2.03% and 2.12% accuracy compared to when graphs are not utilized. This result confirms the effectiveness of constructing graph networks for social bot detection.

2) BECE Has Low Sensitivity to the Number of GNN Layers: We conduct experiments on four heterogeneous GNNs: RGT, RGCN, HGT, and S-HGN. As shown in Fig. 6, our research findings indicate that using a two-layer GNN architecture on top of Twibot-20 achieves the best results. The best performance is achieved in MGTAB using only a one-layer graph structure model. Additionally, we find that the model is less sensitive to the number of GNN layers due to the successful removal of numerous unreliable edges, which reduces the smoothness of node features.

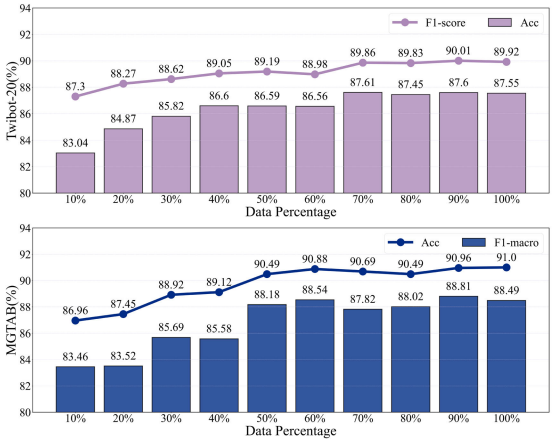

Fig. 7. Model robustness study. We randomly select 10%–100% training sets on TwiBot-20 and MGTAB for training and test the model on the original test set.

F. Data Efficiency Study

Most existing bot detection models are supervised and rely on extensive data annotations. However, due to the large amount of unlabeled data generated by OSNs, it is a great challenge to build a high-quality bot detection system. To verify the efficient data utilization of our proposed model, we conduct a study with different proportions of training sets.

1) Outstanding Performance With Limited Training Data: Specifically, we train the model with a ratio of 10%–100% of the training set and evaluate the performance on the same test set, and for our experiments, we select the HGT aggregator. Fig. 7 shows the experiment results on two datasets, namely Twibot-20 and MGTAB. Experimental results show that using only 70% and 50% of the training data, BECE achieves comparable performance to its full version. Moreover, on Twibot-20 and MGTAB, we acquire better results than the state-of-the-art model RGT [25] with only 30% and 60% of training data used, respectively. These results demonstrate that BECE is less dependent on data, demonstrating the robustness of our model.

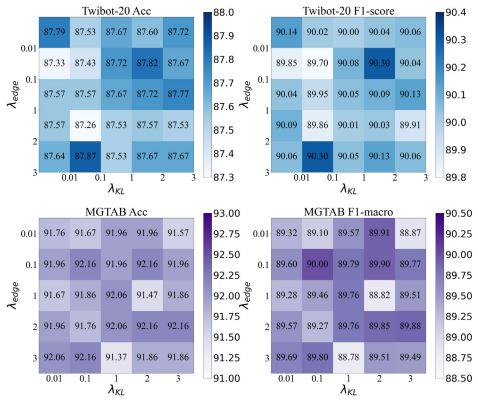

Fig. 8. Hyperparameter study. λ_{edge} is the weight term of edge cross-entropy loss, and λ_{KL} is the KL divergence of edge representation embedding weight item.

Fig. 9. Case study. The confidence of edges connected to humans is much higher than those connected to bots.

G. Hyperparameter Study

In this section, we further conduct experiments on two important balance coefficients λ_{edge} and λ_{KL} to analyze the impact of parameters on the detection performance.

1) Detection Performance Is Less Affected by the Balancing Factors: The impact of balancing factors on detection performance is relatively minor. Specifically, we utilize λ_{edge}, λ_{KL} as regularization terms. λ_{edge} represents the weight term for edge cross-entropy loss, and λ_{KL} is the KL divergence of edge representation weight item. Through the analysis of Fig. 8, we find that in Twibot-20 and MGTAB, the optimal values of the parameters $\{\lambda_{\text{edge}}, \lambda_{\text{KL}}\}$ are $\{3, 0.1\}$ and $\{0.1, 0.1\}$, respectively. In this optimization setting, the cross-entropy loss function of edges allows the model to remove edges with the lowest confidence. The KL divergence loss of edge embedding enables the model to generate a larger variance for edge representations containing more noise and assigns smaller variances to higher-quality edge embeddings. Furthermore, we find that BECE is less affected by these two parameters, which proves the robustness of our proposed model.

H. Case Study

To further understand the impact of the edge confidence evaluation module on assessing the reliability of edges, we present an example of connection evaluation values between bots and humans.

Users With Same Label Exhibit Higher Edge Confidence Than With Different Labels: We evaluate the model using the probability values calculated by (6). Fig. 9 shows two scenarios, scenario ① centered on a human and scenario ② centered on a bot, both of which are connected to two humans and two bots. By examining Fig. 9, it becomes evident that users with the same label have higher confidence in connected edges compared to users with different labels. This proves the effectiveness of the designed edge confidence evaluation model.

VI. Conclusion

We propose BECE, a social bot detection method based on edge confidence evaluation. Specifically, we first design a user information representation module that utilizes attention mechanisms to integrate user profile and text features. Next, we create edge embeddings based on the representations of user nodes and generate robust and distinctive edge representations by reconstructing these embeddings using a parameterized Gaussian approach. We remove these unreliable edges from the graph using the Bernoulli distribution. BECE employs GNN encoders to learn more comprehensive and holistic representations for user nodes on the graph with unreliable edges removed. This helps reduce the smoothness of different types of node representations and enhances bot detection performance. We conduct extensive experiments on three datasets, and the results show that BECE outperforms other comparative methods. Furthermore, by conducting experiments on six different types of graph structures, we demonstrate that our proposed edge confidence evaluation module can be applied as a plug-in to different graph structures.

In the future, we will consider leveraging richer information, such as structural information, to design edge embeddings and propose more effective strategies for removing unreliable edges to assist social bot detection.

References

[1] C. A. Davis, O. Varol, E. Ferrara, A. Flammini, and F. Menczer, "BotOrNot: A system to evaluate social bots," in *Proc. 25th Int. Conf. Companion World Wide Web WWW Companion*, 2016, pp. 273–274.

[2] E. Ferrara, "Social bot detection in the age of ChatGPT: Challenges and opportunities," *FM*, vol. 28, no. 6, Jun. 2023.

[3] Y. Boshmaf, I. Muslukhov, K. Beznosov, and M. Ripeanu, "The socialbot network: When bots socialize for fame and money," in *Proc. 27th Annu. Comput. Secur. Appl. Conf.*, Dec. 2011, pp. 93–102.

[4] E. Ferrara, O. Varol, C. Davis, F. Menczer, and A. Flammini, "The rise of social bots," *Commun. ACM*, vol. 59, no. 7, pp. 96–104, Jun. 2016.

[5] M. Orabi, D. Mouheb, Z. Al Aghbari, and I. Kamel, "Detection of bots in social media: A systematic review," *Inf. Process. Manage.*, vol. 57, no. 4, Jul. 2020, Art. no. 102250.

[6] Z. Ellaky, F. Benabbou, and S. Ouahabi, "Systematic literature review of social media bots detection systems," *J. King Saud Univ. Comput. Inf. Sci.*, vol. 35, no. 5, May 2023, Art. no. 101551.

[7] R. Gilmary, A. Venkatesan, G. Vaiyapuri, and D. Balamurali, "DNA-influenced automated behavior detection on Twitter through relative entropy," *Sci. Rep.*, vol. 12, no. 1, pp. 1–12, May 2022.

[8] L. Luceri, A. Deb, A. Badawy, and E. Ferrara, "Red bots do it better: Comparative analysis of social bot partisan behavior," in *Proc. Companion World Wide Web Conf.*, May 2019, pp. 1007–1012.

[9] G. Guglielmi, "The next-generation bots interfering with the U.S. election," *Nature*, vol. 587, no. 7832, p. 21, Nov. 2020.

[10] E. Ferrara, "What types of COVID-19 conspiracies are populated by Twitter bots?" *FM*, vol. 25, no. 6, May 2020.

[11] M. Himelein-Wachowiak et al., "Bots and misinformation spread on social media: Implications for COVID-19," *J. Med. Internet Res.*, vol. 23, no. 5, May 2021, Art. no. e26933.

[12] S. Cresci, F. Lillo, D. Regoli, S. Tardelli, and M. Tesconi, "FAKE: Evidence of spam and bot activity in stock microblogs on Twitter," in *Proc. Int. AAAI Conf. Web Social Media*, 2018, vol. 12, no. 1, pp. 580–583.

[13] L. Nizzoli, S. Tardelli, M. Avvenuti, S. Cresci, M. Tesconi, and E. Ferrara, "Charting the landscape of online cryptocurrency manipulation," *IEEE Access*, vol. 8, pp. 113230–113245, 2020.

[14] S. Cresci, "A decade of social bot detection," *Commun. ACM*, vol. 63, no. 10, pp. 72–83, Sep. 2020.

[15] D. M. Beskow and K. M. Carley, "Its all in a name: Detecting and labeling bots by their name," *Comput. Math. Org. Theory*, vol. 25, no. 1, pp. 24–35, Mar. 2019.

[16] K.-C. Yang, O. Varol, P.-M. Hui, and F. Menczer, "Scalable and generalizable social bot detection through data selection," in *Proc. AAAI Conf. Artif. Intell.*, vol. 34, no. 1, 2020, pp. 1096–1103.

[17] S. Kudugunta and E. Ferrara, "Deep neural networks for bot detection," *Inf. Sci.*, vol. 467, pp. 312–322, Oct. 2018.

[18] F. Wei and U. T. Nguyen, "Twitter bot detection using bidirectional long short-term memory neural networks and word embeddings," in *Proc. 1st IEEE Int. Conf. Trust, Privacy Secur. Intell. Syst. Appl. (TPS-ISA)*, Dec. 2019, pp. 101–109.

[19] K. Hayawi, S. Mathew, N. Venugopal, M. M. Masud, and P.-H. Ho, "DeeProBot: A hybrid deep neural network model for social bot detection based on user profile data," *Social Netw. Anal. Mining*, vol. 12, no. 1, p. 43, Dec. 2022.

[20] D. Dukić, D. Keča, and D. Stipić, "Are you human? Detecting bots on Twitter using BERT," in *Proc. IEEE 7th Int. Conf. Data Sci. Adv. Anal. (DSAA)*, Oct. 2020, pp. 631–636.

[21] G. Stanton and A. A. Irissappane, "GANs for semi-supervised opinion spam detection," in *Proc. 28th Int. Joint Conf. Artif. Intell.*, Macao, China, Aug. 2019, pp. 5204–5210.

[22] Z. Lei et al., "BIC: Twitter bot detection with text-graph interaction and semantic consistency," in *Proc. 61st Annu. Meeting Assoc. Comput. Linguistics (Long Papers)*, vol. 1, 2023, pp. 10326–10340.

[23] B. Qiao, K. Li, W. Zhou, Z. Yan, S. Li, and S. Hu, "Social bot detection based on window strategy," in *Proc. IEEE Int. Conf. Multimedia Expo. (ICME)*, Jul. 2023, pp. 2201–2206.

[24] S. Feng, H. Wan, N. Wang, and M. Luo, "BotRGCN: Twitter bot detection with relational graph convolutional networks," in *Proc. IEEE/ACM Int. Conf. Adv. Social Netw. Anal. Mining*, Nov. 2021, pp. 236–239.

[25] S. Feng, Z. Tan, R. Li, and M. Luo, "Heterogeneity-aware Twitter bot detection with relational graph transformers," in *Proc. AAAI Conf. Artif. Intell.*, vol. 36, no. 4, 2022, pp. 3977–3985.

[26] Y. Liu, Z. Tan, H. Wang, S. Feng, Q. Zheng, and M. Luo, "BotMoE: Twitter bot detection with community-aware mixtures of modal-specific experts," in *Proc. 46th Int. ACM SIGIR Conf. Res. Develop. Inf. Retr.*, Taipei, Taiwan, Jul. 2023, pp. 485–495.

[27] M. Schlichtkrull, T. N. Kipf, P. Bloem, R. van den Berg, I. Titov, and M. Welling, "Modeling relational data with graph convolutional networks," in *Proc. Eur. Semantic Web Conf.*, 2018, pp. 593–607.

[28] Y. Shi, H. Zhengjie, S. Feng, H. Zhong, W. Wang, and Y. Sun, "Masked label prediction: Unified message passing model for semi-supervised classification," in *Proc. Int. Joint Conf. Artif. Intell. Org.*, Aug. 2021, pp. 1548–1554.

[29] T. N. Kipf and M. Welling, "Semi-supervised classification with graph convolutional networks," in *Proc. 5th Int. Conf. Learn. Represent.*, Toulon, France, Apr. 2017, pp. 1–7

[30] D. Chen, Y. Lin, W. Li, P. Li, J. Zhou, and X. Sun, "Measuring and relieving the over-smoothing problem for graph neural networks from the topological view," in *Proc. AAAI Conf. Artif. Intell.*, vol. 34, no. 4, 2020, pp. 3438–3445.

[31] Y. Yang et al., "RoSGAS : Adaptive social bot detection with reinforced self-supervised GNN architecture search," *ACM Trans. Web*, vol. 17, no. 3, pp. 1–31, Aug. 2023.

[32] D. M. Beskow and K. M. Carley, "Bot conversations are different: Leveraging network metrics for bot detection in Twitter," in *Proc. IEEE/ACM Int. Conf. Adv. Social Netw. Anal. Mining (ASONAM)*, Aug. 2018, pp. 825–832.

[33] S. H. Moghaddam and M. Abbaspour, "Friendship preference: Scalable and robust category of features for social bot detection," *IEEE Trans. Depend. Secure Comput.*, vol. 20, no. 2, pp. 1516–1528, Mar. 2023.

[34] P. G. Efthimion, S. Payne, and N. Proferes, "Supervised machine learning bot detection techniques to identify social Twitter bots," *SMU Data Sci. Rev.*, vol. 1, no. 2, p. 5, 2018.

[35] S. Feng, H. Wan, N. Wang, J. Li, and M. Luo, "SATAR: A self-supervised approach to Twitter account representation learning and its application in bot detection," in *Proc. 30th ACM Int. Conf. Inf. Knowl. Manage.*, Oct. 2021, pp. 3808–3817.

[36] J. Pan, Y. Liu, X. Liu, and H. Hu, "Discriminating bot accounts based solely on temporal features of microblog behavior," *Phys. A, Stat. Mech. Appl.*, vol. 450, pp. 193–204, May 2016.

[37] C. M. Zhang and V. Paxson, "Detecting and analyzing automated activity on Twitter," in *Proc. Int. Conf. Passive Act. Netw. Meas.* Cham, Switzerland: Springer, 2011, pp. 102–111.

[38] P. Shi, Z. Zhang, and K. R. Choo, "Detecting malicious social bots based on clickstream sequences," *IEEE Access*, vol. 7, pp. 28855–28862, 2019.

[39] S. Cresci, R. Di Pietro, M. Petrocchi, A. Spognardi, and M. Tesconi, "DNA-inspired online behavioral modeling and its application to spambot detection," *IEEE Intell. Syst.*, vol. 31, no. 5, pp. 58–64, Sep. 2016.

[40] N. Chavoshi, H. Hamooni, and A. Mueen, "DeBot: Twitter bot detection via warped correlation," in *Proc. IEEE 16th Int. Conf. Data Mining (ICDM)*, vol. 18, Dec. 2016, pp. 28–65.

[41] E. D. Paolo, M. Petrocchi, and A. Spognardi, "From online behaviours to images: A novel approach to social bot detection," 2023, *arXiv:2304.07535*.

[42] A. Addawood, A. Badawy, K. Lerman, and E. Ferrara, "Linguistic cues to deception: Identifying political trolls on social media," in *Proc. Int. AAAI Conf. Web Social Media*, vol. 13, 2019, pp. 15–25.

[43] M. Fazil, A. K. Sah, and M. Abulaish, "DeepSBD: A deep neural network model with attention mechanism for socialbot detection," *IEEE Trans. Inf. Forensics Security*, vol. 16, pp. 4211–4223, 2021.

[44] C. Grimme, J. Pohl, S. Cresci, R. Lüling, and M. Preuss, "New automation for social bots: From trivial behavior to ai-powered communication," in *Proc. Multidisciplinary Int. Symp. Disinformation Open Online Media*. Cham, Switzerland: Springer, 2022, pp. 79–99.

[45] C. Cai, L. Li, and D. Zeng, "Detecting social bots by jointly modeling deep behavior and content information," in *Proc. ACM Conf. Inf. Knowl. Manage.*, Nov. 2017, pp. 1995–1998.

[46] W. Hamilton, Z. Ying, and J. Leskovec, "Inductive representation learning on large graphs," in *Proc. Adv. Neural Inf. Process. Syst.*, vol. 30, 2017, pp. 1–11.

[47] S. Brody, U. Alon, and E. Yahav, "How attentive are graph attention networks?" in *Proc. Int. Conf. Learn. Represent.*, 2021, pp. 1–12.

[48] P. Velickovic et al., "Graph attention networks," *Int. Conf. Learn. Represent.*, vol. 1050, no. 20, 2017, Art. no. 48550.

[49] Z. Wu, S. Pan, F. Chen, G. Long, C. Zhang, and S. Y. Philip, "A comprehensive survey on graph neural networks," *IEEE Trans. Neural Netw. Learn. Syst.*, vol. 32, no. 1, pp. 4–24, Mar. 2020.

[50] S. Ye et al., "HOFA: Twitter bot detection with homophily-oriented augmentation and frequency adaptive attention," 2023, *arXiv:2306.12870*.

[51] S. Li, B. Qiao, K. Li, Q. Lu, M. Lin, and W. Zhou, "Multi-modal social bot detection: Learning homophilic and heterophilic connections adaptively," in *Proc. 31st ACM Int. Conf. Multimedia*, Oct. 2023, pp. 3908–3916.

[52] A. Li, Z. Qin, R. Liu, Y. Yang, and D. Li, "Spam review detection with graph convolutional networks," in *Proc. 28th ACM Int. Conf. Inf. Knowl. Manage.*, Beijing, China, Nov. 2019, pp. 2703–2711.

[53] H. Peng, R. Zhang, Y. Dou, R. Yang, J. Zhang, and P. S. Yu, "Reinforced neighborhood selection guided multi-relational graph neural networks," *ACM Trans. Inf. Syst.*, vol. 40, no. 4, pp. 1–46, Oct. 2022.

[54] Y. Dou, Z. Liu, L. Sun, Y. Deng, H. Peng, and P. S. Yu, "Enhancing graph neural network-based fraud detectors against camouflaged fraudsters," in *Proc. 29th ACM Int. Conf. Inf. Knowl. Manage.*, 2020, pp. 315–324.

[55] T. Zhao, Y. Liu, L. Neves, O. Woodford, M. Jiang, and N. Shah, "Data augmentation for graph neural networks," in *Proc. AAAI Conf. Artif. Intell.*, vol. 35, no. 12, 2021, pp. 11015–11023.

[56] Y. Liu et al., "RoBERTa: A robustly optimized BERT pretraining approach," 2019, *arXiv:1907.11692*.

[57] F. Feng, Y. Yang, D. Cer, N. Arivazhagan, and W. Wang, "Language-agnostic BERT sentence embedding," in *Proc. 60th Annu. Meeting Assoc. Comput. Linguistics*, 2022, pp. 878–891.

[58] J. Chang, Z. Lan, C. Cheng, and Y. Wei, "Data uncertainty learning in face recognition," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jun. 2020, pp. 5710–5719.
[59] D. P. Kingma, T. Salimans, and M. Welling, "Variational dropout and the local reparameterization trick," in *Proc. Adv. Neural Inf. Process. Syst.*, vol. 28, 2015, pp. 1–7.
[60] W. S. Mcculloch and W. Pitts, "A logical calculus of the ideas immanent in nervous activity," *Bull. Math. Biophys.*, vol. 5, no. 4, pp. 115–133, 1943.
[61] G. Shafer, "The significance of Jacob Bernoulli's Ars Conjectandi for the philosophy of probability today," *J. Econometrics*, vol. 75, no. 1, pp. 15–32, Nov. 1996.
[62] S. Kullback and R. A. Leibler, "On information and sufficiency," *Ann. Math. Statist.*, vol. 22, no. 1, pp. 79–86, 1951.
[63] S. Cresci, R. Di Pietro, M. Petrocchi, A. Spognardi, and M. Tesconi, "Fame for sale: Efficient detection of fake Twitter followers," *Decis. Support Syst.*, vol. 80, pp. 56–71, Dec. 2015.
[64] S. Feng, H. Wan, N. Wang, J. Li, and M. Luo, "TwiBot-20: A comprehensive Twitter bot detection benchmark," in *Proc. 30th ACM Int. Conf. Inf. Knowl. Manage.*, Oct. 2021, pp. 4485–4494.
[65] S. Shi et al., "MGTAB: A multi-relational graph-based Twitter account detection benchmark," 2023, *arXiv:2301.01123*.
[66] S. Cresci, R. D. Pietro, M. Petrocchi, A. Spognardi, and M. Tesconi, "The paradigm-shift of social spambots: Evidence, theories, and tools for the arms race," in *Proc. 26th Int. Conf. World Wide Web Companion (WWW Companion)*, 2017, pp. 963–972.
[67] K. Yang, O. Varol, C. A. Davis, E. Ferrara, A. Flammini, and F. Menczer, "Arming the public with artificial intelligence to counter social bots," *Hum. Behav. Emerg. Technol.*, vol. 1, no. 1, pp. 48–61, Jan. 2019.
[68] M. Mazza, S. Cresci, M. Avvenuti, W. Quattrociocchi, and M. Tesconi, "RTbust: Exploiting temporal patterns for botnet detection on Twitter," in *Proc. 10th ACM Conf. Web Sci.*, 2019, pp. 183–192.
[69] S. Feng et al., "TwiBot-22: Towards graph-based Twitter bot detection," in *Proc. 36th Conf. Neural Inf. Process. Syst. Datasets Benchmarks Track*, 2022, pp. 1–25.
[70] M. Fey and J. E. Lenssen, "Fast graph representation learning with PyTorch geometric," 2019, *arXiv:1903.02428*.
[71] Z. Hu, Y. Dong, K. Wang, and Y. Sun, "Heterogeneous graph transformer," in *Proc. Web Conf.*, New York, NY, USA, Apr. 2020, pp. 2704–2710.
[72] Q. Lv et al., "Are we really making much progress? Revisiting, benchmarking and refining heterogeneous graph neural networks," in *Proc. 27th ACM SIGKDD Conf. Knowl. Discovery Data Mining*, Aug. 2021, pp. 1150–1160.

Boyu Qiao received the bachelor's degree from Hebei University of Technology, Tianjin, China, in 2021. She is currently pursuing the Ph.D. degree with the Institute of Information Engineering, Chinese Academy of Sciences, Beijing, China, and the School of Cyber Security, University of Chinese Academy of Sciences, Beijing.

Her main research interests include natural language processing and social network analysis.

Wei Zhou received the Ph.D. degree from the Institute of Information Engineering, Chinese Academy of Sciences, Beijing, China, in 2019.

She is currently a Senior Engineer with the Institute of Information Engineering, Chinese Academy of Sciences. Her main research interests include fake news detection and data mining.

Kun Li received the master's degree from the Institute of Scientific and Technical Information of China, Beijing, China, in 2018. He is currently pursuing the Ph.D. degree with the Institute of Information Engineering, Chinese Academy of Sciences, Beijing, and the School of Cyber Security, University of Chinese Academy of Sciences, Beijing.

His main research interests include natural language processing and social network analysis.

Shilong Li received the bachelor's degree from Jilin University, Changchun, China, in 2022. He is currently pursuing the master's degree with the Institute of Information Engineering, Chinese Academy of Sciences, Beijing, China, and the School of Cyber Security, University of Chinese Academy of Sciences, Beijing.

His main research interests include natural language processing and social network analysis.

Songlin Hu received the Ph.D. degree from Beihang University, Beijing, China, in 2001.

In 2002, he was promoted to Associate Professor at the Institute of Computing Technology, Chinese Academy of Sciences, Beijing. He is currently a Professor at the Institute of Information Engineering, Chinese Academy of Sciences, and the School of Cyber Security, University of Chinese Academy of Sciences, Beijing. His main research interests include misinformation detection, intelligent processing, and knowledge graphs.