Higher-Order Information Matters: A Representation Learning Approach for Social Bot Detection

Min Gao
Shanghai Key Lab of Intelligent Information Processing, College of Computer Science and Artificial Intelligence
Fudan University
Shanghai, China
mgao21@m.fudan.edu.cn

Qiang Duan
Department of Information Sciences and Technology
The Pennsylvania State University
Abington, Pennsylvania, United States
qduan@psu.edu

Boen Liu
Division of Natural and Applied Sciences
Duke Kunshan University
Kunshan, China
bl263@duke.edu

Yu Xiao
Department of Information and Communications Engineering
Aalto University
Espoo, Finland
yu.xiao@aalto.fi

Xin Wang*
Shanghai Key Lab of Intelligent Information Processing, College of Computer Science and Artificial Intelligence
Fudan University
Shanghai, China
xinw@fudan.edu.cn

Yang Chen*
Shanghai Key Lab of Intelligent Information Processing, College of Computer Science and Artificial Intelligence
Fudan University
Shanghai, China
chenyang@fudan.edu.cn

Abstract

Detecting social bots is crucial for mitigating the spread of misinformation and preserving online conversation authenticity. State-of-the-art solutions typically leverage graph neural networks (GNNs) to model user representations from social relationships and metadata. However, these approaches overlook two key factors: the similarity of a user and her neighbors, as well as the coordinated behaviors of social bots, resulting in a suboptimal detection performance. To address these issues, we propose HyperScan, a novel representation learning method for social bot detection. Specifically, we introduce three effective learners to capture pair-wise, hop-wise, and group-wise relations. HyperScan learns pair-wise user representations based on social relations and user features. It then enhances user representations by building hop-wise interactions across the learned pair-wise user representations for capturing the structure-level proximity information. Subsequently, it models user representations by constructing higher-order (group-wise) relations derived from user profiles, tweets, and social relations to capture the feature-level proximity knowledge. By leveraging hop-wise interactions and higher-order relations, HyperScan significantly improves bot detection performance. Our extensive experiments demonstrate that HyperScan outperforms state-of-the-art methods on three benchmark datasets. Additional studies validate the robustness and effectiveness of each component of HyperScan.

CCS Concepts

• **Computing methodologies → Machine learning**; • **Information systems → Information systems applications**.

Keywords

Bot Detection; Long-Range Dependency; Higher-Order Networks

ACM Reference Format:
Min Gao, Qiang Duan, Boen Liu, Yu Xiao, Xin Wang, and Yang Chen. 2025. Higher-Order Information Matters: A Representation Learning Approach for Social Bot Detection. In *Proceedings of the 34th ACM International Conference on Information and Knowledge Management (CIKM'25), November 10–14, 2025, Seoul, Republic of Korea.* ACM, New York, NY, USA, 11 pages. https://doi.org/10.1145/3746252.3761162

1 Introduction

Online social networks (OSNs) [36], such as Twitter/X, Facebook, and Weibo, have amassed a large number of users and have become an integral part of people's daily lives. With the increasing popularity of these platforms, social bots [9, 24, 47, 55] have emerged as automated accounts designed to mimic human behavior and interact with users. Some of them pose threats to our social and political lives. For instance, social bots have been used to spread fake news, steal users' private information, and manipulate online topics [7, 9, 55]. The surge in artificial intelligence technologies has given rise to advanced social bots and amplified these issues, making social bot detection an imperative task [23, 29].

In the early stage of social bot detection, researchers predominantly employed simple feature-based techniques to identify social bots. These early detection methods [11, 51] relied on manually crafted

*Xin Wang and Yang Chen are Corresponding Authors.

features and lacked the flexibility to cope with the evolving sophistication of bot behaviors. Recent studies [17, 31, 44, 68] on graph deep learning for bot detection have achieved success by utilizing graph neural networks (GNNs) to capture user attributes (such as user metadata and tweet content) and social structures together based on original relational data (see Figure 1(a)).

While most existing methods focus on pair-wise interactions, they often overlook two key factors for detecting bots: *(1) The similarity of a user and her neighbors.* As depicted in Figure 1(c), social bots show consistently high similarities with their neighbors, even across multiple hops. In contrast, humans present relatively lower and more dispersed feature similarities. This pattern suggests that bots tend to engage in long-range interactions, indicating a strong covert strategy by spanning multiple hops rather than direct connections. Such a strategy not only expands their influence but also reduces the chance of being detected. However, previous studies [17, 21, 67] have overlooked the similarity of a bot user and her neighbors, which limits their effectiveness in accurately identifying such bots. *(2) Higher-order (group-wise) interactions.* Higher-order (group-wise) interactions exist ubiquitously in OSNs and are indispensable for capturing group-wise behaviors among users [1, 3]. Social bots often operate in a coordinated fashion [12, 15, 71], reflecting a form of collaboration that traditional pair-wise graph-based methods fail to capture. This is because previous studies [19, 44, 46] have ignored higher-order (group-wise) relations among users (see Figure 1(d)), resulting in the ineffective identification of certain coordinated behaviors. To tackle the above limitations, we need to design a method that can simultaneously consider the similarity of a user and her neighbors along with higher-order relations, to obtain an informative user representation for achieving accurate bot detection. Specifically, our method not only needs to identify the similarity of a user and her neighbors, but also needs to learn higher-order relations to understand more comprehensive user interaction patterns. To achieve these goals, two major challenges arise: ***CH1** How to effectively utilize the information of the similarity of a user and her neighbors based on the user's social relations?* ***CH2** How to effectively model and learn higher-order (group-wise) relations among users?*

To address the above challenges, we propose a novel method called HyperScan for social bot detection. HyperScan considers the similarity of a user and her neighbors as well as higher-order interactions, and it learns user representations by modeling pair-wise, hop-wise, and group-wise (higher-order) relationships together to leverage knowledge from both the structure-level and feature-level proximity among users. For addressing the challenge ***CH1***, we first build a multi-relation graph and learn the pair-wise user representations utilizing a relational graph neural network (R-GCN) [54]. Subsequently, we construct the hop-wise interactions among the learned pair-wise user representations and employ the message-passing mechanism on the hop-based user representations. Such a hop-wise design enables independent modeling of the information of the similarity of a user and her neighbors across different hops, allowing for learning refined structural proximity knowledge from the long-range interactions. To tackle the challenge ***CH2***, we construct a hypergraph, a powerful structure to model higher-order (group-wise) interactions from the representation space, and propose a group-wise learner that learns group-wise user representations by

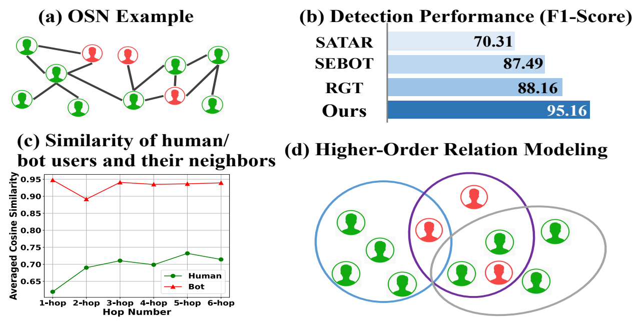


Figure 1: Motivation of HyperScan design. (a) is an online social network example, where green signifies humans and red denotes bots. (b) displays the detection performance of some representative baselines and our method on the TwiBot-20 dataset. (c) We measure the average cosine similarity between each node and its k-hop neighbors ($k = 1$ to 6). The results reveal a clear discrepancy in how feature similarity varies with structure distance between humans and bots. Welch's t-test ($p < 0.001$) confirms the statistical significance, highlighting the need for models that can robustly capture such signals. (d) depicts the example of group-wise relationships among users.

leveraging feature-level proximity knowledge among users. We further incorporate a cross-attention mechanism to effectively fuse user representations from both pair-wise and group-wise relations. This novel fusion makes HyperScan more robust in handling complex relations, including both pair-wise and group-wise relations. Additionally, HyperScan is flexible enough to employ different hypergraph neural networks (HNNs), enhancing its adaptability in various scenario settings. The main contributions of this work are as follows:

- We introduce an approach to effectively utilize higher-order interactions among users, addressing shortcomings of previous studies. We effectively construct higher-order relations by leveraging user metadata, user tweets, and social structure information. To our best knowledge, we are the first to make good use of higher-order (group-wise) relations for social bot detection, which provides an effective structure to comprehensively identify group-wise relations among users.
- We propose HyperScan, a novel bot detection method that considers the similarity of a user and her neighbors as well as higher-order (group-wise) relations. HyperScan uses effective learners to learn pair-wise, hop-wise, and group-wise relations. HyperScan captures pair-wise user representations by considering multiple social relations between users. By investigating the hop-wise interactions across pair-wise user representations, HyperScan enhances user representations by learning structural proximity knowledge within the long-range interactions. The group-wise learner further captures feature-level proximity and acquires informative user representations for detecting bots. Additionally, HyperScan employs the cross-attention mechanism to enhance user representation learning by modeling the complicated correlations across pair-wise and group-wise perspectives. By leveraging these techniques, our method is flexible and robust enough for effective bot detection in various scenarios.
- We conduct extensive experiments on three representative public datasets, and the evaluation results demonstrate the superior

performance of HyperScan over state-of-the-art methods. The effectiveness of each component of the HyperScan design is also verified. These results indicate that our method is both general and robust in different scenarios, highlighting the importance of leveraging both the information of the similarity of a user and her neighbors, along with higher-order relations, for bot detection.

2 Related Work

In this section, we first review existing representative studies for social bot detection in Section 2.1 and then outline the key techniques that inspired the design of HyperScan, including long-range dependency in graph learning in Section 2.2 and higher-order modeling and learning in Section 2.3.

2.1 Social Bot Detection

We review existing representative social bot detection techniques and classify them into two main categories: (1) feature-based methods and (2) deep learning-based models.

Feature-Based Methods. The rise of social bots on social media has prompted researchers to turn to machine learning algorithms, particularly those focused on feature engineering methods [4, 51, 53, 67]. For example, Raffel et al. [51] designed T5 and built user features from tweets and user descriptions. Yang et al. [67] presented SGBot, which primarily extracted user features from profile metadata and employed random forest classifiers for identifying social bots. Additionally, Liu et al. [41] began to focus on the social structure and extracted user features from social community information. However, most of them did not pay attention to the social structure. Even though some approaches have begun to consider the social structure, they typically analyzed them individually and thus overlook the importance of the inconsistency of user features and the social structure.

Deep Learning-Based Methods. Recent deep learning-based works have shown success in identifying bots [6, 17, 19, 44, 68]. Several studies [16, 19] have utilized advanced sequence modeling and natural language processing techniques for social bot detection. For example, Feng et al. [19] developed SATAR, which integrated Word2Vec and Long Short-Term Memory (LSTM) to encode user information, including profiles, tweets, and social relations. Similarly, Fazil et al. [16] introduced DeepSBD, which utilized Convolutional Neural Network (CNN), Bidirectional Long Short-Term Memory (BiLSTM), and a hierarchical attention architecture to identify social bots. Recently, GNN-based methods [6, 17, 21, 46, 57, 68] have been applied to bot detection studies, significantly enhancing social bot detection. Recent studies on social bot detection have explored diverse modeling strategies to capture complex user relationships and community structures [17, 21, 44]. For example, Feng et al. [17] presented RGT model, which modeled relation and influence heterogeneity using a relational graph transformer and semantic attention networks. Liu et al. [44] introduced BotMoE, which employed a community-aware Mixture-of-Experts (MoE) layer to automatically assign users to different communities and leveraged the corresponding expert networks. Yang et al. [68] employed structural entropy to model hierarchical community structures with contrastive learning. Shi et al. [57] enhanced user features by considering the differences between a user's features and neighborhood features. These neighborhood features were based on a hypergraph structure from tweet content and were utilized in GNNs for bot detection. Most existing methods emphasized pair-wise user connections and ignored the higher-order (group-wise) correlations, failing to leverage group-wise interactions with deep learning techniques. Although some have noticed the higher-order correlations, they have solely utilized tweet features to explore these correlations [57]. In contrast, our approach captures richer group-wise interactions by building a hypergraph from multi-modal user data and integrating representations from both pair-wise and hop-wise perspectives, thereby enabling deeper semantic modeling for effective bot detection.

2.2 Long-Range Dependency in Graph Learning

Long-range dependency in graph learning [8, 14, 63] refers to the challenge of effectively capturing interactions between distant nodes in a graph. Traditional GNNs suffer from over-smoothing when propagating features across multiple hops, making it difficult to learn long-range dependencies. Recent solutions include using hierarchical pooling [13], transformer-based architectures [66], and decoupled GNNs [56, 70] to enhance long-range information flow while mitigating over-smoothing. Among them, decoupled GNNs [56, 70] first pre-compute the linear aggregation of k-hop neighbors to generate node features and then utilize the Multi-Layer Perceptron (MLP) for each node without considering the graph structure during training and inference. For example, HopGNN [8] employs the message-passing mechanism among pre-processed multi-hop neighbor features inside each node to overcome the over-smoothing issue. In this work, we treat the similarity of a user and her neighbors across multiple hops as a form of long-range dependency, i.e., hop-wise feature dependency, by building hop-wise interactions and employing GNNs to enhance user representations.

2.3 Higher-Order Information Modeling and Learning

Higher-order information modeling and learning refer to the techniques and methodologies used to understand and predict complex interactions beyond simple pair-wise relationships. Hypergraphs are commonly employed to depict higher-order (group-wise) connections [1, 2]. Unlike traditional graphs, hypergraphs allow hyperedges to connect any number of vertices [2, 40, 69], enabling them to naturally model higher-order relationships among entities. This feature makes hypergraphs valuable in various domains, such as social networks and biological networks [2, 42]. Hypergraph representation learning aims to project the nodes of a hypergraph into a latent space while preserving the structural and relational properties of the network. Various methods have been developed for hypergraph representation learning [10, 22, 27, 33, 37]. In this work, we leverage a hypergraph to model the higher-order (group-wise) relationships among users and employ HNNs [22, 27] to learn informative user representations, subsequently improving bot detection performance.

3 Our Approach: HyperScan

In this section, we first give some preliminary information and define our research problem in Section 3.1. Then, we introduce our approach, HyperScan, in Figure 2. HyperScan contains three modules: (a) feature encoder (in Section 3.2), (b) hybrid-order relation learner (in Section 3.3), and (c) bot detector (in Section 3.4). Firstly, the feature encoder captures user features from user tweets and profile information. Secondly, the hybrid-order relation learner obtains informative user representations based on the user social network and extracted user features. Specifically, the hybrid-order relation learner includes four sub-modules: (b1) pair-wise learner, (b2) hop-wise learner, (b3) group-wise learner, and (b4) cross-attention sub-module. The pair-wise, hop-wise, and group-wise learners obtain pair-wise, hop-wise, and group-wise user representations, respectively. The cross-attention module effectively fuses both pair-wise and group-wise user representations together. Finally, the bot detector is designed for the final prediction.

3.1 Preliminaries

Given a multi-relation social graph $\mathcal{G} = (\mathcal{V}, \mathcal{E}, \mathcal{R}, \mathcal{X}) = (\mathcal{A}, \mathcal{R}, \mathcal{X})$, where $\mathcal{V}$ is the set of nodes with cardinality n, $\mathcal{E} \subseteq \mathcal{V} \times \mathcal{V}$ indicates the set of edges with cardinality m, and $\mathcal{R}$ denotes the set of various edge relations with cardinality r. The binary adjacency matrix $\mathcal{A} \subseteq \{0, 1\}^{n \times n}$ has the (i, j)-th entry $a_{ij} = 1$ if v_i is connected to v_j, and vice versa. $\mathcal{X}$ is the feature matrix where the i-th row x_i is the d_0-dimensional feature vector for node v_i. In this scenario, nodes represent users, including both humans and bots, and edges represent different types of relationships between users. Moreover, we build a hypergraph $\mathcal{H}_{\mathcal{G}}$ based on the k-nearest neighbor (kNN) search algorithm [27, 52]. The hypergraph $\mathcal{H}_{\mathcal{G}}$ has the same node set $\mathcal{V}$ and a new hyperedge set $\mathcal{E}_{\mathcal{H}}$ with size m_h. The detailed process of hypergraph construction can be found in Section 3.3.3. The user labels $\mathcal{Y} = \{0, 1\}$, classifying 0 as humans and 1 as bots.

Problem (Social Bot Detection). Given a graph $\mathcal{G} = (\mathcal{V}, \mathcal{E}, \mathcal{R}, \mathcal{X}) = (\mathcal{A}, \mathcal{R}, \mathcal{X})$ and labels $\mathcal{Y}$ for the nodes, we aim to learn a function f that captures informative user representations to classify social bots from humans.

3.2 Feature Encoder

Previous studies have investigated various types of features from the perspectives of property [19], behavior [21], and network structures [19]. These features have been demonstrated to be effective in distinguishing bots from humans. Therefore, we consider these multi-modal information and extract user features by categorizing them into two types: textual, and property features. Our methods for extracting these features are described in the next two paragraphs.
Textual Encoder. Inspired by [19, 21], we employ the RoBERTa model[1] [43] to transform user descriptions and tweet contents into two semantic feature vectors, referred to as X_{des} and X_{twt}, respectively. These semantic feature vectors provide us with rich information that enables a more accurate understanding of user language characteristics and differentiates between humans and social bots. Subsequently, we encode X_{des} and X_{twt} with a linear layer followed by a leaky-ReLU activation function ϕ, defined as $X_d = \phi(W_d X_{\text{des}} + b_d)$ and $X_t = \phi(W_t X_{\text{twt}} + b_t)$, respectively. W_d, W_t, b_d, and b_t are all learnable parameters.

[1]The RoBERTa model, an advanced variant of the BERT model, is able to learn powerful contextual information by pre-training on large-scale text data. Here, we utilize it to generate general textual features.

Property Encoder. To capture user property features, we utilize both user profiles and network structures. Inspired by previous studies [21, 68], we extract the numeric features X_{num}, such as "num_follower" and "num_following", as well as categorical features X_{cat}, like "protected" and "verified", from user data. These features are commonly used in prior studies [19, 21, 68]. Similarly, we apply a fully connected layer and a leaky-ReLU activation ϕ to capture user representations, including numeric features $X_n = \phi(W_n X_{\text{num}} + b_n)$ and categorical features $X_c = \phi(W_c X_{\text{cat}} + b_c)$. W_n, W_c, b_n, and b_c are all learnable parameters.
Based on the above feature encoding, we obtain an initial node feature $X_{in} = [X_d, X_t, X_n, X_c]$, and the X_{in} is fed into the input of the hybrid-order relation learner module.

3.3 Hybrid-Order Relation Learner

To capture both lower-order (pair-wise) and higher-order (group-wise) relations together and obtain informative user representations, it is important to build implicit higher-order relations according to explicit lower-order relations and the above-encoded user features. Hence, we develop a hybrid-order relation learner to cope with complicated relations among humans and bots and learn informative user representations. Specifically, we first design a pair-wise learner for lower-order (pair-wise) relations in Section 3.3.1 and a hop-wise learner to gather multi-hop user features based on social structures in Section 3.3.2. Then, we build a group-wise learner for higher-order (group-wise) relations in Section 3.3.3. Finally, we leverage the cross-attention mechanism to cope with complicated relations and learn informative hybrid-order user representations in Section 3.3.4.

3.3.1 ***Pair-wise Learner.*** Given the multiple relations between users, we design a multiple relational graph learning module to capture such pair-wise heterogeneous connections. Specifically, we employ the relational graph neural network (R-GCN) [54] as the core technique of our pair-wise user representation module. R-GCN has been demonstrated to be effective in capturing multiple relations for graph data. We utilize l stacked relational graph convolutional layers to cope with the complex multi-relation graphs and learn informative embeddings for modeling pair-wise connections. For the s-th ($s = \{1, 2, \cdots, l-1\}$) layer of R-GCN,

$$x_i^{(s+1)} = \sigma\left(W_0^{(s)} x_i^{(s)} + \sum_{r \in \mathcal{R}} \sum_{j \in \mathcal{N}_i^r} \frac{1}{c_{i,r}} W_r^{(s)} x_j^{(s)}\right), \tag{1}$$

where x_i and x_j represent the extracted features of node i and node j, respectively. $\mathcal{N}_i^r$ denotes the set of neighboring nodes of node i under the relation $r \in \mathcal{R}$, and $c_{i,r} = |\mathcal{N}_i^r|$ indicates a normalization factor. $W_0^{(s)}$ and $W_r^{(s)}$ are learnable weight matrices. The final pair-wise user representations are denoted as $X_p = f_p(X_{in})$ with pair-wise learner f_p and extracted user feature X_{in}.

3.3.2 ***Hop-wise Learner.*** After aggregating information across multiple relations, it is necessary to incorporate the information of a user and her neighbors to enhance user representations with richer semantics. Inspired by [8], we propose a hop-wise learner. Specifically, we first pre-compute multi-hop user representations X_p from the pair-wise user representations with $X_p^{l_h} = \hat{A}^{l_h} X_p$ for

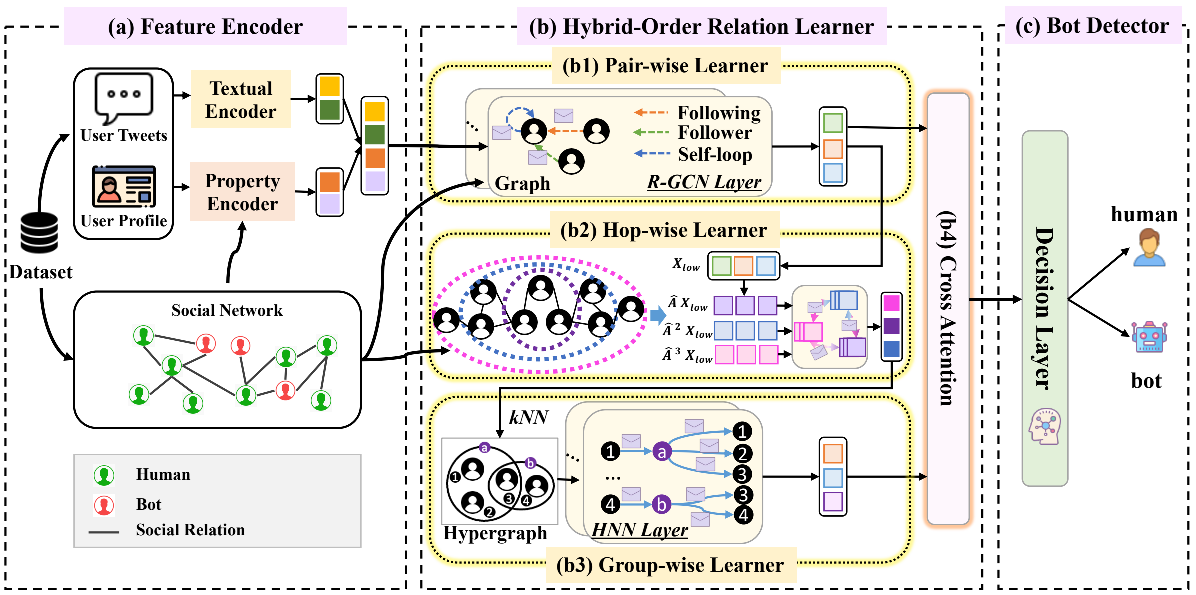


Figure 2: Overview of HyperScan framework. It contains three modules: (a) feature encoder; (b) hybrid-order relation learner with (b1) pair-wise learner, (b2) hop-wise learner, (b3) group-wise learner, and (b4) cross attention module (detailed architecture is illustrated in Figure 3); and (c) bot detector for final prediction.

l_h-hop. Then, we construct a hop-wise graph g_h derived from the user social network and reshape the multi-hop user representations through a linear layer combined with a trainable hop-wise vector V_h, as follows:

$$X_h = [W_{hop}X_p + b_{hop}] + V_h, \tag{2}$$

where W_{hop} and b_{hop} are learnable parameters. Considering the intrinsic co-occurrence of multiple hop features within each user, we seek to leverage GNNs to effectively model multi-hop user representations. Consequently, we employ GNNs to encapsulate the interactions among these multi-hop user representations via $X_h = f_h\,(g_h, X_h)$. Here, we utilize the GAT [61] as f_h to capture hop interactions. Specifically, we adopt a scaled dot-product attention mechanism, which improves efficiency and captures cross-hop dependencies in our task. This computing process can be represented as $X_{hop} = A_h X_h$, where $A_h = \text{Softmax}\left(\frac{\mathcal{Q}_h \mathcal{K}_h^T}{\sqrt{d_h}}\right)\mathcal{V}_h$, $\mathcal{Q}_h = X_h \mathcal{W}_Q^h$, $\mathcal{K}_h = X_h \mathcal{W}_K^h$, and $\mathcal{V}_h = X_h \mathcal{W}_V^h$, with learnable matrices $\mathcal{W}_Q^h$, $\mathcal{W}_K^h$ and $\mathcal{W}_V^h$, respectively. Here, the Softmax is the non-linear activation function, d_h denotes the dimension of X_h, and the $\mathcal{K}_h^T$ represents the transpose of $\mathcal{K}_h$. Furthermore, we have examined other GNN architectures like GCN and GraphSAGE [28, 39] to model hop interactions, which achieved similar results, indicating the adaptability of such design. Then, we apply a mean fusion strategy to derive the final hop-wise user representations $X_h = \text{mean}(X_{hop})$. Similarly, other fusion mechanisms were also explored, but no significant enhancements were observed. After hop interaction, we obtain more distinguishable user representations for the following group-wise relation modeling.

3.3.3 *Group-wise Learner.*

Considering the synergistic nature of social bots (social bots interact with neighboring partners and conspire as a group to achieve common goals), we aim to capture the group-wise correlations by employing the hypergraph structure to represent higher-order interactions among users. A hypergraph is an extension of a graph that facilitates modeling complex higher-order interactions [22]. Therefore, how to construct a hypergraph is critical to model the higher-order interactions between users. To enhance the expression power of our model, we introduce the idea of skip-connection, which concatenates the extracted user features and the learned hop-wise user representations together, formulated as $X = X_{in} + X_h = X_{in} + f_h(f_p(X_{in}))$. Based on these features and representations, we use the k-nearest neighbors (kNN) algorithm [27, 52] to construct a hypergraph $\mathcal{H}_\mathcal{G}$ by grouping each user with its closest $k-1$ neighbors based on user features, including user metadata, tweets, and social relations. The generated hypergraph $\mathcal{H}_\mathcal{G}$ contains nodes representing users (including humans and bots) and hyperedges connecting subsets of similar users. We also explored alternative k-hop based construction techniques, but they did not show notable improvements.

We employ representative HNNs [22, 33, 65] to learn informative group-wise user representations from the hypergraph $\mathcal{H}_\mathcal{G}$. Specifically, we take the learned total user representations $X = \tanh(W_h X + b_h)$ and the constructed hypergraph $\mathcal{H}_\mathcal{G}$ as the inputs to the group-wise learner, where W_h and b_h are learnable parameters. Many popular HNNs, such as HGNN [22], UniSAGE [33], and UniGIN [33], could be utilized in this component. For simplicity without losing generality, we take the HGNN model as our HNN backbone. We follow the two-step message-passing scheme to propagate information on hypergraphs. During the first stage, each hyperedge representation is aggregated from the connected node representations. The second stage involves updating the node representations with their associated hyperedge representations.

Therefore, the process of hypergraph representation can be formulated by

$$X_g = D_i^{-1/2} H W D_e^{-1} H^T D_i^{-1/2} X\Theta, \tag{3}$$

where $D_i^{-1/2}$ and D_e^{-1} denote the inverse square root of the node degree matrix and the inverse of the hyperedge degree matrix, respectively. H stands for the incidence matrix. W is the weight parameter vector to be learned during the training process. Θ represents the hypergraph convolution function used for node feature filtering. X_g corresponds to the learned final group-wise user representations, which can be denoted as $X_g = f_g(X)$ with the group-wise learner f_g and user features X.

3.3.4 ***Hybrid Representation*****.** Given the disparity in embedding spaces of user representations from different perspectives, we have developed a hybrid user representation module leveraging cross-attention. This module adeptly captures the complex correlations and mapping relationships between pair-wise and group-wise user representations through a multi-head cross-attention mechanism [60]. Since we only consider two types of user representations, we revise the original architecture to suit our model. Specifically, the cross-attention architectures for both pair-wise and group-wise representations share the same building blocks and only differ in terms of the Query, Key, and Value matrices. The architecture for pair-wise representations is illustrated in Figure 3. For group-wise representations, the sole modifications involve setting the Query input to X_g and replacing the inputs for the Key and Value with X_p.

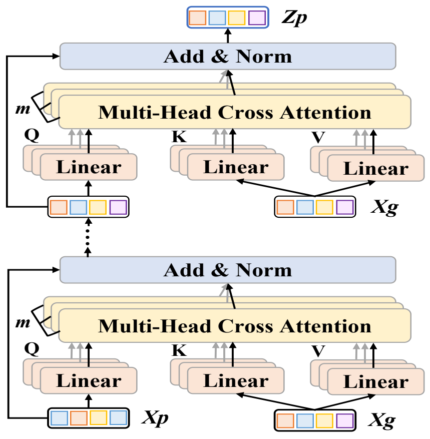


Figure 3: Illustration of the architecture of cross attention module for pair-wise user representations.

Specifically, both pair-wise user representation X_p and group-wise user representation X_g capture valuable knowledge within their respective encoding spaces. Therefore, each of them could integrate complementary information from each other. For simplicity, we take the fused pair-wise representation as an example. Initially, the pair-wise user representation learns correlations and mapping relationships with the group-wise representation, acting as the Query, while the group-wise representation serves as the Key and Value within the multi-head cross-attention mechanism. The computation for the Query, Key, and Value matrices is given by $Q_c^p = X_p W_{Q_c^p}$, $\mathcal{K}_c^g = X_g W_{K_c^g}$, and $\mathcal{V}_c^g = X_g W_{V_c^g}$. Here, $W_{Q_c^p}$, $W_{K_c^g}$, and $W_{V_c^g}$ are learnable parameters. The cross-attention mechanism can be formulated as

$$\mathcal{A}_c^p = \text{Softmax}\left(\frac{Q_c^p \mathcal{K}_c^{g\,T}}{\sqrt{d_{\mathcal{K}_c^g}}}\right)\mathbf{V}_c^g, A_p = \left[\mathcal{A}_c^p \mid c \in \{1, 2, \dots, h\}\right], \tag{4}$$

where Softmax denotes the non-linear activation function, $d_{\mathcal{K}_c^g}$ refers to the dimension of Key matrix $\mathcal{K}_c^g$, and $\mathcal{K}_c^{g\,T}$ represents the transpose of $\mathcal{K}_c^g$. The c refers to the c-th attention head, $[\cdot]$ denotes the concatenation operation, h is the number of attention heads, and A_p represents the cross-attention-based representation from h attention heads.

The output learned in this manner then serves as the new Query with the same group-wise representations as the Key and Value matrices. A feed forward network subsequently acts as a key-value memory. Furthermore, we incorporate residual connection and layer normalization over the output of each multi-head cross-attention module to enhance the training process. This can be represented as

$$Z_p = \text{LayerNorm}\left(X_p + A_p W_p\right), \tag{5}$$

where X_p is the pair-wise user representation, A_p is the learnable cross-attention weights, W_p is the trainable weight matrix and LayerNorm denotes the operation of layer normalization.

Similarly, for the group-wise representation X_g, we generate the Query with X_g itself, while the Key and Value matrices are formed from the pair-wise representation X_p, defined as $\mathbf{Q}_c^g = X_g W_{Q_c^g}$, $\mathcal{K}_c^p = X_p W_{K_c^p}$, and $\mathbf{V}_c^p = X_p W_{V_c^p}$. Here, $W_{Q_c^g}$, $W_{K_c^p}$, and $W_{V_c^p}$ are learnable parameters. The cross-attention weights can then be calculated by $\mathcal{A}_c^g = \text{Softmax}\left(\frac{Q_c^g \mathcal{K}_c^{p\,T}}{\sqrt{d_{\mathcal{K}_c^p}}}\right)\mathbf{V}_c^p$, where $d_{\mathcal{K}_c^p}$ refers to the dimension of Key matrix $\mathcal{K}_c^p$, $\mathcal{K}_c^{p\,T}$ represents the transpose of $\mathcal{K}_c^p$, and c denotes the c-th attention head. Then, with h attention heads, we can obtain $A_g = \left[\mathcal{A}_c^g \mid c \in \{1, 2, \dots, h\}\right]$. Finally, the final cross-attention-based representation can be obtained by $Z_g = \text{LayerNorm}\left(X_g + A_g W_g\right)$, where X_g represents the group-wise user representation, A_g denotes the learnable cross-attention weights, W_g is the trainable weight matrix, and LayerNorm stands for the layer normalization operation. We concatenate Z_p and Z_g to obtain the final user representation via $Z = \left[Z_p, Z_g\right]$, where $[\cdot]$ denotes the concatenation operation.

3.4 Bot Detector

Based on the final user representation Z, HyperScan utilizes a fully connected layer and subsequently applies an activation function sigmoid. The final predicted user label $\hat{Y}$ can be determined by $\hat{Y} = \text{sigmoid}(W_o Z + b_o)$, where W_o and b_o are learnable parameters. Finally, we train our model in a supervised learning process that minimizes the binary cross-entropy loss along with the L2 norm regularization

$$\mathcal{L} = -\sum_{i \in Y} \left[y_i \log\left(\hat{y}_i\right) + \left(1 - y_i\right) \log\left(1 - \hat{y}_i\right)\right] + \lambda \sum_{w \in \theta} w^2, \tag{6}$$

where y_i and $\hat{y}_i$ are the ground-truth label and the predicted label of node i, respectively. θ denotes all trained parameters in the model, and λ is the regularizer parameter.

4 Experiments and Analysis

In this section, we perform an extensive empirical evaluation of HyperScan under various scenarios. Our experiments seek to answer the following questions:
RQ1: How effective is HyperScan for social bot detection?
RQ2: How do HyperScan's components affect its performance?
RQ3: How does HyperScan perform under different hyperparameter settings?
RQ4: Can HyperScan learn more discriminative node representations, and how well does it generalize under varying training data sizes?

4.1 Experiment Setup

Dataset Information. We evaluated the effectiveness of HyperScan on three representative real-world datasets for social bot detection tasks: TwiBot-20 [20], TwiBot-22 [18], and MGTAB-22 [58], which are widely used in [6, 17, 21, 32, 44, 50]. TwiBot-20 contains 229,580 accounts extracted from Twitter and their following and follower interactions. TwiBot-22 contains a heterogeneous Twitter network with 4 types of entities and 14 types of relations. MGTAB-22 consists of 10,199 accounts and 7 types of relations and exhibits high-quality annotation. The statistics of datasets are presented in Table 1.

Table 1: Statistics of datasets

Dataset	#Nodes	#Humans	#Bots	#Edges	#Relations
TwiBot-20	229,580	5,237	6,589	227,979	2
TwiBot-22	1,000,000	860,057	139,943	170,185,937	14
MGTAB-22	10,199	7,451	2,748	1,700,108	7

Baselines. We compare the performance of our model against the following representative baseline models.

- **Feature-based methods:** T5 [51] extracts features from tweets and descriptions, and then utilizes fully connected layers paired with nonlinear activation functions to identify bots. FriendBot [4] explores user features from network, content, and temporal perspectives and utilizes machine learning models to detect bots. SGBot [67] primarily extracts user features from profile metadata and employs random forest classifiers for bot detection.
- **Deep learning-based methods:** GAT [61] uses an attention mechanism during the message-passing process. HGT [30] adopts the relative temporal encoding (RTE) strategy for better heterogeneous graph learning. SimpleHGN [45] is a simple and effective HGNN architecture that pre-computes the neighbor aggregation and combines a Transformer-based module to fuse semantic information. SATAR [19] is a self-supervised representation learning framework that simultaneously leverages semantics, properties, and structural information for specific users. RGT [17] adopts relational graph transformers and semantic attention networks for representation learning and bot detection. SEBOT [68] introduces structural entropy to capture hierarchical community structure information for bot detection in a contrastive learning manner.

Evaluation Metrics. Following [17, 19, 68], we evaluate the performance of each model using four metrics, including accuracy, precision, recall, and F1-score, which are widely used in binary classification tasks.

Implementation Details. We report the mean test performances over 5 trials, with the same training/validation/testing split of 70/20/10 as [21], [18] and [58] for a fair comparison. All features are normalized. The dimension of the node representation is 256. The layers of R-GCN, GAT, and HGNN are configured to be 2. We perform a grid search on the learning rate η in a range of $\{1 \times 10^{-4}, 5 \times 10^{-4}, 1 \times 10^{-3}, 5 \times 10^{-3}\}$. To avoid over-fitting, we apply L2 regularization with weight_decay $\lambda = 5 \times 10^{-4}$. We use EasyGraph [26] to build graphs and adopt PyTorch [48] and PyTorch Geometric [25] to implement all the models and optimize them with the Adam optimizer [38]. We conducted all experiments on a Linux server equipped with the Intel(R) Xeon(R) Gold 6142 CPU @ 2.60GHz, 376GB of memory, and 64 CPU cores.

4.2 Detection Performance (RQ1)

We benchmark the performance of HyperScan in bot detection against representative baselines. The detection results are reported in Table 2. We highlight several findings after analyzing the results:

- HyperScan achieves the best performance across all four metrics on the TwiBot-20 dataset. For the TwiBot-22 dataset, HyperScan achieves the best accuracy and is a runner-up for other metrics. For the MGTAB-22 dataset, HyperScan obtains the best accuracy, precision, and F1-score. Given that each dataset presents a unique scenario, like TwiBot-22 with more nodes and MGTAB-22 featuring higher-quality annotation, these results indicate that our method is both general and robust in different scenarios, highlighting the significance of leveraging hop-wise feature interactions and higher-order information.
- We notice that several baseline models, like SGBot [67], GAT [61], SimpleHGN [45], RGT [17], and SEBOT [68] attain relatively high scores in specific metrics. However, their performances are notably inconsistent across various datasets and even vary among different metrics on the same dataset. For example, RGT excels in terms of accuracy and F1-score, SEBOT achieves higher precision, while SGBot stands out in recall on the TwiBot-20 dataset.
- HyperScan surpasses other feature-based baselines across four metrics by considering the hop-wise feature information, highlighting the significance of integrating graph structure-related design to effectively detect social bots. Compared with some advanced deep learning-based methods, HyperScan exhibits better results in accuracy, precision, and F1-score because it models complex social interactions from a newly introduced higher-order perspective, indicating that such group-wise interactions could further leverage structural information and provide new knowledge for social bot detection.

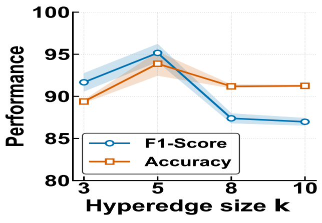
4.3 Ablation Studies (RQ2)

Besides comparing with baseline methods, we also design seven variations of HyperScan by removing or replacing one specific component to assess the effectiveness of different components. The variations are as follows:

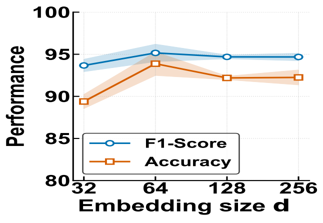
- M1 (w/o Hop-wise Learner): removing the hop-wise learner in HyperScan.

Table 2: The performances (mean ± stdev) of all methods on three datasets. The best-performing method is highlighted in bold, while the second-best performance is underlined. "-" indicates the corresponding method is not scalable or could not be applied due to the absence of some raw data.

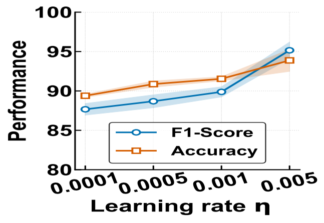
Category	Method	TwiBot-20				TwiBot-22				MGTAB-22			
		Accuracy	F1-Score	Precision	Recall	Accuracy	F1-Score	Precision	Recall	Accuracy	F1-Score	Precision	Recall
Feature-based	T5	65.18$_{\pm0.75}$	49.53$_{\pm3.02}$	68.14$_{\pm1.80}$	39.74$_{\pm4.17}$	72.09$_{\pm0.63}$	21.14$_{\pm1.42}$	64.14$_{\pm0.80}$	12.07$_{\pm2.13}$	89.23$_{\pm0.86}$	79.11$_{\pm3.14}$	81.95$_{\pm5.23}$	77.21$_{\pm8.34}$
	FriendBot	74.00$_{\pm1.68}$	78.46$_{\pm0.12}$	71.49$_{\pm0.03}$	87.64$_{\pm0.03}$	-	-	-	-	-	-	-	-
	SGBot	81.50$_{\pm0.25}$	84.75$_{\pm0.21}$	76.50$_{\pm0.20}$	95.00$_{\pm0.35}$	75.13$_{\pm0.02}$	36.79$_{\pm0.13}$	73.14$_{\pm0.07}$	24.57$_{\pm0.12}$	90.83$_{\pm0.51}$	82.95$_{\pm1.51}$	86.51$_{\pm2.58}$	79.72$_{\pm1.76}$
Deep learning-based	GAT	81.23$_{\pm1.69}$	83.66$_{\pm1.54}$	80.69$_{\pm0.52}$	88.42$_{\pm0.79}$	80.14$_{\pm0.62}$	**56.05$_{\pm0.98}$**	**75.65$_{\pm1.34}$**	**40.65**$_{\pm1.28}$	85.02$_{\pm1.02}$	70.04$_{\pm3.69}$	64.49$_{\pm4.25}$	76.41$_{\pm2.71}$
	HGT	85.31$_{\pm0.46}$	87.31$_{\pm0.67}$	84.32$_{\pm0.96}$	90.14$_{\pm0.72}$	74.90$_{\pm0.15}$	40.12$_{\pm0.93}$	68.24$_{\pm2.15}$	27.59$_{\pm3.06}$	89.60$_{\pm2.01}$	82.25$_{\pm2.03}$	81.04$_{\pm0.70}$	**84.23$_{\pm0.84}$**
	SimpleHGN	85.33$_{\pm0.33}$	87.81$_{\pm0.31}$	84.27$_{\pm0.23}$	92.60$_{\pm0.48}$	75.42$_{\pm0.36}$	46.23$_{\pm1.02}$	73.14$_{\pm2.27}$	33.10$_{\pm1.69}$	91.30$_{\pm1.30}$	84.86$_{\pm2.76}$	85.67$_{\pm3.19}$	84.17$_{\pm2.99}$
	SATAR	58.61$_{\pm0.27}$	70.31$_{\pm0.21}$	59.98$_{\pm0.85}$	81.38$_{\pm2.46}$	-	-	-	-	-	-	-	-
	RGT	86.61$_{\pm0.24}$	88.16$_{\pm0.38}$	84.16$_{\pm0.63}$	90.37$_{\pm1.46}$	76.59$_{\pm0.74}$	43.27$_{\pm1.06}$	74.89$_{\pm0.26}$	31.03$_{\pm0.17}$	88.36$_{\pm0.43}$	80.14$_{\pm1.46}$	80.15$_{\pm2.07}$	80.06$_{\pm0.07}$
	SEBOT	86.07$_{\pm0.35}$	87.49$_{\pm0.23}$	84.92$_{\pm0.13}$	91.57$_{\pm0.27}$	-	-	-	-	90.75$_{\pm1.28}$	82.15$_{\pm2.43}$	82.74$_{\pm2.93}$	81.61$_{\pm0.34}$
Ours	HyperScan	**93.88$_{\pm1.40}$**	**95.16$_{\pm1.04}$**	**95.18$_{\pm1.31}$**	**95.16$_{\pm1.27}$**	**91.57$_{\pm0.19}$**	52.43$_{\pm0.52}$	74.70$_{\pm1.08}$	40.38$_{\pm0.36}$	**93.01$_{\pm0.07}$**	**85.93$_{\pm0.14}$**	**89.94$_{\pm0.33}$**	82.26$_{\pm0.16}$



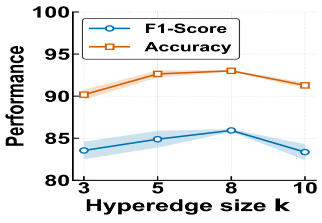
(a) Different sizes of hyperedge k



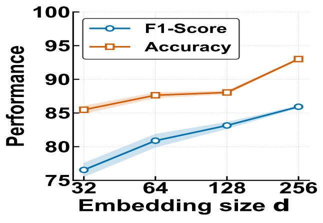
(b) Different embedding sizes d



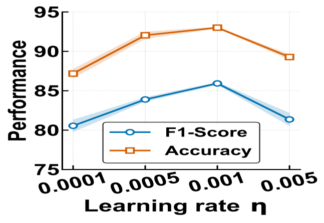
(c) Different learning rates η



(d) Different sizes of hyperedge k



(e) Different embedding sizes d



(f) Different learning rates η

Figure 4: Performance of HyperScan on different datasets w.r.t different settings of hyperparameters. (a)-(c) for TwiBot-20 dataset and (d)-(f) for MGTAB-22 dataset, respectively.

Table 3: Ablation study on datasets reveals that each component impacts and enhances performance. Our model HyperScan (M0) outperforms all other variants.

Settings	TwiBot-20		TwiBot-22		MGTAB-22	
	Accuracy	F1-Score	Accuracy	F1-Score	Accuracy	F1-Score
M0 (Ours)	**93.88$_{\pm1.40}$**	**95.16$_{\pm1.04}$**	**91.57$_{\pm0.19}$**	**52.43$_{\pm0.52}$**	**93.01$_{\pm0.07}$**	**85.93$_{\pm0.14}$**
M1 (w/o Hop-wise Leaner)	90.62$_{\pm0.51}$	89.98$_{\pm1.02}$	88.25$_{\pm3.12}$	48.27$_{\pm1.09}$	91.85$_{\pm2.43}$	84.27$_{\pm1.09}$
M2 (w/o Group-wise Leaner)	90.50$_{\pm0.73}$	92.85$_{\pm0.71}$	87.92$_{\pm0.85}$	49.08$_{\pm0.62}$	88.25$_{\pm2.15}$	84.18$_{\pm1.14}$
M3 (rep HNN w/ UniSAGE)	90.04$_{\pm0.17}$	93.57$_{\pm0.31}$	87.73$_{\pm2.51}$	50.50$_{\pm1.23}$	89.73$_{\pm2.41}$	84.50$_{\pm1.23}$
M4 (rep HNN w/ UniGIN)	90.49$_{\pm0.65}$	93.09$_{\pm0.30}$	89.65$_{\pm1.30}$	50.04$_{\pm0.76}$	91.65$_{\pm0.26}$	85.04$_{\pm0.77}$
M5 (rep CA w/ Con.)	91.23$_{\pm0.01}$	93.38$_{\pm0.01}$	89.48$_{\pm0.74}$	50.35$_{\pm0.39}$	89.65$_{\pm1.57}$	84.04$_{\pm1.09}$
M6 (rep CA w/ Avg.)	89.24$_{\pm0.09}$	90.64$_{\pm0.25}$	87.94$_{\pm1.36}$	48.86$_{\pm0.51}$	87.65$_{\pm1.04}$	84.04$_{\pm0.46}$
M7 (rep CA w/ Con.+ReLU)	90.46$_{\pm0.15}$	92.62$_{\pm0.62}$	86.76$_{\pm1.28}$	48.31$_{\pm0.94}$	88.64$_{\pm1.03}$	83.97$_{\pm0.34}$

- M2 (w/o Group-wise Learner): removing the group-wise learner in HyperScan.
- M3 (rep HNN w/ UniSAGE): replacing the HNN backbone with the UniSAGE in the group-wise learner in HyperScan.
- M4 (rep HNN w/ UniGIN): replacing the HNN backbone with the UniGIN in the group-wise learner in HyperScan.
- M5 (rep CA w/ Con.): replacing the cross attention module with the concatenation operation in HyperScan.
- M6 (rep CA w/ Avg.): replacing the cross attention module with the averaging operation in HyperScan.
- M7 (rep CA w/ Con.+ReLU): replacing the cross attention module with the concatenation operation followed by a ReLU layer in HyperScan.

From the results in Table 3, we can make the following conclusions: (1) Removing each component (M1-M2) would reduce the detection performance because the design of key components of HyperScan contributes to the final detection performance, indicating the importance of considering the information of the similarity of a user and her neighbors, as well as the higher-order knowledge for bot detection. (2) We replace the HNN backbone in the group-wise learner with representative HNNs like UniSAGE and UniGIN (M3-M4). The observed decreased performance indicates that the HNN backbone could be adept at capturing such higher-order interactions by our group-wise learner. (3) We replace the cross attention module with simple fusion manners by adopting concatenation, averaging, and concatenation followed by a ReLU layer, respectively (M5-M7). We notice that the performance of these three variants decreases, which highlights the effectiveness of the cross attention module in capturing the underlying correlations between both pair-wise and group-wise user representations for final detection performance.

4.4 Sensitivity Analysis (RQ3)

In this section, we perform a sensitivity analysis of critical hyperparameters in HyperScan, namely hyperedge size k, embedding size d, and learning rate η. Figure 4 presents the outcomes on different datasets.

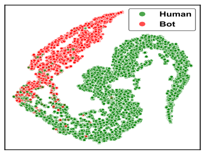
Effect of hyperedge size k. To evaluate the impact of the size of each hyperedge k on model performance, we conduct experiments with k values spanning [3, 5, 8, 10]. Figure 4(a) and Figure 4(d) present the detection performance under different values of k with all other hyperparameters fixed. It can be noted that the proposed HyperScan method delivers the best performance on the TwiBot-20 dataset at $k = 5$ and on the MGTAB-22 dataset at $k = 8$. This indicates that an appropriate k effectively simplifies the higher-order structures, allowing our model to capture effective group-wise interactions for enhanced generalization.

Effect of embedding size d. We study the influence of embedding sizes (from 32 to 256) on the performance of social bot detection. All other hyperparameters are fixed, except for the embedding size. The results are shown in Figure 4(b) and Figure 4(e). One can observe that the performance enhances with larger embedding sizes for both datasets. This improvement could be attributed to the fact that larger embedding sizes allow HyperScan to handle more complex social structures and learn more informative user representations.

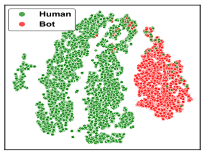
Effect of learning rate η. To evaluate the impact of the learning rate on model performance, we conducted experiments using a range of learning rate values from 0.0001 to 0.005. Figure 4(c) and Figure 4(f) show the detection performance with varying learning rates η with all other hyperparameters fixed. We noticed that an overall improvement in performance for both datasets as the learning rate increases. Notably, HyperScan achieves the best performance with minimal fluctuations at a learning rate of $\eta = 0.005$ on the TwiBot-20 dataset, while it performs less effectively at $\eta = 0.0001$. For the MGTAB-22 dataset, the best performance is achieved at $\eta = 0.001$. This highlights that an appropriate learning rate could accelerate the convergence of our model and ensure stable performance.

4.5 Visualization & Generalization (RQ4)

In this section, we visualize the user representations of the TwiBot-20 dataset generated by HGT and HyperScan using t-SNE [59]. Nodes with the same ground-truth labels are in the same colors. From Figure 5, one could observe that our method effectively differentiates between humans and bots, whereas HGT has a large partial overlap, highlighting that HyperScan learns more distinguishable user representations than HGT.



(a) HGT



(b) HyperScan

Figure 5: Visualization of user embeddings generated by (a) HGT and (b) HyperScan on the TwiBot-20 dataset. Users are colored by their labels.

We further examine the generalizability of our model by adjusting the training ratio on the TwiBot-20 dataset, as depicted in Figure 6. Notably, using just 50% of the training data, HyperScan outperforms other baselines like SGBot and RGT, confirming its strong generalization. Additionally, our findings reveal that more edges and features contribute to enhanced performance, indicating that both factors are beneficial for detecting social bots.

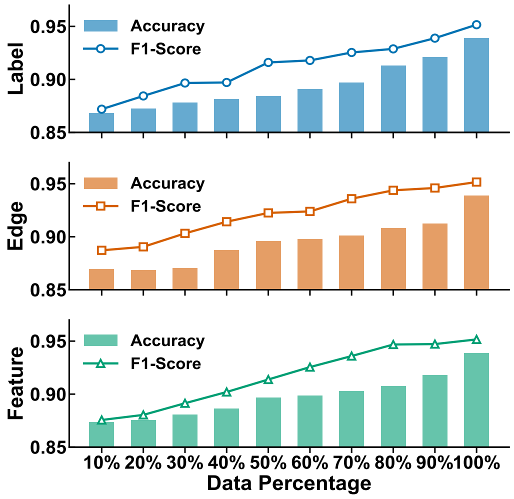


Figure 6: Generalization of the proposed model with varying training ratios.

5 Conclusion and Future Work

Social bot detection is an important task in OSNs, which could prevent the adverse effects of malicious bots on user experience and platform security. Existing solutions focus on the lower-order (pair-wise) relations and ignore the higher-order (group-wise) relations among users, causing unsatisfactory detection performance. To fill this gap, this paper proposes HyperScan, a novel representation learning approach for social bot detection. HyperScan first learns pair-wise user representations by considering multiple relations and integrates hop-wise interactions across pair-wise user representations. Subsequently, HyperScan learns group-wise user representations by constructing group-wise relations derived from user metadata, tweet texts, and social relations. Moreover, HyperScan obtains a hybrid user representation based on the cross-attention mechanism for accurate social bot detection. We have conducted extensive experiments on real-world datasets to demonstrate the superiority of our solution over existing methods.

In future work, we intend to explore the evolution of social bots and humans by building an information propagation graph and considering effective modeling of their online diffusion patterns [5, 49, 62]. Moreover, we plan to extend our method using federated learning [64] across several social platforms to enhance its generalization capabilities by integrating the cross-platform knowledge. Additionally, large language models (LLMs) will be employed to generate semantic knowledge related to higher-order relations to improve the quality of initial node features [34, 35].

Acknowledgments

This work is sponsored by National Natural Science Foundation of China (No. 62072115, No. 62472101), Shanghai Science and Technology Innovation Action Plan Project (No. 22510713600).

GenAI Usage Disclosure

We use generative AI tools to assist with language editing. All AI-generated content has been reviewed and edited by the authors to ensure accuracy and improve clarity.

References

[1] Unai Alvarez-Rodriguez, Federico Battiston, Guilherme Ferraz de Arruda, Yamir Moreno, Matjaž Perc, and Vito Latora. 2021. Evolutionary dynamics of higher-order interactions in social networks. *Nature Human Behaviour* 5, 5 (2021), 586–595.

[2] Federico Battiston, Enrico Amico, Alain Barrat, Ginestra Bianconi, Guilherme Ferraz de Arruda, Benedetta Franceschiello, Iacopo Iacopini, Sonia Kéfi, Vito Latora, Yamir Moreno, Micah M Murray, Tiago P Peixoto, Francesco Vaccarino, and Giovanni Petri. 2021. The physics of higher-order interactions in complex systems. *Nature Physics* 17, 10 (2021), 1093–1098.

[3] Austin R Benson, David F Gleich, and Jure Leskovec. 2016. Higher-order organization of complex networks. *Science* 353, 6295 (2016), 163–166.

[4] David M Beskow and Kathleen M Carley. 2020. You are known by your friends: Leveraging network metrics for bot detection in Twitter. *Open Source Intelligence and Cyber Crime: Social Media Analytics* (2020), 53–88.

[5] Meng Cai, Han Luo, Xiao Meng, Ying Cui, and Wei Wang. 2023. Network distribution and sentiment interaction: Information diffusion mechanisms between social bots and human users on social media. *Information Processing & Management* 60, 2 (2023), 103197.

[6] Zijian Cai, Zhaoxuan Tan, Zhenyu Lei, Zifeng Zhu, Hongrui Wang, Qinghua Zheng, and Minnan Luo. 2024. LMBot: Distilling Graph Knowledge into Language Model for Graph-less Deployment in Twitter Bot Detection. In *Proceedings of WSDM*. 57–66.

[7] Guido Caldarelli, Rocco De Nicola, Fabio Del Vigna, Marinella Petrocchi, and Fabio Saracco. 2020. The role of bot squads in the political propaganda on Twitter. *Communications Physics* 3, 1 (2020), 81.

[8] Jie Chen, Zilong Li, Yin Zhu, Junping Zhang, and Jian Pu. 2023. From node interaction to hop interaction: New effective and scalable graph learning paradigm. In *Proceedings of IEEE/CVF CVPR*. 7876–7885.

[9] Wen Chen, Diogo Pacheco, Kai-Cheng Yang, and Filippo Menczer. 2021. Neutral bots probe political bias on social media. *Nature Communications* 12, 1 (2021), 5580.

[10] Eli Chien, Chao Pan, Jianhao Peng, and Olgica Milenkovic. 2022. You are AllSet: A Multiset Function Framework for Hypergraph Neural Networks. In *Proceedings of ICLR*.

[11] Zi Chu, Steven Gianvecchio, Haining Wang, and Sushil Jajodia. 2012. Detecting Automation of Twitter Accounts: Are You a Human, Bot, or Cyborg? *IEEE Transactions on Dependable and Secure Computing* 9, 6 (2012), 811–824.

[12] Stefano Cresci, Roberto Di Pietro, Marinella Petrocchi, Angelo Spognardi, and Maurizio Tesconi. 2020. Emergent properties, models, and laws of behavioral similarities within groups of Twitter users. *Computer Communications* 150 (2020), 47–61.

[13] Alexandre Duval and Fragkiskos Malliaros. 2022. Higher-order clustering and pooling for graph neural networks. In *Proceedings of ACM CIKM*. 426–435.

[14] Vijay Prakash Dwivedi, Ladislav Rampášek, Michael Galkin, Ali Parviz, Guy Wolf, Anh Tuan Luu, and Dominique Beaini. 2022. Long range graph benchmark. *Advances in Neural Information Processing Systems* 35 (2022), 22326–22340.

[15] Juan Echeverria and Shi Zhou. 2017. Discovery, Retrieval, and Analysis of the 'Star Wars' Botnet in Twitter. In *Proceedings of ASONAM*. 1–8.

[16] Mohd Fazil, Amit Kumar Sah, and Muhammad Abulaish. 2021. DeepSBD: A Deep Neural Network Model with Attention Mechanism for Social Bot Detection. *IEEE Transactions on Information Forensics and Security* 16 (2021), 4211–4223.

[17] Shangbin Feng, Zhaoxuan Tan, Rui Li, and Minnan Luo. 2022. Heterogeneity-aware Twitter Bot Detection with Relational Graph Transformers. In *Proceedings of AAAI*, Vol. 36. 3977–3985.

[18] Shangbin Feng, Zhaoxuan Tan, Herun Wan, Ningnan Wang, Zilong Chen, Binchi Zhang, Qinghua Zheng, Wenqian Zhang, Zhenyu Lei, Shujie Yang, Xinshun Feng, Qingyue Zhang, Hongrui Wang, Yuhan Liu, Yuyang Bai, Heng Wang, Zijian Cai, Yanbo Wang, Lijing Zheng, Zihan Ma, Jundong Li, and Minnan Luo. 2022. Twibot-22: Towards Graph-based Twitter Bot Detection. *Advances in Neural Information Processing Systems* 35 (2022), 35254–35269.

[19] 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 ACM CIKM*. 3808–3817.

[20] Shangbin Feng, Herun Wan, Ningnan Wang, Jundong Li, and Minnan Luo. 2021. Twibot-20: A comprehensive Twitter bot detection benchmark. In *Proceedings of ACM CIKM*. 4485–4494.

[21] Shangbin Feng, Herun Wan, Ningnan Wang, and Minnan Luo. 2021. BotRGCN: Twitter bot detection with relational graph convolutional networks. In *Proceedings of ASONAM*. 236–239.

[22] Yifan Feng, Haoxuan You, Zizhao Zhang, Rongrong Ji, and Yue Gao. 2019. Hypergraph Neural Networks. In *Proceedings of AAAI*, Vol. 33. 3558–3565.

[23] Emilio Ferrara. 2023. Social bot detection in the age of ChatGPT: Challenges and opportunities. *First Monday* 28, 6 (2023).

[24] Emilio Ferrara, Onur Varol, Clayton Davis, Filippo Menczer, and Alessandro Flammini. 2016. The rise of social bots. *Commun. ACM* 59, 7 (2016), 96–104.

[25] Matthias Fey and Jan E. Lenssen. 2019. Fast Graph Representation Learning with PyTorch Geometric. In *Proceedings of ICLR Workshop on Representation Learning on Graphs and Manifolds*.

[26] Min Gao, Zheng Li, Ruichen Li, Chenhao Cui, Xinyuan Chen, Bodian Ye, Yupeng Li, Weiwei Gu, Qingyuan Gong, Xin Wang, and Yang Chen. 2023. EasyGraph: A Multifunctional, Cross-Platform, and Effective Library for Interdisciplinary Network Analysis. *Patterns* 4, 10 (2023), 100839.

[27] Yue Gao, Yifan Feng, Shuyi Ji, and Rongrong Ji. 2022. HGNN+: General Hypergraph Neural Networks. *IEEE Transactions on Pattern Analysis and Machine Intelligence* 45, 3 (2022), 3181–3199.

[28] William L. Hamilton, Rex Ying, and Jure Leskovec. 2017. Inductive representation learning on large graphs. In *Proceedings of NIPS*. 1025–1035.

[29] Chris Hays, Zachary Schutzman, Manish Raghavan, Erin Walk, and Philipp Zimmer. 2023. Simplistic Collection and Labeling Practices Limit the Utility of Benchmark Datasets for Twitter Bot Detection. In *Proceedings of the ACM Web Conference 2023*. 3660–3669.

[30] Ziniu Hu, Yuxiao Dong, Kuansan Wang, and Yizhou Sun. 2020. Heterogeneous Graph Transformer. In *Proceedings of the Web Conference 2020*. 2704–2710.

[31] Di Huang, Jinbao Song, and Xingyu Zhang. 2025. Semi-Supervised Social Bot Detection with Relational Graph Attention Transformers and Characteristics of the Social Environment. *Information Fusion* (2025), 102956.

[32] Haitao Huang, Hu Tian, Xiaolong Zheng, Xingwei Zhang, Daniel Dajun Zeng, and Fei-Yue Wang. 2024. CGNN: A Compatibility-Aware Graph Neural Network for Social Media Bot Detection. *IEEE Transactions on Computational Social Systems* 11, 5 (2024), 6528–6543.

[33] Jing Huang and Jie Yang. 2021. UniGNN: a Unified Framework for Graph and Hypergraph Neural Networks. In *Proceedings of IJCAI*. 2563–2569.

[34] Xuanwen Huang, Kaiqiao Han, Yang Yang, Dezheng Bao, Quanjin Tao, Ziwei Chai, and Qi Zhu. 2024. Can GNN be Good Adapter for LLMs?. In *Proceedings of the ACM Web Conference 2024*. 893–904.

[35] Bowen Jin, Gang Liu, Chi Han, Meng Jiang, Heng Ji, and Jiawei Han. 2024. Large Language Models on Graphs: A Comprehensive Survey. *IEEE Transactions on Knowledge and Data Engineering* 36, 12 (2024), 8622–8642.

[36] Long Jin, Yang Chen, Tianyi Wang, Pan Hui, and Athanasios V. Vasilakos. 2013. Understanding user behavior in online social networks: a survey. *IEEE Communications Magazine* 51, 9 (2013), 144–150.

[37] Sunwoo Kim, Soo Yong Lee, Yue Gao, Alessia Antelmi, Mirko Polato, and Kijung Shin. 2024. A survey on hypergraph neural networks: an in-depth and step-by-step guide. In *Proceedings of ACM SIGKDD*. 6534–6544.

[38] Diederik P. Kingma and Jimmy Ba. 2015. Adam: A Method for Stochastic Optimization. In *Proceedings of ICLR*.

[39] Thomas N. Kipf and Max Welling. 2017. Semi-Supervised Classification with Graph Convolutional Networks. In *Proceedings of ICLR*.

[40] Geon Lee, Jaemin Yoo, and Kijung Shin. 2023. Mining of Real-world Hypergraphs: Patterns, Tools, and Generators. In *Proceedings of ACM SIGKDD*. 5811–5812.

[41] Feng Liu, Chunfang Yang, Zhenyu Li, Daofu Gong, Rui Ma, and Fenlin Liu. 2023. Accou2vec: A Social Bot Detection Model Based on Community Walk. *IEEE Transactions on Dependable and Secure Computing* (2023), 1–17.

[42] Luotao Liu, Feng Huang, Xuan Liu, Zhankun Xiong, Menglu Li, Congzhi Song, and Wen Zhang. 2023. Multi-view contrastive learning hypergraph neural network for drug-microbe-disease association prediction. In *Proceedings of IJCAI*. 4829–4837.

[43] Yinhan Liu, Myle Ott, Naman Goyal, Jingfei Du, Mandar Joshi, Danqi Chen, Omer Levy, Mike Lewis, Luke Zettlemoyer, and Veselin Stoyanov. 2019. RoBERTa: A Robustly Optimized BERT Pretraining Approach. *arXiv preprint arXiv:1907.11692* (2019).

[44] Yuhan Liu, Zhaoxuan Tan, Heng Wang, Shangbin Feng, Qinghua Zheng, and Minnan Luo. 2023. BotMoE: Twitter bot detection with community-aware mixtures of modal-specific experts. In *Proceedings of ACM SIGIR*. 485–495.

[45] Qingsong Lv, Ming Ding, Qiang Liu, Yuxiang Chen, Wenzheng Feng, Siming He, Chang Zhou, Jianguo Jiang, Yuxiao Dong, and Jie Tang. 2021. Are we really making much progress? revisiting, benchmarking and refining heterogeneous graph neural networks. In *Proceedings of ACM SIGKDD*. 1150–1160.

[46] Nuoyan Lyu, Bingbing Xu, Fangda Guo, and Huawei Shen. 2023. DCGNN: Dual-Channel Graph Neural Network for Social Bot Detection. In *Proceedings of ACM CIKM*. 4155–4159.

[47] Diogo Pacheco. 2024. Bots, Elections, and Controversies: Twitter Insights from Brazil's Polarised Elections. In *Proceedings of the ACM Web Conference*. 2651–2659.

[48] Adam Paszke, Sam Gross, Francisco Massa, Adam Lerer, James Bradbury, Gregory Chanan, Trevor Killeen, Zeming Lin, Natalia Gimelshein, Luca Antiga, Alban Desmaison, Andreas Köpf, Edward Yang, Zach DeVito, Martin Raison, Alykhan Tejani, Sasank Chilamkurthy, Benoit Steiner, Lu Fang, Junjie Bai, and Soumith Chintala. 2019. PyTorch: an imperative style, high-performance deep learning

library. In *Proceedings of NIPS*. 8026–8037.

[49] Jinghua Piao, Jiazhen Liu, Fang Zhang, Jun Su, and Yong Li. 2023. Human–AI adaptive dynamics drives the emergence of information cocoons. *Nature Machine Intelligence* 5, 11 (2023), 1214–1224.

[50] Boyu Qiao, Wei Zhou, Kun Li, Shilong Li, and Songlin Hu. 2024. Dispelling the Fake: Social Bot Detection Based on Edge Confidence Evaluation. *IEEE Transactions on Neural Networks and Learning Systems* (2024), 1–14.

[51] Colin Raffel, Noam Shazeer, Adam Roberts, Katherine Lee, Sharan Narang, Michael Matena, Yanqi Zhou, Wei Li, and Peter J. Liu. 2020. Exploring the limits of transfer learning with a unified text-to-text Transformer. *Journal of Machine Learning Research* 21, 1 (2020), 5485–5551.

[52] Hanan Samet. 2007. K-nearest Neighbor Finding Using MaxNearestDist. *IEEE Transactions on Pattern Analysis and Machine Intelligence* 30, 2 (2007), 243–252.

[53] Mohsen Sayyadiharikandeh, Onur Varol, Kai-Cheng Yang, Alessandro Flammini, and Filippo Menczer. 2020. Detection of novel social bots by ensembles of specialized classifiers. In *Proceedings of ACM CIKM*. 2725–2732.

[54] Michael Schlichtkrull, Thomas N Kipf, Peter Bloem, Rianne Van Den Berg, Ivan Titov, and Max Welling. 2018. Modeling relational data with graph convolutional networks. In *Proceedings of ESWC*. Springer, 593–607.

[55] 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, 4787 (2018).

[56] Wei Shen, Mang Ye, and Wenke Huang. 2024. Resisting Over-Smoothing in Graph Neural Networks via Dual-Dimensional Decoupling. In *Proceedings of ACM MM*. 5800–5809.

[57] Shuhao Shi, Yan Li, Zihao Liu, Chen Chen, Jian Chen, and Bin Yan. 2024. Neighborhood Difference-Enhanced Graph Neural Network Based on Hypergraph for Social Bot Detection. In *Proceedings of the Chinese Conference on Pattern Recognition and Computer Vision (PRCV)*. Springer, 76–90.

[58] Shuhao Shi, Kai Qiao, Zihao Liu, Jie Yang, Chen Chen, Jian Chen, and Bin Yan. 2025. MGTAB: A Multi-Relational Graph-Based Twitter Account Detection Benchmark. *Neurocomputing* 647 (2025), 130490.

[59] Laurens Van der Maaten and Geoffrey Hinton. 2008. Visualizing data using t-SNE. *Journal of Machine Learning Research* 9, 11 (2008), 2759–2605.

[60] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. 2017. Attention is all you need. *Advances in Neural Information Processing Systems* 30 (2017), 5998–6008.

[61] Petar Veličković, Guillem Cucurull, Arantxa Casanova, Adriana Romero, Pietro Liò, and Yoshua Bengio. 2018. Graph Attention Networks. In *Proceedings of ICLR*.

[62] Shaomei Wu, Jake M. Hofman, Winter A. Mason, and Duncan J. Watts. 2011. Who says what to whom on Twitter. In *Proceedings of WWW*. 705–714.

[63] Zhanghao Wu, Paras Jain, Matthew Wright, Azalia Mirhoseini, Joseph E Gonzalez, and Ion Stoica. 2021. Representing long-range context for graph neural networks with global attention. *Advances in Neural Information Processing Systems* 34 (2021), 13266–13279.

[64] Han Xie, Li Xiong, and Carl Yang. 2024. Federated Node Classification over Distributed Ego-Networks with Secure Contrastive Embedding Sharing. In *Proceedings of ACM CIKM*. 2607–2617.

[65] Naganand Yadati, Madhav Nimishakavi, Prateek Yadav, Vikram Nitin, Anand Louis, and Partha Talukdar. 2019. HyperGCN: A new method for training graph convolutional networks on hypergraphs. In *Proceedings of NIPS*. 1511–1522.

[66] Jianwei Yang, Chunyuan Li, Pengchuan Zhang, Xiyang Dai, Bin Xiao, Lu Yuan, and Jianfeng Gao. 2021. Focal attention for long-range interactions in vision Transformers. *Advances in Neural Information Processing Systems* 34 (2021), 30008–30022.

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

[68] Yingguang Yang, Qi Wu, Buyun He, Hao Peng, Renyu Yang, Zhifeng Hao, and Yong Liao. 2024. SEBot: Structural Entropy Guided Multi-View Contrastive Learning for Social Bot Detection. In *Proceedings of ACM SIGKDD*. 3841–3852.

[69] Bodian Ye, Min Gao, Xiu-Xiu Zhan, Xinlei He, Zi-Ke Zhang, Qingyuan Gong, Xin Wang, and Yang Chen. 2025. EasyHypergraph: an open-source software for fast and memory-saving analysis and learning of higher-order networks. *Humanities and Social Sciences Communications* 12, 1291 (2025).

[70] Hanqing Zeng, Muhan Zhang, Yinglong Xia, Ajitesh Srivastava, Andrey Malevich, Rajgopal Kannan, Viktor Prasanna, Long Jin, and Ren Chen. 2021. Decoupling the depth and scope of graph neural networks. *Advances in Neural Information Processing Systems* 34 (2021), 19665–19679.

[71] Jinxue Zhang, Rui Zhang, Yanchao Zhang, and Guanhua Yan. 2016. The rise of social botnets: Attacks and countermeasures. *IEEE Transactions on Dependable and Secure Computing* 15, 6 (2016), 1068–1082.