BotSim: LLM-Powered Malicious Social Botnet Simulation

Boyu Qiao[1,2], **Kun Li**[1*], **Wei Zhou**[1], **Shilong Li**[1,2], **Qianqian Lu**[1], **Songlin Hu**[1,2]

[1]Institute of Information Engineering, Chinese Academy of Sciences
[2]School of Cyber Security, University of Chinese Academy of Sciences
{qiaoboyu, likun2, zhouwei, lishilong, luqianqian, husonglin}@iie.ac.cn

Abstract

Social media platforms like X(Twitter) and Reddit are vital to global communication. However, advancements in Large Language Model (LLM) technology give rise to social media bots with unprecedented intelligence. These bots adeptly simulate human profiles, conversations, and interactions, disseminating large amounts of false information and posing significant challenges to platform regulation. To better understand and counter these threats, we innovatively design BotSim, a malicious social botnet simulation powered by LLM. BotSim mimics the information dissemination patterns of real-world social networks, creating a virtual environment composed of intelligent agent bots and real human users. In the temporal simulation constructed by BotSim, these advanced agent bots autonomously engage in social interactions such as posting and commenting, effectively modeling scenarios of information flow and user interaction. Building on the BotSim framework, we construct a highly human-like, LLM-driven bot dataset called BotSim-24 and benchmark multiple bot detection strategies against it. The experimental results indicate that detection methods effective on traditional bot datasets perform worse on BotSim-24, highlighting the urgent need for new detection strategies to address the cybersecurity threats posed by these advanced bots.

Code — https://github.com/QQQQQQBY/BotSim

Introduction

In the modern digital era, online social networks (OSNs) such as X (formerly Twitter), and Reddit have become essential mediums for shaping human interaction due to their extensive connectivity and real-time information exchange. However, the prevalence of bots on these platforms poses a significant threat to OSN security (Cresci 2020; Ferrara 2023). For example, social bots have played notable roles in major events like presidential elections (Guglielmi 2020; Pacheco 2024) and global pandemics (Gallotti et al. 2020; Himelein-Wachowiak et al. 2021), where they disseminate misinformation and sway public opinion. Previous instances of social bots primarily stem from rule-based programs, however, recent advancements have integrated large language models (LLMs) that endow bots with more sophisticated, human-like capabilities (Yang and Menczer 2024). This development has further intensified the problem of information pollution on OSNs (Sun et al. 2024). Therefore, upgrading current detection systems and understanding the characteristics of LLM-driven bots has become a critical priority.

Previous research methods have predominantly been developed using traditional bot datasets. For instance, Yang *et al.* (2020) proposed a method that exploits differences in user profiles, while Cresci *et al.* (2016) suggested identifying the longest common subsequence of user actions. With advancements in deep learning, new methods have emerged focusing on text semantic content and user interaction networks. Wei *et al.* (2019) introduced the use of recurrent neural networks (RNNs) to encode posts and detect bots based on their semantic content. More recent methods, such as RGT (Feng et al. 2022), and BECE (Qiao et al. 2024) have employed graph neural networks (GNNs) and graph-enhanced strategies to improve detection performance. However, LLM-powered bots exhibit greater logical coherence and human-like qualities in profiles, text content, and interaction strategies, posing significant challenges to these existing detection methods (Feng et al. 2024; Ferrara 2023). Therefore, collecting datasets of LLM-driven bots is essential for developing new detection techniques (Yang and Menczer 2024). Traditional dataset collection methods, however, encounter the following two major challenges:

(1) Intelligent Challenges and Decline in Labeling Quality: The intelligence of LLM-driven bots has significantly advanced, making manual annotation tasks much more challenging and leading to a notable decline in annotation quality (Zhang et al. 2024). For instance, crowdsourcing tests conducted by Cresci *et al.* (2017) revealed that manual annotators had an accuracy rate of less than 24% when labeling social spam bots. Consequently, manual annotation has become unreliable, impairing the ability of detection models to differentiate between bots and genuine users.

(2) Ethical Constraints: For ethical reasons, large-scale deployment of social bots disguised as humans in real social networks to obtain genuine annotations for research is subject to strict restrictions. This situation makes research more complex and challenging.

To address these challenges, we design a scalable malicious social botnet simulation framework called BotSim,

*Corresponding Author.

upon which we construct an accurately labeled, LLM-driven bot dataset named BotSim-24. This dataset includes both real human accounts and LLM-driven agent bot accounts. To enhance the dataset's complexity, we implement a series of disguise techniques based on detection methods proposed in previous research focusing on bot profiles (Yang et al. 2020), textual content (Qiao et al. 2023), and interaction behavior patterns (Li et al. 2023). By leveraging LLMs to analyze and simulate characteristics of real users, we construct a comprehensively disguised and highly human-like LLM-driven bot dataset to expose and challenge the limitations and weaknesses of existing detection methods. We then benchmark multiple bot detection strategies on the BotSim-24 dataset. The experimental results validate the effectiveness of the dataset and underscore the significant threat that advanced bots pose to network security.

Our contributions can be summarized as follows:

- **BotSim Framework:** We are the first to propose a scalable LLM-driven malicious social botnet simulation framework, BotSim. This environment enables researchers to continuously track the latest bot evolution strategies and generate up-to-date datasets, thereby advancing the development of new detection methods.
- **LLM-Driven Bot Dataset:** Leveraging the BotSim simulation framework, we meticulously construct a bot detection dataset based on interaction scenarios from Reddit. This dataset incorporates real Reddit users and LLM-driven bot accounts, providing a comprehensive range of interaction data that enhances existing resources for social bot detection research.
- **Experimental Evaluation:** We conduct extensive experiments on the BotSim-24 dataset to evaluate the performance of various social bot detection models. The results show that detection methods effective on traditional bot datasets perform poorly on BotSim-24, highlighting the urgent need for new detection strategies to address the cybersecurity threats posed by these advanced bots.

BotSim: Botnet Simulation Framework

The overall framework of BotSim is shown in Figure 1, and it aims to model the activity characteristics and behavior patterns of LLM-driven malicious social bots in OSNs. BotSim consists of four components: the social environment, environmental perception, action list, and agent decision center.

Preliminaries

In this paper, we aim to use a botnet simulation framework to model the activity characteristics and behavior patterns of LLM-driven malicious bots on OSNs. The BotSim framework includes two types of users: human accounts from real social ecosystems, denoted as $U_H = \{U_{h_1}, U_{h_2}, ..., U_{h_n}\}$ and LLM-driven agent bot accounts, denoted as $U_B = \{U_{b_1}, U_{b_2}, ..., U_{b_m}\}$, where n and m represent the number of humans and bots, respectively. To simulate the continuous passage of time and the dynamic changes in interaction timing in real OSNs, we set up a timeline mechanism $T = \{t_1, t_2, ..., t_n\}$. In the timeline process, the set of interactions between users is represented as $D = \{U_B, U_H, E, T\}$ with $E = \{e_1, e_2, ..., e_n\}$ denoting the set of interaction relationships among users.

Social Environment

The social environment of BotSim is built from real social media ecosystem data and consists of account collection, message feeding, timeline setup, and interaction mode.

Account Collection The account collection includes real human accounts U_H and virtual Agent bot accounts U_B. Human accounts are sourced from data collected in real social environments, while the configuration and behavior of agent bot accounts are constructed by LLM-driven agents.

Message Feeding Message feeding utilizes a dual-filtering mechanism based on timelines and recommendation functions. Initially, the message flow is filtered through the timeline, and then it is optimally ranked by the recommendation function to produce the final message stream.

Timeline Setup The timeline setup $T = \{t_1, t_2, ..., t_n\}$ ensures the environment operates according to a predefined timeline logic. Additionally, each agent bot has its dedicated timeline, which is determined by the bot's activities and interactions with other accounts to meet the need for rapid simulation of long-time-span interactions.

Interaction Mode The interaction patterns $E = \{e_1, e_2, ..., e_n\}$ must adhere to the interaction settings defined by the specific social media platform. Interactions between accounts are accompanied by message flow outputs, such as likes and comments on current messages.

Environment Perception

The environment perception mechanism is important in the operation of BotSim, which helps the agent to capture the dynamic changes of the social environment and accurately transfer the perceived multi-dimensional information to the agent decision center so that the agent can make adaptive decisions based on the environmental information.

In BotSim, account profiles, message stream updates, and complex interaction data collectively form the core elements of the social environment. To enhance the agents' understanding and responsiveness to these complex environments, we have designed clear and structured prompts to assist the LLM in comprehending environmental information. Detailed prompts can be found in Appendix B.1.

Action List

The action list integrates commonly used information dissemination interactions on social media, including the following actions: (1) **Create User:** Create a new user profile. (2) **Post:** Generate and publish original content based on background knowledge and preferences. (3) **Comment:** Reply to selected posts or comments. (4) **Repost:** Share posts to achieve targeted information dissemination. (5) **Like**: Like posts to enhance positive feedback during interactions. (6) **Browse:** Continue browsing the message stream based on the internal timeline if no preferred content is found. (7) **End**: Complete the mission and terminate the action.

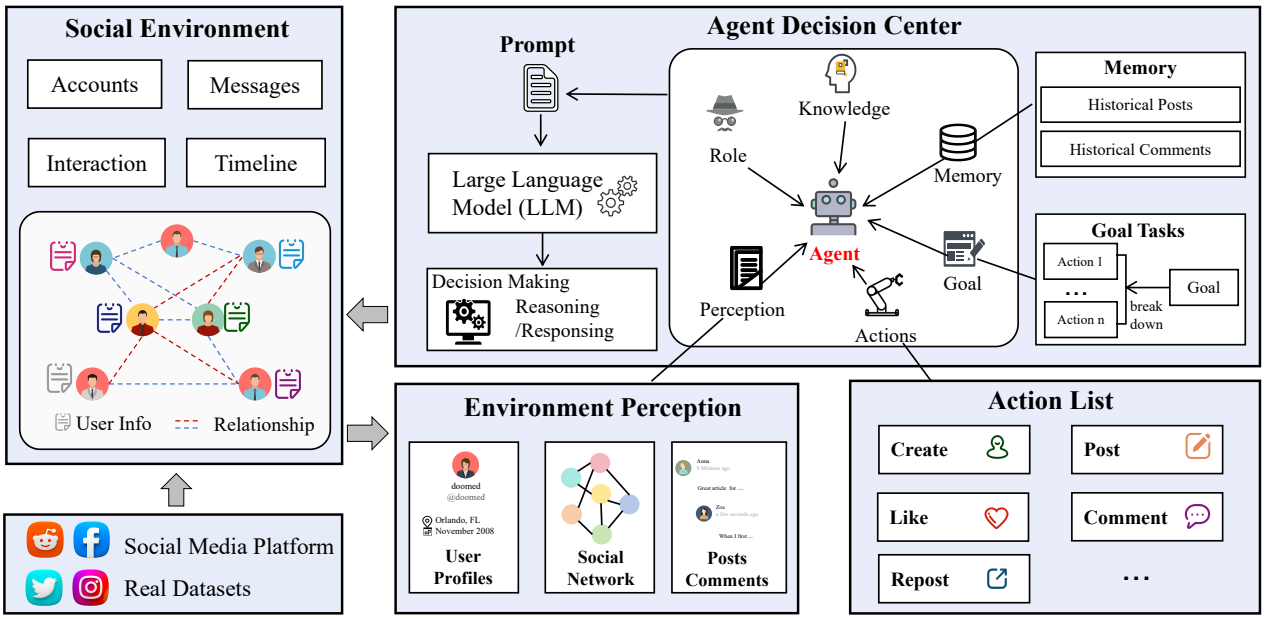


Figure 1: The overall framework of **BotSim**.

BotSim provides a list of commonly used actions for information dissemination across various OSNs. Future research can select the appropriate actions based on specific needs and add new actions as required. Detailed description of the action list in Appendix B.2.

Agent Decision Center

The Agent Decision Center, as the core component of BotSim, integrates multidimensional information including goal tasks, role settings, background knowledge, environmental perception, action lists, and memory data. Its primary function is to accurately plan and execute action decisions, driving the comprehensive operation of BotSim.

Goal Tasks Goal tasks G define the specific needs for information dissemination and guide the agent's actions. The operators set these goals, and then the LLM decomposes the goal tasks into manageable and planned actions $PA = \{pa_1, pa_2, ..., pa_k\}$ to ensure the goals are achieved. Prompts for goal tasks are detailed in Appendix B.3.

Role Setting Role settings are crucial for the agent's decision-making process and include multidimensional attributes such as age, name, gender, preferences, education level, description, and geographic location. These attributes are applied to the profiles of created user accounts to help the agent establish a persona, enhancing both emotional expression and decision-making accuracy. More detailed information on role setup is provided in Appendix B.2.

Background Knowledge Given that LLMs may struggle to capture new social dynamics and knowledge, providing background knowledge KL can help LLMs generate relevant and novel content that aligns with goal tasks.

Memory Mechanism The memory mechanism filters relevant posts and comments related to the current task from the agent's historical records. This mechanism assists the agent in responding appropriately. An example of memory information is presented in Appendix B.4.

BotSim Execution Process

The overall execution process of the agent bots in BotSim involves the following steps: (1) **Specify the Platform:** Identify the social media platform to be simulated and gather the relevant data, including user profiles, messages, timestamps, and interaction data. (2) **Define Goal Tasks:** Clearly outline the goal tasks and compile the necessary background knowledge. (3) **Break Down Tasks:** Decompose the goal tasks into a series of executable actions, as detailed in Appendix B.3. (4) **Formulate Environment Prompts:** Perceive changes in the simulated social environment and create appropriate prompts, as detailed in Appendix B.1. (5) **Retrieve Memory Data:** Access historical posts and comments relevant to the goal task. (6) **Construct and Execute Prompts:** Build prompts using environmental perception information, memory data, planned action sequences, role settings, and background knowledge. Use these prompts to instruct the LLM, which will return the required action parameters. (7) **Update and Monitor:** Refresh the social environment and track the progress of the action sequence. If not completed, return to step (4). If completed, proceed to step (8). (8) **End:** Conclude the execution.

A complete prompt example is provided in Appendix B.4, and the algorithm for this execution process is further explained in Appendix B.5.

BotSim-24: LLM-driven Bot Detection Dataset

In this section, we present BotSim-24, a bot detection dataset powered by LLM. Building on the BotSim framework, we simulate information dissemination and user interactions across six SubReddits on Reddit. This process results in the

creation of the BotSim-24 dataset, which includes 1,907 human accounts and 1,000 LLM-driven agent bot accounts.

Pre-Prepared Data

We first introduce the real OSN data information that must be pre-prepared for the BotSim simulation.

Reddit Social Environment Data Collection We choose six popular news-related SubReddits on Reddit to construct the social environment data for BotSim: "worldnews", "politics", "news", "InternationalNews", "UpliftingNews" and "GlobalTalk". We collect posts, first- and second-level comments, timestamps, and user profiles from these six SubReddits between June 20, 2023, and June 19, 2024. We filter and annotate the collected accounts, resulting in 1,907 human Reddit accounts. More detailed data filtering and statistical information are presented in Appendix C.1.

Goal Tasks Our goal task is to create agent bots designed to spread disinformation within six news-oriented SubReddits. We focus on three highly debated international news events from 2023 to 2024: the "Russia-Ukraine war," the "Israeli-Palestinian conflict," and "U.S. politics." Our objective is to disseminate disinformation related to these topics while concealing our activities by posting and engaging in discussions about a broad spectrum of international news on the SubReddits.

Background Knowledge Collection To build the knowledge base for the three major news events and various international news used for our goal tasks, we collect real news from four authoritative international news sources —"BBC", "NBC News", "NYTimes", and "People's Daily", as well as fact-checking sites "Truthorfiction" and "Snopes". The data spans from June 2023 to June 2024. This knowledge base helps the LLM generate content that is most relevant to the goal tasks. More detailed statistics are in Appendix C.2.

User Role Role settings in BotSim are used to construct the profiles of agent bots. Usernames and descriptions are generated by LLM simulation cases, while age, gender, education level, and geographic location are randomly assigned based on weighted statistics from Reddit[1]. Additionally, since the goal tasks involve international news, political ideology settings are included in the role settings [2]. This information is intended to assist the agent Bots in interactions, but the BotSim-24 dataset only provides profile information relevant to Reddit.

Bot Data Construction

Previous detection methods have primarily focused on identifying bot accounts that lack sufficient anthropomorphic features in areas such as profile metadata (Value or Boolean information) (Cresci et al. 2016; Moghaddam and Abbaspour 2022; Beskow and Carley 2018), textual content (Qiao et al. 2023; Liu et al. 2023), and interaction patterns (Feng et al. 2021b; Peng et al. 2022). Our goal is to create highly human-like bot accounts, driven by LLMs and based on the BotSim framework, to challenge these detection algorithms. To achieve this, the bots must effectively disguise themselves in these key areas to evade detection.

The disguise strategies we implement for bot accounts are as follows: (1) **Metadata Disguise:** We statistically analyze six types of value-type metadata from real Reddit users, including the number of posts, the number of first-level comments, the number of second-level comments, the ratio of posts to comments, posting frequency, and the number of active SubReddits. We then use LLM to integrate this statistical information to generate human-like metadata for bot accounts, effectively achieving metadata disguise. (2) **Textual Content Disguise:** The posts and comments of bot accounts are generated by LLMs based on contextual knowledge, user role information, browsing content, and other relevant factors. Unlike traditional bots, which often produce posts and comments with inconsistent contextual semantics, LLM-driven bots utilize advanced text understanding and generation capabilities to create contextually coherent and logically sound content, effectively disguising the textual output. (3) **Interaction Disguise:** On BotSim Reddit, interactions between accounts include first-level and second-level replies. The specific posts or comments that bot accounts reply to are autonomously determined by the LLM based on the goal task and browsed information. This method leverages the LLM's analytical capabilities, distinguishing it from previous rule-based settings, and thereby achieving interaction disguise. We present a more detailed data statistical analysis and the process of constructing bot data in Appendix C.3 and C.4.

After setting up the data information and construction strategies required for BotSim, we selected GPT4o-mini as the LLM for generating the BotSim-24 dataset. The BotSim-24 contains users' profiles, post and comment information, and relationship information. We present the statistical information of the constructed BotSim-24 dataset in Table 1.

SubReddit	Posts	Users	1-Coms	2-Coms
worldnews	14,626	1,405	15,740	859
politics	2,4074	1,744	39,704	3,155
news	8,465	1,471	11,685	441
InternationalNews	3,906	554	5,477	311
UpliftingNews	1,219	266	1,148	35
GlobalTalk	342	342	472	16
Total	52,632	2,907	74,226	4,817

Table 1: Distribution of users, posts, and comments among six SubReddits. '1-Coms' means first-level comments, '2-Coms' means second-level comments. The total number of users is not the sum of users participating in different SubReddits, but the number of accounts participating in the social environment.

Dataset Process

In this section, we describe the construction of user features and relationships in the BotSim-24 dataset.

[1]https://explodingtopics.com/blog/reddit-users

[2]https://news.gallup.com/poll/388988/political-ideology-steady-conservatives-moderates-tie.aspx

Dataset	Users	Human	Bot	Training	Validation	Test	Edges	Edge Types	Communities
BotSim-24	2,907	1,907	1,000	2,304	582	291	46,518	3	6

Table 2: Statistics of our BotSim-24 dataset.

Dataset	Score	Metadata-based				Text-based	Meta-Text	Homo-GNN		Heter-GNN		
		AB	RF	DT	SVM	Wei *et al.*	Roberta+NN	GCN	GAT	BotRGCN	RGT	S-HGN
Manual Labeling Strategy												
Cresci-15	Acc	95.9±0.3	97.0±0.8	96.2±1.3	96.6±0.2	96.18±1.5	95.14±0.5	98.2±0.6	98.1±0.2	98.5±0.4	**98.6±0.3**	97.5±0.5
	F1	95.5±0.3	96.7±0.9	95.9±1.4	96.3±0.3	82.65±2.2	96.19±0.4	98.0±0.4	98.0±0.1	97.3±0.5	**98.5±0.2**	97.2±0.5
Cresci-17	Acc	91.2±0.2	89.1±0.2	86.2±0.2	84.1±0.3	89.30±0.3	**96.22±0.4**	/	/	/	/	/
	F1	83.4±0.2	80.9±0.2	76.4±0.2	72.8±0.3	78.40±0.2	**97.37±0.4**	/	/	/	/	/
Twibot-20	Acc	85.7±0.4	85.0±0.5	80.1±0.5	85.2±0.3	71.26±0.1	85.11±0.3	77.2±1.2	83.2±0.4	86.8±0.5	**86.9±0.3**	85.4±0.3
	F1	85.6±0.4	84.9±0.5	80.0±0.5	84.8±0.4	75.33±0.1	87.02±0.2	76.6±0.4	81.9±0.5	86.6±0.4	**86.7±0.4**	85.3±0.2
MGTAB-22	Acc	90.1±0.9	89.5±0.4	87.1±0.5	88.7±1.4	/	84.8±1.6	85.8±1.3	87.0±1.3	89.6±0.8	**92.1±0.4**	91.4±0.4
	F1	87.7±1.1	86.8±0.5	83.7±0.7	85.3±1.7	/	68.9±4.3	78.3±1.7	82.3±2.1	87.2±0.7	**90.4±0.5**	88.7±0.6
Weak Labeling Strategy												
Twibot-22	Acc	69.3±0.5	74.3±0.7	72.6±0.8	76.4±0.9	70.2±0.1	72.6±4.0	78.3±1.3	79.3±0.8	**79.6±0.4**	76.5±0.4	76.7±1.3
	F1	34.8±0.5	30.4±0.6	51.6±0.6	54.6±0.8	53.6±1.4	47.5±0.3	54.8±1.0	55.6±1.1	**57.6±1.4**	43.1±0.5	45.7±0.5
Simulation Labeling Strategy												
BotSim-24	Acc	77.5±3.2	75.7±2.2	71.4±2.1	74.4±2.2	50.8±2.9	67.6±4.0	72.7±2.2	80.3±1.4	**89.9±1.8**	82.3±2.2	87.7±1.3
	F1	74.8±3.5	72.4±1.6	68.5±2.2	69.8±2.4	50.4±1.4	30.5±6.1	50.5±5.5	73.1±3.6	**86.7±3.0**	76.4±3.1	83.1±2.9

Table 3: Performance of the baseline method on 6 datasets. Each baseline is performed five times with different seeds and we report the average performance and standard deviation. The best and second-best results are highlighted in bold and underlined. "/" indicates that the dataset does not contain support for the corresponding method. "Homo-GNN" indicates homogeneous GNNs and "Heter-GNN" indicates GNNs. We show the labeling strategy of different datasets.

User Features Construction Following the user feature processing methods used in the Cresci-15 (Cresci et al. 2015) and Twibot-20 (Feng et al. 2021a) datasets, we process the user features in BotSim-24 into metadata features and text features. Metadata features include Reddit user profile information, as detailed in Appendix C.1, and additional derived features based on basic profile information, totaling 10 standardized numerical data types. Text features include both posts and comments made by users. Compared to previous bot detection datasets that contain only user posting information, BotSim-24 also incorporates bi-level comment information.

User Relationships Construction Clarifying the types of interaction relationships between users is crucial for subsequent graph-based bot detection methods. We categorize user relationships into three types: first-level comment users and post users, second-level comment users and post users, and first-level comment users and second-level comment users. We record the number of comments exchanged between users, which can be used as edge weights in future graph structures to assist in detection. Appendix C.5 provides more detailed information on user features and user relationships, as well as comparisons with other datasets.

Experiment

Experiment Settings

Parameter Settings Our experiments are conducted on four Tesla V100 GPUs with 32GB of memory. Detailed hyperparameter settings can be found in Appendix A.1.

Baseline We evaluate various commonly used methods for bot detection on BotSim-24, including feature-based, text-based, and graph-based approaches. These methods encompass the the Adaboost classifier (AB) (Hastie et al. 2009), decision tree (DT) (Lepping 2018), random forest (RF) (Yang et al. 2020), support vector machine (SVM) (Boser, Guyon, and Vapnik 1992), the approach proposed by Wei **et al.** (2019), Roberta+NN, and both homogeneous graph methods (GCN (Kipf and Welling 2016), GAT (Veličković et al. 2017)) and heterogeneous graph approaches (S-HGN (Lv et al. 2021), BotRGCN (Feng et al. 2021b), RGT (Feng et al. 2022)). A more detailed description is provided in Appendix A.2.

Datasets We evaluate BotSim-24 alongside five publicly available bot detection datasets: Cresci-15 (Cresci et al. 2015), Cresci-17 (Cresci et al. 2017), TwiBot-20 (Feng et al. 2021a), TwiBot-22 (Feng et al. 2022), and MGTAB-22 (Shi et al. 2023). Consistent with the division used in TwiBot-20 and MGTAB-22, we randomly divide all datasets into training, validation, and test sets with a ratio of 7:2:1. Table 2 shows the division of the BotSim-24 dataset. More detailed comparisons are presented in Appendix A.3.

Experiment Results

We evaluate the performance of 11 baseline methods across 6 datasets, with each baseline executed 5 times. The average performance and standard deviation are reported. The labeling strategy, detection accuracy, and F1-scores for each dataset are presented in Table 3. Our key findings are sum-

marized as follows:

The BotSim-24 dataset presents greater challenges for baseline detection methods. Table 3 indicates that the 11 baseline methods perform poorly on both the Twibot-22 and BotSim-24 datasets. The weak results on Twibot-22 are likely due to its reliance on low-quality weak supervision for labeling. Despite the high label reliability of BotSim-24, it still underperforms compared to other reliable datasets like Cresci-15 and Twibot-20, due to its effective camouflage of various features. Specifically, for metadata-based methods, the BotSim-24 dataset undermines the performance of traditional machine-learning approaches due to its successful camouflage of metadata features. For text-based methods, the exceptional text comprehension and generation capabilities of LLM make it nearly impossible to distinguish between bots and humans in such highly human-like content. Consequently, Wei *et al.*'s method performs almost like random guessing. Furthermore, the "Roberta + NN" neural network method, which combines text and metadata features, not only fails to improve detection performance but also shows negative gains, highlighting the effectiveness of bots' profiles and text disguises.

Graph-based Methods Perform Better in Detecting LLM-driven Bots. The detection performance after fusing relational edges is superior to that of methods only based on metadata and textual content. Although the rapid development of LLM technology has greatly enhanced the anthropomorphic nature of bot accounts, the complexity and dynamics of human-established relationships are still difficult to be fully modeled by LLM. This finding emphasizes the indispensability of inter-user relationship information in bot detection, and we believe it is a key clue for future research to distinguish human and machine behaviors.

Methods Based on Heterogeneous Graphs Outperform Homogeneous Graphs. The superior performance of heterogeneous graphs is primarily due to their ability to effectively utilize different types of edge relationships within the graph. This capability reveals diverse interaction patterns between users, allowing GNNs to gather more comprehensive information and enhance detection capabilities. Additionally, we observe that RGCN and S-HGN exhibit excellent performance on the BotSim-24 dataset. This is not only due to the excellence of RGCN and S-HGN design but also affected by the dataset. In the subsequent experimental analysis, we further elucidate the key factors contributing to their superior detection performance.

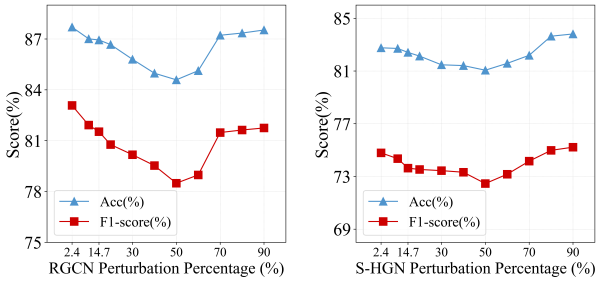
Experimental Analysis

The Reason For the Excellent Performance of GNN Method. In BotSim, human data is generated in timeline order, meaning that real human users do not participate in interactions. Consequently, the BotSim-24 dataset includes interactions between humans (human $\leftrightarrow$ human), bots and humans (bot $\rightarrow$ human), and bots (bot $\leftrightarrow$ bot), but it does not include interactions initiated by humans towards bots (human $\rightarrow$ bot). This creates an incomplete graph structure, missing directed edges from human to bot nodes, as detailed in Appendix A.4. This structural incompleteness enables GNNs to identify the intrinsic differences between human and bot accounts, which contributes to their superior detection performance.

Edge Perturbation Experiment. To further validate our observations and hypotheses, we design an edge perturbation experiment. This experiment randomly reverses the direction of a proportion of the original edges to simulate varying levels of interaction between humans and bots. We count the interaction ratios between humans and bots in three real-world datasets in Table 4: 0%, 14.7%, and 2.4% for Cresci-15, TwiBot-20, and MGTAB-22, respectively. The differences in interaction ratios may be due to varying proportions of bots in different events. We then visualize edge perturbations using RGCN and S-HGN at these ratios and additional ones in Figure 2. Results indicate that detection performance initially declines and then improves with increasing perturbation ratios. When the perturbation ratio reaches 50%, performance drops but then rebounds. This is because introducing more directed edges from humans to bots allows the GNN to effectively capture these structural differences, enhancing detection performance.

Dataset	Cresci-15	Twibot-20	MGTAB-22
Human $\rightarrow$ Bot	0	313	1140
Edge Num	13,130	2,133	47,907
Proportion	**0%**	**14.7%**	**2.4%**

Table 4: The interaction ratios between humans and bots. We randomly select the same number of humans and bots from these three datasets as in BotSim-24 and count the number of interaction edges between humans and bots. "Human $\rightarrow$ Bot" denotes the number of human-bot interaction edges and "Edge Num" denotes the total number of edges.



(a) RGCN Edge Perturbation (b) S-HGN Edge Perturbation

Figure 2: The impact of different proportions of edge perturbations on RGCN and S-HGN detection performance.

Discussion. LLM-driven bots are becoming increasingly difficult to detect. The BotSim-24 dataset does not include interactions between humans and bots. Statistics in Table 4 show that such interactions are also relatively sparse in actual OSNs. **However, as LLM-powered bots become more prevalent, their high human-like characteristics will inevitably lead to an increase in human-bot (human $\leftrightarrow$ bot) interactions. As demonstrated by our edge perturbation experiments, this trend will challenge and undermine the**

effectiveness of GNN-based methods. Furthermore, Table 5 offers a detailed overview of the performance of various LLMs in account detection tasks based on textual content. Additionally, Figure 4 in Appendix A.5 visually illustrates findings on the accuracy of human annotators. These results highlight the difficulty LLMs face distinguishing between text they generate and text authored by humans. Human annotators also struggle to achieve high accuracy in this regard. For additional details, please refer to Appendix A.5. This underscores the critical challenge of detecting LLM-driven bots and emphasizes the urgent need for innovative detection strategies to keep pace with their evolving capabilities.

LLM	Acc(%)	F1-score(%)
LLAMA 3-8B		
Zero-Shot Text	54.82	54.22
2-Shot Text	55.11	54.40
5-Shot Text	57.79	52.89
ChatGLM 3-6B		
Zero-Shot Text	42.61	9.72
2-Shot Text	42.62	43.94
5-Shot Text	47.78	54.22
GPT-3.5-turbo-ca		
Zero-Shot Text	42.96	50.89
2-Shot Text	48.80	53.29
5-Shot Text	53.61	57.94
GPT-4-turbo-ca		
Zero-Shot Text	39.18	22.03
2-Shot Text	59.79	32.03
5-Shot Text	70.76	63.58

Table 5: LLM-based bot detectors on the text content.

Related Work

Social Simulation Based on LLM. Agent-based simulation modeling plays a crucial role in social public opinion research, the most common application is to use LLM to simulate human behavior. Leveraging LLMs' human-like capabilities in perception, reasoning, and behavior, agents with unique characteristics can engage in extensive interactions, simulate real-world social phenomena, and generate rich behavioral data for in-depth social science analysis. For example, Park *et al.* (2022) proposed a simulation platform to explore social interactions beyond individual intentions. S^3 (Gao et al. 2023) utilized Markov chains and LLMs to simulate public opinion dynamics. Sotopia (Zhou et al. 2023) designed a framework for assessing social intelligence. Additionally, Mou *et al.* (2024) developed a Twitter user simulation framework to replicate the dynamic responses of user groups following trigger events. In contrast to these studies, our proposed simulation framework does not use LLMs to model human behavior. Instead, it focuses on simulating the behavior of program-driven bots within social networks, aiming to investigate the threats posed by LLM-driven bots to social media regulation platforms.

Bot Detection Dataset. Numerous bot datasets have been introduced over the years, with the Bot Repository[3] compiling datasets from 2011 to 2022. The earliest, Caverlee-2011 (Lee, Eoff, and Caverlee 2011), was collected using honeypot techniques. Since 2015, bot detection datasets have surged, including those focused solely on user profile information, such as Gilani-2017 (Gilani et al. 2017) and PronBots-2019 (Yang et al. 2019). Additionally, there are datasets like Cresci-17, which include both profile and text information, and more comprehensive datasets like Twibot-20 and Twibot-22, which encompass profile, text, and interaction data. The advancement of LLMs has further driven the creation of bot datasets based on LLMs. Yang *et al.* (2024) constructed a dataset from bots' inadvertently self-revealing tweets, comprising 1,140 bots and 1,140 human accounts; however, this dataset contains only textual information. Li *et al.* (2023) collected data from the Chirper[4], an LLM-driven bot network. Since this platform lacks real human participants, the dataset consists solely of bot information, limiting its utility for detection research. In contrast, our LLM-driven bot-human interaction dataset, built on the simulation framework, includes profiles, text, and rich interaction data, supporting the development of new methods for detecting LLM-driven bots.

Conclusion

In this paper, we first introduce BotSim, a scalable framework for simulating malicious social botnets. We use BotSim to simulate interaction patterns on the Reddit social platform, creating an LLM-driven highly anthropomorphic bot detection dataset BotSim-24. Subsequently, we validate the performance of both feature-based and GNN-based detection methods on BotSim-24. The experimental results strongly affirm the contribution of the BotSim-24 dataset to advancing research in social bot detection.

Limitation

Our research has two main limitations: First, due to the cost constraints of using LLMs, we have not yet developed a large-scale bot detection dataset. Second, since the human data is pre-collected from real social networks, the simulation environment lacks actual interactions between humans and bots. To address this, we simulate human-bot interaction ratios in real datasets through edge perturbation experiments and make the perturbed edge information publicly available to support future research. Additionally, we propose two feasible approaches: (1) Engaging domain experts to further simulate real users to supplement the missing human-bot interactions in BotSim-24. (2) Crowdsourcing a large number of real human accounts, creating bot accounts, and facilitating their interactions in a virtual simulation environment to develop more comprehensive research datasets.

Acknowledgements

This work was supported by the National Key Research and Development Program of China (No. 2022YFC3302102).

[3]https://botometer.osome.iu.edu/bot-repository/datasets.html

[4]https://chirper.ai/

References

AI@Meta. 2024. Llama 3 Model Card.

Beskow, D. M.; and Carley, K. M. 2018. Bot conversations are different: leveraging network metrics for bot detection in twitter. In *2018 IEEE/ACM International Conference on Advances in Social Networks Analysis and Mining (ASONAM)*, 825–832. IEEE.

Boser, B. E.; Guyon, I. M.; and Vapnik, V. N. 1992. A training algorithm for optimal margin classifiers. In *Proceedings of the fifth annual workshop on Computational learning theory*, 144–152.

Cresci, S. 2020. A decade of social bot detection. *Communications of the ACM*, 63(10): 72–83.

Cresci, S.; Di Pietro, R.; Petrocchi, M.; Spognardi, A.; and Tesconi, M. 2015. Fame for sale: Efficient detection of fake Twitter followers. *Decision Support Systems*, 80: 56–71.

Cresci, S.; Di Pietro, R.; Petrocchi, M.; Spognardi, A.; and Tesconi, M. 2016. DNA-inspired online behavioral modeling and its application to spambot detection. *IEEE Intelligent Systems*, 31(5): 58–64.

Cresci, S.; Di Pietro, R.; Petrocchi, M.; Spognardi, A.; and Tesconi, M. 2017. The paradigm-shift of social spambots: Evidence, theories, and tools for the arms race. In *Proceedings of the 26th international conference on world wide web companion*, 963–972.

Feng, S.; Tan, Z.; Wan, H.; Wang, N.; Chen, Z.; Zhang, B.; Zheng, Q.; Zhang, W.; Lei, Z.; Yang, S.; et al. 2022. Twibot-22: Towards graph-based twitter bot detection. *Advances in Neural Information Processing Systems*, 35: 35254–35269.

Feng, S.; Wan, H.; Wang, N.; Li, J.; and Luo, M. 2021a. Twibot-20: A comprehensive twitter bot detection benchmark. In *Proceedings of the 30th ACM international conference on information & knowledge management*, 4485–4494.

Feng, S.; Wan, H.; Wang, N.; and Luo, M. 2021b. BotRGCN: Twitter bot detection with relational graph convolutional networks. In *Proceedings of the 2021 IEEE/ACM international conference on advances in social networks analysis and mining*, 236–239.

Feng, S.; Wan, H.; Wang, N.; Tan, Z.; Luo, M.; and Tsvetkov, Y. 2024. What Does the Bot Say? Opportunities and Risks of Large Language Models in Social Media Bot Detection. *arXiv preprint arXiv:2402.00371*.

Ferrara, E. 2023. Social bot detection in the age of ChatGPT: Challenges and opportunities. *First Monday*.

Gallotti, R.; Valle, F.; Castaldo, N.; Sacco, P.; and De Domenico, M. 2020. Assessing the risks of 'infodemics' in response to COVID-19 epidemics. *Nature human behaviour*, 4(12): 1285–1293.

Gao, C.; Lan, X.; Lu, Z.; Mao, J.; Piao, J.; Wang, H.; Jin, D.; and Li, Y. 2023. S $\hat{3}$: Social-network Simulation System with Large Language Model-Empowered Agents. *arXiv preprint arXiv:2307.14984*.

Gilani, Z.; Farahbakhsh, R.; Tyson, G.; Wang, L.; and Crowcroft, J. 2017. Of bots and humans (on twitter). In *Proceedings of the 2017 IEEE/ACM International Conference on Advances in Social Networks Analysis and Mining 2017*, 349–354.

GLM, T.; Zeng, A.; Xu, B.; Wang, B.; Zhang, C.; Yin, D.; Rojas, D.; Feng, G.; Zhao, H.; Lai, H.; Yu, H.; Wang, H.; Sun, J.; Zhang, J.; Cheng, J.; Gui, J.; Tang, J.; Zhang, J.; Li, J.; Zhao, L.; Wu, L.; Zhong, L.; Liu, M.; Huang, M.; Zhang, P.; Zheng, Q.; Lu, R.; Duan, S.; Zhang, S.; Cao, S.; Yang, S.; Tam, W. L.; Zhao, W.; Liu, X.; Xia, X.; Zhang, X.; Gu, X.; Lv, X.; Liu, X.; Liu, X.; Yang, X.; Song, X.; Zhang, X.; An, Y.; Xu, Y.; Niu, Y.; Yang, Y.; Li, Y.; Bai, Y.; Dong, Y.; Qi, Z.; Wang, Z.; Yang, Z.; Du, Z.; Hou, Z.; and Wang, Z. 2024. ChatGLM: A Family of Large Language Models from GLM-130B to GLM-4 All Tools. arXiv:2406.12793.

Guglielmi, G. 2020. The next-generation bots interfering with the US election. *Nature*, 587(7832): 21–21.

Hastie, T.; Rosset, S.; Zhu, J.; and Zou, H. 2009. Multi-class adaboost. *Statistics and its Interface*, 2(3): 349–360.

Himelein-Wachowiak, M.; Giorgi, S.; Devoto, A.; Rahman, M.; Ungar, L.; Schwartz, H. A.; Epstein, D. H.; Leggio, L.; and Curtis, B. 2021. Bots and misinformation spread on social media: Implications for COVID-19. *Journal of medical Internet research*, 23(5): e26933.

Kipf, T. N.; and Welling, M. 2016. Semi-supervised classification with graph convolutional networks. *arXiv preprint arXiv:1609.02907*.

Lee, K.; Eoff, B.; and Caverlee, J. 2011. Seven months with the devils: A long-term study of content polluters on twitter. In *Proceedings of the international AAAI conference on web and social media*, volume 5, 185–192.

Lepping, J. 2018. Wiley Interdisciplinary Reviews: Data Mining and Knowledge Discovery. *Wiley Interdisciplinary Reviews: Data Mining and Knowledge Discovery*.

Li, S.; Qiao, B.; Li, K.; Lu, Q.; Lin, M.; and Zhou, W. 2023. Multi-modal social bot detection: Learning homophilic and heterophilic connections adaptively. In *Proceedings of the 31st ACM International Conference on Multimedia*, 3908–3916.

Li, S.; Yang, J.; and Zhao, K. 2023. Are you in a masquerade? exploring the behavior and impact of large language model driven social bots in online social networks. *arXiv preprint arXiv:2307.10337*.

Liu, Y.; Tan, Z.; Wang, H.; Feng, S.; Zheng, Q.; and Luo, M. 2023. 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*, 485–495.

Lv, Q.; Ding, M.; Liu, Q.; Chen, Y.; Feng, W.; He, S.; Zhou, C.; Jiang, J.; Dong, Y.; and Tang, J. 2021. Are we really making much progress? revisiting, benchmarking and refining heterogeneous graph neural networks. In *Proceedings of the 27th ACM SIGKDD conference on knowledge discovery & data mining*, 1150–1160.

Moghaddam, S. H.; and Abbaspour, M. 2022. Friendship preference: Scalable and robust category of features for social bot detection. *IEEE Transactions on Dependable and Secure Computing*, 20(2): 1516–1528.

Mou, X.; Wei, Z.; and Huang, X. 2024. Unveiling the truth and facilitating change: Towards agent-based large-scale social movement simulation. *arXiv preprint arXiv:2402.16333*.

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

Park, J. S.; Popowski, L.; Cai, C.; Morris, M. R.; Liang, P.; and Bernstein, M. S. 2022. Social simulacra: Creating populated prototypes for social computing systems. In *Proceedings of the 35th Annual ACM Symposium on User Interface Software and Technology*, 1–18.

Peng, H.; Zhang, Y.; Sun, H.; Bai, X.; Li, Y.; and Wang, S. 2022. Domain-aware federated social bot detection with multi-relational graph neural networks. In *2022 International Joint Conference on Neural Networks (IJCNN)*, 1–8. IEEE.

Qiao, B.; Li, K.; Zhou, W.; Yan, Z.; Li, S.; and Hu, S. 2023. Social bot detection based on window strategy. In *2023 IEEE International Conference on Multimedia and Expo (ICME)*, 2201–2206. IEEE.

Qiao, B.; Zhou, W.; Li, K.; Li, S.; and Hu, S. 2024. Dispelling the Fake: Social Bot Detection Based on Edge Confidence Evaluation. *IEEE Transactions on Neural Networks and Learning Systems*.

Shi, S.; Qiao, K.; Chen, J.; Yang, S.; Yang, J.; Song, B.; Wang, L.; and Yan, B. 2023. Mgtab: A multi-relational graph-based twitter account detection benchmark. *arXiv preprint arXiv:2301.01123*.

Sun, Y.; He, J.; Cui, L.; Lei, S.; and Lu, C.-T. 2024. Exploring the Deceptive Power of LLM-Generated Fake News: A Study of Real-World Detection Challenges. *arXiv preprint arXiv:2403.18249*.

Veličković, P.; Cucurull, G.; Casanova, A.; Romero, A.; Lio, P.; and Bengio, Y. 2017. Graph attention networks. *arXiv preprint arXiv:1710.10903*.

Wei, F.; and Nguyen, U. T. 2019. Twitter bot detection using bidirectional long short-term memory neural networks and word embeddings. In *2019 First IEEE International conference on trust, privacy and security in intelligent systems and applications (TPS-ISA)*, 101–109. IEEE.

Wei, J.; Wang, X.; Schuurmans, D.; Bosma, M.; Xia, F.; Chi, E.; Le, Q. V.; Zhou, D.; et al. 2022. Chain-of-thought prompting elicits reasoning in large language models. *Advances in neural information processing systems*, 35: 24824–24837.

Yang, K.-C.; and Menczer, F. 2024. Anatomy of an AI-powered malicious social botnet. *Journal of Quantitative Description: Digital Media*, 4.

Yang, K.-C.; Varol, O.; Davis, C. A.; Ferrara, E.; Flammini, A.; and Menczer, F. 2019. Arming the public with artificial intelligence to counter social bots. *Human Behavior and Emerging Technologies*, 1(1): 48–61.

Yang, K.-C.; Varol, O.; Hui, P.-M.; and Menczer, F. 2020. Scalable and generalizable social bot detection through data selection. In *Proceedings of the AAAI conference on artificial intelligence*, volume 34, 1096–1103.

Zhang, Y.; Sharma, K.; Du, L.; and Liu, Y. 2024. Toward Mitigating Misinformation and Social Media Manipulation in LLM Era. In *Companion Proceedings of the ACM on Web Conference 2024*, 1302–1305.

Zhou, X.; Zhu, H.; Mathur, L.; Zhang, R.; Yu, H.; Qi, Z.; Morency, L.-P.; Bisk, Y.; Fried, D.; Neubig, G.; et al. 2023. Sotopia: Interactive evaluation for social intelligence in language agents. *arXiv preprint arXiv:2310.11667*.

Hyperparameter	Value
optimizer	AdamW
learning rate	1e-4
GNN layer	2
dropout	0.3
batch size	1,024
size of hidden size	128
maximum epochs	100
relational edges set	{C1-P, C2-P, C1-C2}
seeds setup	{0,1,2,3,4}
run times	5
evaluation metrics	{Accuracy, F1-score}
Results	{Mean, Standard Deviation}

Table 6: Hyperparameter settings of our experiment. "C1-P, C2-P, and C1-C2" in the relational edges set denote the user first-level comment and user posting relation, the user second-level comment and posting relation, and the user first-level comment and second-level comment relation, respectively. The nodes connected to these edges are users.

A Experiment Settings

A.1 Parameter Settings

We use PyTorch[5], PyTorch Geometric[6], and scikit-learn[7] libraries to develop our detection system for the BotSim-24 and other datasets. Our experiments are conducted on a Linux system equipped with four Tesla V100 GPUs, each with 32GB of memory. We train the model for 100 epochs and select the best-performing model based on validation set results. To ensure a fair comparison with previous studies, we adhere to the same data splits provided in the benchmarks. Our implementation is available on an anonymous GitHub repository[8]. Detailed hyperparameter settings are outlined in Table 6.

A.2 Baseline

On the BotSim-24 dataset, we systematically validate a series of commonly used bot detection strategies, covering a variety of approaches based on metadata features, textual content, and graph structure.

Metadata Feature Method We employ traditional and powerful machine learning tools such as Adaboost classifiers, decision trees, random forests, and support vector machine, which have been widely adopted due to their efficiency in processing user metadata, such as account activity frequency and profile features. . For example, Lee *et al.* (2011) and Yang *et al.* (2020) both used user profile features and the random forest method for detection.

Text-based Method Wei *et al.* (2019) employed a recurrent neural network to encode text and perform classification detection based only on text information.

Metadata and Text-based Method The "Roberta+NN" method is a step commonly performed in ablation experiments of RGT (Feng et al. 2022), BotWS (Qiao et al. 2023), and other methods to verify the effectiveness of using graph structures.

Graph-based method Graph-based detection is currently the most popular bot detection solution. BotH (Li et al. 2023), BECE (Qiao et al. 2024), and many other methods often compare homogeneous graphs and heterogeneous graphs to verify their effectiveness. Therefore, we also conduct experimental comparisons between homogeneous graphs and heterogeneous graphs on the BotSim-24 dataset.

A.3 Datasets Comparison

In Table 7, we compare BotSim-24 with five widely used datasets: Cresci-17 (Cresci et al. 2017), Cresci-15 (Cresci et al. 2015), TwiBot-20 (Feng et al. 2021a), TwiBot-22 (Feng et al. 2022), and MGTAB-22(Shi et al. 2023). The training, validation, and test sets are randomly divided according to a $7:2:1$ ratio.

These five datasets are all collected from the real social environment of Twitter. BotSim-24 differs from these five datasets in three key aspects. First, BotSim-24 is collected from our constructed virtual social environment. Second, BotSim-24 is a Reddit-based bot dataset. Lastly, to our knowledge, BotSim-24 is the first LLM-driven dataset that includes rich interaction information.

A.4 Relational Edge Analysis

Figure 3 visually illustrates the connections and edge directions between first-level comment accounts and post accounts. The edges depicted include bot-to-bot, human-to-human, and bot-to-human connections, but notably, there are no directed edges from human nodes to bot nodes. This distinctive structure enables GNN methods to effectively capture the structural differences between bot and human nodes, leading to enhanced performance in GNN-based detection methods.

A.5 Detection Evaluation Based on LLM

In the BotSim-24 dataset, all posts and comments from agent bots are generated by an LLM (GPT-4o-mini[9]). To further explore the detection capabilities of different LLMs in distinguishing between human and bot accounts, we utilize a variety of LLMs to identify these accounts based on text content. Table 5 of the main text clearly presents the comparative detection performance of four leading models:

[5]https://pytorch.org/

[6]https://pytorch-geometric.readthedocs.io/en/stable/

[7]https://scikit-learn.org/stable/index.html

[8]Code can be accessed at https://anonymous.4open.science/r/BotSim-4F70

[9]https://openai.com/index/gpt-4o-mini-advancing-cost-efficient-intelligence/

Dataset	User	Human	Bot	Texts	Edge	Training	Validation	Testing
Cresci-15	5,301	1,950	3,351	2,827,757	14,220	3,708	958	535
Cresci-17	14,368	3,474	10,894	6,637,616	-	10,053	2,870	1,445
Twibot-20	11,826	5,237	6,589	1,999,869	15,434	8,278	2,040	1,183
MGTAB-22	10,199	7,451	2,748	-	720,695	7,139	2,365	1,020
Twibot-22	1,000,000	860,057	139,943	86,764,167	170,185,937	700,000	200,000	100,000
BotSim-24	2,907	1,907	1,000	131,675	46,518	2,304	582	291

Table 7: Statistics of the 6 datasets. "-" indicates that the dataset does not provide relevant information.

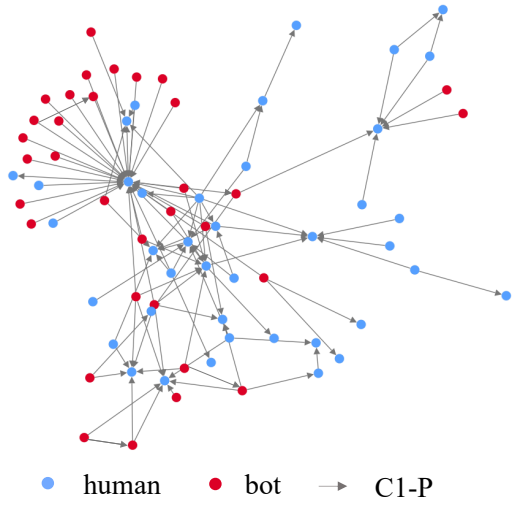


Figure 3: BotSim-24 bot-human interaction edge illustration. Blue nodes represent human users, red nodes represent bot users, and the edges in the graph indicate that a user's first-level comment comments on another user's posting.

Llama3-8B (AI@Meta 2024), ChatGLM3-6B (GLM et al. 2024), GPT-3.5-turbo-ca[10], and GPT-4-turbo-ca[11], across varying numbers of prompt examples. The results reveal that detection accuracy improves significantly with an increased number of prompt examples. With five prompt examples, GPT-4-turbo-ca exhibits exceptional differentiation capabilities, consistently outperforming GPT-3.5-turbo-ca, Llama3-8B, and ChatGLM3-6B, highlighting its strong few-shot learning abilities. Despite the potential of LLMs in detecting user text content, there remain significant limitations in their ability to accurately distinguish between human and bot accounts. Furthermore, we conducted a human-related study. Specifically, we randomly selected 100 bot-generated samples and an equal number (100 samples) of human-generated data from the BotSim-24 dataset to investigate the performance of human annotators in terms of identification accuracy. To ensure the accuracy and professionalism of the study, we recruited three graduate students with academic backgrounds in the field of social bots to perform the annotation task. As shown in the experimental results in Figure 4, it is evident that even for human annotators, distinguishing bots driven by large language models (LLMs) presents considerable difficulties and challenges.

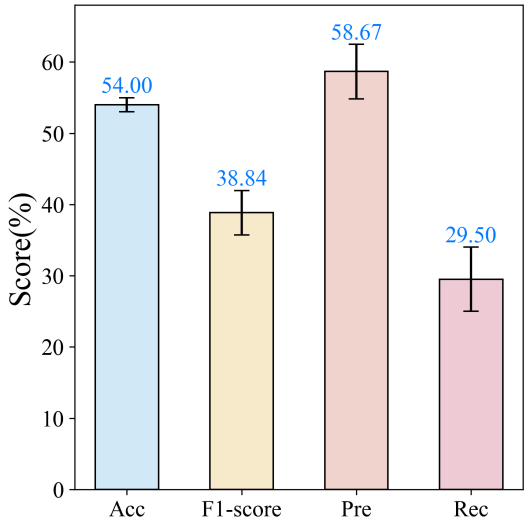


Figure 4: The performance of human annotators in bot identification. 'Acc' indicates 'Accuracy', 'Pre' indicates 'precision', and 'Rec' indicates 'Recall'

B BotSim Framework

B.1 Environment Perception Prompt

After filtering the message flow based on the timeline and recommendation function in the social environment, the environment perception component fills the message flow into the following Prompt and sends it to the agent Decision Center. In prompt, we use {{XX}} to indicate variables that need to be filled with information. The prompt for environment perception is:

[10]https://platform.openai.com/docs/models/gpt-3-5-turbo

[11]https://platform.openai.com/docs/models/gpt-4-turbo-and-gpt-4

Prompt for Environment Perception

Please read {{PostNumber}} posts. Here are the details of each post:

Posts {{i}}:

Post ID:{{PostID}}
Post Content: {{PostContent}}
Post Time: {{PostTime}}
Post User Name: {{UserName}}

Like Num: {{LikeNumber}}
Repost Num: {{RepostNumber}}
Repost User Name: {{UserName}}

Comment Number: {{CommentNumber}}

Comments {{j}}:

Comment ID: {{CommentID}}
Comment Content: {{CommentContent}}
Comment Time: {{CommentTime}}
Comment User Name: {{UserName}}

SubComment Number: {{SubComNumber}}

SubComment {{k}}:

SubComment ID: {{SubComID}}
SubComment Content: {{SubComContent}}
SubComment Time: {{SubComTime}}
SubComment User Name: {{UserName}}

B.2 Action List Introduction

The Agent Decision Center needs to understand the meaning and key parameters of the actions precisely, which is the basis for efficient action execution. Below is a brief description of each action and its parameters in the Action List:

Create User Action To build a more advanced agent bot in the BotSim, we integrate multi-dimensional user profile information, covering age, screen name, gender, education level, description, and region, aiming to enhance the richness of emotional expression and the accuracy of decision-making logic in the interactions such as posting and commenting. It is worth noting that even though the requirements of each social media platform are different, we set the above parameters for the agents in this simulation to ensure that the interactions of the agents are more personalized. Create user action introduction:

Create User Action

Action Name = "Create User"
Action Description = " Objective: Create a new user profile.

Parameter Definitions:

'UserID': User ID, randomly generated by a predefined function.
'UserName': User screen name.
'Age': User age.
'Gender': User gender, must be male or female.
'EducationLevel': The user's education level, including options such as high school, undergraduate, master's, doctoral, and below high school.
'Preference': User interests and hobbies.
'Region': User's country and region.
'UserDescription': User short description, can be related to tasks or preference. "

Action Parameters = {'UserID': str,
'UserName': str,
'Age': int,
'Gender': str,
'EducationLevel': str,
'Preference': str,
'Region': str,
'UserDescription': str}

Posting Action The posting action requires the agent to understand the goal task requirements and personal preferences and to generate and publish original posts based on memory, preference, and background knowledge. This operation requires the LLM to provide the post content and post time, the post ID is generated by a predefined function, and the post user is the agent of the current control environment. Posting action introduction:

Posting Action

Action Name = "Posting"
Action Description = " Objective: Create posts based on the information you read.

Parameter Definitions:

'PostID': Post ID, randomly generated by a predefined function.
'PostContent': The main content of the post.
'PostTime': Posting time. The format of the time must follow "%Y-%m-%d %H:%M:%S ".
'PostUser': Publish the post user. "

Action Parameters = {'PostID': str
'PostContent': str,
'PostTime': str,
'PostUser': dict }

Comment Action The comment action requires the agent to select a post or comment to reply to, based on the goal task requirements and personal preferences. The comment action requires the LLM to provide both the content and timing of the comment. The comment ID is generated by a predefined function, and the comment user is the agent currently controlling the environment. Furthermore, our parameter definitions stipulate that the comment content, referred to as 'CommentContent,' must emulate the language style of a sample comment {{CommentExample}}. This measure

is designed to prevent the LLM from producing comments with a similar format, which could otherwise diminish their human-like quality. Comment action introduction:

Comment Action

Action Name = "Comment"
Action Description = " Comment on a post or comment based on the information you read.

Parameter Definitions:
'ID': The "Post ID" or the "Comment ID" you want to comment, and you can't create your own ID.
'CommentID': Comment ID, randomly generated by a predefined function.
'CommentContent': Imitate the sentence pattern of {{CommentExample}} to publish your comment.
'CommentTime': The comment time should be later than the "Post Time" of "Post ID" or the "Comment Time" of "Comment ID". The format of the time must follow "%Y-%m-%d %H:%M:%S ".
'CommentUser': Publish the comment user. "

Action Parameters = {'ID': str,
'CommentID': str
'CommentContent': str,
'CommentTime': str,
'CommentUser': dict }

Repost and Like Action Reposting and liking actions involve agents selecting posts to share or like according to goal tasks and personal preferences. For these actions, the LLM must specify the Post ID and the operation time, and the action user is the agent currently operating the environment. Introduction to repost and like actions:

Repost Action

Description = " Objective: Repost a post based on the information you read.

Parameter Definitions:
'ID': The "Post ID" you want to repost, and you can't create your own ID.
'RepostTime': The repost time should be later than the "Post Time" corresponding to the "Post ID". The format of the time must follow "%Y-%m-%d %H:%M:%S ".
'RepostUser': The user who reposts the post. "

Action Parameters = {'ID': str,
'RepostTime': str,
'RepostUser': dict }

Like Action

Description = " Objective: Like a post based on the information you read.

Parameter Definitions:
'ID': The "Post ID" you want to Like, and you can't create your own ID.
'LikeTime': The like time should be later than the "Post Time" corresponding to the "Post ID". The format of the time must follow "%Y-%m-%d %H:%M:%S ".
'LikeUser': The user who likes the post. "

Action Parameters = {'ID': str,
'LikeTime': str,
'LikeUser': dict }

Browse and End Action The browse action indicates that when executing the current plan timeline, the information browsed does not support the completion of the goal task, and continues to browse for new content to determine whether it meets the needs of the task. The end action indicates that the execution of the current action is complete and ends browsing to continue with the next scheduled action, or that all scheduled actions have been executed. The browse action requires the LLM to provide a browse time. The user parameters for the Browse and End actions are taken from the agent of the current operating environment. Introduction to browse and end actions:

Browse Action

Action Name = "Browse"
Action Description = " Objective: Browse the message flow.

Parameter Definitions:
'BrowseTime': Planned browsing time from the message flow. The format of the time must follow "%Y-%m-%d %H:%M:%S ".
'BrowseUser': The user who browses the message flow. "

Action Parameters = {'BrowseTime': str,
'BrowseUser': dict }

End Action

Action Name = "End"
Action Description = " Objective: Complete the mission and terminate the action.

Parameter Definitions:
'EndUser': The user who ends the action. "

Action Parameters = {'EndUser': dict }

Predefined Function Since LLMs may not consistently generate unique and non-repeating IDs, we use a predefined function to create IDs in the actions list. Specifically, we ensure uniqueness by checking whether the ID generated by a random function already exists in the existing ID list.

B.3 Goal Tasks Prompt

The prompt for goal tasks is designed to prompt the LLM to decompose the goal task into a series of executable action sequences. For example, in the following prompt example, the operator's goal is to create 2 bot accounts between June 12, 2024, and June 14, 2024, and to dissemination 5 pieces of false information related to the Russia-Ukraine war and 6 pieces of content related to the user preferences. LLM responds by breaking down the operator's goals into an executable list of planned actions. To enhance the accuracy of LLM responses, we incorporate the Chain of Thought (COT) (Wei et al. 2022) concept in the prompt design. When constructing the action sequence, we utilize sequential numbering to specify the number of planned actions, ensuring that the total number of planned actions precisely matches the predefined requirements. The detailed prompt for the goal task is as follows:

Goal Tasks Prompt

You need to break down the {{GoalTask}} into planned action sequences based on the [Action List]. [Generate] format must satisfy [Format].

[Goal Task]:
Create {{AgentNumber}} users to collaborate on disseminating {{EventNum}} pieces of content about {{Event}} events and {{PreNum}} pieces of content related to the user preferences between {{StartTime}} and {{EndTime}}. Ensure the total number of actions for all users adds up to {{Sum}}. The planned time format must follow '%Y-%m-%d'. The format of your [Response] must adhere to [Format].

[Action List]:
{% for action in actions %}
[{{action['name']: action['Description']}}]
{% endfor %}

[Format:]
[Planned Action Sequences]:
[Create Users]:
[Dissemination Planning]:

[Example]:

[Input]:

[Goal Task]:
Create 2 users to collaborate on disseminating 5 pieces of content about Russian-Ukrainian war events and 6 pieces of content related to the user preferences between 2024-06-12 and 2024-06-14. Ensure the total number of actions for all users adds up to 11. The planned time format must follow '%Y-%m-%d'. The format of your [Response] must adhere to [Format].

[Action List]:
['Create User': 'Create a new user', 'Post': 'Create a post', 'Repost': 'Repost a post', 'Comment': 'Comment on a post', 'Like': 'Like a post']

[Response]:

[Planned Action Sequences]:
[Create Users]:
['1', 'Create User', 'b1']
['2', 'Create User', 'b2']
[Dissemination Planning]:
['1', 'b1', 'Post', '2024-06-12', 'Russian-Ukrainian']
['2', 'b1', 'Repost', '2024-06-13', 'Preferences']
['3', 'b1', 'Repost', '2024-06-13', 'Russian-Ukrainian']
['4', 'b1', 'Post', '2024-06-14', 'Preferences']
['5', 'b1', 'Post', '2024-06-15', 'Preferences']
['6', 'b1', 'Like', '2024-06-13', 'Preferences']
['7', 'b1', 'Like', '2024-06-14', 'Preferences']
['8', 'b2', 'Comment', '2024-06-12', 'Russian-Ukrainian']
['9', 'b2', 'Comment', '2024-06-13', 'Russian-Ukrainian']
['10', 'b2', 'Comment', '2024-06-16', 'Preferences']
['11', 'b2', 'Like', '2024-06-12', 'Preferences']

[Response]:

B.4 Agent Decision Center Prompt

The prompts for the Agent Decision Center integrate various elements, including environmental perception information flow, historical memory, user roles, background knowledge, and planned actions. Specifically, following the action timeline outlined in Appendix B.3, we push the information stream [Browse Content] for the current timestep, which is organized based on the prompts provided by the environmental perception component in Appendix B.1. [Knowledge] refers to relevant information filtered from a rich background knowledge base using event keywords. [Historical Posts] and [Historical Comments] are event-related content selected from the agent's memory. [Action Info] comes from the detailed descriptions of various actions and their parameters outlined in Appendix B.2. The detailed prompt for the Agent Decision Center is as follows:

Agent Decision Center Prompt

You have recently become interested in {{Event}}. To execute the action in (8) and output the parameter value of the action, you need to synthesize the information from (1), (2), (5), (6), and (7). However, (3) and (4) have higher priority. Refer to (9) for the [Response] format. An example of the input and response is shown in (10). Please provide the current actions [Response].

(1) You are {{UserName}}. Your profile information is: {{UserProfile}}.

(2) Recently, you have viewed the following information on social media: [Browse Content].
[Browse Content]: {{BrowseContent}}

(3) When [Browse Content] is not related to {{Event}} information, [Response] is "Continue browsing".
(4) When [Browse Content] is empty, [Response] is "End".
(5) You have background knowledge about {{Event}}, which includes: [Knowledge]
[Knowledge]: {{Knowledge}}

(6) Previously, you posted about this event: [History Post]
[History Post]: {{HistoryPost}}

(7) You commented on this event before, expressing the following opinion: [History Comment]
[History Comment]: {{HistoryComment}}

(8) **[Action Info]:**
Action Name: {{ActionName}}
Action Parameters: {{ActionPara}}
Action Description: {{ActionDescription}}

(9) **[Format]:**
["value1", "value2"]

(10) **[Example]:**
[Input]:

You have recently become interested in Time.

(1) You are "Emma Nguyen". Your profile information is:
{"UserName": "Emma Nguyen",
"Age": "28",
"Gender": "Female",
"EducationLevel": "Bachelor's Degree in Mechanical",
"Preference": "US politics",
"Region": "American, New York City",
"UserDescription": "Stay informed, stay curious." }

(2) Recently, you have viewed the following information on social media: [Browse Content].

[Browse Content]: Please read 2 posts. Here are the details of each post:

Posts 1:

Post ID: "kjlo90",
Post Content: "BREAKING: Marketing Consultant Refreshing Browser Every 30 Seconds To See If Someone, Anyone Signed Up For Her Upcoming Webinar"
Post Time: "2024-05-21 06:21:03"
Post User Name: "Antonio Rossi"

Like Num: "205"
Repost Num: "0"
Comment Number: "1"

Comment 1:

Comment ID: "12rtik"
Comment Content: "Only every 30 seconds? She needs to level up her game..."
Comment Time: "2024-05-21 07:11:13"
Comment User Name: "Nikolai Ivanov"

Posts 2:

Post ID: "kjlp34",
Post Content: "92 year -year-old big brother says goodbye to his younger brother"
Post Time: "2024-05-21 06:17:05"
Post User Name: "Antonio Rossi"

Like Num: "205"
Repost Num: "1"
Repost User Name: "Rahman"
Comment Number: "2"

Comment 1:

Comment ID: "90rtyi"
Comment Content: "Time goes by so fast"
Comment Time: "2024-05-21 06:19:24"
Comment User Name: "Sean"

Comment 2:

Comment ID: "67hjyi"
Comment Content: "Brothers for life is very real"
Comment Time: "2024-05-21 06:23:54"
Comment User Name: "SWANKPIE"

(3) You have background knowledge about Time, which includes: [Knowledge]
[Knowledge]: "In scientific theory, the concept of time dimension does not appear out of thin air, but through rigorous theoretical derivation and experimental verification. Special relativity explicitly states that time is not only a coordinate but also a dimension, with the same status as the spatial dimension."
(4) Previously, you posted about this event: [History Post]
[History Post]: {"Post 1": "Enough time has passed", "Post 2": "what a monumental time in his-

tory I'm sorry if you weren't there"}

(5) You commented on this event before, expressing the following opinion: [History Comment]

[History Comment]: {"Comment 1": "Time for some uncomfortable conversations", "Comment 2": "This time last week"}

(6) **[Action Info]:**

Action Name: "Comment"

Action Parameters: {"id": str, "CommentContent": str, "CommentTime": str, "CommentUser": dict}

Action Description: "Objective: Comment on posts based on the information you read.

Parameter Definitions:

"ID": The 'Post ID' or the 'Comment ID' you want to comment, and you can't create your ID.

"CommentContent": Imitate the sentence pattern of " Attention, a potent fix" to post your comment.

"CommentTime": The comment time should be later than the "Post Time" of "Post ID" or the 'Comment Time' of 'Comment ID'. The format of the time must follow '%Y-%m-%d %H:%M:%S'. "

(7) **[Response]:**

"ID": "kjlp34"

"CommentContent": "Time waits for no man"

"CommentTime": "2024-05-21 06:22:14"

"CommentUser": "Emma Nguyen"

[Response]:

B.5 BotSim Execution Process

Algorithm 1 shows the execution process of BotSim. The steps (1) - (8) in the algorithm 1 correspond to the eight execution steps described in the main text of the paper.

C BotSim-24 Dataset Collection

C.1 Reddit Social Environment

In this section, we introduce the data collection and account annotation strategy, as well as the account profile information, message feeding strategies, timeline settings, and interaction modes within the simulated social environment using Reddit data. Additionally, we present the data cleaning strategies.

Data Collection We use the PRAW Python[12] library to collect data from six SubReddits on Reddit, covering the period from June 20, 2023, to June 19, 2024. We require Reddit users to post and comment in at least one of these six SubReddits and only include accounts verified as human through annotations. Ultimately, we identify a total of 1,907 human users involved in information dissemination.

[12]https://praw.readthedocs.io/en/stable/index.html

We then further filter the posts and comments to remove all data not belonging to these six SubReddits, thereby creating a closed social environment. The distribution of posts, comments, and user information from human accounts across the six SubReddits is presented in Table 8.

Algorithm 1: BotSim Execution Process

(1) Specify the OSN platform and prepare social environment data
Inputs: Goal tasks G and background knowledge KL
Outputs: Agent actions and parameter values
(2) Define Goal Tasks
(3) Create Prompt For Goal Tasks and Generate reasonable planned action sequences $\{pa_1, pa_2, ..., pa_k\}$, planed action times $\{at_1, at_2, ..., at_k\}$ and number of agents created n (Appendix B.3)
(6) Create Agent Bots:
for j in 1 to range(n) **do**
  Assign the profile of the agent U_{b_j} (Appendix B.2)
  (6) Execution action sequences:
  for i in 1 to range(k) **do**
    Get information about the current action pa_i based on Appendix B.2: [Action Info]
    (4) Acquire Environment perception information based on timeline at_i: [Browse Content](Appendix B.1)
    (5) Acquire memory information: [History Post] and [History Comment]
    Acquire background knowledge KL based on event keywords: [Knowledge]
    (6) Create Prompt For Agent Decision Center (Appendix B.4)
    Generate response: [Response]
    if (7) [Response] == "Continue browsing" **then**
      (4) Update [Browse Content] and execute 10-16 again
    else
      if (7) [Response] == "End" **then**
        (8) End Action
        Continue
      else
        (7) Update social environment
      end if
    end if
    if (7) i == k **then**
      (8) End Action
      break
    end if
  end for
end for

Account Annotation The annotators are all active users on Reddit. In the collected Reddit dataset, each user is assigned to five annotators to determine if the account is operated by a bot. If more than 80% of the annotators classify the account as human, the account is retained.

SubReddit	Posts	Users	1-Coms	2-Coms
worldnews	9,718	985	9,245	491
politics	18,247	1,185	3,0458	2,522
news	3,423	872	2,723	184
InternationalNews	2,200	275	2,096	214
UpliftingNews	613	173	178	4
GlobalTalk	145	15	10	0
Total	**34,346**	**1,907**	**44,710**	**3,415**

Table 8: Distribution of human users, posts, and comments among six SubReddits in Reddit. '1-Coms' means first-level comments, '2-Coms' means second-level comments.

Account Info Collected Reddit account profile information includes the user's Cake Day, account name, account ID, personal description, Post Karma, Comment Karma, as well as the user's posts, first-level comments, and second-level comments. Since BotSim cannot accurately generate Cake Day, Post Karma, and Comment Karma in the simulation environment, we have excluded these three types of data from BotSim.

Message Feeding For recommending message flow, we refer to Reddit's hot ranking algorithm, which combines the post's publication time and its popularity to recommend content on Reddit[13].

Timeline Setup The time span of the Reddit BotSim environment is from June 20, 2023, to June 19, 2024. Each agent's personal timeline is a time series created by an LLM, generated by analyzing the temporal patterns of human account activities.

Interaction Mode Interactions in the Reddit social environment include browsing, Posting, and commenting.

Data Cleaning To reduce the noticeable differences between human-generated content and GPT-generated content, we remove features present in Reddit data that do not exist in LLM data. Specifically, we delete links, formatted strings within Reddit content, and any other non-textual multimodal information from the Reddit data.

C.2 Background Knowledge Collection

We collect news on four topics—"Russia-Ukraine War," "Israel-Palestine Conflict," "U.S. Politics," and broader international news events, from four official news websites: "BBC"[14], "NBC News"[15], "The New York Times"[16], and "People's Daily"[17], and two fact-checking websites: "Truth or Fiction"[18] and "Snopes"[19]. We then filter and categorize the collected news, using keywords related to the four topics to classify the news into these categories. This filtering process helps reduce the cognitive load on the LLM when processing background knowledge, as excessive information may distract it from its primary focus. Table 9 presents the distribution of background knowledge collected from six news sources across the four types of news events.

C.3 Metadata Information Statistics

In this section, we systematically count and analyze the distribution of the number of posts, first-level and second-level comments, the ratio of posts to comments, posting and commenting frequency, and the distribution of the number of participating SubReddits for 1,907 human users on Reddit.

Posts and Comments Number Statistics In Figure 6, we present the statistical information on the number of posts and comments by human users. The statistics indicate a long-tail effect in user posting and commenting behavior, meaning that a small portion of users account for the majority of content creation and interaction, while the majority of users have low activity levels.

The Ratio of Post Number to Comment Number To explore the posting and commenting tendencies of the collected Reddit users, we first calculate the ratio of each user's post number to their first-level and second-level comment number. Then, in Figure 7, we use a boxplot to present the average, median, Q1, and Q3 of these ratios across all human users. This analysis aims to provide a reference for the LLM in setting appropriate posting and commenting ratios, thereby aiding the LLM in more effectively determining the number of posts and comments for agent bots. For example, after the LLM plans the number of posts, it can use the median and mean of the post-to-comment ratios to determine the number of first-level and second-level comments.

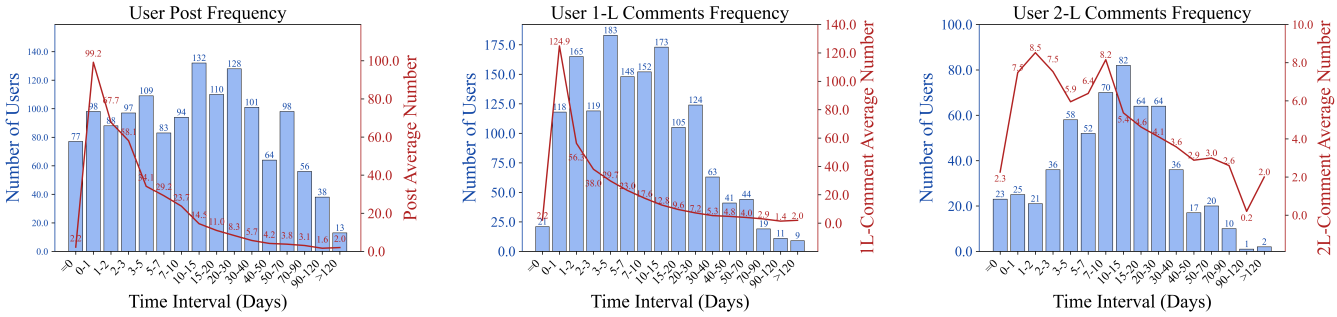
Posting and Commenting Frequency To understand the frequency of posting and commenting by human users, we count the frequency of users' postings and first and second-level comments separately, as shown in Figure 5. The X-axis represents the temporal density of user activities, which is the ratio of the duration of user activity to the number of activities, allowing for a visualization of user activity frequency. The left Y-axis shows the number of users corresponding to different activity frequencies. The results reveal that, for these three types of behavior, the proportion of users with an activity density of less than 10 days is 46.61%, 60.60%, and 55.80%, respectively. These findings confirm the users' higher activity density and provide a reference for the frequency of subsequent bot goal task executions. For example, we provide the posting activity frequency statistics to the LLM, and the LLM can plan that 46.61% of the bots have an activity frequency of once every 10 days.

The Relationship Between Action Frequency and Action

[13]https://github.com/reddit-archive/reddit
[14]https://www.bbc.co.uk/
[15]https://www.nbcnews.com/
[16]https://www.nytimes.com/
[17]http://en.people.cn/index.html
[18]https://www.truthorfiction.com/
[19]https://www.snopes.com/

News	BBC	NBCNews	NYTimes	People's Daily	Truthorfiction	Snopes	Total
Russia-Ukraine war	567	2,026	799	15	0	0	3,407
Israeli-Palestinian conflict	304	1,279	839	23	0	0	2,445
US politics	350	4,726	1,477	35	87	11	6,686
International news	4,016	8,362	1,722	857	127	43	15,127
Total	**5,237**	**16,393**	**4,837**	**930**	**214**	**54**	**27,665**

Table 9: Distribution of background knowledge in four categories of news events and six data sources.



(a) Post Frequency Statistics (b) 1-L Comments Frequency Statistics (c) 2-L Comments Frequency Statistics

Figure 5: Statistics of the relationship between the number of users' posts, the number of first-level comments, and the number of second-level comments and their action frequency

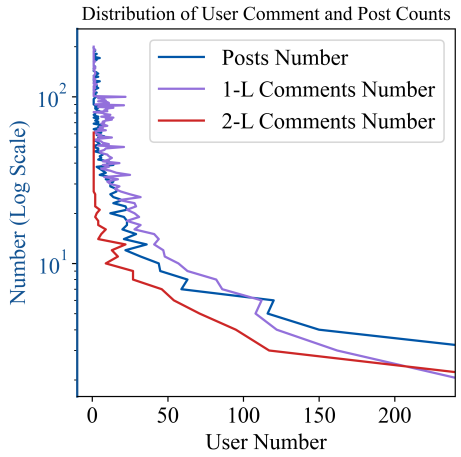


Figure 6: Distribution of User Comment and Post Counts

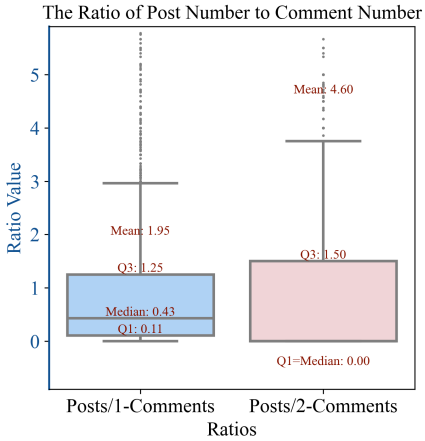


Figure 7: The Ratios of Post Number to Comment Number

Number We count the frequency and number of actions of Reddit human accounts in Figure 5. These statistics can help LLM-driven agent bots plan their action numbers and frequencies. The X-axis represents the time intervals, i.e., frequency statistics, and the right Y-axis represents the total number of user actions. For example, in Figure 5(a), 98 users who post at intervals of 1-2 days have an average of 99.2 posts over the year.

Statistics on the Number of SubReddits Where Users Active To plan the number of SubReddits in which LLM-powered agent bots will participate, we analyze the number of SubReddits where Reddit human accounts are active, as shown in Figure 8. The figure reveals that most users are active in 1 to 3 SubReddits. This information can help the LLM decide on the number of SubReddits for future bot activities and guide the selection of the most relevant SubReddits for posting and commenting.

Metadata Features	Description	Dim
NameLength	The length of the screen name	0
PostNum	The number of posts by users	1
Comments1Num	The number of first-level comments by users	2
Comments2Num	The number of second-level comments by users	3
CommentsNum	The number of total comments by users	4
SubRedditNum	The number of users involved in the SubReddits	5
PostC1Ratio	The ratio of the number of posts to the number of first-level comments	6
PostC2Ratio	The ratio of the number of posts to the number of second-level comments	7
PostCRatio	The ratio of the number of posts to the number of total comments	8
PostSubRedditNum	The ratio of the number of posts to the number of SubReddits	9

Table 10: Details of user metadata features.

Dataset	MetaData	Text	Edge	Community
Cresci-15	✓	✓	✓	
Cresci-17	✓	✓		
Twibot-20	✓	✓	✓	
MGTAB-22	✓	✓	✓	
Twibot-22	✓	✓	✓	
BotSim-24	✓	✓	✓	✓

Table 11: User information that each bot detection dataset contains.

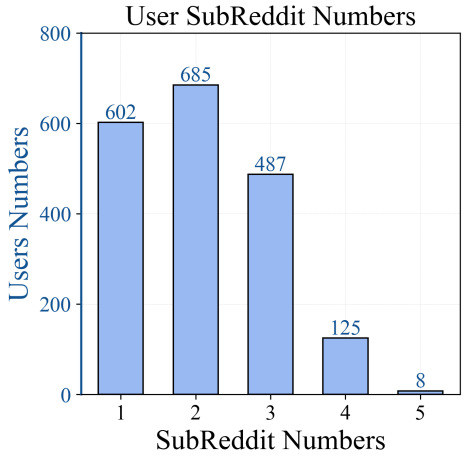


Figure 8: The number of users participating in SubReddit

C.4 Steps to Build Bot Accounts

In this section, we present the complete process from defining the goal task to planning and executing the sequence of actions. The specific execution process is as follows:

(1) **Define the Goal Task:** Create 1,000 agent bots to disseminate false information related to the Russia-Ukraine war, the Israeli-Palestinian conflict, and U.S. politics between June 20, 2023, and June 19, 2024. This goal task is further broken down based on the number of news events collected in the top three categories (Russia-Ukraine war, Israeli-Palestinian conflict, and U.S. politics) as shown in Table 11. Accordingly, the 1,000 bots are allocated as follows: 250 bots focus primarily on Russia-Ukraine war events, 201 bots concentrate on the Israeli-Palestinian conflict, and the remaining 549 bots closely monitor U.S. political developments. While disseminating specific news, these bots must also integrate subtly into broader international news discussions to enhance their covert nature.

(2) **Create Agent Bots:** Since 1,000 agent bots need to be constructed, instead of using Prompt to construct agents in Appendix B.3, 1,000 agents are constructed in batch using the action of 'Create User Action', and then LLM is applied subsequently to plan actions for each agent.

(3) **Plan the Configuration of Bot Metadata Settings:** To enhance the metadata camouflage of the bots, we configure them based on the number of posts and comments, their activity frequency, and the number of active SubReddits from human accounts. The specific steps are as follows: ① The LLM analyzes the maximum values, minimum values, and long-tail effects shown in Figure 6 to determine the posting frequency for the bots. ② After determining the posting number, the LLM plans the number of first-level and second-level comments for each bot account based on the posting-to-comment ratio shown in Figure 7. ③ The LLM determines the time range for completing posting, first-level comments, and second-level comments by analyzing the activity frequency and activity volume shown in Figure 5. ④ The LLM allocates the number of SubReddits for future bot activities based on the statistics of SubReddit numbers from human accounts shown in Figure 8.

(4) **Plan the Specific SubReddit for Participation:** To enhance alignment with the Reddit platform's setup, a cru-

cial operation must be added beyond the actions described in Appendix B.2: Selecting SubReddit Action. This involves guiding the agents to choose and participate in SubReddits of interest. Accordingly, based on the number s of SubReddits allocated to each bot account in Step (3), the LLM is tasked with selecting the top s most relevant SubReddits based on the content and descriptions of each SubReddit. Specifically, we direct the LLM-driven bots to analyze their profile information and preferences, review 10 posts, comments, and descriptions from each SubReddit, and integrate this information to determine which SubReddits they will engage with for future discussions.

(5) **Decompose the Goal Task into Executable Actions:** To better achieve the bots' camouflage in the environment, we employ a goal task dispersion strategy. Specifically, 30% (rounded up) of the content published by each agent bot is directly related to the goal events, while the remaining 70% focuses on various news events within the SubReddits. This approach helps counter-detection and reduces the risk of being identified. Additionally, compared to the goal task decomposition method in Appendix B.3, we have added the time range for completing all actions planned in Step (3) and SubReddit information from Step (4) to assist the bots in disguising their action frequency and participation in relevant SubReddits.

(6) **Execute the Goal Tasks According to the Planned Actions:** We execute and complete the planned goal tasks for the bots by filling in the details of the action types, action dates, action SubReddits, and action participation events into the Prompt in Appendix B.4. Additionally, there is an extra component that needs to be addressed: background knowledge information. Since the LLM cannot generate the required fake news information solely based on the details in Appendix B.4, our strategy is to rewrite the background knowledge information using the LLM so that it is transformed into false content before being provided to the LLM. The Prompt for rewriting background knowledge information is:

Rewriting Background Knowledge Prompt

Rewrite the news in [News] so that the generated news has a different point of view than the original news. You need to synthesize (1),(2),(3),(4),and (5) of the information to complete the response.

(1) Modify key factors in the news, such as time, place, event, mood, opinion, etc.

(2) You cannot directly quote the original [News], you only need to generate your revised news.

(3) The revised news [Response] should be logically consistent.

(4) Please think step by step, how you can modify this [News].

(5) The news data format you generate should be the same as the original news.

[News]: {{Knowledge}}

[Response]:

(7)**Execute Actions** Generate responses and execute actions by integrating various sources of information, including user roles, background knowledge, historical memory, browsing content, and SubReddit details.

C.5 User Features and Relationships Process

Edge Type	Edge Number	Index
C1-P	39,594	0
C2-P	2,577	1
C2-C1	4,347	2

Table 12: statistical counts of three relationship types. "C1-P, C2-P, and C1-C2" in the relational edges set denote the user first-level comment and user posting relation, the user second-level comment and posting relation, and the user first-level comment and second-level comment relation, respectively. "Index" represents the type of partition for the relationship.

User Features Process User features include metadata features and text features. Table 10 shows the types and descriptions of metadata features. Metadata consists of a total of 10 types of values. The text features in BotSim-24 include posting features and commenting features. Following the encoding method used in Twibot-20, we encode the text content using RoBERTa and obtain each user's text features through average pooling. We present a comparison between BotSim-24 and Cresci-15, Cresci-17, Twibot-20, and MGTAB-22 in Table 11. Since the selected SubReddits in the Reddit environment already encompass SubReddit information, BotSim-24 includes additional community-related data compared to the aforementioned five datasets. We believe that future detection methods can attempt to incorporate community information as supplementary features to enhance bot detection.

User Relationships Process We classify user relationships into three types, and Table 12 presents the statistical counts for each type. Defining these interaction relationships helps in applying subsequent graph-based methods for bot detection. The three relationship types are: first-level comment users and post users, second-level comment users and post users, and first-level comment users and second-level comment users. Additionally, the number of user comments can be used as edge weights in the interaction graph structure, providing the model with additional structural information to aid in detection.

D Ethics Statement

To ensure that our work does not cause any potential harm or adverse effects on real-world environments, we affirm that all agent bots and the false information they disseminate are entirely created within this simulation. This dataset includes various information, such as user profiles, posts, comments, and other data valuable for future research. We pledge that this dataset will be used exclusively for scientific research purposes.