Social Bot Detection via Heterogeneous Graph Learning and Sample Balancing Strategies

Chenbo Fu[1,2,3(✉)], Jianquan Qiu[1,2,6], Feifan Xiong[1,2,6], Yuchen Xiong[1,2,3], Qifan Zhao[1,2,3], Shanqing Yu[1,2,3], Qi Xuan[1,2,3], and Yong Min[1,4,5]

[1] Binjiang Institute of Artificial Intelligence, Zhejiang University of Technology, Hangzhou 310056, China
cbfu@zjut.edu.cn
[2] Institute of Cyberspace Security, Zhejiang University of Technology, Hangzhou 310023, China
[3] College of Information Engineering, Zhejiang University of Technology, Hangzhou 310023, China
[4] Computational Communication Research Center, Beijing Normal University, Zhuhai 519087, China
[5] School of Journalism and Communication, Beijing Normal University, Beijing 100875, China
[6] College of Computer Science and Technology, Zhejiang University of Technology, Hangzhou 310023, China

Abstract. The influence of social media platforms has fundamentally transformed modern communication paradigms while introducing new security challenges. While these platforms offer users rich social connections and interactive experiences, malicious social bots are also covertly exploiting their unique features to engage in harmful activities, such as spreading misinformation, manipulating public opinion, and disrupting the digital ecosystem. However, conventional detection methods relying on static feature engineering or homogeneous network analysis prove inadequate against continuously evolving bot mimicry techniques and imbalanced dataset attacks. To address these limitations, this work proposes a hierarchical detection framework integrating heterogeneous graph representation learning with dynamic sample balancing strategies. By constructing multi-dimensional interaction graphs to model user behavior and fusing them with similarity-based affinity graphs, our approach enables graph neural networks to identify latent bot behavioral patterns effectively. Furthermore, the hybrid sampling mechanism rebalances sample distributions to mitigate data imbalance. Evaluations on real-world datasets verify the effectiveness of our proposed method. Our study not only advances methodological innovation in computational social science but also provides theoretical foundations for platform governance and digital policy-making.

Keywords: Social Bot Detection · Heterogeneous Graph Learning · Data Imbalance · Graph Neural Networks

X. Meng et al. (Eds.): BDSC 2025, CCIS 2622, pp. 217–234, 2026.
https://doi.org/10.1007/978-981-95-0880-8_18

1 Introduction

In the current digital era, social media platforms have transformed the landscape of information consumption by shattering the constraints of temporal and spatial, thus allowing people to communicate, interact, and access information with great convenience [1–5]. Yet, like shadows cast by sunlight, this deluge of information has brought with it turbulent undercurrents.

Malicious users manipulate social bots to engage in harmful activities, such as spreading misinformation, malicious marketing, and cyber violence [6,7].

For example, Alessandro et al. have studied Twitter data during the 2016 US presidential election and found that approximately 15% of the accounts were identified as social bots and approximately 19% of the tweets originated from them [8]. These figures demonstrate that social bots play a significant role in disrupting public debate and swaying public opinion. Furthermore, social bots also post provocative online comments, eliciting adverse feelings like anger and fear deep within the public, which leads to skepticism, distrust, and irrational behavior, facilitating the spreading of misinformation [9,10].

Traditional bot detection primarily relied on supervised machine learning for the analysis of individual accounts [11,12], i.e., given a set of accounts, detectors are applied to each account and assigned a binary label (bot or not bot). Current techniques for identifying social bots can generally be divided into three main categories: methods based on features, text and graph structures [13]. Feature-based methods manually extract user features, such as account creation date, number of followers, and other metadata [14–16]. These methods also use textual content, such as tweet content and reply [17]. However, social bots are becoming increasingly sophisticated, and they can bypass feature-based detection by creating deceptive accounts with fabricated metadata and copied tweets [18]. Text-based methods leverage natural language processing (NLP) techniques to identify social bots by analyzing their tweets and profile descriptions. For example, a BERT-based model has been used to analyze user-generated tweets for bot detection [19]. Similarly, Heidari et al. [17] utilized deep contextualized word embeddings, specifically the ELMO model, to create user profiles from tweet texts and improve bot detection accuracy. However, when advanced social bots plagiarize tweets and descriptions from real users, text-based methods can be easily misled [18]. Graph-based methods focus on detecting social bots by examining the graph structure of users' social connections. For example, Dehghan et al. [20] have employed node centrality metrics, and Magelinski et al. [21] have leveraged latent local graph features. Recent studies using Graph Neural Networks (GNNs) [22,23] construct heterogeneous graphs from interactions (e.g., follows, followers) and learn user representations through GNNs, achieving significant bot detection performance. However, as bot operators increasingly create complex botnets to manipulate network structures [18,24], methods relying solely on graph information face limitations in robustness, and these methods usually fail to consider the interaction between structural and attribute-based features.

In this study, we propose a bot detection framework based on sample balancing strategies and heterogeneous graphs, which integrates heterogeneous graph learning with sample balancing. Specifically, we first construct a multi-relational graph based on user interactions, such as follows and replies, to capture explicit social connections and latent behavioral correlations. Subsequently, we generate a similarity graph by establishing edges between nodes based on their pairwise similarity to strengthen connections among bot accounts with analogous behavioral features.

Finally, by combining follow graphs and similarity graphs into a heterogeneous graph, together with the reply graph as input, we feed it into a graph neural network for detection. Meanwhile, to mitigate the issue of imbalanced sample distribution, we implement a hybrid balancing strategy that integrates SMOTE [25] (Synthetic Minority Over-sampling Technique) for bot sample augmentation and TomekLinks [26] undersampling to remove ambiguous instances near the classification boundary between real users and social bots.

Our main contributions are summarized as follows:

- We propose to combine the follow relationship between users with the user similarity graph to construct an information-rich heterogeneous graph.
- We introduce a strategy that integrates both oversampling and undersampling techniques to enhance the model's robustness and accuracy by balancing the proportion of social bot and real user samples within the dataset.
- We evaluate the effectiveness of our proposed method using two publicly available datasets: Twibot-20 and Cresci-15.

The remainder of this paper is structured as follows: Sect. 2 reviews the related work, Sect. 3 elaborates on our proposed method, Sect. 4 provides detailed experimental procedures and results, and Sect. 5 summarizes our contributions and findings.

2 Related Work

2.1 Social Bot Detection Techniques

Traditional bot detection methods predominantly utilize feature-based methods, which involve manually extracting features from user metadata and behavioral interactions. For instance, Miller et al. [27] have utilized tweet content features combined with anomaly detection to identify social bots, and Beskow et al. [28] have used the metadata information on the user's profile for detection. However, such methods are vulnerable to evasion as social bots evolve to mimic real user behavior by manipulating metadata [18]. Text-based methods emerged to address this limitation by analyzing linguistic patterns using natural language processing. Methods like BERT [29] and RoBERTa [30] have been employed to detect social bots through semantic analysis of user-generated text. While effective, these methods struggle when social bots repurpose authentic content generated by real users [18]. Graph-based methods have recently gained prominence for their ability to model complex social relationships. Dehghan et

al. [20] have used node centrality metrics to identify social bots, Magelinski et al. [21] have leveraged local graph features and Kudugunta et al. [31] have used recurrent neural networks to capture the tweet features. Advanced frameworks like BotRGCN [32] and RGT [23] have employed heterogeneous graph neural networks to capture multi-relational interactions (e.g., follows, replies). Despite their success, these methods often overlook the interplay between structural and attribute-based features, limiting their robustness against sophisticated bot networks [18]. In addition, Zhang et al. [33] introduced a Transformer-based framework that enhances social graph representation by reconstructing masked node features and topological relationships. Yang et al. [34] generated robust training signals for graph contrastive learning by synthesizing counterfactual hard negative samples, addressing the data sparsity issue in social bot detection. Additionally, Sun et al. [35] leveraged adaptive structural learning to model high-order interactions in dynamic networks, offering new insights into real-time social bot detection. These works have provided new methods and ideas for social bot detection from different perspectives, further advancing the development of this field.

2.2 Data Imbalance in Bot Detection

Class imbalance, where real user accounts significantly outnumber bot samples, remains a critical challenge. Traditional oversampling techniques like SMOTE [25] and undersampling methods such as TomekLinks [26] have been applied to balance datasets. However, SMOTE may introduce synthetic noise near class boundaries, while TomekLinks risks removing informative majority samples. Recent studies [36] highlight the need for hybrid strategies that combine oversampling and undersampling to preserve data integrity while enhancing model generalization.

2.3 Graph Neural Networks for Social Network Analysis

GNNs analyze social networks by encoding topological and attribute information. Graph Convolutional Networks (GCNs) [37] aggregate neighborhood features through message passing, while Graph Attention Networks (GATs) [38] dynamically weight neighbor contributions via attention mechanisms. For large-scale graphs, GraphSAGE [39] employs neighbor sampling to improve scalability. Heterogeneous GNNs [40] further extend these frameworks to multi-relational graphs, enabling nuanced modeling of user interactions. Despite these advancements, existing graph-based bot detection models often fail to integrate auxiliary signals (e.g., user similarity) or address data imbalance, leading to suboptimal performance in real-world scenarios.

3 Method

The architecture of our model is shown in Fig. 1, which integrates sample balancing strategies and heterogeneous graph learning through four interconnected

components: the sample balancing module, the heterogeneous graph construction module, the feature extraction module and the classifier. The sample balancing module mitigates class imbalance by dynamically adjusting the ratio of bot and real user samples in the training dataset. The heterogeneous graph construction module integrates user relationships, such as follow and reply, and combines them with the user similarity graph to construct the heterogeneous graph, presenting user associations. The feature extraction module employs a hierarchical Graph Neural Network (GNN) to aggregate structural and semantic features from the heterogeneous graph, emphasizing discriminative patterns that differentiate social bots from real users. Finally, the fully-connected neural network is employed as the classifier to process these refined features and compute the prediction results.

3.1 Problem Formulation

The social bot detection framework is formalized as a graph-structured dataset $\mathcal{G} = (\mathcal{V}, \mathcal{E}, \mathbf{X}, \mathcal{Y})$, where $\mathcal{V} = \{v_1, v_2, ..., v_N\}$ denotes the set of N user nodes with $\mathbf{X} \in \mathbb{R}^{N \times D}$ representing their multidimensional feature matrix encompassing profile attributes (username, registration date, geolocation), textual content (tweet semantics, hashtag usage patterns), metadata (verification status, follower-to-following ratios), and temporal activity patterns (posting frequency, login regularity). The edge set $\mathcal{E} \subseteq \mathcal{V} \times \mathcal{V}$ captures directed social relationships between users, including follow interactions, comment exchanges, and retweet behaviors, where each edge $e_{ij} = (v_i, v_j)$ denotes a specific interaction type between user v_i and v_j. The label set $\mathcal{Y} = \{y_1, y_2, ..., y_N\}$ contains binary annotations with $y_i \in \{0, 1\}$ distinguishing real user accounts ($\mathcal{C}_0$) from bot profiles ($\mathcal{C}_1$). The classification task aims to learn a predictive function $f_\theta : \mathcal{G} \rightarrow \{0, 1\}^M$ that generates probabilistic labels $\hat{y}_u$ for target users.

3.2 User Node Representation

In social bot detection, textual features are key sources of information. Our study leverages pre-trained language models to extract semantic information from user-posted tweets and profile descriptions. We employ the pre-trained RoBERTa [30] as the encoder for the text module to encode a user's sequence of textual content $\{w_1^i, w_Q^i\}_{i=0}^K$. Text embeddings are derived through an average pooling operation:

$$\{\hat{t}_0, \ldots, \hat{t}_K\} = \text{AvgPool}\left(\text{LM}\left(\{w_1^i, w_Q^i\}_{i=0}^K\right)\right) \tag{1}$$

where AvgPool$(\cdot)$ denotes the average pooling layer, and $\hat{t}_i$ represents the textual feature vector for the i-th post. These embeddings are further averaged to form a comprehensive user semantic representation: $\hat{T}_i = \text{AvgPool}(\{\hat{t}_0, \ldots, \hat{t}_K\})$.

The goal of node representation is to transform users' information into effective feature vectors. First, semantic information is captured by encoding textual content using a pre-trained language model. The embeddings are consolidated

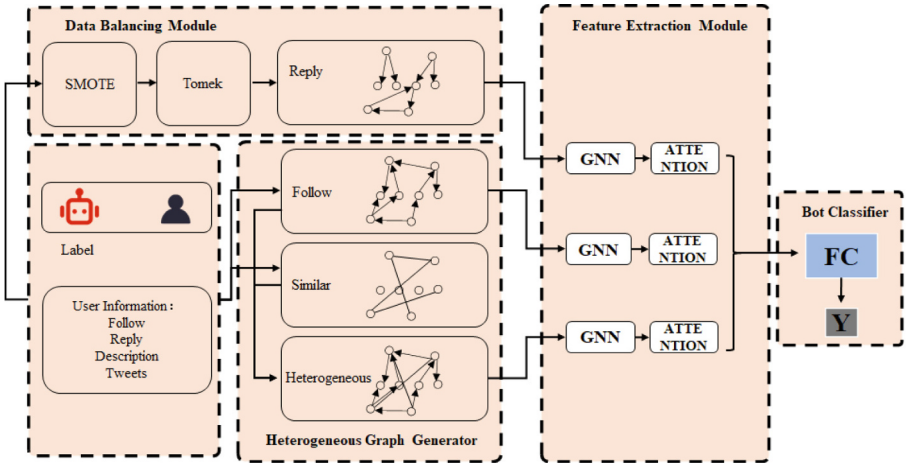

Fig. 1. Architecture of the bot detection model based on data-balancing strategy and heterogeneous graph.

via average pooling to generate a unified semantic representation $\hat{T}_i$, followed by parameterization: $\hat{T}_i = \sigma(W_s\hat{T}_i)$. User attribute information, including numerical data and Boolean data, is extracted from metadata. These attributes undergo separate parameterization processes, such as $\hat{V}_i = \sigma(W_v V_i)$ and $\hat{B}_i = \sigma(W_b B_i)$, to better integrate attribute information into node representations.

Topological information implies the user's connectivity in the social network. In our work, topological information is incorporated into node representations by constructing multi-relational and similarity graphs. In multi-relational graphs, neighbor node features are aggregated through graph convolution operations, enabling nodes to learn features from surrounding nodes. Similarity graphs establish additional connections based on node similarities. Finally, the processed semantic information $\hat{Y}_i$, attribute information $\hat{V}_i$, and $\hat{B}_i$ are concatenated to form the final node representation:

$$u_i = \text{Concat}\left([\hat{V}_i; \hat{B}_i; \hat{T}_i]\right)_v \tag{2}$$

3.3 Data Balancing

Imbalanced class distributions in social bot detection datasets often degrade the model's overall performance and generalization ability. To address this issue, our study proposes a hybrid sampling strategy that combines oversampling and undersampling strategies. Our approach balances the quantity of bot samples (minority class) and real user samples (majority class). Specifically, after extracting user features, oversampling synthesizes additional minority-class samples using interpolation, while undersampling selectively removes redundant majority-class samples near decision boundaries.

Oversampling Strategy: The Synthetic Minority Oversampling Technique (SMOTE) [25] is employed to augment minority-class samples, ensuring the model captures nuanced patterns of bot behavior. The process begins with metadata normalization, which standardizes heterogeneous features (i.e., follower count, user description) into comparable numerical ranges. For a given metadata feature x, normalization is performed as:

$$x_{\text{norm}} = \frac{x - x_{\min}}{x_{\max} - x_{\min}} \tag{3}$$

where $x_{\min}$ and $x_{\max}$ denote the minimum and maximum values of the feature. Normalized metadata vectors enable consistent distance calculations across features. Next, minority-class samples (bots) are identified. For each minority sample x_a, the k-nearest neighbors ($k = 3$) within the majority class (real users) are computed using the Euclidean distance metric:

$$d(i,j) = \sqrt{\sum_{m=1}^{n} (x_{im} - x_{jm})^2} \tag{4}$$

Here, $d(i,j)$ represents the distance between samples im and jn, where x_{im} and x_{jm} are the m-th feature values of sample i and sample j, and n is the dimensionality of the features. For the current minority class sample, we first randomly select a majority class sample (real user sample) from 3 nearest neighbors. Then, by performing random interpolation between the selected nearest neighbor and the current sample, we generate a new minority class sample. Suppose the current minority class sample is x_a, and the selected nearest neighbor sample is x_b, the formula for generating the new sample x_k is:

$$x_k = (1 - \delta) \cdot x_a + \delta \cdot x_b \tag{5}$$

where $\delta \in [0, 1]$ is a random interpolation coefficient. This process enriches the minority class with synthetic samples that preserve the underlying feature distribution, directly enhancing training data diversity without reconstructing raw user data.

Undersampling Strategy: The TomekLinks [26] is employed as undersampling strategy in our approach. To counteract overfitting to majority-class patterns, undersampling removes ambiguous majority samples near class boundaries. After metadata normalization, Tomek pairs–defined as pairs of minority and majority samples with minimal inter-class Euclidean distance–are identified. Specifically, for a minority sample i and majority sample j, (i,j) constitutes a Tomek pair if:

$$d(i,j) = \min_{k \in \text{Majority}} d(i,k) \tag{6}$$

$$d(i,j) = \min_{i \in \text{Minority}} d(i,j) \tag{7}$$

Here, majority and minority denote the majority and minority sample sets, respectively. The majority of samples involved in Tomek pairs are removed,

as they often reside in regions of overlapping class distributions. This pruning sharpens decision boundaries and reduces noise, enabling the model to focus on discriminative features.

The synergy of oversampling and undersampling addresses both class imbalance and boundary ambiguity. Oversampling mitigates the underrepresentation of social bots by generating synthetic samples, while undersampling eliminates confusing real user samples near bot clusters. This dual approach optimizes the dataset's class distribution, ensuring balanced and high-quality training data.

3.4 Heterogeneous Graph Construction

Social networks facilitate diverse user interactions, including following, retweeting, and commenting. The fundamental challenge in constructing multi-relational graphs involves effectively aggregating these heterogeneous node relationships and representing them through a topological structure.

Consider users as nodes, represented by the set $V = \{v_1, v_2, ..., v_n\}$, where n denotes the total number of user nodes. Each node possesses unique attributes and features, collectively forming the node's feature set X. The structural information between users–namely, their diverse interactions–is represented as edges in the graph. Different types of relationships are distinguished by subscripts, $r \in \{0, 1, ..., R\}$, where R represents the total number of relationship types. For instance, $r = 0$ denote a "follow" relationship, $r = 1$ a "comment" relationship, and so on. Thus, the edge set can be expressed as $E_r|_{r=0}^{R}$, where E_r denotes the set of edges corresponding to a specific relationship. In our work, we only consider two relationships, i.e., reply and follow relationships. Reply relationship means one user comments on another user, and follow relationship represents the behavior of one user choosing to follow another.

To explore the similarity between nodes, we consider both the structural features of nodes in the follow-relationship graph and their intrinsic attribute features. For each node v_i, its follow list and follower list are treated as part of its structural features, alongside the previously obtained user node feature vector u_i. The cosine similarity is employed to measure the similarity between two nodes. For nodes v_i and v_j, their similarity sim_{ij} is calculated as:

$$\text{sim}_{ij} = \frac{u_i^T u_j}{\|u_i\| \cdot \|u_j\|} \tag{8}$$

Here, $u_i^T u_j$ is the dot product of the two feature vectors, reflecting their similarity in the feature space, while $\|u_i\|$ and $\|u_j\|$ are the L2 norms of vectors u_i and u_j, respectively, ensuring the similarity value is normalized within the range $[-1, 1]$.

Furthermore, to account for the influence of follow relationships, we introduce a follow-structural similarity $\text{sim}_{s_{ij}}$. Assuming node v_i's follow set is F_i, its follower set is B_i, node v_j's follow set is F_j, and its follower set is B_j, the follow-structural similarity is defined as:

$$\text{sim}_{s_{ij}} = \frac{|F_i \cap F_j| + |B_i \cap B_j|}{|F_i \cup F_j| + |B_i \cup B_j|} \tag{9}$$

where $|\cdot|$ denotes the cardinality of a set. By combining both feature similarity and follow-structural similarity, we obtain the final node similarity measure $\text{sim}_{\text{final}_{ij}}$:

$$\text{sim}_{\text{final}_{ij}} = \alpha \times \text{sim}_{ij} + (1 - \alpha) \times \text{sim}_{\text{s}_{ij}} \tag{10}$$

where $\alpha \in [0, 1]$ serves as a weighting coefficient that balances the importance between feature similarity and follow-structural similarity. The value of α can be adjusted according to specific dataset characteristics and task requirements to achieve flexible weight allocation.

After computing the pairwise node similarities, for each node v_i, we select its top-k most similar neighbors $\{v_{i1}, v_{i2}, \cdots, v_{ik}\}$ based on $\text{sim}_{\text{final}_{ij}}$. Here, k represents a hyperparameter that requires tuning according to the specific dataset and task. Using these selected similar node pairs, we construct the similarity-based edges E_s. The resulting similarity graph is defined as $G_s = \{V, E_s\}$, where V denotes the node set (identical to that in the follow-relationship graph) and E_s represents the edge set built upon similarity measures.

To emphasize the greater importance of relationship graphs compared to similarity graphs, we apply a weight reduction to similarity graph edges. Given initial edge weights w^s_{ij}, the reduced weights $w^s_{\text{new}_{ij}}$ are computed as:

$$w^s_{\text{new}_{ij}} = \beta \times w^s_{ij} \tag{11}$$

where $\beta \in (0, 1)$ serves as the edge weight reduction coefficient. By adjusting β, we can flexibly control the degree of weight reduction for the graph's edges. During construction, we regard the edges in the similarity graph as a novel type of special relationship. Based on our assumption that homophilic connections–edges linking nodes of the same type–display similar node attributes, all edges in the similarity graph are treated as homophilic connections.

The node similarity graph effectively captures similarity relationships between different nodes. Connecting similar nodes enables more accurate modeling of complex relationships in the graph. For instance, in social networks, even if two users lack direct connections in the follow-relationship graph, they will be linked in the similarity graph when their feature similarity and follow-structural similarity are sufficiently high. The construction of the similarity graph in this work helps discover social robots hidden in real user communities, and also helps to identify potential neighboring nodes, boosting the representational capacity of neural networks. Graph neural networks can leverage these similar connections to better learn node features and improve social bot detection capability.

To fully exploit the advantages of multi-relational and similarity graphs, we integrate connections from the follow-relationship and similarity graphs. While the follow-relationship graph contains rich, authentic social connection information, the node similarity graph focuses on capturing node-to-node similarities. Their combination yields more comprehensive information:

$$E_c|_{r=0}^{|R+1|} = E_r|_{r=0}^{|R|} \cup E_s \tag{12}$$

where $E_c|_{r=0}^{|R+1|}$ represents the combined edge set in the heterogeneous graph, and $E_r|_{r=0}^{|R|}$ denotes the edge set from the follow-relationship graph. In Eq. 12, $|R|$ indicates the number of relationships in the follow relationship graph. During the combination process, it is important to note that some overlapping connections may exist between the two graphs. For these duplicate connections, since their relationship types might differ between the follow relationship graph and the similarity graph, we remove such connections from the node similarity graph while retaining those in the follow relationship graph. This approach preserves the structural integrity of the follow relationship graph while incorporating novel connections introduced by the node similarity graph, thereby enriching the overall graph information. Furthermore, by reducing the edge weights of homogeneous connections, we ensure the dominant role of follow relationships in the global information fusion, providing stronger support for subsequent social bot detection tasks. By using this integration strategy, the structural information from the follow relationship graph and the information of similar nodes in the similarity graph are effectively combined, resulting in a more comprehensive and valuable heterogeneous graph for the model. This further enhances the accuracy and reliability of social bot detection.

3.5 Feature Extraction

To more comprehensively exploit the information embedded in heterogeneous graphs, we employ multiple graph neural networks to train graphs and aggregate the training results to obtain the final features for users. For each specific relation graph G_r, we utilize a graph neural network-based embedding layer to derive node representations. Specifically, we compute the first-order neighbor set N_r^i of node v_i in relation graph G_r, and take the mean of these neighbors' representations from the $(l-1)$-th GNN layer, $z_r^{(l-1)}(v_j)$, as follows:

$$z_r^{(l)}(N_r^i) = \text{mean}\left(\left[z_r^{(l-1)}(v_j), \forall v_j \in \{v_i\} \cup N_r^i\right]\right) \tag{13}$$

where N_r^i denotes the first-order neighbor set of v_i in relation graph G_r. Subsequently, using the node features $\tilde{X}_r$ of G_r as initial representations, the representation of node v_i in the l-th GNN layer is computed via:

$$z_r^{(l)}(v_i) = \sigma\left(W_r^{(l)} \cdot z_r^{(l)}(N_i^r) + b_r^{(l)}\right) \tag{14}$$

Here, $W_r^{(l)}$ and $b_r^{(l)}$ are learnable parameters, and $\sigma(\cdot)$ represents the activation function, for which we adopt the tanh function in this work. And the final representation of node v_i in the GNN-based embedding layer is denoted as $z_r(v_i)$.

Given that different relations may contribute unequally to user node representations across multiple GNN-based embedding layers, we introduce a semantic attention layer to fuse the representations of each user node under different relations. First, we define a learnable relation preference vector $a_r \in \mathbb{R}^{1\times d}$ for each relation. The contribution weight of the representation $z_r(v_i)$ under a specific relation to the final representation depends on the similarity between $z_r(v_i)$

and a_r. To compute this weight, we first transform the d-dimensional $z_r(v_i)$ into $h_r(v_i)$:

$$h_r(v_i) = \sigma(W'_r \cdot z_r(v_i) + b'_r) \tag{15}$$

where $\sigma(\cdot)$ is the nonlinear activation function (tanh function). Subsequently, we compute the similarity $w_r(v_i)$ between a_r and $h_r(v_i)$:

$$w_r(v_i) = \frac{a_r^T \cdot h_r(v_i)}{\|a_r\| \cdot \|h_r(v_i)\|} \tag{16}$$

where $\|\cdot\|$ denotes the L2-norm of a vector. Then, we normalize the weights assigned to node v_i for relation r using the softmax function:

$$\alpha_r(v_i) = \frac{\exp(w_r(v_i))}{\sum_{r' \in R} \exp(w_{r'}(v_i))} \tag{17}$$

Finally, we aggregate the representations of node v_i across all relations:

$$z(v_i) = \sum_{r \in R} \alpha_r(v_i) z_r(v_i) \tag{18}$$

Through the heterogeneous graph embedding layer and semantic attention layer, comprehensive representations are obtained for each user node.

3.6 Bot Classifier

After completing the user representation learning module, we obtain the final representation z_v for each user node v. In this work, we employ a fully connected neural network as the bot classifier. The final representation z_v of a user node is fed into this network to compute the prediction result. The detailed process is as follows.

$$\hat{y}_v = \mathrm{FC}(z_v) \tag{19}$$

Here, the predicted label $\hat{y}_v$ for node v takes a value of 0 for real users and 1 for bots, and the FC represents fully connected neural networks. The cross-entropy loss function is used to measure the discrepancy between predictions and ground truth labels. Assuming there are N user node samples in the training set, with $y_{v_i} \in \{0, 1\}$ being the ground truth label for node v_i, the cross-entropy loss L is defined as:

$$L = -\frac{1}{N} \sum_{i=1}^{N} [y_{v_i} \log(\hat{y}_{v_i}) + (1 - y_{v_i}) \log(1 - \hat{y}_{v_i})] \tag{20}$$

4 Experimental Results and Analysis

4.1 Datasets

Our method utilizes two publicly available datasets, i.e., Twibot-20 [13] and Cresci-15 [41], to assess the performance of the proposed bot detection model as well as the baseline models. The statistical details of these datasets are presented in Table 1.

Table 1. Statistics of the datasets

Statistics	Twibot-20	Cresci-15
Number of users	11,826	5,301
Number of edges	15,434	14,220
Real user samples	5,237	1,950
Bot samples	6,589	3,351

The Twibot-20 dataset comprises 5,237 real user accounts and 6,589 bot accounts with their metadata, where all accounts can be categorized into four domains: politics, business, entertainment, and sports. The Cresci-15 dataset contains 5,301 Twitter users and their network structures. We maintain consistent data partitioning after the original dataset splits for comparison with previous models. Both datasets include various relationships, such as follow and retweet relations, which facilitate the construction of multi-relational graphs.

4.2 Baseline Models

We compare our proposed model with seven baseline models. Brief descriptions of these baselines are as follows:

(1) **Botometer** [42]: It incorporates over 1,000 features derived from user metadata, content, and interaction patterns. The model is trained using a compilation of existing datasets.
(2) **RoBERTa** [30]: A text-based detection approach that analyzes bot-generated comments and other textual content.
(3) **BotRGCN** [32]: It builds heterogeneous graphs from layered networks and uses relational graph convolutional networks for learning user representations and detecting social bots.
(4) **BotBuster** [43]: A social bot detection system that leverages mixture-of-experts models to analyze user metadata and text data, thereby improving cross-platform detection performance.
(5) **RGT** [44]: This model captures the inherent heterogeneity in stacked spaces to enhance social bot detection.

(6) **SEBOT** [45]: A multi-view based social bot detection model that employs multi-view contrastive learning and multi-task learning to enhance detection performance.
(7) **BotMoE** [36]: This approach integrates multiple information modalities such as user metadata, textual content, and network structure. It employs community-aware mixture-of-experts layers to automatically classify users into different communities and merge multimodal representations.

The detailed hyperparameter configurations are presented in Table 2. Our model utilizes the Adam optimizer [23], with a learning rate of 10^{-4} and a hidden layer dimension of 256. For the remaining baseline models, we retain their default parameter settings.

Table 2. Hyperparameter settings

Parameter	TwiBot-20	Cresci-15
Optimizer	Adam	Adam
Learning rate	10^{-4}	10^{-4}
L2 regularization	10^{-6}	10^{-6}
BL coefficient	10^{-2}	10^{-2}
Dropout rate	0.3	0.3
Hidden layer size	256	256
Max training epochs	400	400
Transformer attention heads	2	2

We use Accuracy and F1-score (F_1) as evaluation metrics to measure model performance. The calculations for these metrics are as follows:

$$\text{Accuracy} = \frac{TP + TN}{TP + TN + FN + FP} \tag{21}$$

$$F_1 = \frac{2 \times \text{Precision} \times \text{Recall}}{\text{Precision} + \text{Recall}} \tag{22}$$

In bot detection evaluation, four key metrics are crucial: True Positives (TP) are real users correctly identified as such; False Positives (FP) are real users incorrectly classified as bots; False Negatives (FN) are bots incorrectly classified as real users; and True Negatives (TN) are bots correctly identified. Accuracy measures the ratio of correctly predicted samples to the total number of samples, providing an intuitive measure of the model's overall prediction accuracy. The F1-score, which is the weighted harmonic mean of Precision and Recall, is crucial for evaluating model performance.

Table 3. Bot detection performance comparison

Model	TwiBot-20		Cresci-15	
	Accuracy	F_1	Accuracy	F_1
Botometer	53.10	53.10	60.10	66.90
RoBERTa	75.45	73.21	96.80	95.86
BotRGCN	85.70	87.25	96.50	97.20
BotBuster	78.24	81.18	95.80	96.63
RGT	86.60	88.01	97.20	97.78
SEBOT	86.24	87.61	98.46	98.71
BotMoE	86.76	**88.22**	98.50	98.82
Ours	**86.89**	87.98	**98.58**	**98.85**

4.3 Results

Table 3 presents the experimental results of all models on the TwiBot-20 and Cresci-15 datasets, with the best results highlighted in bold.

Compared with baseline models, our proposed model achieves 86.89% accuracy and 87.98% F1-score on the TwiBot-20 dataset, outperforming Botometer, RoBERTa, and BotBuster. While showing comparable performance with RGT, SEBOT and BotMoE in accuracy, it slightly trails BotMoE and RGT in F1-score. On the Cresci-15 dataset, our model attains 98.58% accuracy and 98.85% F1-score, achieving state-of-the-art performance. This indicates superior capability in both prediction accuracy and precision-recall balance. Graph-based models (BotRGCN, RGT, SEBOT, BotMoE) generally perform well on both datasets, confirming the effectiveness of graph-structured approaches for relational pattern mining in bot detection. The results on different datasets demonstrate the effectiveness of our approach on social media.

4.4 Ablation Study

To comprehensively examine the influence of the critical components in our proposed model, we conducted ablation experiments by constructing four model variants. The experimental results are shown in Table 4:

(1) **Ours w/o Follow:** This variant removes the follow graph entirely to examine how explicit social connections affect detection performance. The follow graph represents direct user-to-user attention relationships, which may reveal bot coordination patterns.
(2) **Ours w/o Reply:** This version eliminates the reply graph to assess the importance of interaction patterns. Reply relationships capture conversational dynamics that often differ between bots and real users.
(3) **Ours w/o Similar:** This configuration discards the similarity graph to evaluate the contribution of latent feature relationships. The similarity graph connects users with comparable behavioral or content patterns.

(4) **Ours w/o Balance:** This variant disables all data balancing operations during training to quantify the impact of our hybrid (SMOTE + TomekLinks) balancing strategy on handling class imbalance.

Table 4. Ablation Study Results

Dataset	Metric	Ours w/o Follow	Ours w/o Reply	Ours w/o Similar	Ours w/o Balance	Ours
TwiBot-20	Accuracy	85.23	84.56	84.47	85.68	**86.89**
	F_1	86.15	85.72	85.38	87.21	**87.98**
Cresci-15	Accuracy	97.37	96.59	96.62	97.25	**98.58**
	F_1	97.63	97.56	96.78	97.67	**98.85**

The experimental results demonstrate that each component contributes significantly to the model's performance. The follow graph shows the strongest individual impact (1.66% accuracy drop on Cresci-15 when removed), while the similarity graph proves most crucial for F1-score (2.07% decrease). Our hybrid balancing strategy contributes 0.9–1.3% improvements across all metrics compared to unbalanced training.

To further investigate the importance of data sample balancing, we conducted experiments to evaluate the impact of different balancing methods on bot detection performance. We constructed two variants of the comparative model:

(1) **Ours w/o SMOTE:** Removes SMOTE oversampling to assess its individual contribution
(2) **Ours w/o Tomek:** Eliminates TomekLinks undersampling to analyze its specific effects

Table 5. Data balancing experiment results

Dataset	Metric	Model Variants			
		w/o SMOTE	w/o Tomek	w/o Balance	Full Model
TwiBot-20	Accuracy	86.12	85.90	85.68	**86.89**
	F_1	87.35	87.10	87.21	**87.98**
Cresci-15	Accuracy	97.80	97.62	97.25	**98.58**
	F_1	98.20	98.05	97.67	**98.85**

The comparative results presented in Table 5 reveal that while both SMOTE and TomekLinks undersampling individually improve model performance, neither single method achieves the detection accuracy of the complete hybrid model. This empirical evidence confirms that: (1) data class imbalance significantly impacts bot detection effectiveness, and (2) a strategic combination of complementary balancing techniques produces superior performance compared to any individual approach.

5 Conclusion

In this work, we proposed a bot detection model based on the sample balancing strategy and the heterogeneous graph. Experiments on real datasets such as TwiBot-20 and Cresci-15 show that the proposed method performs better in evaluation indicators and can detect bots more accurately than baseline models. The ablation experiments verify the importance of each module of our proposed method. In addition, the ablation experiment on the data balancing strategy shows that although the use of the SMOTE or TomekLinks method alone could improve the model performance, the model effect is best when multiple sample balancing strategies are used in combination. In summary, our proposed method not only expands existing social bot detection methods but also emphasizes the important roles of heterogeneous graphs and sample balancing strategies in identifying social bots. In the future, we may explore newer methods, such as graph prompt learning [46], and exploring more efficient sample balancing strategies may improve the model's detection performance.

Acknowledgements. This work was supported by the Zhejiang Provincial Natural Science Foundation of China under Grant (ZCLY24F0302), the Zhejiang Fundamental Public Welfare Research Project under Grant (LGF21G010003), the National Natural Science Foundation of China under Grant (62103374 and U21B2001), the Key R&D Program of Zhejiang under Grant (2022C01018), and the Key Research and Development Program of Zhejiang Province under Grant (2024C01025), the Baima Lake Laboratory Joint Fund of Zhejiang Provincial Natural Science Foundation of China under Grant(LBMHZ25F020002), the Fundamental Research Funds for the Central Universities, China under Grant (1243200012), and 2023 General Project of Ministry of Education Foundation on Humanities and Social Sciences under Grant (23YJA860011).

References

1. Lazer, D., et al.: Meaningful measures of human society in the twenty-first century. Nature **595**(7866), 189–196 (2021)
2. Chenbo, F., et al.: A novel spatiotemporal behavior-enabled random walk strategy on online social platforms. IEEE Trans. Comput. Soc. Syst. **9**(3), 807–817 (2021)
3. Chenbo, F., Luo, H., Liang, X., Min, Y., Xuan, Q., Chen, G.: Mutual influence in citation and cooperation patterns. IEEE Trans. Comput. Soc. Syst. **11**(3), 3851–3861 (2023)
4. Chenbo, F., Yue, X., Shen, B., Shanqing, Yu., Min, Y.: Patterns of interest change in stack overflow. Sci. Rep. **12**(1), 11466 (2022)
5. Zhang, C., et al.: Polarization in social media. Phys. A Stat. Mech. Appl., 130487 (2025)
6. Fu, C., Chen, K., Pan, X., Yu, S., Ni, J., Min, Y.: Rumor detection based on the temporal sentiment. In: China National Conference on Big Data and Social Computing, pp. 275–290. Springer (2022)
7. Fu, C., et al.: Rumor detection based on conflict and bot features. In: China National Conference on Big Data and Social Computing, pp. 279–297. Springer (2024)

8. Bessi, A., Ferrara, E.: Social bots distort the 2016 US presidential election online discussion. First Monday **21**(11-7) (2016)
9. Shao, C., Ciampaglia, G.L., Varol, O., Yang, K.-C., Flammini, A., Menczer, F.: The spread of low-credibility content by social bots. Nat. Commun. **9**(1), 4787 (2018)
10. Subrahmanian, V.S., et al.: The DARPA Twitter bot challenge. Computer **49**(6), 38–46 (2016)
11. Lee, K., Caverlee, J., Webb, S.: Uncovering social spammers: social honeypots + machine learning. In: Proceedings of the 33rd International ACM SIGIR Conference on Research and Development in Information Retrieval, pp. 435–442 (2010)
12. Grimme, C., Preuss, M., Adam, L., Trautmann, H.: Social bots: human-like by means of human control? Big Data **5**(4), 279–293 (2017)
13. Feng, S., Wan, H., Wang, N., Li, J., Luo, M.: TwiBot-20: a comprehensive Twitter bot detection benchmark. In: Proceedings of the 30th ACM International Conference on Information & Knowledge Management, pp. 4485–4494 (2021)
14. Hays, C., Schutzman, Z., Raghavan, M., Walk, E., Zimmer, P.: Simplistic collection and labeling practices limit the utility of benchmark datasets for Twitter bot detection. In: Proceedings of the ACM Web Conference, pp. 3660–3669 (2023)
15. Wu, J., Ye, X., Mou, C.: BotShape: a novel social bots detection approach via behavioral patterns. arXiv preprint arXiv:2303.10214 (2023)
16. Yang, K.-C., Varol, O., Hui, P.-M., Menczer, F.: Scalable and generalizable social bot detection through data selection. In: Proceedings of the AAAI Conference on Artificial Intelligence, vol. 34, pp. 1096–1103 (2020)
17. Heidari, M., Jones, J.H.: Using BERT to extract topic-independent sentiment features for social media bot detection. In: 2020 11th IEEE Annual Ubiquitous Computing, Electronics & Mobile Communication Conference (UEMCON), pp. 0542–0547. IEEE (2020)
18. Cresci, S.: A decade of social bot detection. Commun. ACM **63**(10), 72–83 (2020)
19. Dukić, D., Keča, D., Stipić, D.: Are you human? Detecting bots on Twitter using BERT. In: 2020 IEEE 7th International Conference on Data Science and Advanced Analytics (DSAA), pp. 631–636. IEEE (2020)
20. Dehghan, A., Siuta, K., Skorupka, A., et al.: Detecting bots in social-networks using node and structural embeddings. J. Big Data **10**(1), 119 (2023)
21. Magelinski, T., Beskow, D., Carley, K.M.: Graph-Hist: graph classification from latent feature histograms with application to bot detection. In: Proceedings of the AAAI Conference on Artificial Intelligence, vol. 34, pp. 5134–5141 (2020)
22. Alhosseini, S.A., Tareaf, R.B., Najafi, P., Meinel, C.: Detect me if you can: spam bot detection using inductive representation learning. In: Companion Proceedings of the 2019 World Wide Web Conference, pp. 148–153 (2019)
23. Feng, S., Tan, Z., Li, R., Luo, M.: Heterogeneity-aware Twitter bot detection with relational graph transformers. In: Proceedings of the AAAI Conference on Artificial Intelligence, vol. 36, pp. 3977–3985 (2022)
24. Chenbo, F., Zheng, Y., Liu, Y., Xuan, Q., Chen, G.: NES-TL: network embedding similarity-based transfer learning. IEEE Trans. Netw. Sci. Engineer. **7**(3), 1607–1618 (2019)
25. Chawla, N.V., Bowyer, K.W., Hall, L.O., Kegelmeyer, W.P.: SMOTE: synthetic minority over-sampling technique. J. Artif. Intell. Res. **16**, 321–357 (2002)
26. Ivan, T.: Two modifications of CNN. IEEE Trans. Syst. Man Commun. SMC **6**, 769–772 (1976)
27. Miller, Z., Dickinson, B., Deitrick, W., Hu, W., Wang, A.H.: Twitter spammer detection using data stream clustering. Inf. Sci. **260**, 64–73 (2014)

28. Beskow, D.M., Carley, K.M.: Bot-hunter: a tiered approach to detecting & characterizing automated activity on twitter. In: Conference Paper. SBP-BRiMS: International Conference on Social Computing, Behavioral-Cultural Modeling and Prediction and Behavior Representation in Modeling and Simulation, vol. 3 (2018)
29. Dukić, D., Keča, D., Stipić, D.: Are you human? Detecting bots on Twitter using BERT. In: 2020 IEEE 7th International Conference on Data Science and Advanced Analytics (DSAA), pp. 631–636 (2020)
30. Liu, Y., et al.: RoBERTa: a robustly optimized BERT pretraining approach. arXiv preprint arXiv:1907.11692 (2019)
31. Kudugunta, S., Ferrara, E.: Deep neural networks for bot detection. Inf. Sci. **467**, 312–322 (2018)
32. Feng, S., Wan, H., Wang, N., Luo, M.: 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, ASONAM 2021, pp. 236–239. ACM, November 2021
33. Zhang, S., Chen, H., Yang, H., Sun, X., Yu, P.S., Xu, G.: Graph masked autoencoders with transformers. arXiv preprint arXiv:2202.08391 (2022)
34. Yang, H., et al.: Generating counterfactual hard negative samples for graph contrastive learning. In: Proceedings of the ACM Web Conference 2023, WWW 2023, pp. 621–629. ACM, April 2023
35. Sun, X., et al.: Structure learning via meta-hyperedge for dynamic rumor detection. IEEE Trans. Knowl. Data Eng. **35**(9), 9128–9139 (2023)
36. Liu, Y., Tan, Z., Wang, H., Feng, S., Zheng, Q., Luo, M.: BotMoE: Twitter bot detection with community-aware mixtures of modal-specific experts. In: Proceedings of the 46th International ACM SIGIR Conference on Research and Development in Information Retrieval, pp. 485–495 (2023)
37. Kipf, T.N., Welling, M.: Semi-supervised classification with graph convolutional networks. arXiv preprint arXiv:1609.02907 (2016)
38. Veličković, P., Cucurull, G., Casanova, A., Romero, A., Lio, P., Bengio, Y.: Graph attention networks. arXiv preprint arXiv:1710.10903 (2017)
39. Hamilton, W., Ying, Z., Leskovec, J.: Inductive representation learning on large graphs. In: Advances in Neural Information Processing Systems, vol. 30 (2017)
40. Zhang, C., Song, D., Huang, C., Swami, A., Chawla, N.V.: Heterogeneous graph neural network. In: Proceedings of the 25th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining, KDD 2019, pp. 793–803. Association for Computing Machinery, New York, NY, USA (2019)
41. Cresci, S., Di Pietro, R., Petrocchi, M., Spognardi, A., Tesconi, M.: Fame for sale: efficient detection of fake twitter followers. Decis. Support Syst. **80**, 56–71 (2015)
42. Yang, K.-C., Ferrara, E., Menczer, F.: Botometer 101: social bot practicum for computational social scientists. J. Comput. Soc. Sci. **5**(2), 1511–1528 (2022)
43. Ng, L.H.X., Carley, K.M.: BotBuster: multi-platform bot detection using a mixture of experts. In: Proceedings of the International AAAI Conference on Web and Social Media, vol. 17, pp. 686–697 (2023)
44. Feng, S., Tan, Z., Li, R., Luo, M.: Heterogeneity-aware Twitter bot detection with relational graph transformers. In: Proceedings of the AAAI Conference on Artificial Intelligence, vol. 36, no. 4, pp. 3977–3985 (2022)
45. Yang, Y., et al.: SeBot: structural entropy guided multi-view contrastive learning for social bot detection. In: Proceedings of the 30th ACM SIGKDD Conference on Knowledge Discovery and Data Mining, pp. 3841–3852 (2024)
46. Sun, X., Zhang, J., Wu, X., Cheng, H., Xiong, Y., Li, J.: Graph prompt learning: a comprehensive survey and beyond. arXiv preprint arXiv:2311.16534 (2023)