Unsupervised Social Bot Detection via Structural Information Theory

HAO PENG*, Beihang University, China
JINGYUN ZHANG, Beihang University, China
XIANG HUANG, Beihang University, China
ZHIFENG HAO, University of Shantou, China
ANGSHENG LI, Beihang University, China
ZHENGTAO YU, Kunming University of Science and Technology, China
PHILIP S. YU, University of Illinois at Chicago, USA

Research on social bot detection plays a crucial role in maintaining the order and reliability of information dissemination while increasing trust in social interactions. The current mainstream social bot detection models rely on black-box neural network technology, e.g., Graph Neural Network, Transformer, etc., which lacks interpretability. In this work, we present UnDBot, a novel unsupervised, interpretable, yet effective and practical framework for detecting social bots. This framework is built upon structural information theory. We begin by designing three social relationship metrics that capture various aspects of social bot behaviors: *Posting Type Distribution*, *Posting Influence*, and *Follow-to-follower Ratio*. Three new relationships are utilized to construct a new, unified, and weighted social multi-relational graph, aiming to model the relevance of social user behaviors and discover long-distance correlations between users. Second, we introduce a novel method for optimizing heterogeneous structural entropy. This method involves the personalized aggregation of edge information from the social multi-relational graph to generate a two-dimensional encoding tree. The heterogeneous structural entropy facilitates decoding of the substantial structure of the social bots network and enables hierarchical clustering of social bots. Thirdly, a new community labeling method is presented to distinguish social bot communities by computing the user's stationary distribution, measuring user contributions to network structure, and counting the intensity of user aggregation within the community. Compared with ten representative social bot detection approaches, comprehensive experiments demonstrate the advantages of effectiveness and interpretability of UnDBot on four real social network datasets.

CCS Concepts: • **Information systems → Information systems applications**; • **Social and professional topics → User characteristics**; • **Computing methodologies → Artificial intelligence**.

*This is the corresponding author.

Authors' addresses: H. Peng, School of Cyber Science and Technology, Beihang University, No. 37 Xue Yuan Road, Haidian District, Beijing, 100191, China, and State Key Laboratory of Public Big Data, Guizhou University, No. 2708, South Section of Huaxi Avenue, Huaxi District, Guiyang City, Guizhou Province, 550025, China, and Guangxi Key Lab of Multi-source Information Mining & Security, Guangxi Normal University, Guilin 541004, China, and Yunnan Key Laboratory of Artificial Intelligence, Kunming University of Science and Technology, Kunming 650500, China; email: penghao@buaa.edu.cn; J. Zhang, School of Cyber Science and Technology, Beihang University, No. 37 Xue Yuan Road, Haidian District, Beijing, 100191, China; email: zhangjingyun@buaa.edu.cn; X. Huang, School of Cyber Science and Technology, Beihang University, No. 37 Xue Yuan Road, Haidian District, Beijing, 100191, China; email: huang.xiang@buaa.edu.cn; Z. Hao, College of Science, University of Shantou, No. 243, University Road, Shantou, 515063, China; email: haozhifeng@stu.edu.cn; A. Li, School of Computer Science and Engineering, Beihang University, No. 37 Xue Yuan Road, Haidian District, Beijing, 100191, China; email: angsheng@buaa.edu.cn; Z. Yu, Faculty of Information Engineering and Automation, and Yunnan Key Laboratory of Artificial Intelligence, Kunming University of Science and Technology, Kunming 650500, China; email: ztyu@hotmail.com; P. S. Yu, Department of Computer Science, University of Illinois at Chicago, Chicago 60607, IL; email: psyu@uic.edu.

1046-8188/2024/1-ART1 $15.00
https://doi.org/10.1145/3660522

Additional Key Words and Phrases: Social bot detection, structural entropy, multi-relational graph, stationary distribution, interpretability

ACM Reference Format:
Hao Peng, Jingyun Zhang, Xiang Huang, Zhifeng Hao, Angsheng Li, Zhengtao Yu, and Philip S. Yu. 2024. Unsupervised Social Bot Detection via Structural Information Theory. *ACM Trans. Inf. Syst.* 1, 1, Article 1 (January 2024), 42 pages. https://doi.org/10.1145/3660522

1 INTRODUCTION

Social bots are usually controlled by programs, pretending to be humans to publish harmful and low-credibility information [55, 67, 70], and even manipulate or guide public behavior in social networks [39, 81] to reduce social trust and disrupt the orderly dissemination of information. For example, spreading misinformation about COVID-19 has triggered an "information epidemic" [23, 34]. Nowadays, the development of Artificial Intelligence Content Generator technology [99, 102] also makes the competition between social bot detection and anti-detection more intense [43, 83, 92, 97]. It has been shown that the more human-like a bot account behaves, the more likely users are to interact with it [82], which makes it harder for social bots to be detected from social networks. Even in public emergencies, the significant volume of low-credibility information and objectionable content spread by social bots has the potential to manipulate public emotions and disrupt the trajectory of Internet public opinion [2, 14, 72]. Studies also have shown that the content disseminated by social bots primarily adopts a critical, correct, and questioning tone [12, 73], aiming to polarize netizens' speech and disrupt the orderly dissemination of information [62]. Therefore, the research of detecting social bots is of great significance to protecting the orderly dissemination of information and maintaining social trust [29, 41, 51].

Effective and reliable social bot detection approaches need adequate modeling, representation, and analysis of social user behaviors [86]. However, accomplishing such a task is challenging due to the varied and dynamic interactive behaviors of social bots [15, 16, 58]. Consequently, relying solely on social bot detection methods based on manual feature engineering [5, 38, 40] and traditional machine learning classifiers, such as Logistic regression [5], K-means [56], SVM [24], naive Bayesian [25], Random forest [90], etc., practical detection accuracy becomes bottlenecked. Expressly, limited by the simple and discrete low-dimensional manual features, such as *username length*, *the number of followers*, *the number of tweets*, *the number of likes*, etc., the above model's performance falls short of the ideal [27]. Deep learning-based social bot detection models have made significant advancements in learning feature embeddings from metadata, including Graph Neural Network (GNN)-based [1, 4, 6, 28, 32, 49, 77, 91] and Transformer-based [26, 33, 53] models. The former enables improving social bot detection performance by understanding semantic relationships in the neighborhood through transmitting information between users and learning embedded user characteristics. The latter aggregates user influence through multiple relationships and learns critical discriminative features for bot detection through self-attention mechanisms. Moreover, they are capable of better understanding contextual information. Additionally, the federated knowledge distillation-based bot detection model [92] is employed for the cross-platform and cross-language social bots. While the aforementioned data-driven representative models are deemed sufficient, they necessitate a substantial number of labeled social user samples, posing a risk to their generalization ability. The behavior of social bots evolves in a realistic environment, making the labeling process challenging and rendering supervised/semi-supervised models less effective for out-of-sample data [88].

Existing unsupervised social bot detection models [3, 10, 13, 52, 54, 56] typically rely on identifying time series, simple clustering methods, or explicit behavioral markers that are exclusive to social bots. These markers include *highly synchronized*, *repeated tweets*, *URL shortening services*, *retweeting behaviors*, etc. However, contemporary social bots have exhibited enhanced intelligence and proficiency in disguising their objectives and concealing their authentic identity. Therefore, the behavioral indicators for detecting bots are only sometimes apparent or discernible. Moreover, the behavioral markers are always low-order and discrete, often describing superficial

behaviors that undermine the accuracy of detecting social bots. Therefore, it is essential to develop effective, practical, interpretable, and unsupervised social bot detection models by utilizing user behavior data to their fullest potential to govern social bots effectively.

The significance of social network structural features in social bot detection cannot be underestimated, as they offer valuable insights into user interaction patterns and information dissemination. The local social network contains valuable information that is crucial for identifying potential social bots [22]. Despite this, structure-based detection methods are not yet prevalent, and the primary strategy for social bot detection is random walk propagation of user labels [37, 75]. To create directed graphs, social network modeling primarily relies on direct social relationships between users, such as following, commenting, liking, and sharing. However, most downstream tasks after graph modeling still rely on neural networks [46, 50], which have black-box characteristics [47, 69] and lack interpretability due to the absence of a direct causal logical relationship between input and output features. In addition, developing interpretable social bot detection models can significantly enhance the detection process's reliability by improving our understanding of how the model works. Therefore, it is highly imperative to design a new unsupervised social bot detection framework based on user behavior data and graph structure with high effectiveness and interpretability.

In this work, we propose UnDBot, an effective, practical, Unsupervised, and interpretable Detection framework of social Bots based on structural information theory. Our framework seeks to reveal the significant structure of social bot networks, thereby achieving hierarchical clustering from graph to tree and detection in an unsupervised manner. Firstly, we construct a multi-relational graph from a social bot's perspective, representing users as nodes and depicting heterogeneous social behavior commonalities as edges. This approach enables us to define new types of social relationships that represent the similarity of users in their behavioral characteristics, such as posting type distribution, posting influence, and follow-to-follower ratio. Unlike traditional graphs that rely on direct user interactions like following and retweeting, this multi-relational graph gives more weight to the hidden commonalities of social behaviors among users. Secondly, we present a new heterogeneous structural entropy optimization method to partition social users into distinct communities. By aggregating the various types of relationships in the multi-relational graph of social users, we extend the structural entropy [45] from the simple graph to the multi-relational graph. A new encoding tree is constructed and optimized to minimize the multi-relational graph's structural entropy, from which hierarchical community partitions of social users are provided. Thirdly, we propose a novel community labeling method combining community influence and cohesion to identify social bot communities. We employ Multirank [59] to calculate the co-ranking of vertices and multi-relational edges in the graph, obtaining a stationary distribution to quantify the influence of social user communities. The entropy of community nodes on the encoding tree is also utilized to quantify community cohesion. Combining community influence and cohesion distinguishes social bot communities from normal human communities.

We conduct extensive experiments on four datasets, Cresci-2015 [17], Cresci-2017 [20], Pronbots-2019, and Botwiki-2019 [90], to demonstrate the effectiveness, interpretability, and efficiency of UnDBot. More additional human accounts and tweets are added to the Pronbots-2019 and Botwiki-2019 datasets to make the detection experiments more realistic. First, the comparative experimental results indicate that UnDBot demonstrates superior overall performance compared to existing unsupervised social bot detection models and models based on unsupervised network representation learning. It significantly enhances the accuracy of social bot detection. Second, a series of ablation experiments demonstrate the necessity and rationality of the graph modeling introduced in the UnDBot. Each type of edge contributes to the performance improvement of UnDBot, and the proposed multi-relational graph significantly enhances accuracy compared to other modeling approaches. Third, the experiment of time analysis further illustrates the balance between accuracy and efficiency of UnDBot. Finally,

visualizations for the model effect showcase the interpretability of UnDBot. All codes and datasets of this work are publicly available at GitHub [1].

The main contributions of this work are summarized as follows:

- An unsupervised and interpretable social bot detection framework is proposed with high accuracy that decodes the significant structural features of the network using structural information theory.
- A new, unified, and weighted social multi-relational graph is devised based on social bot behavioral similarity, which breaks through the traditional approach solely on direct user interactions and better models the activity of social bot users.
- A new heterogeneous structural entropy optimization method is proposed that aggregates different types of edges by assigning personalized weights to edges to achieve hierarchical community partitioning of social users.
- A new community labeling method is developed that involves integrating community influence (measured by stationary distribution) and community cohesion (measured by node entropy) to identify social bot communities with higher accuracy.
- A series of comparative and analytical experiments demonstrate that UnDBot achieves high detection accuracy and comprehensively analyzes the model's interpretability.

The structure of this paper is as follows: Section 2 outlines the background and preliminaries of our work. In Section 3, we describe the technical details of the proposed framework, named UnDBot. Section 4 presents the experimental setup, and Section 5 discusses the experiment's results. Section 6 provides an overview of related works. Finally, we conclude the paper in Section 7.

2 BACKGROUND AND PRELIMINARIES

In this section, we first summarize the problems and challenges of social bot detection. Then we introduce the structural information theory used in our social bot detection framework and elaborate on the basic concept of structural entropy. The comprehensive list of the primary symbols used throughout this paper is presented in Table 1.

2.1 Problem and Challenges

Intuitively, we model with multi-relational graphs for practical tasks to transform the social bot detection into an unsupervised vertex classification problem. The vertices in the graph can be hierarchically clustered using a two-dimensional structural entropy minimum algorithm. Once the social user community is formed, the next task is to classify the community. Overall, to achieve effective, unsupervised, and interpretable user vertices classification, social bot detection faces the following 3 main challenges.

Challenge 1: How to model social user networks to serve social bot clustering task?

Most of the current graph modeling approaches create edges between users based on their direct interactions on social networks, such as liking, commenting, retweeting, favoriting, following/being followed, and mentioning. Although these techniques can intuitively capture the interactions among social users and facilitate neighbor aggregation of the graph neural network to learn high-order user features, they have minimal impact on detecting social bots from a network structure perspective. Given the increasing intelligence of social bots, their interactions with actual human users are becoming more commonplace. Many social bots infiltrate typical human social networks, which are hard to detect via social network structure information modeled solely by interaction relationships [29]. Hence, it is necessary to define new types of social relationships and model a social user network from the hidden behavior commonality of social bots.

Challenge 2: How to achieve adaptive hierarchical clustering of social users?

[1] https://github.com/SELGroup/UnDBot

Currently, two types of clustering algorithms are used in social bot detection: feature-based clustering algorithms and network structure-based community detection algorithms [11], such as K-nearest neighbors, density clustering, maximum flow minimum cut theory, etc. However, these traditional models exhibit a singular level of user clustering, wherein the predetermined number of clusters can potentially influence the efficacy of clustering. Deep clustering methods based on neural networks lack interpretability. Additionally, the distance measurement method employed may not be entirely suitable. The structural information theory provides an adaptive

Table 1. Forms and interpretations of notations.

Symbol	Definition
$\mathcal{G}; \mathcal{G}^r$	Homogeneous Graph; Multi-relational Graph (heterogeneous graph).
$\mathcal{V}; \mathcal{E}; \mathcal{E}_k$	Vertex set; Edge set of Homogeneous Graph; Edge set under k-th relationship.
E	The connected user pairs in graph $\mathcal{G}^r$.
R	Total number of relationships.
$v; N$	Vertex in graph; Total number of vertices.
$e^k_{i,j}$	Edge between vertex i and vertex j under k-th relationship.
$w^k_{i,j}$	Weight of edge $e^k_{i,j}$ between vertex i and vertex j under the k-th relationship.
$\mathcal{T}; \lambda$	Encoding tree; The root node of the encoding tree.
$\alpha; T_\alpha$	Node on encoding tree; Label of node α.
$\alpha_i; \alpha^-$	i-th child node of node α; Parent node of node.
$d_i; g_\alpha$	Degree of vertex i; Number of cutting edges of node α.
$vol(\mathcal{G}^r); vol(\alpha)$	Volume of Graph $\mathcal{G}$; Volume of node α.
$H^{\mathcal{T}}(\mathcal{G})$	The structural entropy of $\mathcal{G}$ under encoding tree $\mathcal{T}$.
$H_k(\mathcal{G})$	The k-dimensional structure entropy.
$H(\mathcal{G}^r; \alpha)$	The structural entropy of node α on an encoding tree.
$Mg(\mathcal{T}; \alpha, \beta)$	Merging operator between node α and node β.
$\mathcal{T}_{mg}$	Encoding tree after Merging operator.
$\Delta^{Mg}_{\mathcal{G}^r}(\mathcal{T}; \alpha, \beta)$	Difference of Structural entropy after merging node α and β.
p	The maximum scale ratio for parallel merge operators.
$Pt; Inf; ff$	Distribution of posting types; Influence of posting; Following-followers ratio.
$d_{i,j}$	Manhattan distance between posting type distribution between user i and user j.
$\Delta_{i,j}$	Deviation ratio of influence between user i and user j.
ξ	Threshold of feature similarity in modeling.
$\mathcal{A}$	A three-dimensional tensor of multi-relational graph $\mathcal{G}^r$.
$\mathcal{O}$	Tensor of transition probabilities to reach different vertices.
$\mathcal{S}$	Tensor of transition probabilities through different edges.
$o_{i,j,k}$	The possibility of reaching vertex v_j, from vertex v_i through k-th edge.
$s_{i,j,k}$	The possibility of going through k-th edge, from vertex v_i to vertex v_j.
ρ	The stop threshold in random walk.
$\mathcal{X}; \mathcal{Y}$	Stationary distribution of users; stationary distribution of edges.
x^α_{comm}	The community influence.
$Ev(\alpha)$	Evaluation score of the community represented by node α.
θ	Threshold of evaluation index for communities.
π	Weighted parameters of influence and cohesion.

hierarchical community division of social users by building a structural entropy encoding tree and decoding the essential structure of social networks. However, the current structural entropy optimization strategy [45] is aimed at homogeneous networks and does not consider the heterogeneity of user relationships in social network environments. Therefore, an effective structure entropy optimization strategy under multi-relational graphs is needed to achieve adaptive hierarchical clustering of social users.

Challenge 3: How to identify social user communities and implement binary classification?

Theoretically, the encoding tree of a multi-relational graph can only achieve the effect of clustering users. It cannot directly separate users into two large social bots and human communities. Moreover, the number of communities formed on the encoding tree depends on the structure of the multi-relational graph. Although humans and social bots occupy separate communities on the encoding tree, it is necessary to study the differences between different types of user communities and employ appropriate discrimination methods to explicitly identify social bot communities in the absence of anchor markers or any labels. Unsupervised and interpretable discriminative behavioral features play a more important role in realistic social bot detection tasks.

2.2 Structural Information Theory

Structural information theory [45] was originally proposed in 2016 for measuring the structural information contained within a graph. Specifically, this theory aims to calculate the structural entropy of the homogeneous graph $\mathcal{G} = (\mathcal{V}, \mathcal{E})$, which reflects its uncertainty when undergoing hierarchical division. In our work, the hierarchical partitions are represented by a tree structure known as the encoding tree. We introduce encoding trees and k-dimensional structural entropy below.

Encoding tree. Similar to the previous study [96], an encoding tree of a graph $\mathcal{G}$ is defined as a rooted tree $\mathcal{T}$ with the following properties:

(1) For each node α in the encoding tree $\mathcal{T}$, there is a subset $T_\alpha \in \mathcal{V}$ of vertices in the graph $\mathcal{G}$ corresponding to it.
(2) For the root node λ in the encoding tree, $T_\lambda = \mathcal{V}$.
(3) The children of node α are denoted as α_i and sorted from left to right as i increases. The parent node of α_i is denoted as $\alpha_i^- = \alpha$.
(4) If node α has L children, then the vertex subset T_{α_i} of child nodes is mutually exclusive, and $T_\alpha = \cup T_{\alpha_i}$.
(5) Each leaf node v in the tree, v corresponds to a single vertex in the vertex set $\mathcal{V}$ in the graph $\mathcal{G}$.

The k-dimensional encoding tree means that the tree's height is k (the height of the root node is 0). Intuitively, the encoding tree embodies the hierarchical community division of graph vertices, the parent node is a large community, and the child nodes are small communities in the large community.

Structure entropy. The structural information of the homogeneous graph $\mathcal{G}$ determined by the encoding tree $\mathcal{T}$ is defined as:

$$H^{\mathcal{T}}(\mathcal{G}) = -\sum_{\alpha \in \mathcal{T}, \alpha \neq \lambda} \frac{g_\alpha}{vol(\mathcal{G})} log \frac{vol(\alpha)}{vol(\alpha^-)}, \tag{1}$$

where $vol(\mathcal{G})$ is the sum of the degrees of all vertices in the graph $\mathcal{G}$. $vol(\alpha)$ is the volume of T_α and is the sum of the degrees of all vertices in the vertex subset T_α. g_α is the sum of weights of all edges from vertex subset T_α to vertex subset $\mathcal{V}/T_\alpha$, which can be understood as the total weight of the edges from the vertices outside the vertex subset T_α to the vertices inside the vertex T_α, or the total weight of the cut edges. $\frac{g_\alpha}{vol(\mathcal{G})}$ represents the probability that the random walk enters T_α. The structural entropy $H(\mathcal{G})$ of graph $\mathcal{G}$ is the minimum $H^{\mathcal{T}}(\mathcal{G})$. Let $\mathcal{T}_k$ be encoding trees whose height is not greater than k, then the k-dimensional structural entropy of $\mathcal{G}$ is defined as follows:

$$H_k(\mathcal{G}) = min H^{\mathcal{T}_k}(\mathcal{G}). \tag{2}$$

Furthermore, one-dimensional structural entropy is special as there are only root nodes and leaf nodes in the encoding tree of one layer. All the vertices in the graph $\mathcal{G}$ belong to a large community λ under the one-dimensional encoding tree, which is unique in terms of community division so that the one-dimensional structural entropy can be directly expressed as:

$$H_1(\mathcal{G}) = -\sum_{i=1}^{n} \frac{d_i}{vol(\mathcal{G})} log \frac{d_i}{vol(\mathcal{G})}, \tag{3}$$

where d_i is the sum of weights of all edges connected to vertex v_i in graph $\mathcal{G}$ and is called the degree of vertex v_i. One-dimensional structural entropy measures the uncertainty of graph $\mathcal{G}$ without layering.

3 METHODOLOGY

In this section, we will introduce the social bot detection framework UnDBot based on the structural information theory and the multi-relational graph. As shown in Figure 1, UnDBot consists of three key modules: Multi-relational Graph Construction, User Community Division, and Community Binary Classification. **(1) Multi-relational Graph Construction.** To begin with, social users are constructed as a multi-relational graph based on the similarity of bot behavioral characteristics. (Section 3.1) **(2) User Community Division.** Following the graph construction process, an optimal two-dimensional encoding tree is created based on the principle of structural entropy minimization. Additionally, social users are allocated to different subtrees on the encoding tree, resulting in community division. (Section 3.2) **(3) Community Binary Classification.** For each community, the stationary distribution and community entropy are utilized to quantify community influence and cohesion, which are then used for binary classification to bot or human accounts. (Section 3.3)

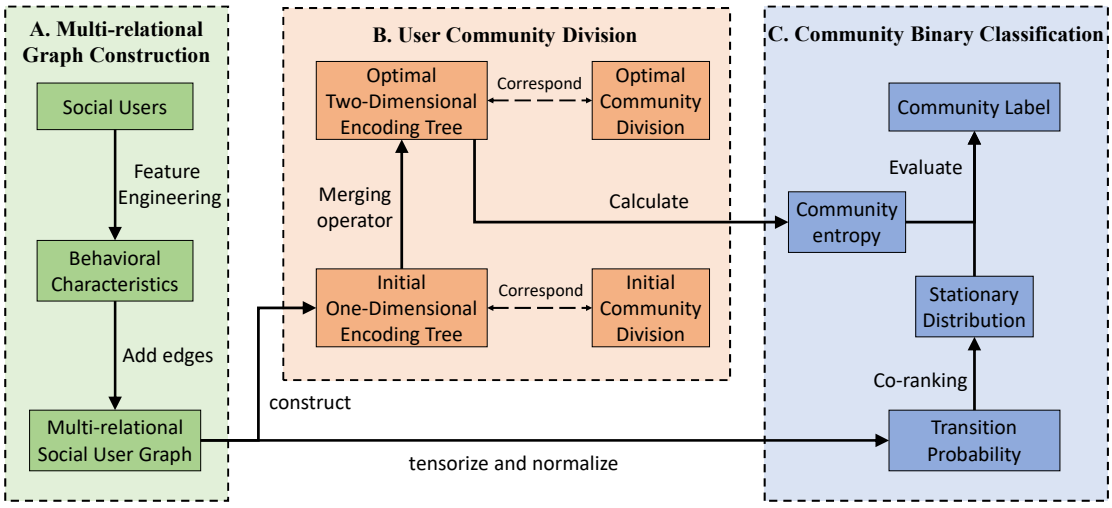


Fig. 1. The overall framework of UnDBot.

3.1 Multi-relational social user graph

In this subsection, we first define the presented multi-relational social user graph and compare it to the relationships used in other models. Subsequently, a detailed description of constructing a multi-relational graph in UnDBot will be provided.

Definition 3.1. Multi-relational graph in the social bot detection task.

The multi-relational graph in social bot detection, denoted as $\mathcal{G}^r = \{\mathcal{V}, \mathcal{E}_k|_{k=1}^R\}$, is defined in terms of $\mathcal{V}$ and $\mathcal{E}_k$. $\mathcal{V}$ represents the set of users $\{v_1, \ldots, v_N\}$, and N denotes the total number of users in the social network. $\mathcal{E}_k$ represents a set of edges $e_{i,j}^k = (v_i, v_j, w_{ij}^k) \in \mathcal{E}_k$ with weight between users under k-th relationship. R is the total number of different relationships, and weight w_{ij}^k is the similarity under k-th relationship.

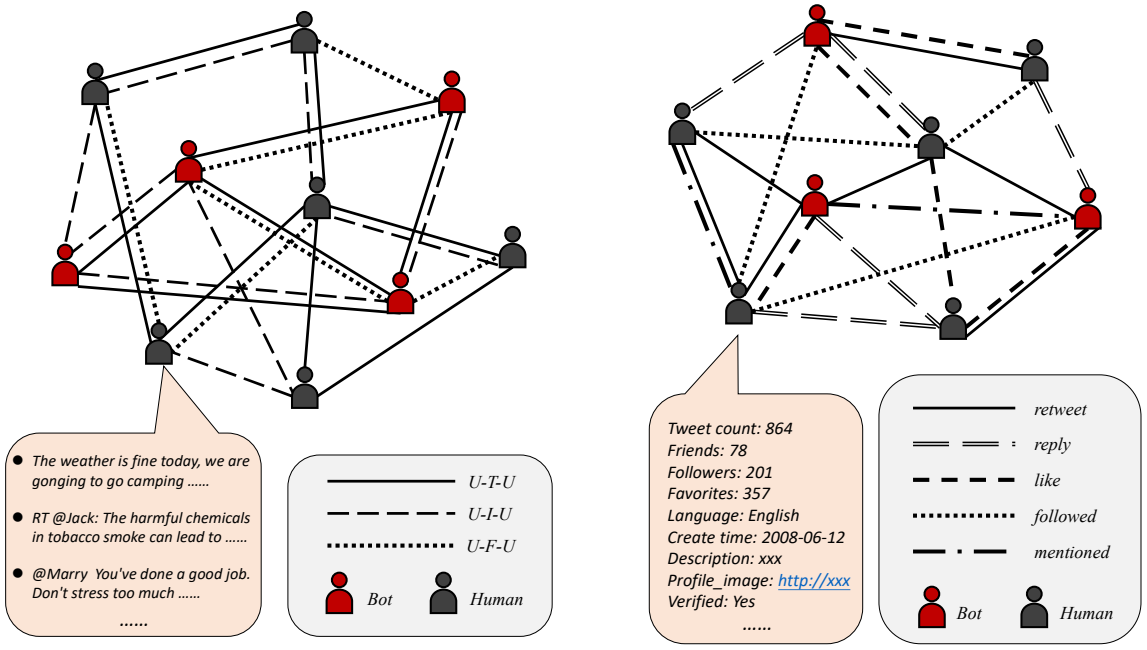
When constructing a multi-relational graph for a social bot detection task, relationships are always direct interactions between users on social networks, such as following, replying, retweeting, mentioning, and liking, as shown in Figure 2(b). This modeling method may be suitable for neighbor aggregation in architectures based on graph neural networks, but it is not beneficial for structure-based detection models. Due to the intelligence of social bots, their interactions with humans are becoming more and more frequent. It is difficult to directly identify a social bot in a complex network through interactive relationships or basic user information such as their ID, profile, username length, and the number of tweets they have made. Because the social bot will disguise itself by interacting with normal human users, there are also rich connections between social bots and human users in terms of following, replying, etc. Nevertheless, the existence of social bots is always for certain purposes, like spreading malicious news, guiding public opinion, or acting as fake fans. These purposes are always achieved through social behaviors such as tweeting, following, commenting, retweeting, and liking. Traditional social behaviors-based multi-relationship graph modeling in practice makes it difficult to achieve effective bot detection performance.

As shown in Figure 2(a), we abandon the traditional multi-relational graph modeling method and focus on the commonality of social behaviors from the perspective of social bots to construct a multi-relational graph, thereby transforming the social bot detection task into a vertex binary classification problem. The connections we selected are closely related to social bots, as shown in Table 2. We divide social behaviors into posting type (tweeting, retweeting, and commenting), posting influence (retweeting, liking, commenting), and follow-to-follower ratio (following). The above social behaviors include active and passive behaviors. For example, retweeting and commenting in the posting type are the actions of the user, while retweeting and commenting in the posting influence are received by the user, that is, the actions of other users towards the user. The connections formed include having the same *posting type distribution* (U-T-U), having the same *posting influence* (U-I-U), and having the same *follow-to-follower ratio* (U-F-U). Meanwhile, the closeness of different types of connections can be formalized as the weight of the edges. Intuitively, the more prominent the social bot behavioral characteristics, the richer the information around the suspected nodes in the graph.

Table 2. Relationships in UnDBot.

Relation	Illustration	Social Behaviors
User-*posting Type distribution*-User (*U-T-U*)	It connects two users who have the same distribution of posting types.	tweeting, retweeting, and commenting
User-*posting Influence*-User (*U-I-U*)	It connects two users who have the same posting influence on social networks.	retweeting, liking, and commenting
User-*Follow-to-follower ratio*-User (*U-F-U*)	It connects users who have the same ratio of followings to followers.	following

Definition 3.2. Multi-relational Graph Construction.



(a) Multi-relational graph in UnDBot

(b) Multi-relational graph in other models

Fig. 2. Multi-relational graph in social bot detection.

In terms of posting behavior, the posting type can be divided into original tweets, retweets, and comments. Social spambots typically disseminate a substantial volume of spam tweets and promotional content to lure internet users. Conversely, certain types of social bots engage in excessive retweeting and commenting on tweets about specific subjects to influence the trend of public opinion topics. Therefore, we argue that ***Posting Type Distribution*** is a characteristic closely related to social bots. To this end, we propose a new kind of edge to aggregate the proportion information for each type of tweet. We utilize a vector $Pt_i = [pt_i^1, pt_i^2, pt_i^3]$ to represent the distribution of posting types for each user, where pt_i^1 signifies the proportion of original tweets, pt_i^2 indicates the proportion of retweets and pt_i^3 denotes the proportion of comments. Since the *posting type distribution* is not a discrete value, we also use the Manhattan distance [71] to calculate the similarity between two users' posting type distribution vectors, thereby determining whether the posting type distributions are similar. The distance of Pt_i and Pt_j and the weight are as follows:

$$d_{i,j} = \mid pt_i^1 - pt_j^1 \mid + \mid pt_i^2 - pt_j^2 \mid + \mid pt_i^3 - pt_j^3 \mid, \quad w_{ij}^1 = 1 - d_{i,j}, \tag{4}$$

where $d_{i,j} \leq 1$ is the Manhattan distance of the *posting type distribution* and w_{ij}^1 is the weight of edge between user v_i and user v_j defined by the similarity of *posting type distribution*. The larger the value of $d_{i,j}$, the smaller the weight of the edge w_{ij}^1. When the Manhattan distance $d_{i,j}$ is less than the threshold ξ, we consider that the two users have similar *posting type distribution*, indicating they have similar posting preferences. In this case, an edge $e_{i,j}^1 = (v_i, v_j, w_{ij}^1)$ is added on the multi-relational graph $\mathcal{G}^r$.

In terms of social influence, the influence on social networks is manifested in the number of comments, likes, and retweets received by users' tweets. Generally speaking, the most influential users are some official media or social media influencers, but social bots lurk in social networks, accumulating certain influence in long-term

interactions with normal users. Therefore, the ***Posting Influence*** is also an important focus of social bot detection. In our work, we define the influence of a user's post Inf_i as the sum of the average number of comments, likes, and retweets of original tweets. Unlike the *posting type distribution* Pt_i, there is no upper limit to the Inf_i defined here. If normalization is performed before utilizing the Manhattan distance to determine the similarity of influence between two users, it may disadvantage users with significantly fewer likes, comments, and retweets than the maximum value. This is because normalization narrows the influence gap between users. Therefore, we use the deviation ratio of influence between users to calculate the weight as follows:

$$\Delta_{i,j} = \frac{\mid Inf_i - Inf_j \mid}{max(Inf_i, Inf_j)}, \quad w_{ij}^2 = 1 - \Delta_{i,j}, \tag{5}$$

where $\Delta_{i,j}$ is the deviation ratio of the influence of the two users, and w_{ij}^2 is the weight of the edge between user v_i and user v_j defined by the similarity of *posting influence*. When the deviation ratio is less than the threshold ξ, we consider that the *posting influences* of these two users are similar. In this case, an edge $e_{i,j}^2 = (v_i, v_j, w_{ij}^2)$ is added on the multi-relational graph $\mathcal{G}^r$.

Apart from tweeting, the act of following other users also has an impact on social networks. However, some unethical individuals or commercial organizations have recognized the commercial value of having many fans. "Fake fans" refers to followers that are artificially created through the use of robots, virtual identities, and other methods. These followers typically do not interact with other accounts. Therefore, the ***Follow-to-follower Ratio*** is also an important indicator for analyzing social user behavior. To fit users who do not have any followers, we define the *follow-to-follower ratio* as follows:

$$ff_i = \frac{num\text{-}following_i + 1}{num\text{-}follower_i + 1}, \tag{6}$$

where ff_i is the *follow-to-follower ratio*, $num\text{-}following_i$ and $num\text{-}follower_i$ refer to the number of followings and fans of user v_i, respectively. Similar to the *posting influence*, there is also no upper limit to ff_i, so we use the deviation ratio to calculate the weight as follows:

$$w_{ij}^3 = 1 - \frac{\mid ff_i - ff_j \mid}{max(ff_i, ff_j)}. \tag{7}$$

Here w_{ij}^3 is the weight of the edge between user v_i and user v_j defined by the similarity of *follow-to-follower ratio*. When the deviation ratio is less than ξ, we consider that the *follow-to-follower ratio* of these two users are similar. To prioritize the following behavior of users, we implement a restriction φ on the rules for building edges: only if both the ff_i and ff_j of user pair (i, j) are greater than φ, an edge $e_{i,j}^3 = (v_i, v_j, w_{ij}^3)$ is added to the multi-relational graph $\mathcal{G}^r$.

Overall, we design the above three social relationship metrics that capture various aspects of social bot behaviors: *Posting Type Distribution*, *Posting Influence*, and *Follow-to-follower Ratio*, to model the multi-relational social user graph.

3.2 User Community Division

The encoding tree clustering method demonstrates superior adaptability in selecting the number and size of communities on graphs [9, 44, 45, 94], compared to other clustering methods. However, it is important to note that this approach is restricted to homogeneous and simple graphs. To tackle the challenge of constructing encoding trees and partitioning communities on multi-relational graphs, we first expand the concept of structural entropy to encompass multi-relational graphs defined in Section 3.1. Then, we construct the optimal encoding tree for multi-relational graphs to partition the social user community effectively.

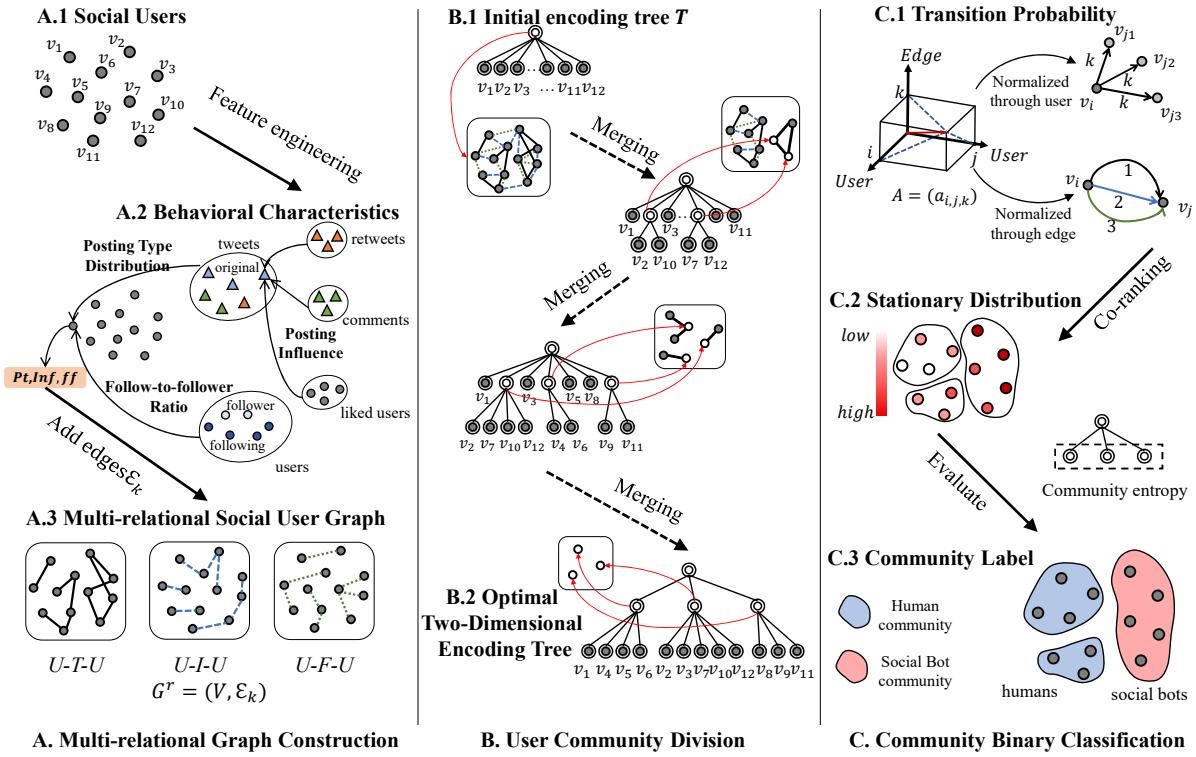


Fig. 3. The Structural Information Theory-Based Social Bot Detection.

Multi-Relational Graph Structural Entropy. The definition of traditional structural entropy is given in Section 2.2 After constructing a multi-relational social user graph based on the similarity of social behaviors, our objective is to classify users with similar social behaviors into the same community using structural information theory. However, the initial structural entropy analysis is conducted on the homogeneous graph. Therefore, we design an optimized structural entropy of the multi-relational graph $H^{\mathcal{T}}(\mathcal{G}^r)$ under the encoding tree $\mathcal{T}$. This is achieved by representing all variables of edges as the summation of the weights across the three kinds of relationships, which include the degree of vertices (d_i), community volume ($vol(\alpha)$), and cut edges (g_α):

$$\begin{cases} d_i = \omega_1 d_i^1 + \omega_2 d_i^2 + \omega_3 d_i^3. \\ vol(\alpha) = \omega_1 vol^1(\alpha) + \omega_2 vol^2(\alpha) + \omega_3 vol^3(\alpha). \\ g_\alpha = \omega_1 g_\alpha^1 + \omega_2 g_\alpha^2 + \omega_3 g_\alpha^3. \end{cases} \tag{8}$$

The resulting structural entropy integrates the three newly defined social relationships, providing a more comprehensive depiction of the structure of the multi-relational graph.

Optimized Encoding Tree. The definition of encoding tree is also given in Section 2.2 An optimal hierarchical encoding tree is constructed during the computation of the structural entropy. The leaf nodes of this encoding tree represent the vertices in the graph. If two leaf nodes share the same parent node, it signifies that the corresponding graph vertices belong to the same community. The initial one-dimensional encoding tree represents the simplest two-level structure, where the leaf nodes in the graph are directly connected to the root node, as shown in Figure 3 B.1. The *Merge operator* combines the nodes, and a greedy search strategy is employed to construct an

optimal encoding tree. The *Merge operator* combines two subtrees under the same parent node, resulting in a single subtree. This is visualized in the graph as the merging of two small communities. In the optimal encoding tree, the graph structure exhibits minimal uncertainty, and the nodes achieve a balanced and stable state, leading to the optimal partitioning of user vertices. The steps of the *Merge operator* are outlined as follows:

(1) Let label $T_\alpha = \{x_1, x_2 ... x_M, y_1, y_2, ..., y_N\}$, and merge the label of node β into node α;
(2) For every subtree $\alpha_s, s \in \{1, 2..., M\}$, define $T_{\alpha_s} = \{x_s\}$, assign the level $h(\alpha_s) \leftarrow h(\alpha) + 1$;
(3) For every subtree $\alpha_t, M + 1 \leq t \leq M + N$, define $T_{\alpha_t} = \{y_{t-M}\}$, assign the level $h(\alpha_t) \leftarrow h(\alpha) + 1$;
(4) Delete subtree node β and its all subtrees.

Step (1) updates the label of node α to the merged one, and steps (2) and (3) define the child nodes of the merged node α. Recording $\mathcal{T}_{mg}(\alpha, \beta)$ as the encoding tree after $\mathcal{T}$ runs $Mg(\mathcal{T}; \alpha, \beta)$, the difference in structural entropy of the graph $\mathcal{G}^r$ determined by the two encoding trees $\mathcal{T}_{mg}(\alpha, \beta)$ and $\mathcal{T}$ is:

$$\begin{aligned}\Delta^{Mg}_{\mathcal{G}^r}(\mathcal{T}; \alpha, \beta) = &\left(H^{\mathcal{T}_{mg}(\alpha,\beta)}(\mathcal{G}^r; \alpha) + \sum_{\delta^- = \alpha} H^{\mathcal{T}_{mg}(\alpha,\beta)}(\mathcal{G}^r; \delta) \right) \\ &- \left(H^{\mathcal{T}}(\mathcal{G}^r; \alpha) + H^{\mathcal{T}}(\mathcal{G}^r; \beta) + \sum_{\delta^- = \alpha or \delta^- = \beta} H^{\mathcal{T}}(\mathcal{G}^r; \delta) \right),\end{aligned} \tag{9}$$

where $H^{\mathcal{T}}(\mathcal{G}^r; \alpha)$ is the structural information of subtree α, $H^{\mathcal{T}}(\mathcal{G}^r; \beta)$ is the structural information of subtree β, $H^{\mathcal{T}}(\mathcal{G}^r; \delta)$ is the structural information of α's and β's subtree δ; $H^{\mathcal{T}_{mg}(\alpha,\beta)}(\mathcal{G}^r; \alpha)$ is the structural information of subtree α after merging subtree β into α; Similarly, $H^{\mathcal{T}_{mg}(\alpha,\beta)}(\mathcal{G}^r; \delta)$ is the structure information of the subtree δ after merging subtree β into α. Eq. 9 calculates the change in the related subtree structure entropy before and after running the Merge operator on nodes α and β. If $\Delta^{Mg}_{\mathcal{G}^r}(T; \alpha, \beta) \leq 0$, then the Merging operator runs successfully, denoted as $Mg(\mathcal{T}; \alpha, \beta) \downarrow$. According to Eq. 9 $\Delta^{Mg}_{\mathcal{G}^r}(\mathcal{T}; \alpha, \beta)$ is locally computable. The Merging operator only merges two subtrees into one subtree, and other nodes in the encoding tree do not change. For a two-dimensional encoding tree, the structural entropy difference after the Merging operator is expanded as:

$$\begin{aligned}\Delta^{Mg}_{\mathcal{G}^r}(\mathcal{T}; \alpha, \beta) = &-\frac{g_{\alpha'}}{vol(\mathcal{G}^r)} log_2 \frac{vol(\alpha')}{vol(\mathcal{G}^r)} + \sum_{v_i \in \alpha'} -\frac{d_i}{vol(\mathcal{G}^r)} log_2 \frac{d_i}{vol(\alpha')} \\ &- \left(-\frac{g_\alpha}{vol(\mathcal{G}^r)} log_2 \frac{vol(\alpha)}{vol(\mathcal{G}^r)} + \sum_{v_i \in \alpha} -\frac{d_i}{vol(\mathcal{G}^r)} log_2 \frac{d_i}{vol(\alpha)} \right) \\ &- \left(-\frac{g_\beta}{vol(\mathcal{G}^r)} log_2 \frac{vol(\beta)}{vol(\mathcal{G}^r)} + \sum_{v_i \in \beta} -\frac{d_i}{vol(\mathcal{G}^r)} log_2 \frac{d_i}{vol(\beta)} \right),\end{aligned} \tag{10}$$

where g_α is the sum of the cut edge weights of vertex subset T_α, d_i represents the degree of vertex v_i, which is the sum of the weights of its edges. $vol(\mathcal{G}^r)$ represents the volume of the root node and is the sum of the degrees of all vertices; $vol(\alpha)$ represents the volume of node α, which is the sum of degrees of vertex subset T_α.

The initial one-dimensional encoding tree implies that each user vertex v_i forms an independent community as illustrated in Figure 3 B.1. Next, we calculate the difference in structural entropy before and after executing the Merge operator on any two nodes, and select no more than the maximum number of pairs, denoted as $max\_operate$, from the pairs of nodes whose structural entropy is reduced the most to run the Merge operator. Continue this process until no pair of nodes is found that allows the Merge operator to execute successfully. The value of $max\_operate$ for each iteration is determined by the Equation $ceil((current\_num - 1) * p)$, where $current\_num$ represents the current number of communities/nodes, and p is a hyperparameter between 0 and

1 that controls the speed of parallel operations for the Merge operator. The operator ceil is a mathematical function that rounds a given number up to the nearest integer. The final optimal two-dimensional encoding tree partitions users into distinct communities, as illustrated in Figure 3 B.2. Like information entropy, which measures information uncertainty, structural entropy quantifies the uncertainty resulting from graph partitioning. A lower value of structural entropy indicates reduced uncertainty and greater stability in the graph structure, thereby achieving the most favorable division of user nodes.

3.3 Community Binary Classification

Up to this point, we have successfully partitioned social users on the multi-relational graph into distinct communities. This subsection introduces a method for quantifying community influence in a stationary distribution. We propose a community labeling strategy by combining community influence and cohesion.

Stationary Distribution. MultiRank [59] is a framework designed for the co-ranking of vertices and edges in multi-relational graphs. It assesses the significance of users and relationships when computing the probability distribution for multi-relational data. Users on the multi-relational graph are connected through different relationships, so we represent the multi-relational graph $\mathcal{G}^r$ as a three-dimensional tensor $\mathcal{A} = (a_{i,j,k})$, where $a_{i,j,k}$ is the weight between nodes v_i and v_j under k-th relationship. When user vertices v_i and v_j have no edge under k-th relationship, the (i, j, k) term is zero. And this step is called tensorization and corresponds to 'tensorize' in Figure 1:

$$\mathcal{A} = (a_{i,j,k}) = \begin{cases} 1, & \exists e^k_{i,j}. \\ 0, & \nexists e^k_{i,j}. \end{cases} \tag{11}$$

To obtain the influence of social users on the entire multi-graph through co-ranking, we normalize the tensor $\mathcal{A}$ based on vertices and relationships separately, as shown in Figure 3 C.1 of the UnDBot architecture which corresponds to 'normalize' in Figure 1. This process results in two tensors, O and S:

$$\begin{cases} O = (o_{i,j,k}) = \begin{cases} \frac{a_{i,j,k}}{\sum_{j=1}^{N} a_{i,j,k}}, & \exists e^k_{i,j}, \\ \frac{1}{N}, \forall j \in \{1, 2, ..., N\}, & \nexists e^k_{i,j}, \end{cases} \\ S = (s_{i,j,k}) = \begin{cases} \frac{a_{i,j,k}}{\sum_{k=1}^{3} a_{i,j,k}}, & \exists e^k_{i,j}. \\ \frac{1}{3}, \forall k \in \{1, 2, 3\}, & \nexists e^k_{i,j}. \end{cases} \end{cases} \tag{12}$$

Here $o_{i,j,k}$ represents the possibility of reaching vertex v_j from vertex v_i and k-th edge. And $s_{i,j,k}$ represents the possibility of going through k-th edges from vertex v_i to vertex v_j. In particular, if the vertex v_i does not have any edge under the k-th relationship ($\sum_{j=1}^{N} a_{i,j,k} = 0$), or if the vertex pair (v_i, v_j) has no edge under any relationship ($\sum_{k=1}^{3} a_{i,j,k} = 0$), the average value is used instead. The tensors O and S represent the transition probability of a random walk on the multi-relational graph $\mathcal{G}^r$. We use $t-1$ to represent the state of the random walk at the previous moment and use t to represent the state after a random walk. Then the possibility $Prob_O[j;t]$ of reaching user v_j and the possibility $Prob_S[k;t]$ of passing the k-th edge at the current moment are formalized by the following Equation:

$$\begin{cases} Prob_O[j;t] = \sum_{i=1}^{N} \sum_{k=1}^{3} o_{i,j,k} \times Prob_O[i;t-1] \times Prob_S[k;t], \\ Prob_S[k;t] = \sum_{i=1}^{N} \sum_{j=1}^{N} s_{i,j,k} \times Prob_O[i;t-1] \times Prob_O[j;t]. \end{cases} \tag{13}$$

The possibility of reaching user v_j at time t ($Prob_O[j;t]$) is expressed as the accumulation of the possibility of reaching user v_i at the previous moment $t-1$ ($Prob_O[i;t-1]$) and passing edge k from user v_i at the current moment t ($Prob_S[k;t]$). The possibility of passing the k-th edge at time t ($Prob_S[k;t]$) is expressed as the accumulation of the possibility of reaching user v_i at the previous moment $t-1$ ($Prob_O[i;t-1]$) and reaching user v_j at the current moment t ($Prob_O[j;t]$). After the continuous random walk, the possibility of arriving at user v_j and passing k-th edge reaches a steady state, i.e., $Prob_O[j;t] \approx Prob_O[j;t-1]$. Denoting $\overline{x_j} = \lim_{t\to\infty} Prob_O[j;t]$ and $\overline{y_k} = \lim_{t\to\infty} Prob_S[k;t]$, Eq. 13 is formalized as follows when the time zone is infinite:

$$\begin{cases} \overline{x_j} = \sum_{i=1}^{N}\sum_{k=1}^{3} o_{i,j,k} \times \overline{x_i} \times \overline{y_k}, \\ \overline{y_k} = \sum_{i=1}^{N}\sum_{j=1}^{N} s_{i,j,k} \times \overline{x_i} \times \overline{x_j}. \end{cases} \tag{14}$$

We represent the steady state as a stationary distribution in the tensor form $\mathcal{X} = [\overline{x_1}, \overline{x_2}, \ldots, \overline{x_N}]$ and $\mathcal{Y} = [\overline{y_1}, \overline{y_2}, \overline{y_3}]$. The random walk process is simulated to iterate the initial distribution tensor until the two tensors tend to be stable. The stationary distribution $\mathcal{X}$ can thus be obtained from the system of equations:

$$\begin{cases} \mathcal{X} = \mathcal{X}\mathcal{O}\mathcal{Y}^\top, \\ \mathcal{Y} = \mathcal{X}\mathcal{S}\mathcal{X}^\top. \end{cases} \tag{15}$$

Constantly update tensor $\mathcal{X}$ and tensor $\mathcal{Y}$ through the equation, and a stabilize threshold ρ is set to stop updating when $|\mathcal{X}_{new} - \mathcal{X}_{old}| + |\mathcal{Y}_{new} - \mathcal{Y}_{old}| < \rho$.

Community Label. The stationary distribution $\mathcal{X}$ represents the distribution of influence that each user vertex has on the entire network, as depicted in Figure 3 C.2. The average influence of users quantifies community influences x^{α}_{comm}:

$$x^{\alpha}_{comm} = avg(\overline{x_i}, v_i \in T_\alpha), \tag{16}$$

where x^{α}_{comm} is the community influence, T_α is the community represented by node α, and avg represents a mathematical function that calculates the average value. In this work, since the multi-relational graph constructed in section 3.2 is based on the similarity of behavior between users, the distribution of vertex influence $\mathcal{X}$ is interpreted as the similarity between each user and other users. The degree of community influence is interpreted as the degree of similarity of user behavior within the community.

In addition to community influence, internal cohesion within a community is an important indicator for analyzing the community. The entropy $H(\mathcal{G}^r;\alpha)$ of the community node on the encoding tree quantifies community cohesion:

$$H(\mathcal{G}^r;\alpha) = -\frac{g_\alpha}{vol(\mathcal{G}^r)} log_2 \frac{vol(\alpha)}{vol(\mathcal{G}^r)}. \tag{17}$$

The higher the similarity among users within a community, the more interconnected edges exist within the community, resulting in a higher entropy of community nodes and stronger cohesion of the community, resembling a social bot community. We combine community influence and community cohesion to define the community label function as follows:

$$Ev(\alpha) = (1-\pi)\frac{x^{\alpha}_{comm}}{x^{\lambda}_{comm}} + \pi\frac{H(\mathcal{G}^r;\alpha)}{\sum_{\beta^-=\lambda} H(\mathcal{G}^r;\beta)}. \tag{18}$$

If the evaluation index $Ev(\alpha)$ of the community α is greater than the threshold θ, the community α is judged as a social bot community. Otherwise, it is a human community, as shown in Figure 3 C.3.

Algorithm 1: The overall process of UnDBot.

Input: Tweet information of users: Pt; Influence Information of tweets: Inf; following Information of users: ff; threshold of the similarity for three features in modeling: ξ; the stop threshold in random walk: ρ; the community evaluation threshold: θ; the parameters for parallel Merge Operators: p; weighted parameters of influence and cohesion: π.

Construct the multi-relational graph $\mathcal{G}^r$ via Eq. 4, Eq. 5 and Eq. 7;
$comms \leftarrow$ the initial division, $current\_num \leftarrow$ number of users;; `// Each user forms a community`
Initialize $edge$ with user pairs that connected in $\mathcal{G}^r$, $merge\_all \leftarrow False$;
while *not merge_all* **do**
    $max\_operate \leftarrow (current\_num - 1) * p$;`// The maximum number of operator runs`
    for (u, v) *in edge* **do**
        $dH(u, v) \leftarrow \Delta_{\mathcal{G}}^{Mg}(\mathcal{T}; comm_u, comm_v)$ via Eq. 10;
    end
    $op\_edge \leftarrow (u, v)$ with $dH(u, v) > median(dH > 0)$;`// Select edges that can be merged`
    if *len(op_edge)>max_operate* **then**
        $op\_edge \leftarrow (u, v)$ with top $max\_operate$ dH;
    end
    if *len(op_edge)=0* **then**
        $merge\_all = True$;
    end
    $current_num \leftarrow current_num-$len($op\_edge$);`// Update the number of communities`
    update $edge$ and $comms$;`// Running Merge Operator on` op_edge
end
Initialize $\mathcal{A} \leftarrow$Eq. 11, $\mathcal{O}, \mathcal{S} \leftarrow$Eq. 12; `// Normalized`
Initialize the distribution tensor $\mathcal{X}_{old}$ and $\mathcal{Y}_{old}$;
while *True* **do**
    $\mathcal{X}_{new} \leftarrow \mathcal{X}_{old}\mathcal{O}\mathcal{Y}_{old}^{\top}$, $\mathcal{Y}_{new} \leftarrow \mathcal{X}_{old}\mathcal{S}\mathcal{X}_{old}^{\top}$ via Eq. 15;
    if $|| \mathcal{X}_{new} - \mathcal{X}_{old} || + || \mathcal{Y}_{new} - \mathcal{Y}_{old} || < \rho$ **then**
        Break; `// The distribution tensor tends to be stable`
    else
        $\mathcal{X}_{old} \leftarrow \mathcal{X}_{new}, \mathcal{Y}_{old} \leftarrow \mathcal{Y}_{new}$; `// Update`
    end
end
$\mathcal{X} \leftarrow \mathcal{X}_{old}$;
for *comm in comms* **do**
    Calculate x_α and H_α via Eq. 16 and Eq. 17; `// Calculate community influence and cohesion`
    $E(comm) \leftarrow (1 - \pi)x_\alpha + \pi H_\alpha$ via Eq. 18; `// Calculate the evaluation score`
    if $E(comm) > \theta$ **then**
        Users in *comm* are social bots;
    else
        Users in *comm* are humans;
    end
end

3.4 Put them together

Algorithm 1 outlines the overall detection process of the proposed UnDBot. The process includes the construction of a multi-relational graph, the user community division using structural entropy, and a community labeling process that utilizes stationary distribution and community node entropy. We identify three new types of social relationships based on the definition in Section 3.1. Using these relationships, we construct a social user multi-relational graph based on the similarity of social behavior (Line 1). We use the supplemented multi-relational graph-based structural information theory to construct a two-dimensional encoding tree with structural entropy minimization. To speed up, we run the *Merge operator* in parallel that select no more than $max\_operate$ edges (Lines 5-12) to merge in each round until the *Merge operator* fails (Lines 13-15). This process partitions user vertices into communities. We then translate the multi-relational graph as a three-dimensional tensor (Line 19), as described in Section 3.3, and calculate the stationary distribution using co-ranking (Lines 20-28). Once we obtain the user influence distribution vector $\mathcal{X}$ (Line 29), we calculate the influence x_α and cohesion H_α of each community to distinguish the social bot community from the human community (Lines 30-38).

3.5 Time Complexity

The entire UnDBot model is divided into three parts: multi-relational graph construction, user community division, and community binary classification. Among them, modeling the social user multi-relational graph takes $O(N^2)$,where N is the total number of social users. Constructing the structural entropy encoding tree takes $O(|E|log\frac{N-1}{1-p})$, where $|E|$ is the number of connected user pairs in graph MG. Specifically, during each round of execution, the time complexity of calculating the difference in structural entropy for each pair of nodes after merging is $O(|E_i|)$, where E_i represents the current inter-community edges. The maximum number of rounds of execution is $log_{1-p}\frac{1-p}{N-1}$, where p is a hyperparameter controlling the parallel operation of Merge operators and can be regarded as a constant. So the maximum time complexity of user community division is $O(|E| * log_{1-p}\frac{1-p}{N-1}) = O(|E|(1 + logN))$. The MultiRank module (lines 21-30 in Algorithm 1) takes $O(k(N + 3))$. k is the number of iterations related to the graph structure in the MultiRank calculation process. Generally, k is a small number, so the time complexity of community binary classification is simplified to $O(N)$. Consequently, the total time complexity of UnDBot is $O(N^2 + N) + O(|E|(1 + logN)) = O(N^2 + |E|(1 + logN))$.

4 EXPERIMENTAL SETUP

4.1 Software and Hardware

We implement all models with Python 3.10. All experiments are executed on a Linux server with a 128-core Intel Xeon Platinum 8336C CPU, 503GB of RAM, and an NVIDIA A800-SXM4-80GB. As for baselines, we utilize open-source implementation of node2vec from the library PecanPy [2], as well as the codes provided by the authors for other baselines.

4.2 Datasets

We employ four publicly available social user datasets to assess the performance of the models in the context of social bot detection. The original *Cresci-2015* and *Cresci-2017* datasets consist of user tweet data and user attribute data. As UnDBot requires user tweet information in the dataset to analyze posting type and posting influence distribution, the experiments conducted in this study on the *Cresci-2017* dataset exclude users who have no tweet data. Furthermore, we collect tweets from users in the *Botwiki-2019* and *Pronbots-2019* datasets and add human users to build datasets required for the experiment to make it more consistent with real scenarios. Table 3 presents various statistical information about the datasets we utilized and the number of connected edges

[2]https://github.com/krishnanlab/PecanPy

Table 3. Statistics of datasets.

Datasets	Human	Bot	User	Tweet	Edges	Feature Similarity
Cresci-2015	1,950	3,351	5,301	2,827,757	*U-F-U*: 726,323	0.3735
					U-T-U: 2,970,033	0.2791
					U-I-U: 412,741	0.2960
Cresci-2017	3,474	9,263	12,737	6,637,616	*U-F-U*: 2,279,920	0.3215
					U-T-U: 16,652,890	0.3323
					U-I-U: 1,026,120	0.3133
Pronbots-2019	1,481	17,882	19,363	231,224	*U-F-U*: 66,979	0.2775
					U-T-U: 131,853,041	0.7091
					U-I-U: 15,015	0.7604
Botwiki-2019	65	698	763	357,851	*U-F-U*: 68	0.2119
					U-T-U: 176,210	0.6145
					U-I-U: 5,561	0.1778

for different relationships on the datasets under the graph construction method of UnDBot. Additionally, we calculated the average feature similarity based on three characteristics of user nodes, as defined in Section 3.1. The detailed descriptions of these four datasets are as follows:

- ***Cresci-2015*** [17]. The Cresci-2015 dataset contains both benign Twitter users and fake accounts on Twitter. Among them, datasets *E13* and *TFP* contain 1481 and 469 active accounts from different social classes and backgrounds, manually verified as benign accounts by social scientists. Datasets *FSF*, *INT*, and *TWT* are fake accounts purchased from different online marketplaces, containing 1169, 1337, and 845 users, respectively.
- ***Cresci-2017*** [19]. The Cresci-2017 dataset contains benign users and three types of fake users on Twitter. The dataset *genuine* contains 3474 benign accounts that have been manually verified. The dataset *social spambot1* contains 991 automated accounts related to the mayoral election of Rome. These social bots are similar to real accounts in profile, tweeting behavior, and friending behavior. The dataset *social spambot2* contains 3457 social spambots related to the *Talnts* mobile application. The dataset *social spambot3* contains 464 social spambots related to advertising products for sale on amazon.com. The dataset *fake followers* contains 3351 fake follower accounts purchased from different online marketplaces. The dataset *traditional spambot1* contains 1000 traditional spambots from [87].
- ***Pronbots-2019*** [89]. The original Pronbots-2019 dataset contains 17882 Twitter bots related to advertising scam sites. The dataset was first shared by Andy Patel (github.com/r0zetta/pronbot2) and collected for training Botometer models. In this experiment, to make the dataset more consistent with real scenarios without destroying the structure of the original *Pronbots-2019* dataset, we add 1481 human users from *E13* of *Cresci-2015*.
- ***Botwiki-2019*** [90]. The Botwiki-2019 dataset contains 698 self-identified bots from botwiki.org, which is an open catalog that includes bots designed for social media platforms, messaging apps, websites, and other online platforms. The dataset retains only active users of the Twitter platform. In this experiment, to make the dataset more consistent with real scenarios without destroying the structure of the original *Botwiki-2019* dataset, we add 65 human users from *TFP* of *Cresci-2015*.

4.3 Baselines and Variations

Baselines. To verify the effectiveness of the proposed UnDBot for social bot detection tasks, we compare it with various unsupervised machine learning-based and network embedding-based baselines. All models based on

unsupervised network embedding are constructed on the multi-relational graph described in Section 3.1, followed by binary classification using K-means. To demonstrate the effectiveness of multi-relational graph modeling, we also conduct experiments on homogeneous graphs for comparison. The homogeneous graph consists of two types: one is constructed based on the action of following other users, which is contained in the original dataset Cresci-2015; the other follows anomaly detection [68] and constructs the graph based on attribute similarity.

- **K-means** [56]. K-means is a classic clustering method. After predefining the number of clusters and the center point, users with similar characteristics are grouped into spherical clusters based on Euclidean distance. The improved streaming algorithm [56] detects abnormal users (social bots) unsupervised by calculating the shortest distance between newly arrived users and core clusters.
- **DNA** [18]. This social user behavior modeling method is based on digital DNA technology, which simulates user behavior and interaction through strings. The model uses the similarity of DNA sequences to discover user groups with highly similar behaviors and unsupervised identify social bot groups.
- **DeepWalk** [65]. DeepWalk is an unsupervised structure-preserving network embedding learning algorithm. It uses the random walk method of depth-first traversal to sample neighbor nodes in the graph and utilizes the co-occurrence relationship between nodes in the graph to learn the vector representation of nodes.
- **LINE** [74]. LINE uses breadth-first sampling of neighbor nodes and defines the first-order and second-order similarity of nodes. The first-order and second-order similarities are optimized separately to learn vector representations of nodes.
- **Node2vec** [31]. Node2vec is an extension of DeepWalk, incorporating a biased walk that comprehensively considers breadth-first and depth-first for domain sampling. The objective is to optimize the likelihood of encountering neighboring nodes within the specified constraints of each vertex.
- **SDNE** [76]. SDNE is an extension of LINE that utilizes an auto-encoder structure to simultaneously optimize first- and second-order similarity. The learned node vector preserves the local and global structure.
- **GraRep** [7]. GraRep is a network embedding learning model that maps the k-order information of nodes to distinct subspaces, thereby computing k models individually, with each model capturing information of different orders. Embeddings learned by each model are concatenated as an embedding that captures all k-level information of a node. It excels in distinguishing the neighbors of different orders of nodes in representation learning and can be extended to capture neighbors of any order.
- **DNGR** [8]. DNGR is a graph representation learning model that uses the random surfing model to extract the graph structure information directly. This allows it to more accurately and directly learn the graph structure information of weighted graphs. Recovering vertex representations from Positive Pointwise Mutual Information (PPMI) matrices can capture potentially complex, non-linear relationships between different vertices.
- **HOPE** [60]. HOPE is a graph representation learning model based on matrix decomposition. It leverages the relationship between graphs, matrices, and discrete Fourier transforms to effectuate the mapping of nodes from high-dimensional spaces to low-dimensional spaces. This process preserves the integrity of high-order similarity relationships among nodes.
- **GAE** [100]. GAE is an unsupervised neural network for graph-structured representation learning. This work uses GCN as its encoder to learn low-dimensional features of nodes and graphs. The decoder reconstructs the original features from the embedded features and is trained to preserve useful structural features in a low-dimensional space for node classification.

Variations. We generate several variants of the full UnDBot model to understand how each module works within the overall detection framework and to assess better how each module individually contributes to detection performance improvements. Because the modeling in the first step directly affects the effect of structural entropy user community division, the key part of UnDBot lies in the construction of multi-relational graphs and the

evaluation indicators in community marking. We selectively enable or disable some of these modules for ablation studies. The details of these changes are described below:

- **UnDBot-FT**. This variant only uses the *posting type distribution* and *posting influence* to construct the multi-relational graph. When calculating the structural entropy, the weight ratio of the two types of edges is $1:1$. Only these two relationships are used to quantify community influence when calculating the stationary distribution. The community evaluation index weighs community influence and community node entropy.
- **UnDBot-FI**. This variant only uses the *posting type distribution* and *following-follower ratio* to construct the multi-relational graph. When calculating the structural entropy, the weight ratio of the two types of edges is $1:1$. Only these two relationships are used to quantify community influence when calculating the stationary distribution. The community evaluation index weighs community influence and community node entropy.
- **UnDBot-TI**. This variant only uses the *posting influence* and *following-follower ratio* to construct the multi-relational graph. When calculating the structural entropy, the weight ratio of the two types of edges is $1:1$. Only these two relationships are used to quantify community influence when calculating the stationary distribution. The community evaluation index weighs community influence and community node entropy.
- **UnDBot-G**. This variant replaces the multi-relational graph MG with a homogeneous graph G. For datasets with primitive graph structures, the following relation in the dataset is used for modeling, while for other datasets, attribute similarity is used for modeling.
- **UnDBot-mg**. This variant replaces the multi-relational graph MG with another multi-relation graph mg. mg is a multi-relational graph constructed from simplified user behavior similarities. It only contains the user's direct active behavior, including the number of tweets (*U-twe-U*), the number of friends (*U-fri-U*), and the number of likes (*U-fav-U*). The specific construction method is the same as MG.

4.4 Model Running

We use the following settings: similarity threshold (ξ=0.01 for *Pronbots-2019* and ξ=0.1 for other datasets) of *posting type distribution*, *posting influence*, and *follow-to-follower ratio* for multi-relational graph construction. For user community division, we adopt the ratio of three edge weights ($1:1:1$) in the multi-relational graph structure entropy and the maximum scale ratio for parallel merge operators ($p = 0.05$ for *Pronbots-2019* and $p = 0.15$ for other datasets). For community binary classification, we set the stabilize threshold (ρ=0.004) for the distribution tensor, the weight of community influence (1-π), the weight of community entropy ($\pi = 0.6$ for *Botwiki-2019* and $\pi = 0.4$ for other datasets), and the threshold of evaluation index for communities ($\theta = 0.60$ for *Pronbots-2019*, $\theta = 0.55$ for *Botwiki-2019*, and $\theta = 1$ for other datasets). For baselines, we uniformly set the epochs in network embedding learning (50), the number of clusters in K-means (2), and the embedding size (128).

4.5 Evaluation Metrics

As the social bot detection task is essentially a binary classification problem, and the number of benign accounts and malicious accounts in the data set is relatively balanced, we use the ACCURACY rate to evaluate the overall performance of the classifier:

$$ACC = \frac{TP + TN}{TP + TN + FP + FN}, \tag{19}$$

where TP is True Positive, TN is True Negative, FP is False Positive, FN is False Negative. In addition, to measure the Pertinency Factor of the social bot detection model, we use the Precision rate as an evaluation index, that is,

the actual proportion of social bots in the samples judged as social bots:

$$Precision = \frac{TP}{TP + FP}. \tag{20}$$

To assess the comprehensiveness of social bot detection, we use the Recall rate evaluation index, that is, the proportion of correctly identified in the social bot samples.

$$Recall = \frac{TP}{TP + FN}. \tag{21}$$

Furthermore, we employ a balanced F-score for a more comprehensive measure of effectiveness.

$$F1 = \frac{2 \times Precision \times Recall}{Precision + Recall}. \tag{22}$$

5 RESULTS AND DISCUSSION

This section conducts several experiments to evaluate the performance of UnDBot. We mainly answer the following questions:

- Q1: How do different models perform under different datasets, i.e., the algorithm's effectiveness (Section 5.1)?
- Q2: How do the multi-relational graph (*MG*), each individual edge within UnDBot, and the hyperparameters contribute to the overall effectiveness (Section 5.2)?
- Q3: How does UnDBot work in terms of interpretability (Section 5.3)?
- Q4: How efficient different baselines and UnDBot can attain (Section 5.4)?
- Q5: How to explore the detection result and gain deeper insights into the distinctions in human perspective (Section 5.5).

5.1 Model Effectiveness

In this section, we conduct experiments on four publicly available social bot detection datasets (Cresci-2015, Cresci-2017, Pronbots-2019, and Botwiki-2019) to evaluate the effectiveness of our social bot detection model UnDBot. Table 4 presents the test accuracy (ACC), precision (P), recall (R), and F1 score of the unsupervised baselines and our model UnDBot. All baselines that rely on unsupervised graph representation learning are evaluated on the social user multi-relational graph (models denoted as $baseline_{MG}$) and the homogeneous graph (models denoted as $baseline_G$). Our model demonstrates a generally superior level of effectiveness compared to the baseline methods. UnDBot outperforms all baselines on the evaluation indices ACC and F1 for the Cresci-2015, Cresci-2017, Pronbots-2019, and botwiki-2019 datasets. Although it does not achieve the highest precision and recall on three datasets, it still has an advantage over most baselines and achieves a balance between precision and recall. Overall, UnDBot significantly outperforms most unsupervised social bot detection methods regarding detection accuracy, precision, recall, and F1. This demonstrates the effectiveness of UnDBot in the social bot detection scenario. Compared with existing unsupervised social bot detection methods (K-means and DNA), the detection accuracy on the four datasets is increased by at least 12.61%, 19.66%, 12.40% and 10.36% respectively. Compared to methods based on unsupervised graph representation learning, the detection accuracy is improved by at least 1.37%, 3.13%, 0.14%, and 2.23% on the four datasets respectively.

For the baselines, the first two methods in Table 4 are existing unsupervised models for the social bot detection task, while the third method is an unsupervised model based on autoencoders. *DNA* consistently performs the worst, even lower than 50%, except in the botwiki-2019 dataset. This is because *DNA* encodes social user behavior into sequences of behavior strings using genetic techniques and identifies social bots based on the presence of common behavior substrings. This method only encodes behaviors related to tweeting and does not consider other social behaviors of users. Additionally, quantifying behavior similarity based on the longest common behavior substring is prone to false positives due to the influence of the order of action behaviors. *K-means* treats

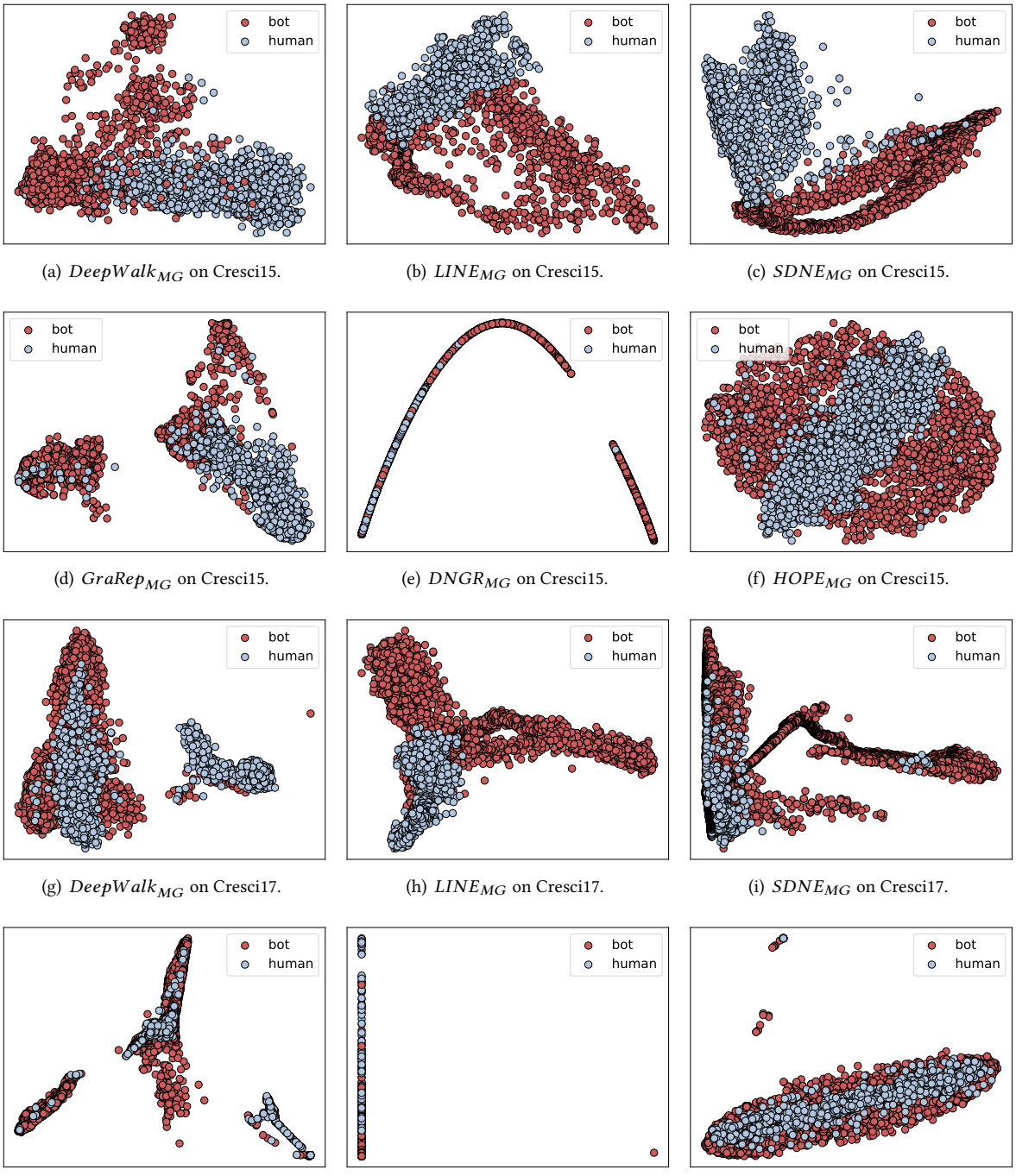
Table 4. Comparison of the average *ACC*, *Precision*, *Recall*, and *F*1 across different methods for social bot detection (unit:%). The best results are bolded, and the second-best results are underlined.

Method	Cresci-2015				Cresci-2017				Pronbots-2019				botwiki-2019			
	ACC	**P**	**R**	**F1**	**ACC**	**P**	**R**	**F1**	**ACC**	**P**	**R**	**F1**	**ACC**	**P**	**R**	**F1**
K-means	76.51	73.22	99.07	84.21	73.62	76.86	93.31	84.29	80.28	97.71	80.53	88.30	77.85	95.84	79.23	86.75
DNA	36.22	0.87	32.95	1.69	33.09	14.39	93.71	23.34	7.80	66.67	0.34	0.67	82.83	96.85	83.95	89.95
GAE	74.83	72.09	98.21	83.15	73.80	95.58	76.17	84.53	53.68	94.99	52.62	67.73	64.74	96.52	63.75	76.79
DeepWalk$_{MG}$	87.75	92.93	87.25	90.00	90.33	89.38	98.39	93.67	91.49	99.98	90.80	95.17	82.18	90.72	89.68	90.20
LINE$_{MG}$	55.30	62.01	78.38	69.01	72.72	72.72	**99.99**	84.20	92.36	92.36	**100**	96.03	71.17	97.05	70.63	81.76
Node2vec$_{MG}$	73.85	89.25	66.67	76.32	90.32	89.38	98.38	93.66	91.24	99.50	90.97	95.05	82.18	90.72	89.68	90.20
SDNE$_{MG}$	74.54	98.32	60.76	75.09	66.59	83.45	72.04	75.37	91.31	99.69	90.87	95.08	71.69	**100**	69.05	81.69
GraRep$_{MG}$	80.04	98.89	69.20	81.43	61.63	**99.70**	47.38	64.24	91.51	**100**	90.81	95.18	84.67	99.15	83.95	90.92
DNGR$_{MG}$	83.06	**98.92**	74.00	84.67	73.71	73.59	99.72	84.67	92.54	99.56	92.33	95.81	82.18	90.61	89.83	90.22
HOPE$_{MG}$	51.01	67.60	43.21	52.72	60.21	87.40	52.92	65.92	81.66	**100**	80.14	88.98	82.83	90.67	90.54	90.61
DeepWalk$_{G}$	76.08	79.80	82.88	81.41	62.71	73.52	79.41	76.33	63.14	99.47	60.41	75.17	77.72	99.81	75.79	86.16
LINE$_{G}$	64.80	66.67	88.61	76.09	58.80	96.78	47.23	63.48	92.34	92.35	99.99	96.02	61.07	98.08	58.59	73.36
Node2vec$_{G}$	64.34	63.94	99.99	78.00	50.42	71.77	57.06	63.57	66.99	99.23	64.76	78.37	82.44	100	80.80	89.38
SDNE$_{G}$	68.63	66.84	**100**	80.12	72.20	77.46	89.88	82.96	87.48	91.96	94.72	93.32	53.87	86.34	58.88	70.01
GraRep$_{G}$	67.65	66.15	**100**	79.62	57.48	71.78	72.39	72.08	50.04	**100**	45.90	62.92	66.71	**100**	63.61	77.76
DNGR$_{G}$	60.24	64.72	81.58	72.17	75.80	75.82	99.96	86.23	92.36	92.37	99.98	96.03	90.96	91.55	**99.28**	95.26
HOPE$_{G}$	66.67	65.47	**100**	79.14	50.40	75.92	50.65	60.76	50.46	94.03	49.50	64.85	53.34	**100**	48.99	65.77
UnDBot	**89.12**	96.11	86.27	**90.93**	**93.46**	98.07	92.83	**95.38**	**92.68**	92.75	99.88	**96.18**	**93.18**	97.78	94.70	**96.22**

social bot identification as anomaly detection and labels users whose feature properties deviate from the core population as social bots. Although *K-means* demonstrates higher detection accuracy compared to *DNA*, it is sensitive to the initial clustering centers and struggles to handle noisy data effectively. Furthermore, clustering social users solely based on directly obtained user features easily results in highly disguised social bots being wrongly assigned to human clusters, thus affecting accuracy. Therefore, the accuracy of *K-means* is only around 75% in the four dataset scenarios. *GAE* combines GCNs and autoencoders to learn low-dimensional feature embeddings for unlabeled samples. However, its performance is influenced by the choice of autoencoder and graph structure, and the overall effect is not as good as *K-means*. In comparison, the advantage of UnDBot lies in its ability to analyze user behavior characteristics while considering the network structure. Unlike *K-means* and *DNA*, which directly determine individual social bots based on the numerical values of raw feature values or action sequences, UnDBot identifies social bot communities through behavioral similarity within the entire social network. It comprehensively considers the relationships with other users, enhancing detection effectiveness.

Next are seven models based on unsupervised graph representation learning implemented on the multi-relation graph *MG* constructed in Section 3.1. *DeepWalk* utilizes a random walk approach with depth-first traversal to sample neighboring nodes in a graph and learn node embeddings that capture structural information. *Node2vec* improves upon *DeepWalk*'s random walk method by incorporating breadth-first traversal. *LINE*, on the other hand, employs breadth-first sampling of neighboring nodes and combines first-order and second-order similarities to learn node embeddings. *SDNE* extends *LINE* by incorporating autoencoders to optimize the similarities in LINE. On our multi-relation graph constructed based on behavior similarity, *DeepWalk* achieves higher accuracy by employing depth-first sampling of neighboring nodes. However, *Node2vec*, influenced by limited information from breadth-first search, exhibits slightly lower accuracy than *DeepWalk* regarding identification. The baselines *LINE* and *SDNE*, which rely on breadth-first sampling, demonstrate generally lower detection performance compared to *Node2vec*. The accuracy of *LINE* on the Cresci-2015 dataset is only slightly higher than 50%, and they only learn appropriate node embeddings to distinguish social bots in the experimental setting on the Pronbots-2019 dataset. *GraRep*, *DNGR*, and *HOPE* are all graph representation learning models based on matrix factorization. The difference lies in their approaches. *GraRep* can be extended to capture higher-order proximity information and preserve the graph structure in low-dimensional representations, but it overlooks the contextual information of some nodes. *DNGR* utilizes the PPMI matrix to capture potential complex nonlinear relationships between different vertices. *GraRep* and *DNGR* demonstrate similar performance, with overall detection effectiveness superior to *Node2vec* but inferior to *DeepWalk*, especially with low accuracy on the Cresci-2017 dataset. *HOPE* maintains higher-order similarity relationships between nodes through matrix factorization of higher-order proximity matrices. It has limited expressive power and scalability and exhibits the worst performance among unsupervised graph representation learning baselines in social bot detection. However, it performs well on the Botwiki-2019 dataset. In comparison, UnDBot has the advantage of utilizing structural information theory to effectively leverage the structural information of the multi-relational graph of social users. This approach results in hierarchical community partitioning, leading to more accurate classification and identification of social users.

To demonstrate the impact of graph construction on the effectiveness of the baselines, we additionally conduct experiments on the seven unsupervised graph representation learning models using the homogeneous graph G. The homogeneous graph G is described in detail in Section 4.3. On the Cresci-2015 dataset, the unsupervised graph representation learning models in the homogeneous graph G exhibit low discriminability for social bots. The best-performing model, *DeepWalk*, achieves only 76% accuracy, while the accuracies of other baseline models are below 70%. However, unlike using the multi-relational graph *MG*, these models demonstrate relatively high recall rates, with *SDNE*, *GraRep*, and *HOPE* reaching a Recall of 100%. For the other three datasets, except for *DNGR*, the unsupervised graph representation learning models on the homogeneous graph G generally exhibit poor performance in detecting social bots. We notice that *DNGR* shows relatively high recall rates on the Cresci-2017 dataset and even outperforms all other baseline models in effectiveness on Pronbots-2019 and Botwiki-2019



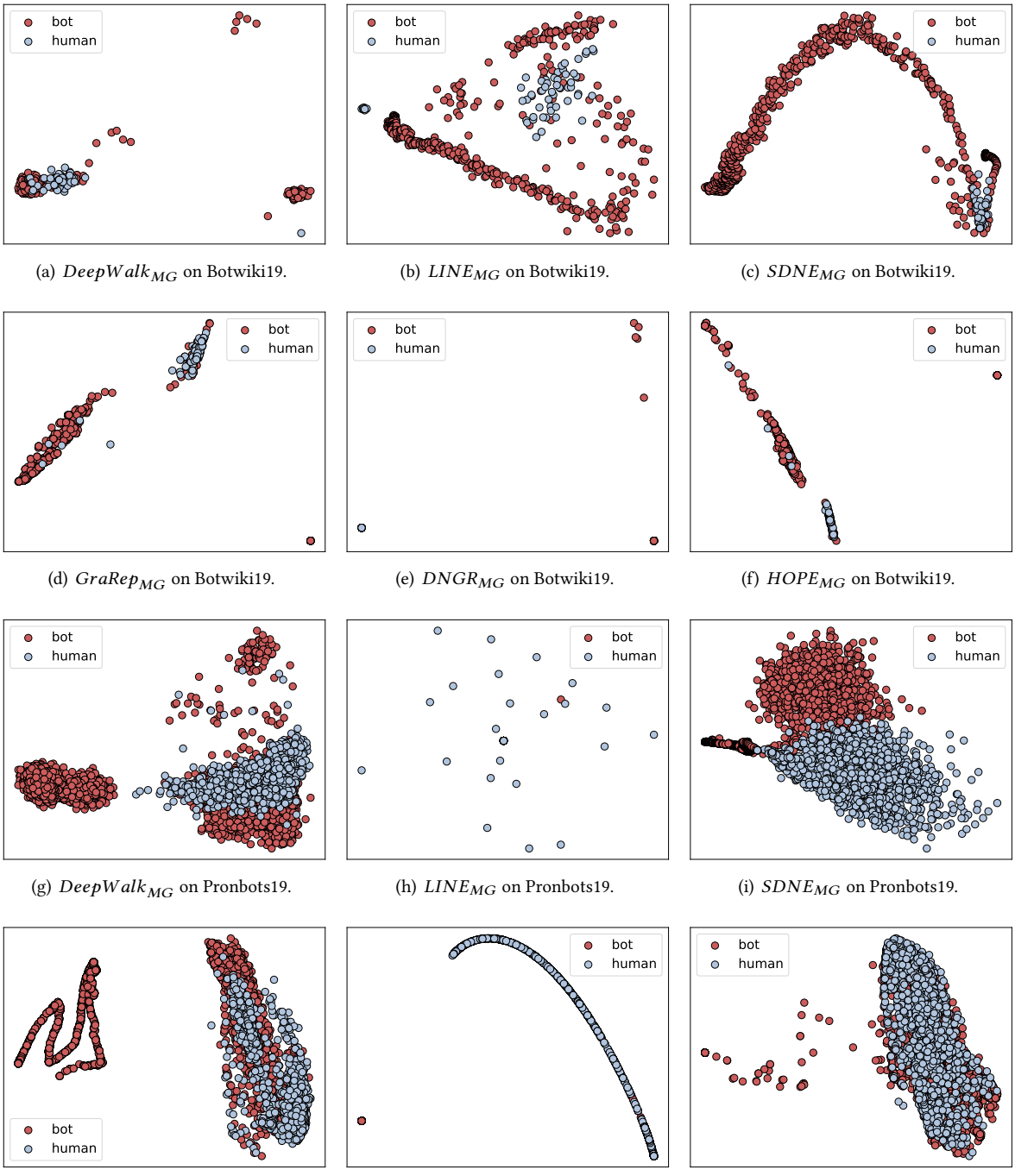
(a) $DeepWalk_{MG}$ on Cresci15. (b) $LINE_{MG}$ on Cresci15. (c) $SDNE_{MG}$ on Cresci15.

(d) $GraRep_{MG}$ on Cresci15. (e) $DNGR_{MG}$ on Cresci15. (f) $HOPE_{MG}$ on Cresci15.

(g) $DeepWalk_{MG}$ on Cresci17. (h) $LINE_{MG}$ on Cresci17. (i) $SDNE_{MG}$ on Cresci17.

(j) $GraRep_{MG}$ on Cresci17. (k) $DNGR_{MG}$ on Cresci17. (l) $HOPE_{MG}$ on Cresci17.

Fig. 4. Distribution of user embeddings generated by unsupervised graph learning models on the Cresci-2015 and Cresci-2017 datasets.



(a) $DeepWalk_{MG}$ on Botwiki19. (b) $LINE_{MG}$ on Botwiki19. (c) $SDNE_{MG}$ on Botwiki19.

(d) $GraRep_{MG}$ on Botwiki19. (e) $DNGR_{MG}$ on Botwiki19. (f) $HOPE_{MG}$ on Botwiki19.

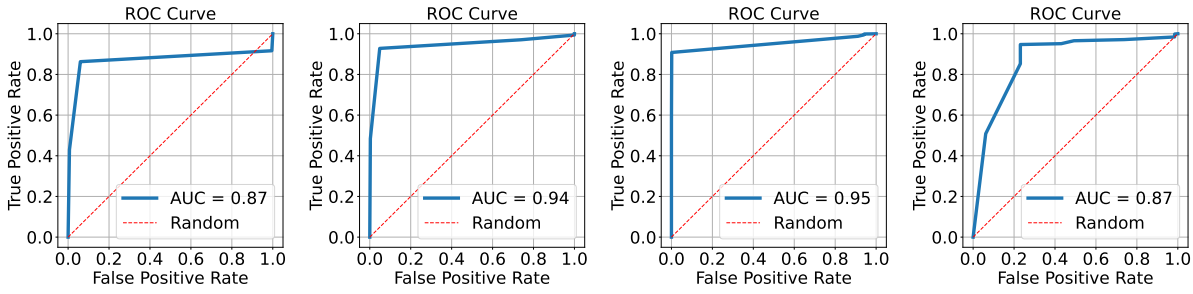
(g) $DeepWalk_{MG}$ on Pronbots19. (h) $LINE_{MG}$ on Pronbots19. (i) $SDNE_{MG}$ on Pronbots19.

(j) $GraRep_{MG}$ on Pronbots19. (k) $DNGR_{MG}$ on Pronbots19. (l) $HOPE_{MG}$ on Pronbots19.

Fig. 5. Distribution of user embeddings generated by unsupervised graph learning models on the Botwiki-2019 and Pronbots-2019 datasets.

datasets. However, considering the four datasets together, its overall performance is unsatisfactory. In comparison, UnDBot achieves high accuracy on all four datasets and maintains a balance between precision and recall. Each unsupervised graph representation learning baseline achieved significant improvement in detecting social bots on certain datasets after using *MG*. In the Cresci-2015 dataset, the detection accuracy or precision of the five baselines (*DeepWalk*, *Node2vec*, *SDNE*, *GraRep*, *DNGR*) greatly improved after using *MG*. In the Cresci-2017 dataset, the detection accuracy or precision of the five baselines (*DeepWalk*, *LINE*, *Node2vec*, *GraRep*, *HOPE*) greatly improved after using *MG*. In the Pronbots-2019 dataset, the detection accuracy or precision of the six baselines (*DeepWalk*, *LINE*, *Node2vec*, *SDNE*, *GraRep*, *DNGR*, *HOPE*) greatly improved after using *MG*. In the Botwiki-2019 dataset, the detection accuracy or precision of the four baselines (*DeepWalk*, *LINE*, *SDNE*, *GraRep*, *HOPE*) greatly improved after using *MG*. Overall, the effectiveness of social bot detection using the social user graph *MG* far surpasses that of using graph *G* on baselines. We will delve deeper into the effectiveness of the multi-relational graph *MG* and analyze the impact of each type of edge within it in Section 5.2.

To gain a deeper understanding of the effectiveness of the learned representation vectors, we employed Principal Component Analysis (PCA) to reduce the dimensionality of embeddings learned by unsupervised graph representation learning-based baselines on the multi-relational graph (*MG*) to a two-dimensional vector and visualize them on a plane graph. We evaluated six baseline models based on four datasets separately. Figure 4 depicts the dimensionality reduction effect of user representation vectors on the Cresci-2015 and Cresci-2017 datasets. Figure 5 illustrates the dimensionality reduction effect of user representation vectors on the Pronbots-2019 and Botwiki-2019 datasets. Blue dots represent humans, while red dots represent social bots. The unsupervised graph representation learning models embed network structure data into a low-dimensional space reflected in each user's embeddings. As illustrated in Figures 4(d), 4(f), 4(i) and 4(k), these embeddings exhibit a limited distinction between social bots and human entities, which leads to the challenge of discerning between social bots and humans. Besides, as shown in Figures 5(a), 5(c), 5(d), 5(g), and 5(l), on the Pronbots -2019 and Botwiki-2019 datasets, both of which consist mostly social bots and a small number of humans, the learned embeddings do not exhibit concentration, with high overlap between human and bot embeddings. Furthermore, the learned embeddings and the two-dimensional vectors obtained by PCA dimensionality reduction lack interpretability. In contrast, UnDBot utilizes structural information theory to decode the network structure into a hierarchical encoding tree, resulting in high discriminability for social bots and interpretability as detailed in Section 5.3.



(a) Cresci15. (b) Cresci17. (c) Pronbots19. (d) Botwiki19.

Fig. 6. AUC and ROC curves on four datasets.

To evaluate the stability of the UnDBot with learned community label index, we treat $E_v(\alpha)$ equally as the predicted value for each user and report the area under the ROC curves. As shown in Figure 6, UnDBot achieves good performances in all four datasets, especially for imbalanced datasets like Pronbots19 and Botwiki19. These

results demonstrate that by combining community influence and community cohesion to label social bots, we can effectively deal with unequal distributions of bots and users in the real world.

5.2 Ablation Study

In this section, we conduct ablation experiments on the graph modeling of UnDBot to demonstrate the rationality of social users' multi-relational graph *MG*. Additionally, we perform experiments by varying three hyperparameters of UnDBot to analyze the sensitivity of the model to hyperparameters.

Table 5. The total number of edges, the average number of edges per user, and the average connection degree of users in Multi-relation Graph, Homogeneity Graph, and Ordinary Multi-relation Graph for different datasets.

Datasets	Multi-relation Graph (MG)			Homogeneity Graph (G)			Ordinary MG (mg)		
	Total	**Avg.**	**Conn.**	**Total**	**Avg.**	**Conn.**	**Total.**	**Avg.**	**Conn.**
Cresci-2015	4,109,097	516.7	9.75%	4,276	1.6	0.03%	1,966,693	247.3	4.67%
Cresci-2017	19,958,930	1044.6	8.20%	12,326,304	1715.7	11.94%	9,749,755	510.3	4.01%
Pronbots-2019	131,935,035	4542.5	23.46%	40,244,245	4156.8	21.44%	2,896,693	99.7	0.52%
Botwiki-2019	181,839	158.9	20.85%	107,326	281.3	36.92%	35,294	30.8	4.05%

5.2.1 ***Graph Effectiveness***. To demonstrate the rationality of the social user multi-relational graph *MG* proposed in Section 3.1 for social bot detection, we conduct comparative experiments on UnDBot using the homogeneous graph *G* (denoted as UnDBot-G) and an ordinary Multi-relation graph *mg* (denoted as UnDBot-mg). Besides, to evaluate the positive impact of the edges used in the modeling of the social user multi-relation graph in UnDBot on the performance of social bot detection, we also construct three variations by removing one type of edge from the multi-relation graph for ablation experiments (denoted as UnDBot-FT, UnDBot-FI, and UnDBot-TI). Except for the graph construction, all other experimental settings remain the same as described in Section 4.4. All experimental results across the four datasets are reported in Table 6, Table 7, Table 8, and Table 9, respectively. Table 5 lists the total edges, the average number of edges under one type per user, and the average connection degree of users in the four datasets under graph *MG*, graph *G*, and graph *mg*. In the case of the Cresci-2015 dataset, the homogeneous graph *G* is sparse due to the use of the action of following between users. For the other datasets, there is no significant difference in the average connectivity level between the homogeneous graph *G* and the multi-relational graph *MG*. In the case of the Pronbots-2019 dataset, the ordinary Multi-relation graph *mg* is sparse due to the less similarity between direct user behaviors. The other three datasets have similar degrees of connectivity, but are all sparser than *MG*.

Table 6 shows that UnDBot-*G* with sparse edges leads to many communities and low accuracy after clustering based on structural entropy. Besides, as shown in Table 7 and Table 9, the accuracy of using graph *G* is much lower compared to the *MG*. This is because social bots make their attributes indistinguishable from humans through camouflage. Therefore, utilizing the proposed social user multi-relational graph *MG* for UnDBot is more rational. To further analyze *MG* on UnDBot, we also investigate the efficacy of three types of edges within the multi-relational graph *MG* as well as the results of using other types of edges. For *MG* in the variation UnDBot-FT, only the edges related to the *following-follower ratio* and *posting type distribution* are retained. When the influence of *posting influence* is excluded in the Cresci-2015, Cresci-2017, and Botwiki-2019 datasets, the detection accuracy significantly decreases, but the recall rate increases to 98.07% on Cresci-2017 and 95.13% on Botwiki-2019. However, in the Pronbots-2019 dataset, the accuracy, precision, and F1 score of the variation UnDBot-FT increases by 3.71%,

Table 6. Comparison with the ACC, Precision, Recall, F1 and the number of communities of different variations on Cresci-2015 dataset (unit:%).

Variation	Cresci-2015				
	ACC	**Precision**	**Recall**	**F1**	**Num_comm**
UnDBot-G	27.69	37.17	20.83	26.70	3731
UnDBot-mg	63.08	63.17	**99.79**	77.36	9
UnDBot-FT	58.06	61.76	88.36	72.71	8
UnDBot-FI	64.97	64.48	99.28	78.18	65
UnDBot-TI	59.71	61.97	93.85	74.65	10
UnDBot	**89.12**	**96.11**	86.27	**90.93**	**4**
Gain	**24.15-61.43**↑	**31.63-58.94**↑	13.52↓	**12.75-64.23**↑	**4-3727**↓

Table 7. Comparison with the ACC, Precision, Recall, F1 and the number of communities of different variations on Cresci-2017 dataset (unit:%).

Variation	Cresci-2017				
	ACC	**Precision**	**Recall**	**F1**	**Num_comm**
UnDBot-G	68.79	74.59	89.24	81.26	15
UnDBot-mg	67.30	71.25	92.28	80.41	10
UnDBot-FT	77.58	77.24	**98.07**	86.42	19
UnDBot-FI	86.50	87.02	95.71	91.16	2524
UnDBot-TI	75.83	76.61	96.12	85.26	13
UnDBot	**93.46**	**98.07**	92.83	**95.38**	**6**
Gain	**6.96-26.16**↑	**11.05-26.82**↑	5.24↓	**4.22-14.97**↑	**4-2518**↓

Table 8. Comparison with the ACC, Precision, Recall, F1, and the number of communities of different variations on Pronbots-2019 dataset (unit:%).

Variation	Pronbots-2019				
	ACC	**Precision**	**Recall**	**F1**	**Num_comm**
UnDBot-G	93.52	95.73	97.33	96.52	**13**
UnDBot-mg	92.33	92.36	**99.96**	96.01	40
UnDBot-FT	**96.39**	**96.83**	99.34	**98.07**	928
UnDBot-FI	16.89	69.13	18.07	28.66	14827
UnDBot-TI	93.01	93.24	99.66	96.34	101
UnDBot	92.68	92.75	99.88	96.18	58
Gain	3.71↓	4.08↓	0.08↓	1.89↓	45↑

Table 9. Comparison with the ACC, Precision, Recall, F1, and the number of communities of different variations on Botwiki-2019 dataset (unit:%).

Variation	Botwiki-2019				
	ACC	**Precision**	**Recall**	**F1**	**Num_comm**
UnDBot-G	86.37	92.79	92.26	92.53	14
UnDBot-mg	87.42	91.46	**95.13**	93.26	18
UnDBot-FT	90.43	94.45	**95.13**	94.79	29
UnDBot-FI	75.23	90.33	81.66	85.78	153
UnDBot-TI	84.80	**97.86**	85.24	91.11	**11**
UnDBot	**93.18**	97.78	94.70	**96.22**	**11**
Gain	**2.75-17.95**↑	0.08↓	0.43↓	**1.43-10.44**↑	**0-142**↓

4.08%, and 1.89% compared to the original UnDBot. For *MG* in the variation UnDBot-FI, only the edges related to the *following-follower ratio* and *posting influence* are retained. This significantly decreases the detection accuracy and precision in all four datasets. Especially on the Pronbots-2019 dataset, the accuracy rate is even less than 20%. Additionally, due to the lack of rich connectivity structure related to *posting type distribution*, the variation UnDBot-FI forms a larger number of communities with a smaller average size, resulting in a more dispersed social bot community. It can be observed that the number of communities increases most compared with UnDBot. For *MG* in the variation UnDBot-TI, only the edges related to *posting type distribution* and *posting influence* are retained. In the Pronbots-2019 datasets, the accuracy, precision, recall, F1 score, and the number of communities are all comparable to or slightly higher than the original UnDBot model. However, in the Cresci-2015, Cresci-2017, and Botwiki-2019 datasets, the accuracy and precision drop significantly even with less than 60% accuracy on Cresci-2015. For ordinary Multi-relation graph *mg* in the variation UnDBot-mg, three new relations (*U-twe-U*, *U-fri-U*, *U-fav-U*) is constructed based on direct user behaviors to prove the validity of the three relationships selected in *MG*. UnDBot-mg only reflects a higher recall rate. In addition, the accuracy, precision, and F1 on the four datasets are far inferior to UnDBot. Overall, integrating three types of edges can significantly enhance the detection accuracy of UnDBot. Therefore, it is necessary to jointly model the multi-relational social user graph of the three relationships to unleash the full potential of UnDBot's performance optimization.

5.2.2 ***Hyperparameter Sensitivity.*** For the three hyperparameters involved in UnDBot (similarity threshold ξ, parallel ratio p, cohesive weight π), we conduct a series of hyperparameter sensitivity experiments by varying individual parameters at a time and repeating the experiments on UnDBot. Specifically, similarity threshold ξ represents the threshold for user behavior feature bias during the graph construction process. parallel ratio p represents the maximum scale ratio for parallel merge operators, indicating the proportion of community numbers that can be fused compared to the maximum fusion limit. Lastly, cohesive weight π represents the community cohesion proportion in the evaluation metric. The experimental results, as shown in Figure 7, indicate that the accuracy of social bot detection is significantly influenced by the similarity threshold ξ parameter on the first three datasets. In particular, the detection performance is better when similarity threshold ξ is around 0.1. This is because a very small ξ value leads to a lack of important connections in the constructed graph, while a very large ξ value introduces excessive noise into the graph. However, this has little impact on Pronbots-2019, on the contrary, the minimum ξ of 0.01 achieves the best performance. This may be because the users themselves in this dataset have high similarity and the smaller ξ filters out redundant information. In the context of parallel operations for fusion operators, apart from the Pronbots-2019 dataset, selecting a relatively small value for parallel

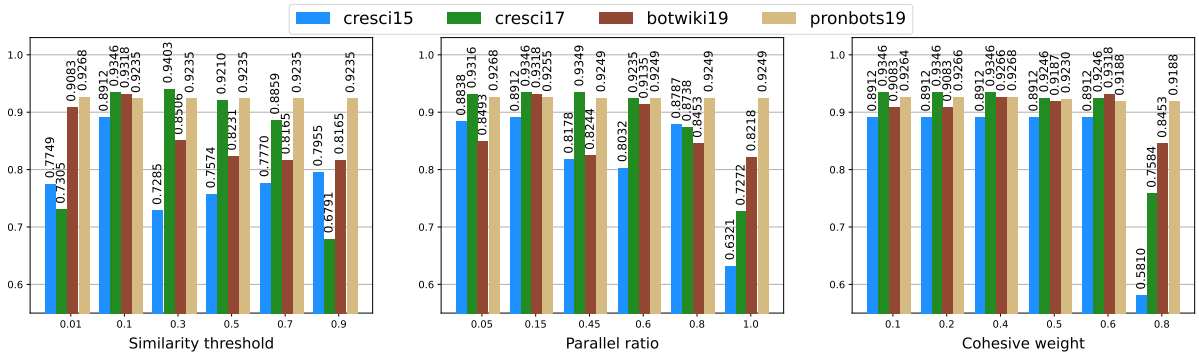


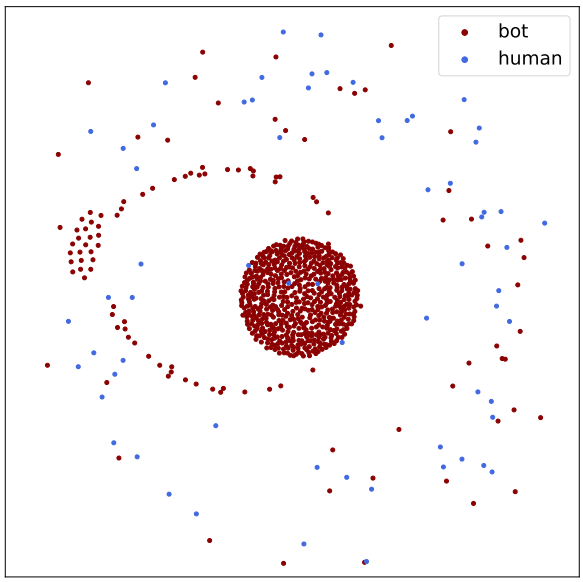
Fig. 7. Hyperparameter sensitivity.

ratio p can improve the detection performance. This is because having an excessive number of parallel operands can potentially lead to the merging of suboptimal communities, resulting in a suboptimal encoding tree and a less stable partitioning of communities. However, setting parallel ratio p to a lower value will also increase the number of rounds for fusion operator execution, thereby reducing the model's efficiency. Hence, we uniformly set p to 0.05 for Pronbots-2019 and 0.15 for other datasets, considering both performance and efficiency factors. Besides, we also find that the cohesive weight π parameter significantly impacts the Botwiki-2019 dataset, while its value does not affect the detection accuracy of Cresci-2015, Cresci-2017, and Pronbots-2019 datasets under non-extreme conditions.

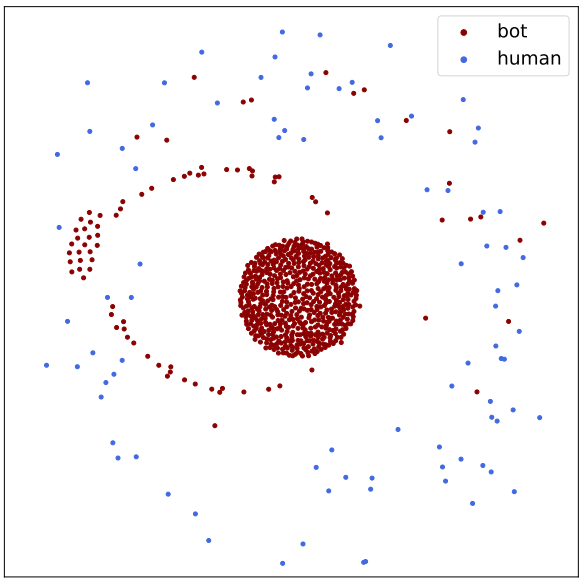
5.3 Interpretability

Interpretability refers to the extraction of knowledge about the relationships contained in the data from a model, providing insights to the audience regarding the detection of social bots [57]. To gain a deeper understanding of the interpretability of the UnDBot, we visualize social user network graphs and structural entropy encoding trees on four datasets. The social user networks in Figure 8, Figure 9, Figure 10, and Figure 11 utilize the spring_layout algorithm [42] to calculate the positional parameters of each user node, taking into account both the nodes and edges. This algorithm uses Fruchterman-Reingold [30] to arrange the nodes, introducing a physical model into the nodes so that the overall layout achieves a dynamic balance by minimizing the system's total energy, resulting in nodes with connected edges being closer in position. In social user networks, red dots represent social bots, and blue dots represent genuine users. Figure 8(a), Figure 9(a), Figure 10(a), and Figure 11(a) show the network with true labels of users, while Figure 8(b), Figure 9(b), Figure 10(b) and Figure 11(b) depict the network with labels obtained through the detection of UnDBot. In the structure entropy encoding tree, square tree nodes represent social user communities, and circular leaf nodes represent social users. Blue nodes represent human communities (users), while red nodes represent social bot communities (users). The numbers above the communities indicate the number of users in each community, representing the community size.

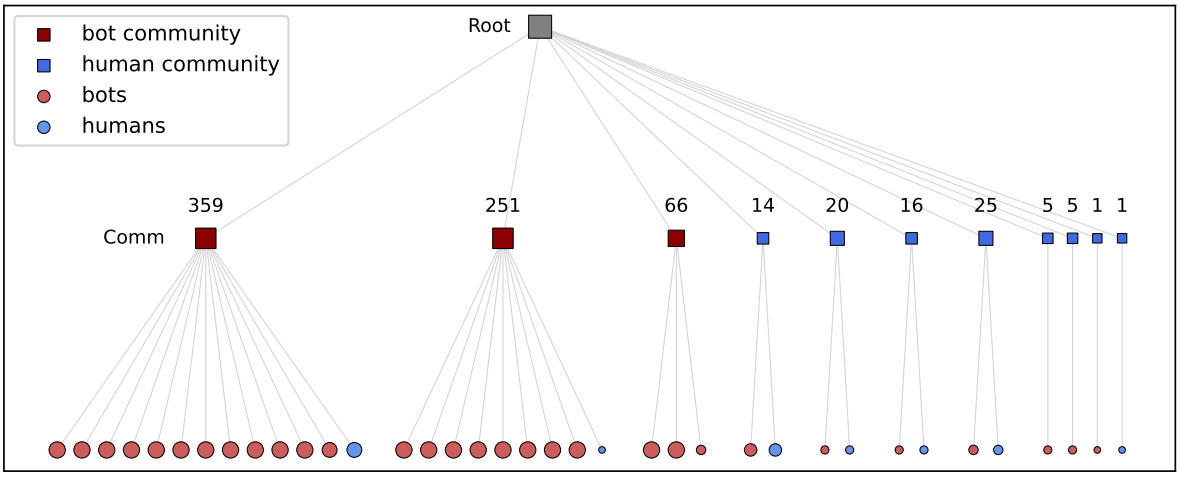
Figure 8(a), Figure 9(a), Figure 10(a), and Figure 11(a) illustrate that social bots are concentrated in the distribution of the multi-relational graph (MG) constructed based on social behavior similarity while humans are distributed more widely on the periphery. Social bots are typically controlled by machines or programs, serving specific purposes such as advertising, information theft, or inciting public opinion. So their behavior tends to be more similar compared to humans, resulting in a higher density of connections among social bots in the MG, whereas connections between humans or between humans and social bots are relatively sparse. UnDBot intends to identify communities with a higher average number of connections (greater influence) and a higher number of



(a) Network with true label.



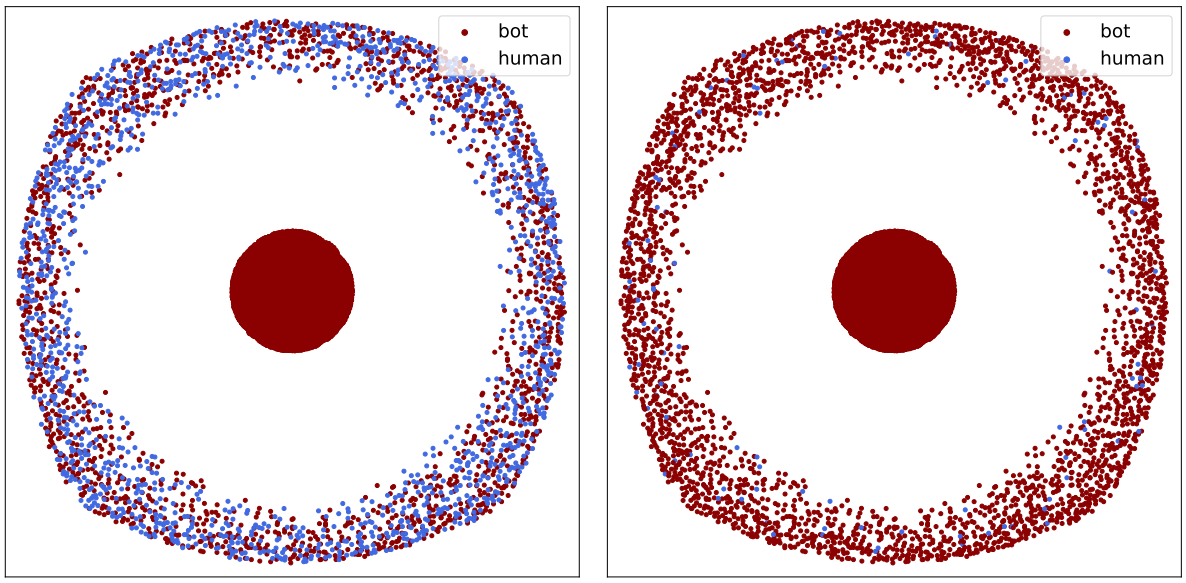
(b) Network with pred label.



(c) The optimal encoding tree of Botwiki-2019 dataset in UnDBot (square tree nodes are communities, circular leaf nodes are social users, and the size of the nodes indicates the number of social users. For visualization, each large leaf node represents 20 social users of Botwiki-2019).

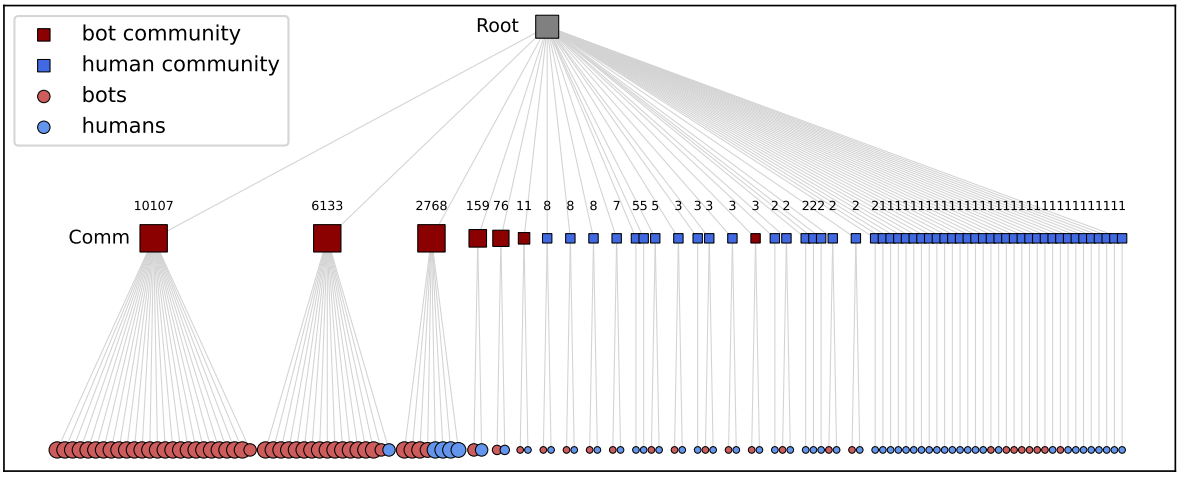
Fig. 8. Renderings of Botwiki-2019.

internal connections (stronger cohesion) as social bot communities. Figure 8(b), Figure 9(b),Figure 10(b), and Figure 11(b) illustrate the identification results of UnDBot for social users. Users with dense distributions are classified as social bots, while users with relatively dispersed distributions are identified as humans.



(a) Network with true label.

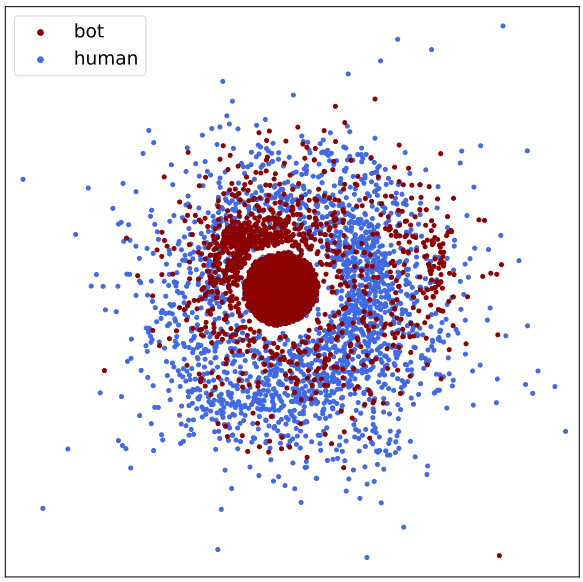
(b) Network with pred label.



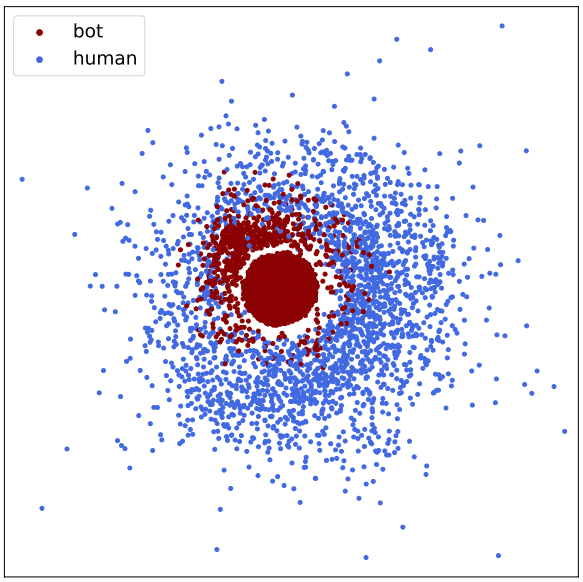
(c) The optimal encoding tree of Pronbots-2019 dataset in UnDBot (square tree nodes are communities, circular leaf nodes are social users, and the size of the nodes indicates the number of social users. For visualization, each large leaf node represents 200 social users of Pronbots-2019).

Fig. 9. Renderings of Pronbots-2019.

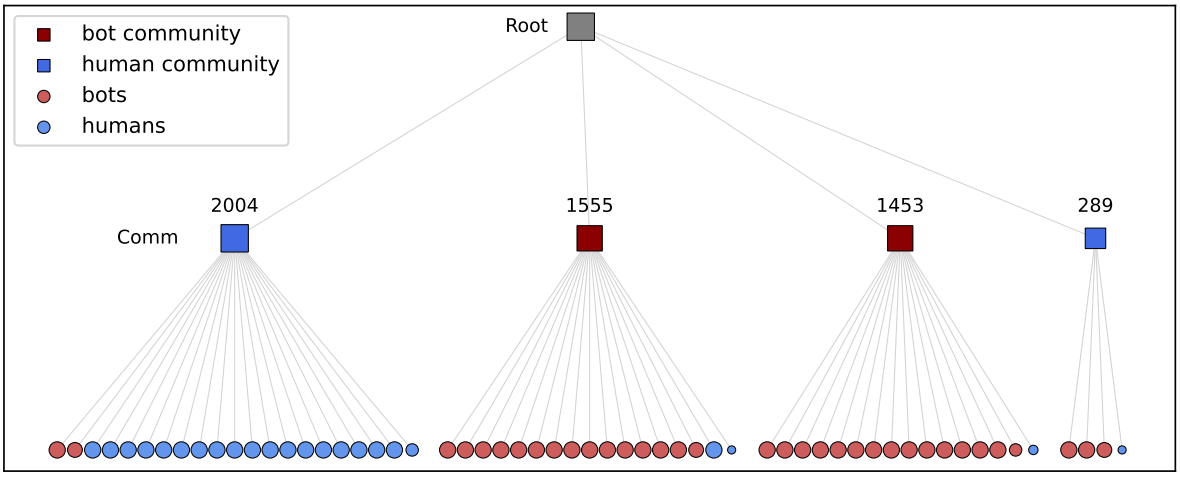
Figure 8(c), Figure 9(c), Figure 10(c), and Figure 11(c) depict the community division on the encoding tree, respectively. To provide a clearer representation, each large circular leaf node represents 30 users in Botwiki-2019, 300 users in Pronbots-2019, 100 users in Cresci-2015, and 200 users in Cresci-2017. The smaller the diameter of the node, the fewer users it represents. The encoding tree partitions social bots and humans into different



(a) Network with true label.



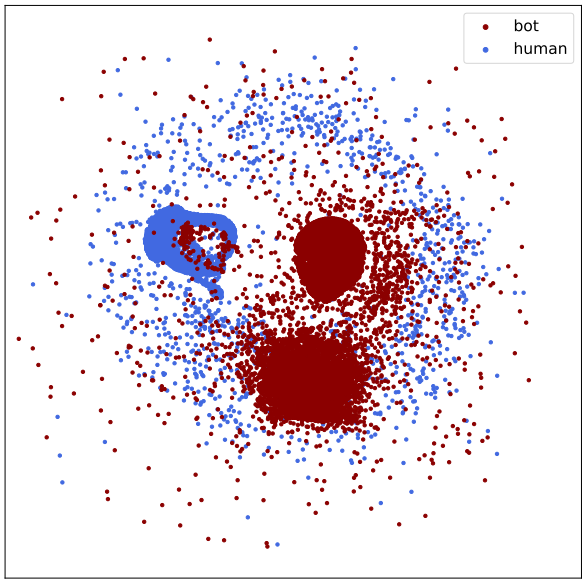
(b) Network with pred label.



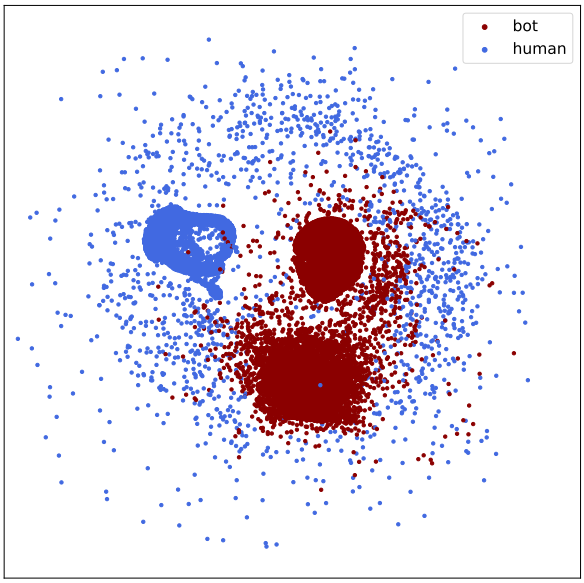
(c) The optimal encoding tree of Cresci-2015 dataset in UnDBot (square tree nodes are communities, circular leaf nodes are social users, and the size of the nodes indicates the number of social users. For visualization, each large leaf node represents 100 social users of Cresci-2015).

Fig. 10. Renderings of Cresci-2015.

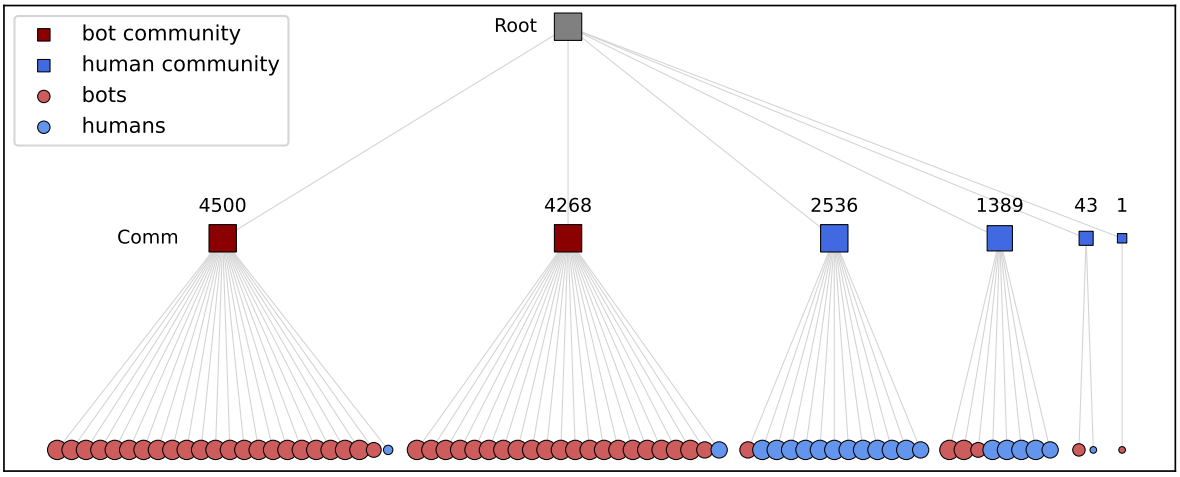
communities. In Figure 8(c) and Figure 9(c), the communities exhibit a long-tail distribution, with fewer large-scale communities and more small-scale communities. The communities marked as social bot communities in Botwiki-2019 and Pronbots-2019 are predominantly large-scale communities. On the other hand, in the encoding



(a) Network with true label.



(b) Network with pred label.



(c) The optimal encoding tree of Cresci-2017 dataset in UnDBot (square tree nodes are communities, circular leaf nodes are social users, and the size of the nodes indicates the number of social users. For visualization, each large leaf node represents 200 social users of Cresci-2017).

Fig. 11. Renderings of Cresci-2017.

trees of Cresci-2015 and Cresci-2017, the distribution of community sizes is relatively uniform, and there is a high proportion of single-type user components within each community.

UnDBot employs structural information theory to decode the essential structure of the social bot network from the multi-relationship graph. It extracts knowledge of user community division from the encoding tree.

From the perspective of the network modeled by behavioral similarity, the distinction between social bots and humans lies in the fact that social bots exhibit dense distributions and strong intra-community cohesion. This provides interpretable insights for the detection of social bots by UnDBot.

5.4 Time Analysis

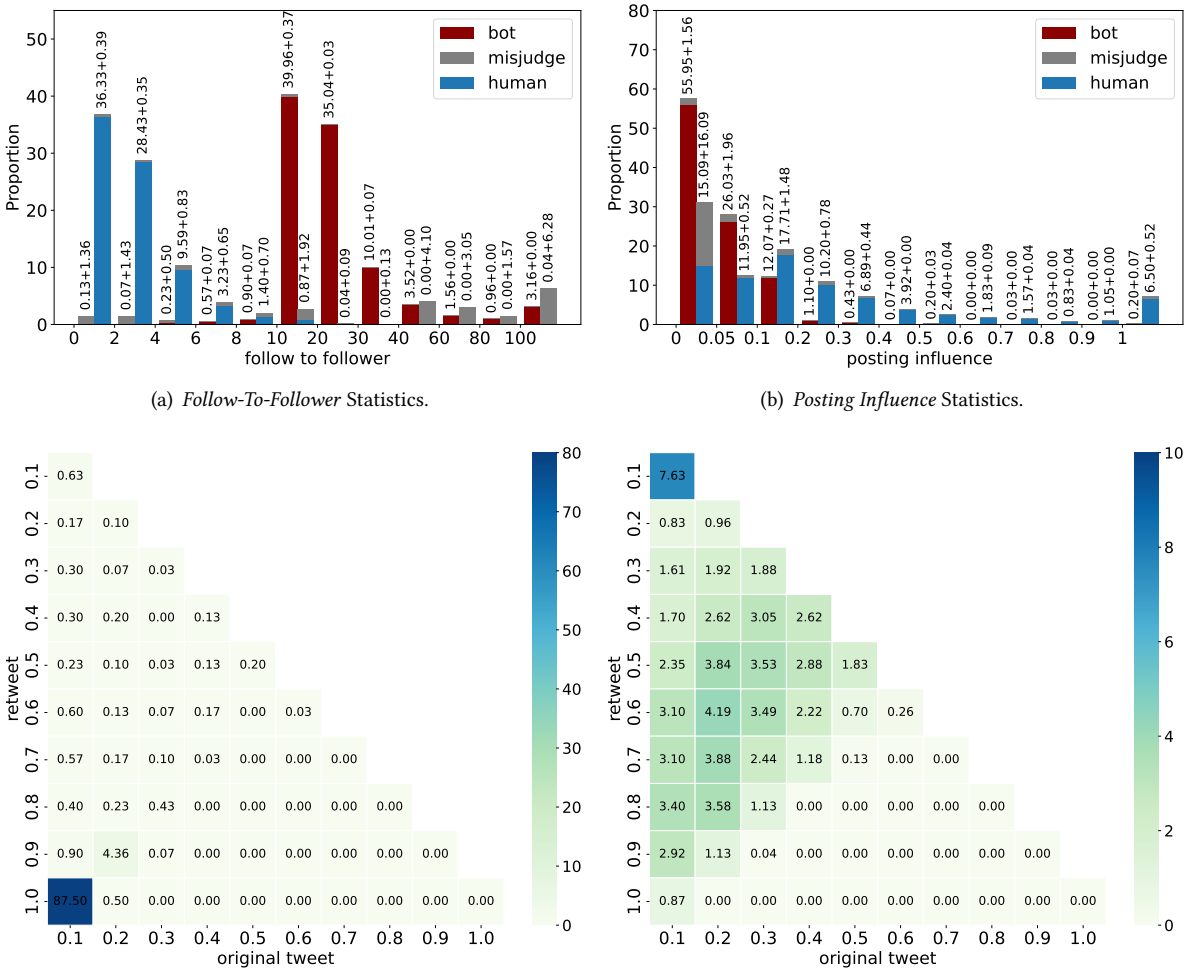
Table 10. Average time per run for UnDBot and all baselines (Unit: Second).

Method	**Cresci-2015**	**Cresci-2017**	**Pronbots-2019**	**Botwiki-2019**
K-means	4.39	8.76	5.78	4.05
DNA	63052.45	224525.64	43832.82	419.31
GAE	2.36	1.63	2.07	0.39
DeepWalk	60.78	308.22	1841.80	2.78
LINE	859.57	4542.45	49379.05	214.55
Node2vec	39.38	165.18	1308.81	8.47
SDNE	22.12	127.65	792.07	1.48
GraRep	179.92	925.80	3394.07	4.85
DNGR	47.44	209.86	1346.57	1.64
HOPE	24.42	141.69	1167.06	1.15
UnDBot	7.37	55.98	541.86	1.12

This section evaluates the efficiency of UnDBot and various baselines. We primarily assess efficiency by measuring the runtime of the models. To facilitate a fairer comparison for unsupervised graph learning-based models, we conduct temporal analysis on the multi-relational social user graph MG. Table 10 presents the average runtime of each model over 10 runs. Overall, UnDBot achieves a balance between runtime and effectiveness. Although the *K-means* model is too simplistic and the *GAE* package allows for parallel computation to accelerate the training process, resulting in less required running time, the accuracy and precision achieved by both of them are significantly lower than UnDBot across all datasets, particularly on Pronbots-2019 and Botwiki-2019 where the data types are relatively homogeneous. It is worth noting that, compared to other baselines, UnDBot's runtime is generally reduced by over threefold. The runtime of the *DNA* model is the longest, as it involves considering all possible combinations of actions for tweeting.

5.5 Case Study

To further understand the distinctions between social bots and humans, we compare various characteristics under UnDBot, such as the follower-to-following ratio, posting type distribution, and posting influence. We collate the statistical distribution of identified social bots and humans for each feature value separately. This analysis aims to determine whether these social users align with human judgment. Figure 12 displays the results of the statistical analysis of Cresci-2015 based on the identification performance of UnDBot. For one-dimensional features such as *follow-to-follower ratio* and *posting influence*, bar charts Figures 12(a) and 12(b) illustrate the proportion of users falling into each interval, with the horizontal axis representing the value intervals of the feature values and the vertical axis representing the proportion of users. Red represents the social bots identified by UnDBot, blue represents humans, and gray represents misjudged users. Each bar consists of two components:



(a) *Follow-To-Follower* Statistics.

(b) *Posting Influence* Statistics.

(c) *Posting Type Distribution* Statistics on Social Bot.

(d) *Posting Type Distribution* Statistics on Social Human.

Fig. 12. Feature Statistics

the proportion of social bots (humans) and the misjudged ratio. Heat maps Figures 12(c) and 12(d) depict the proportions of social bots and humans falling into each interval of the *posting type distribution*. The horizontal axis represents the proportion of original tweets, while the vertical axis represents the proportion of retweets. Darker colors indicate a higher proportion of users. It can be found from the results that the social bots identified by UnDBot generally have a high follow-to-following ratio, very low posting influence, and the type of posting is single (such as retweeting), which is consistent with the human judgment of social bots. Furthermore, the main reason for misjudgments is that the characteristics of some social bots have not been identified. This is because they are relatively similar to humans in certain characteristics or have a smaller scale. This also provides us with inspiration for future research directions. When the behavior of social bots under AI control is almost

indistinguishable from humans or when the behavior similarity between social bots is not high, how do we explore more effective features to identify them?

6 RELATED WORK

In this section, we summarize the related literature, baselines, and approaches related to the proposed framework. It is primarily divided into three parts: social bot detection, network embedding, and structural information theory based applications.

6.1 Social Bot Detection

Early social bot detection methods were mainly based on the user's inherent characteristics and text content, such as the length of the username, number of tweets, number of fans, number of friends, number of likes in the username information, part-of-speech features, frequency features in the tweet information, emotional characteristics, etc. The earliest method [56] treated spam identification as anomaly detection to handle the streaming nature of tweets effectively. Subsequently, a method of analyzing digital DNA sequences in user behavior to model online user behavior [18] provided a new idea for behavioral representation. And fully analyzing users' basic information and social activities [24] has been proven to improve the accuracy. A later proposed multi-model "toolbox" approach [5] focused on random string detection in usernames, while language-independent different feature groups based on specific characteristics [40] demonstrated the effectiveness of a small number of expressive features was demonstrated. However, the performance of traditional machine learning methods is often limited due to its low-dimensional and low-order features. With the development of graph neural networks, deep learning technologies are applied to social bot detection tasks. BiLSTM [80] proposed the use of bidirectional long short-term memory recurrent neural network to effectively capture the characteristics of tweets. The recent BotRGCN [28] and Bot-AHGCN [101] used graph convolutional network training data to enhance the ability to capture social bots with multiple disguises. In addition, large language models have also been applied in bot detection. BGSRD [32] combined large-scale pre-training and transduction learning to symmetrically combine BERT and the graph convolution network GCN for joint training. However, the above data-driven models require a large number of labeled social user samples and even run the risk of weak generalization ability due to the lack of interpretability. Here the proposed work addresses this issue by remodeling the multi-relational social user graph and designing user community division and classification indicators based on structural entropy.

6.2 Network Embedding

The term "network embedding" mentioned here mainly refers to embedding methods that preserve the structure of a network (excluding node attributes and side information). It is an important tool for studying complex structural systems and processing network data. Common models for network embedding include methods based on random walks, matrix factorization, and deep neural networks [21]. Like Word2vec, random walk-based methods focus on learning the neighborhood structure by maximizing the probability of neighboring nodes during the random walk process. The latest Bot2vec [66] model designs a new network representation learning method specifically for social bot detection, which automatically preserves the neighbor relationship and community structure of users. Matrix factorization methods represent the network by learning the low-rank space of the adjacency matrix. Deep neural network methods, such as SDAE [8], SINE [78], and LIME [64] introduce nonlinear functions to learn highly complex and nonlinear networks. Network embedding provides important support for downstream tasks such as node classification, clustering, anomaly detection [35], and link prediction. In node classification [63, 65, 74], network embedding provides effective feature representations, thereby improving the effectiveness of node classification. It has been widely applied in fields such as social networks [36], citation

networks [61], and biological networks [31]. In anomaly detection, the advantage of network embedding lies in its ability to encode the structural information of the network into vector representations, thereby better capturing the relationships between nodes. Compared to traditional anomaly detection methods, network embedding can more accurately identify abnormal nodes and can handle large-scale network data. In addition, network embedding also has another important function, which is network visualization. Network embedding can generate meaningful node layouts by providing low-dimensional representations of nodes and displaying complex networks in two-dimensional space for further research and analysis.

6.3 Structural Information Theory based Applications

The Structural Information Theory decoding network's ability to capture the structure's essence has been validated in many applications. Introducing structural entropy in neural networks captures the underlying connectivity graph and reduces random interference [79]. The hierarchical nature of the structure entropy encoding tree provides new methods for hierarchical structure pooling in graph neural network [85], unsupervised image segmentation [94], dimension estimation [93], state abstraction [98] in reinforcement learning, ensemble of constraints in semi-supervised clustering [95], and unsupervised social event detection [9]. Additionally, reconstructing the graph structure on the hierarchical encoding tree suppresses edge noise and enhances the learning ability of the graph structure [103, 104]. Furthermore, modifying the network structure based on minimizing structural entropy achieves maximum deception of community structure [48]. Similarly, the anchor view, guided by the principle of minimizing structural entropy, improves the performance of graph contrastive learning [84].

7 CONCLUSION

This paper investigates a framework for supporting the unsupervised detection of social bots. The proposed UnDBot framework adaptively performs hierarchical community partitioning and identifies social bot communities during social bot detection. By modeling social user networks based on social behavior similarity, we effectively enhance the connections between social bots. The parallel execution of fusion operators during community partitioning maintains a stable community structure while ensuring high operational efficiency. Unsupervised detection is accomplished by employing a community binary classification that differentiates social bots from humans based on influence and cohesion. Experimental results demonstrate that UnDBot outperforms all unsupervised models in terms of accuracy, achieving a balance between operational efficiency and performance while exhibiting strong interpretability. Our work demonstrates the potential of unsupervised social bot detection and may open up new directions for research in social bot detection. In the future, our goal is to investigate how graph structure modeling with more behavioral and semantic features can improve the detection effectiveness of the model and expand UnDBot to other detection domains.

ACKNOWLEDGMENTS

This work is supported by the National Key R&D Program of China through grant 2022YFB3104700, NSFC through grants 62322202, 61932002, U21B2027, U23A20388, and 62266028, Beijing Natural Science Foundation through grant 4222030, Guangdong Basic and Applied Basic Research Foundation through grant 2023B1515120020, Shijiazhuang Science and Technology Plan Project through grant 231130459A, Foundation of State Key Laboratory of Public Big Data through grant PBD2022-04, Research Fund of Guangxi Key Lab of Multi-source Information Mining & Security (MIMS23-M-01), Yunnan Provincial Major Science and Technology Special Plan Projects through grants 202302AD080003, 202202AD080003 and 202303AP140008, General Projects of Basic Research in Yunnan Province through grant 202301AS070047, 202301AT070471, and the Fundamental Research Funds for the Central Universities. Philip S. Yu was partly supported by NSF under grant III-2106758, and POSE-2346158.

REFERENCES

[1] Seyed Ali Alhosseini, Raad Bin Tareaf, Pejman Najafi, and Christoph Meinel. 2019. Detect me if you can: Spam bot detection using inductive representation learning. In *Companion Proceedings of The 2019 World Wide Web Conference*. 148–153.

[2] Jon-Patrick Allem and Emilio Ferrara. 2018. Could Social Bots Pose a Threat to Public Health? *American journal of public health* 108 (08 2018), 1005–1006.

[3] Ahmed Anwar and Ussama Yaqub. 2020. Bot detection in twitter landscape using unsupervised learning. In *Proceedings of the 21st Annual International Conference on Digital Government Research*. 329–330.

[4] Efe Arin and Mucahid Kutlu. 2023. Deep learning based social bot detection on twitter. *IEEE Transactions on Information Forensics and Security* 18 (2023), 1763–1772.

[5] David M Beskow and Kathleen M Carley. 2019. Its all in a name: detecting and labeling bots by their name. *Computational and mathematical organization theory* 25, 1 (2019), 24–35.

[6] Adam Breuer, Roee Eilat, and Udi Weinsberg. 2020. Friend or faux: Graph-based early detection of fake accounts on social networks. In *Proceedings of the Web conference*. 1287–1297.

[7] Shaosheng Cao, Wei Lu, and Qiongkai Xu. 2015. Grarep: Learning graph representations with global structural information. In *Proceedings of the 24th ACM international on conference on information and knowledge management*. 891–900.

[8] Shaosheng Cao, Wei Lu, and Qiongkai Xu. 2016. Deep neural networks for learning graph representations. In *Proceedings of the AAAI conference on artificial intelligence*. 1145–1152.

[9] Yuwei Cao, Hao Peng, Zhengtao Yu, and Philip S. Yu. 2024. Hierarchical and Incremental Structural Entropy Minimization for Unsupervised Social Event Detection. In *Proceedings of the AAAI Conference on Artificial Intelligence*, Vol. 38. 8255–8264.

[10] Nikan Chavoshi, Hossein Hamooni, and Abdullah Mueen. 2016. Debot: Twitter bot detection via warped correlation.. In *Proceedings of the IEEE ICDM*, Vol. 18. 28–65.

[11] Hongxu Chen, Hongzhi Yin, Xue Li, Meng Wang, Weitong Chen, and Tong Chen. 2017. People opinion topic model: opinion based user clustering in social networks. In *Proceedings of the 26th International Conference on World Wide Web Companion*. 1353–1359.

[12] Wen Chen, Diogo Pacheco, Kai-Cheng Yang, and Filippo Menczer. 2021. Neutral bots probe political bias on social media. *Nature Communications* 12, 1 (2021), 1–10.

[13] Zhouhan Chen, Rima S Tanash, Richard Stoll, and Devika Subramanian. 2017. Hunting malicious bots on twitter: An unsupervised approach. In *Proceedings of the Social Informatics: 9th International Conference*. Springer, 501–510.

[14] Chun Cheng, Yun Luo, and Changbin Yu. 2020. Dynamic mechanism of social bots interfering with public opinion in network. *Physica A: statistical mechanics and its applications* 551 (2020), 1–8.

[15] Stefano Cresci. 2019. Detecting Malicious Social Bots: Story of a Never-Ending Clash. In *Proceedings of the Disinformation in Open Online Media: First Multidisciplinary International Symposium, MISDOOM 2019*. Springer-Verlag, 77–88.

[16] Stefano Cresci. 2020. A decade of social bot detection. *Commun. ACM* 63, 10 (2020), 72–83.

[17] Stefano Cresci, Roberto Di Pietro, Marinella Petrocchi, Angelo Spognardi, and Maurizio Tesconi. 2015. Fame for sale: Efficient detection of fake Twitter followers. *Decision Support Systems* 80 (2015), 56–71.

[18] Stefano Cresci, Roberto Di Pietro, Marinella Petrocchi, Angelo Spognardi, and Maurizio Tesconi. 2016. DNA-inspired online behavioral modeling and its application to spambot detection. *IEEE Intelligent Systems* 31, 5 (2016), 58–64.

[19] Stefano Cresci, Roberto Di Pietro, Marinella Petrocchi, Angelo Spognardi, and Maurizio Tesconi. 2017. The paradigm-shift of social spambots: Evidence, theories, and tools for the arms race. In *Proceedings of the 26th international conference on world wide web companion*. 963–972.

[20] Stefano Cresci, Roberto Di Pietro, Marinella Petrocchi, Angelo Spognardi, and Maurizio Tesconi. 2017. Social fingerprinting: detection of spambot groups through DNA-inspired behavioral modeling. *IEEE Transactions on Dependable and Secure Computing* 15, 4 (2017), 561–576.

[21] Peng Cui, Xiao Wang, Jian Pei, and Wenwu Zhu. 2018. A survey on network embedding. *IEEE Transactions on Knowledge and Data Engineering* 31, 5 (2018), 833–852.

[22] Ashkan Dehghan, Kinga Siuta, Agata Skorupka, Akshat Dubey, Andrei Betlen, David Miller, Wei Xu, Bogumił Kamiński, and Paweł Prałat. 2023. Detecting bots in social-networks using node and structural embeddings. *Journal of Big Data* 10, 1 (2023), 119.

[23] Zening Duan, Jianing Li, Josephine Lukito, Kai-Cheng Yang, Fan Chen, Dhavan V Shah, and Sijia Yang. 2022. Algorithmic Agents in the Hybrid Media System: Social Bots, Selective Amplification, and Partisan News about COVID-19. *Human Communication Research* (2022).

[24] Phillip George Efthimion, Scott Payne, and Nicholas Proferes. 2018. Supervised machine learning bot detection techniques to identify social twitter bots. *SMU Data Science Review* 1, 2 (2018), 1–70.

[25] Mohd Fazil and Muhammad Abulaish. 2017. Identifying active, reactive, and inactive targets of socialbots in twitter. In *Proceedings of the International Conference on Web Intelligence*. 573–580.

[26] Shangbin Feng, Zhaoxuan Tan, Rui Li, and Minnan Luo. 2022. Heterogeneity-aware twitter bot detection with relational graph transformers. In *Proceedings of the AAAI Conference on Artificial Intelligence*, Vol. 36. 3977–3985.
[27] Shangbin Feng, Herun Wan, Ningnan Wang, Jundong Li, and Minnan Luo. 2021. Satar: A self-supervised approach to twitter account representation learning and its application in bot detection. In *Proceedings of the 30th ACM International Conference on Information & Knowledge Management*. 3808–3817.
[28] Shangbin Feng, Herun Wan, Ningnan Wang, and Minnan Luo. 2021. BotRGCN: Twitter bot detection with relational graph convolutional networks. In *Proceedings of the 2021 IEEE/ACM International Conference on Advances in Social Networks Analysis and Mining*. 236–239.
[29] Emilio Ferrara, Onur Varol, Clayton Davis, Filippo Menczer, and Alessandro Flammini. 2016. The rise of social bots. *Commun. ACM* 59, 7 (2016), 96–104.
[30] Petr Gajdoš, Tomáš Ježowicz, Vojtěch Uher, and Pavel Dohnálek. 2016. A parallel Fruchterman–Reingold algorithm optimized for fast visualization of large graphs and swarms of data. *Swarm and evolutionary computation* 26 (2016), 56–63.
[31] Aditya Grover and Jure Leskovec. 2016. node2vec: Scalable feature learning for networks. In *Proceedings of the 22nd ACM SIGKDD international conference on Knowledge discovery and data mining*. 855–864.
[32] Qinglang Guo, Haiyong Xie, Yangyang Li, Wen Ma, and Chao Zhang. 2021. Social Bots Detection via Fusing BERT and Graph Convolutional Networks. *Symmetry* 14, 1 (2021), 1–14.
[33] Maryam Heidari and James H Jones. 2020. Using bert to extract topic-independent sentiment features for social media bot detection. In *Proceedings of the 2020 11th IEEE Annual Ubiquitous Computing, Electronics & Mobile Communication Conference (UEMCON)*. IEEE, 0542–0547.
[34] McKenzie Himelein-Wachowiak, Salvatore Giorgi, Amanda Devoto, Muhammad Rahman, Lyle Ungar, H Andrew Schwartz, David H Epstein, Lorenzo Leggio, Brenda Curtis, et al. 2021. Bots and misinformation spread on social media: Implications for COVID-19. *Journal of Medical Internet Research* 23, 5 (2021), e26933.
[35] Renjun Hu, Charu C Aggarwal, Shuai Ma, and Jinpeng Huai. 2016. An embedding approach to anomaly detection. In *Proceedings of the 2016 IEEE 32nd International Conference on Data Engineering (ICDE)*. IEEE, 385–396.
[36] Xiao Huang, Jundong Li, and Xia Hu. 2017. Label informed attributed network embedding. In *Proceedings of the tenth ACM international conference on web search and data mining*. 731–739.
[37] Jinyuan Jia, Binghui Wang, and Neil Zhenqiang Gong. 2017. Random walk based fake account detection in online social networks. In *2017 47th annual IEEE/IFIP international conference on dependable systems and networks (DSN)*. IEEE, 273–284.
[38] Mücahit Kantepe and Murat Can Ganiz. 2017. Preprocessing framework for Twitter bot detection. In *Proceedings of the 2017 International conference on computer science and engineering (ubmk)*. IEEE, 630–634.
[39] Tobias R Keller and Ulrike Klinger. 2019. Social bots in election campaigns: Theoretical, empirical, and methodological implications. *Political Communication* 36, 1 (2019), 171–189.
[40] Jürgen Knauth. 2019. Language-agnostic twitter-bot detection. In *Proceedings of the International Conference on Recent Advances in Natural Language Processing (RANLP 2019)*. 550–558.
[41] Nils Köbis, Jean-François Bonnefon, and Iyad Rahwan. 2021. Bad machines corrupt good morals. *Nature Human Behaviour* 5, 6 (2021), 679–685.
[42] Stephen G Kobourov. 2012. Spring embedders and force directed graph drawing algorithms. *arXiv preprint arXiv:1201.3011* (2012).
[43] Thai Le, Long Tran-Thanh, and Dongwon Lee. 2022. Socialbots on Fire: Modeling Adversarial Behaviors of Socialbots via Multi-Agent Hierarchical Reinforcement Learning. In *Proceedings of the ACM Web Conference*. 545–554.
[44] Angsheng Li, Jiankou Li, and Yicheng Pan. 2015. Discovering natural communities in networks. *Physica A: Statistical Mechanics and its Applications* 436 (2015), 878–896.
[45] Angsheng Li and Yicheng Pan. 2016. Structural information and dynamical complexity of networks. *IEEE Transactions on Information Theory* 62, 6 (2016), 3290–3339.
[46] Zhao Li, Biao Wang, Jiaming Huang, Yilun Jin, Zenghui Xu, Ji Zhang, and Jianliang Gao. 2024. A graph-powered large-scale fraud detection system. *International Journal of Machine Learning and Cybernetics* 15, 1 (2024), 115–128.
[47] Yu Liang, Siguang Li, Chungang Yan, Maozhen Li, and Changjun Jiang. 2021. Explaining the black-box model: A survey of local interpretation methods for deep neural networks. *Neurocomputing* 419 (2021), 168–182.
[48] Yiwei Liu, Jiamou Liu, Zijian Zhang, Liehuang Zhu, and Angsheng Li. 2019. REM: From structural entropy to community structure deception. *Proceedings of the Advances in Neural Information Processing Systems* 32 (2019), 1–11.
[49] Ziqi Liu, Chaochao Chen, Xinxing Yang, Jun Zhou, Xiaolong Li, and Le Song. 2018. Heterogeneous graph neural networks for malicious account detection. In *Proceedings of the 27th ACM international conference on information and knowledge management*. 2077–2085.
[50] Wai Weng Lo, Gayan Kulatilleke, Mohanad Sarhan, Siamak Layeghy, and Marius Portmann. 2023. XG-BoT: An explainable deep graph neural network for botnet detection and forensics. *Internet of Things* 22 (2023), 1–10.
[51] Kinga Makovi, Anahit Sargsyan, Wendi Li, Jean-François Bonnefon, and Talal Rahwan. 2023. Trust within human-machine collectives depends on the perceived consensus about cooperative norms. *Nature Communications* 14, 1 (2023), 1–12.

[52] Lorenzo Mannocci, Stefano Cresci, Anna Monreale, Athina Vakali, and Maurizio Tesconi. 2022. MulBot: Unsupervised Bot Detection Based on Multivariate Time Series. In *2022 IEEE International Conference on Big Data (Big Data)*. IEEE, 1485–1494.
[53] David Martín-Gutiérrez, Gustavo Hernández-Peñaloza, Alberto Belmonte Hernández, Alicia Lozano-Diez, and Federico Álvarez. 2021. A deep learning approach for robust detection of bots in twitter using transformers. *IEEE Access* 9 (2021), 54591–54601.
[54] Michele Mazza, Stefano Cresci, Marco Avvenuti, Walter Quattrociocchi, and Maurizio Tesconi. 2019. Rtbust Exploiting temporal patterns for botnet detection on twitter. In *Proceedings of the 10th ACM conference on web science*. 183–192.
[55] Marcelo Mendoza, Eliana Providel, Marcelo Santos, and Sebastián Valenzuela. 2024. Detection and impact estimation of social bots in the Chilean Twitter network. *Scientific Reports* 14, 1 (2024), 6525.
[56] Zachary Miller, Brian Dickinson, William Deitrick, Wei Hu, and Alex Hai Wang. 2014. Twitter spammer detection using data stream clustering. *Information Sciences* 260 (2014), 64–73.
[57] W James Murdoch, Chandan Singh, Karl Kumbier, Reza Abbasi-Asl, and Bin Yu. 2019. Definitions, methods, and applications in interpretable machine learning. *Proceedings of the National Academy of Sciences* 116, 44 (2019), 22071–22080.
[58] Lynnette Hui Xian Ng and Kathleen M Carley. 2023. Botbuster: Multi-platform bot detection using a mixture of experts. In *Proceedings of the International AAAI Conference on Web and Social Media*, Vol. 17. 686–697.
[59] Michaek Kwok-Po Ng, Xutao Li, and Yunming Ye. 2011. Multirank: co-ranking for objects and relations in multi-relational data. In *Proceedings of the 17th ACM SIGKDD international conference on Knowledge discovery and data mining*. 1217–1225.
[60] Mingdong Ou, Peng Cui, Jian Pei, Ziwei Zhang, and Wenwu Zhu. 2016. Asymmetric transitivity preserving graph embedding. In *Proceedings of the 22nd ACM SIGKDD international conference on Knowledge discovery and data mining*. 1105–1114.
[61] Shirui Pan, Jia Wu, Xingquan Zhu, Chengqi Zhang, and Yang Wang. 2016. Tri-party deep network representation. In *Proceedings of the Twenty-Fifth International Joint Conference on Artificial Intelligence*. 1895–1901.
[62] Javier Pastor-Galindo, Félix Gómez Mármol, and Gregorio Martínez Pérez. 2022. Profiling users and bots in Twitter through social media analysis. *Information Sciences* 613 (2022), 161–183.
[63] Hao Peng, Jianxin Li, Hao Yan, Qiran Gong, Senzhang Wang, Lin Liu, Lihong Wang, and Xiang Ren. 2020. Dynamic network embedding via incremental skip-gram with negative sampling. *Science China Information Sciences* 63 (2020), 1–19.
[64] Hao Peng, Renyu Yang, Zheng Wang, Jianxin Li, Lifang He, Yu Philip S., Albert Y Zomaya, and Rajiv Ranjan. 2021. Lime: Low-cost and incremental learning for dynamic heterogeneous information networks. *IEEE Trans. Comput.* 71, 3 (2021), 628–642.
[65] Bryan Perozzi, Rami Al-Rfou, and Steven Skiena. 2014. Deepwalk: Online learning of social representations. In *Proceedings of the 20th ACM SIGKDD international conference on Knowledge discovery and data mining*. 701–710.
[66] Phu Pham, Loan TT Nguyen, Bay Vo, and Unil Yun. 2022. Bot2Vec: A general approach of intra-community oriented representation learning for bot detection in different types of social networks. *Information Systems* 103 (2022), 101771.
[67] Rachel Pozzar, Marilyn J Hammer, Meghan Underhill-Blazey, Alexi A Wright, James A Tulsky, Fangxin Hong, Daniel A Gundersen, Donna L Berry, et al. 2020. Threats of bots and other bad actors to data quality following research participant recruitment through social media: cross-sectional questionnaire. *Journal of Medical Internet Research* 22, 10 (2020), e23021.
[68] Durgesh Samariya and Amit Thakkar. 2023. A comprehensive survey of anomaly detection algorithms. *Annals of Data Science* 10, 3 (2023), 829–850.
[69] Wojciech Samek, Thomas Wiegand, and Klaus-Robert Müller. 2017. Explainable artificial intelligence: Understanding, visualizing and interpreting deep learning models. *ITU Journal: ICT Discoveries* (2017), 1–10.
[70] Chengcheng Shao, Giovanni Luca Ciampaglia, Onur Varol, Kai-Cheng Yang, Alessandro Flammini, and Filippo Menczer. 2018. The spread of low-credibility content by social bots. *Nature communications* 9, 1 (2018), 1–9.
[71] Amit Singla and Mr Karambir. 2012. Comparative analysis & evaluation of euclidean distance function and manhattan distance function using k-means algorithm. *International Journal of Advanced Research in Computer Science and Software Engineering (IJARSSE)* 2, 7 (2012), 298–300.
[72] Massimo Stella, Emilio Ferrara, and Manlio De Domenico. 2018. Bots increase exposure to negative and inflammatory content in online social systems. *Proceedings of the National Academy of Sciences* 115, 49 (2018), 12435–12440.
[73] Victor Suarez-Lledo, Javier Alvarez-Galvez, et al. 2022. Assessing the Role of Social Bots During the COVID-19 Pandemic: Infodemic, Disagreement, and Criticism. *Journal of Medical Internet Research* 24, 8 (2022), e36085.
[74] Jian Tang, Meng Qu, Mingzhe Wang, Ming Zhang, Jun Yan, and Qiaozhu Mei. 2015. Line: Large-scale information network embedding. In *Proceedings of the 24th international conference on world wide web*. 1067–1077.
[75] Binghui Wang, Jinyuan Jia, Le Zhang, and Neil Zhenqiang Gong. 2018. Structure-based sybil detection in social networks via local rule-based propagation. *IEEE Transactions on Network Science and Engineering* 6, 3 (2018), 523–537.
[76] Daixin Wang, Peng Cui, and Wenwu Zhu. 2016. Structural deep network embedding. In *Proceedings of the 22nd ACM SIGKDD international conference on Knowledge discovery and data mining*. 1225–1234.
[77] Jianyu Wang, Rui Wen, Chunming Wu, Yu Huang, and Jian Xiong. 2019. Fdgars: Fraudster detection via graph convolutional networks in online app review system. In *Companion proceedings of the 2019 World Wide Web conference*. 310–316.

[78] Suhang Wang, Jiliang Tang, Charu Aggarwal, Yi Chang, and Huan Liu. 2017. Signed network embedding in social media. In *Proceedings of the 2017 SIAM international conference on data mining*. SIAM, 327–335.

[79] Yifei Wang, Yupan Wang, Zeyu Zhang, Song Yang, Kaiqi Zhao, and Jiamou Liu. 2023. User: Unsupervised structural entropy-based robust graph neural network. *Proceedings of the AAAI Conference on Artificial Intelligence*, 10235–10243.

[80] Feng Wei and Uyen Trang Nguyen. 2019. Twitter bot detection using bidirectional long short-term memory neural networks and word embeddings. In *2019 First IEEE International Conference on Trust, Privacy and Security in Intelligent Systems and Applications (TPS-ISA)*. IEEE, 101–109.

[81] Zixuan Weng and Aijun Lin. 2022. Public opinion manipulation on social media: Social network analysis of twitter bots during the covid-19 pandemic. *International journal of environmental research and public health* 19, 24 (2022), 1–17.

[82] Magdalena Wischnewski, Thao Ngo, Rebecca Bernemann, Martin Jansen, and Nicole Krämer. 2022. "I agree with you, bot!" How users (dis) engage with social bots on Twitter. *New Media & Society* (2022), 14614448211072307.

[83] Bin Wu, Le Liu, Yanqing Yang, Kangfeng Zheng, and Xiujuan Wang. 2020. Using improved conditional generative adversarial networks to detect social bots on Twitter. *IEEE Access* 8 (2020), 36664–36680.

[84] Junran Wu, Xueyuan Chen, Bowen Shi, Shangzhe Li, and Ke Xu. 2023. SEGA: Structural Entropy Guided Anchor View for Graph Contrastive Learning. In *Proceedings of the International Conference on Machine Learning*. PMLR, 1–20.

[85] Junran Wu, Xueyuan Chen, Ke Xu, and Shangzhe Li. 2022. Structural entropy guided graph hierarchical pooling. In *Proceedings of the International Conference on Machine Learning*. PMLR, 24017–24030.

[86] Jun Wu, Xuesong Ye, and Chengjie Mou. 2023. Botshape: A Novel Social Bots Detection Approach Via Behavioral Patterns. In *Proceedings of the 12th International Conference on Data Mining & Knowledge Management Process*. 45–60.

[87] Chao Yang, Robert Harkreader, and Guofei Gu. 2013. Empirical evaluation and new design for fighting evolving twitter spammers. *IEEE Transactions on Information Forensics and Security* 8, 8 (2013), 1280–1293.

[88] Jingkang Yang, Kaiyang Zhou, Yixuan Li, and Ziwei Liu. 2021. Generalized out-of-distribution detection: A survey. *arXiv preprint arXiv:2110.11334* (2021).

[89] Kai-Cheng Yang, Onur Varol, Clayton A Davis, Emilio Ferrara, Alessandro Flammini, and Filippo Menczer. 2019. Arming the public with artificial intelligence to counter social bots. *Human Behavior and Emerging Technologies* 1, 1 (2019), 48–61.

[90] Kai-Cheng Yang, Onur Varol, Pik-Mai Hui, and Filippo Menczer. 2020. Scalable and generalizable social bot detection through data selection. In *Proceedings of the AAAI conference on artificial intelligence*, Vol. 34. 1096–1103.

[91] Yingguang Yang, Renyu Yang, Yangyang Li, Kai Cui, Zhiqin Yang, Yue Wang, Jie Xu, and Haiyong Xie. 2023. RoSGAS: Adaptive Social Bot Detection with Reinforced Self-Supervised GNN Architecture Search. *ACM Transactions on the Web* 17, 3 (2023), 1–31.

[92] Yingguang Yang, Renyu Yang, Hao Peng, Yangyang Li, Tong Li, Yong Liao, and Pengyuan Zhou. 2023. FedACK: Federated Adversarial Contrastive Knowledge Distillation for Cross-Lingual and Cross-Model Social Bot Detection. In *Proceedings of the Web conference*. 1314–1323.

[93] Zhenyu Yang, Ge Zhang, Jia Wu, Jian Yang, Quan Z Sheng, Hao Peng, Angsheng Li, Shan Xue, and Jianlin Su. 2023. Minimum entropy principle guided graph neural networks. In *Proceedings of the Sixteenth ACM International Conference on Web Search and Data Mining*. 114–122.

[94] Guangjie Zeng, Hao Peng, Angsheng Li, Zhiwei Liu, Chunyang Liu, Yu Philip S., and Lifang He. 2023. Unsupervised Skin Lesion Segmentation via Structural Entropy Minimization on Multi-Scale Superpixel Graphs. In *Proceedings of the IEEE International Conference on Data Mining (ICDM)*. 768–777.

[95] Guangjie Zeng, Hao Peng, Angsheng Li, Zhiwei Liu, Runze Yang, Chunyang Liu, and Lifang He. 2024. Semi-Supervised Clustering via Structural Entropy with Different Constraints. In *Proceedings of the 2024 SIAM International Conference on Data Mining (SDM)*. SIAM, 208–216.

[96] Xianghua Zeng, Hao Peng, and Angsheng Li. 2023. Effective and stable role-based multi-agent collaboration by structural information principles. In *Proceedings of the Thirty-Seventh AAAI Conference on Artificial Intelligence*. AAAI Press, 11772–11780.

[97] Xianghua Zeng, Hao Peng, and Angsheng Li. 2024. Adversarial socialbots modeling based on structural information principles. In *Proceedings of the AAAI Conference on Artificial Intelligence*, Vol. 38. 392–400.

[98] Xianghua Zeng, Hao Peng, Angsheng Li, Chunyang Liu, Lifang He, and Philip S. Yu. 2023. Hierarchical State Abstraction based on Structural Information Principles. In *Proceedings of the Thirty-Second International Joint Conference on Artificial Intelligence, IJCAI-23*, Edith Elkind (Ed.). International Joint Conferences on Artificial Intelligence Organization, 4549–4557.

[99] Chaoning Zhang, Chenshuang Zhang, Sheng Zheng, Yu Qiao, Chenghao Li, Mengchun Zhang, Sumit Kumar Dam, Chu Myaet Thwal, Ye Lin Tun, Le Luang Huy, et al. 2023. A Complete Survey on Generative AI (AIGC): Is ChatGPT from GPT-4 to GPT-5 All You Need? *arXiv preprint arXiv:2303.11717* (2023).

[100] Ziwei Zhang, Peng Cui, and Wenwu Zhu. 2020. Deep learning on graphs: A survey. *IEEE Transactions on Knowledge and Data Engineering* 34, 1 (2020), 249–270.

[101] Jun Zhao, Xudong Liu, Qiben Yan, Bo Li, Minglai Shao, and Hao Peng. 2020. Multi-attributed heterogeneous graph convolutional network for bot detection. *Information Sciences* 537 (2020), 380–393.

[102] Ce Zhou, Qian Li, Chen Li, Jun Yu, Yixin Liu, Guangjing Wang, Kai Zhang, Cheng Ji, Qiben Yan, Lifang He, et al. 2023. A comprehensive survey on pretrained foundation models: A history from bert to chatgpt. *arXiv preprint arXiv:2302.09419* (2023).

[103] Dongcheng Zou, Hao Peng, Xiang Huang, Renyu Yang, Jianxin Li, Jia Wu, Chunyang Liu, and Philip S. Yu. 2023. SE-GSL: A General and Effective Graph Structure Learning Framework through Structural Entropy Optimization. In *Proceedings of the ACM Web Conference 2023*. 499–510.

[104] Dongcheng Zou, Senzhang Wang, Xuefeng Li, Hao Peng, Yuandong Wang, Chunyang Liu, Kehua Sheng, and Bo Zhang. 2024. Multispans: A multi-range spatial-temporal transformer network for traffic forecast via structural entropy optimization. In *Proceedings of the 17th ACM International Conference on Web Search and Data Mining*. 1032–1041.

Received August 21, 2023; Revised February 24, 2024; Accepted April 15, 2024