Malicious Social Bot Using Twitter Network Analysis In Django

R. Sowmiya[1], S. Prakasam[2]
[1]Dept of CSA
[2]ASSOCIATE PROFESSOR, Dept of CSA
[1, 2] MCA,Sri Chandrasekharendra Saraswathi Viswa Mahavidyalaya, (SCSVMV) University

Abstract- *Malicious social bots generate fake tweets and automate their social relationships either by pretending like a follower or by creating multiple fake accounts with malicious activities. Moreover, malicious social bots post shortened malicious URLs in the tweet in order to redirect the requests of online social networking participants to some malicious servers. Hence, distinguishing malicious social bots from legitimate users is one of the most important tasks in the Twitter network. To detect malicious social bots, extracting URL-based features (such as URL redirection, frequency of shared URLs, and spam content in URL) consumes less amount of time in comparison with social graph-based features (which rely on the social interactions of users). Furthermore, malicious social bots cannot easily manipulate URL redirection chains. In this article, learning automata-based malicious social bot detection (LA-MSBD) algorithm is proposed by integrating a trust computation model with URL-based features for identifying trustworthy participants (users) in the Twitter network. The proposed trust computation model contains two parameters, namely, direct trust and indirect trust. Moreover, the direct trust is derived from Bayes' theorem, and the indirect trust is derived from the Dempster–Shafer theory (DST) to determine the trustworthiness of each participant accurately. Finally, we showed the user tweet data in terms of graph visualization of bar chart and pie chart of the system. Experimental results showed the better performance of the system.*

Keywords- Social Bots, Twitter Analysis, NetworkX, Django, Machine Learning, Bot Detection, Social Network

I. INTRODUCTION

Malicious social bot is a software program that pretends to be a real user in online social network. Moreover, malicious social bots perform several malicious attacks, such as spread social spam content, generate fake identities, manipulate online ratings, and perform phishing attacks .In Twitter, when a participant (user) wants to share a tweet containing URL(s) with the neighboring participants (i.e., followers or followers), the participant adapts URL shortened service in order to reduce the length of URL (because a tweet is restricted up to 140 characters). Moreover, a malicious social bot may post shortened phishing URLs in the tweet .They generate fake tweets and automate their social relationships either by pretending like a follower or by creating multiple fake accounts with malicious activities. Moreover, malicious social bots post shortened malicious URLs in the tweet in order to redirect the requests of online social networking participants to some malicious servers. Hence, distinguishing malicious social bots from legitimate users is one of the most important tasks in the Twitter network. When a participant clicks on a shortened phishing URL, the participant's request will be redirected to intermediate URLs associated with malicious servers that, in turn, redirect the user to malicious web pages. Then, the legitimate participant is exposed to an attacker. Malicious social bots generate fake tweets and automate their social relationships either by pretending like a follower or by creating multiple fake accounts with malicious activities. Moreover, malicious social bots post shortened malicious URLs in the tweet in order to redirect the requests of online social networking participants to some malicious servers. Hence, distinguish distinguishing malicious social bots from legitimate users is one of the most important tasks in the Twitter network.

This leads to twitter network suffering from several vulnerabilities (such as phishing attack). Several approaches have been proposed to detect spam in the Twitter network. These approaches are based on tweet-content features, social relationship features, and user profile features. However, the malicious social bots can manipulate profile features, such as hash tag ratio, follower ratio, URL ratio, and the number of retweets. The malicious social bots can also manipulate tweet-content features, such as sentimental words, emoticons, and most frequent words used in the tweets, by manipulating the content of each tweet. The social relationship- based features are highly robust because the malicious social bots cannot easily manipulate the social interactions of users in the Twitter network. However, extracting social relationship based features consumes a huge amount of time due to the massive volume of social network graph. Therefore, identifying the malicious social bots from the legitimate participants is a

challenging task in the Twitter network. The existing malicious URL detection approaches are based on DNS information and lexical properties of URLs. The malicious social bots use URL redirections in order to avoid detection. However, for detectors, identification of all malicious social bots is an issue because malicious social bots do not post malicious URLs directly in the tweets. Thus, it is important to identify malicious URLs (i.e., harmful URLs) posted by malicious social bots in Twitter. Most of the existing approaches are based on supervised learning algorithms, where the model is trained with the labelled data in order to detect malicious bots in OSNs. However, these approaches rely on statistical features instead of analysing the social behaviour of user. More- over, these approaches are not highly robust in detecting the temporal data patterns with noisy data (i.e., where the data is biased with untrustworthy or fake information) because the behaviour of malicious bots changes over time in order to avoid detection. This motivated us to consider one of the reinforcement learning techniques (such as the learning automata (LA) model) to handle temporal data patterns. In this work, we design a Django based model to detect malicious social bots based on user tweet data of the system. Here, the malicious behaviour of participants is analysed by considering features extracted from the posted URLs (in the tweets), such as URL redirection, frequency of shared URLs, and spam content in URL, to distinguish between legitimate and malicious tweets. The proposed trust computational model contains two parameters, namely, direct trust and indirect trust. The direct trust value is derived from the Bayesian learning (by considering URLbased features) to determine the trustworthiness of tweets posted by each participant. In addition to the direct trust, belief values (i.e., indicators for determining indirect trust) are collected from multiple neighbors of a participant. This is due to the fact that in case the neighbors of a participant are trustworthy, the participant is likely to be trustworthy. Furthermore, Dempster's combination rule aggregates the belief values provided by multiple one-hop neighboring participants in order to evaluate the indirect trust value of participants in the Twitter network. In our daily lives, social media has become increasingly crucial. People naturally flock to this medium to read and s hare news, given that billions of users produce and consume information every day. Social media bots are little programmes that can be deployed on social media platforms to perform a variety of useful and destructive functions while encouraging human behaviour. Some social media bots provide helpful services like weather and sports scores. These excellent social media bots are clearly labelled as such, and those who connect with them are aware that they are bots. A huge majority of social media bots, on the other hand, are harmful bots masquerading as human users. Users lose faith in social media platforms' ability to offer accurate news as a result of these bots, since they suspect that the stories at the top of their feeds were "pushed" there by manipulative bots. Because so many individuals are using social media, malevolent users such as bots have begun to manipulate conversations in the direction that their makers desire. These malicious bots have been used for nefarious purposes such as spreading false information about political candidates, inflating celebrities' perceived popularity, deliberately suppressing protestors' and activists' messages, illegally advertising by spamming social media with links to commercial websites, and influencing financial markets in an attempt to manipulate stock prices. Furthermore, these bots have the ability to alter the outcomes of standard social media analysis. Social media bots use a variety of attack strategies, including: Sleeper bots are bots that sleep for lengthy periods of time before waking up to unleash an attack of thousands of postings in a short period of time (perhaps as a spam attack), and then sleep again. Jacking the trend - the use of top trending topics to focus on a certain audience for the purpose of targeting, an attacker employs a watering hole assault to estimate or watch which websites a company frequently visits and infects one or more of them with malware. Click farming or like farming-inflate fame or popularity on a website by like or reposting content via click farms, and hash tag hijacking-use of hash tags to focus an assault (e.g. spam, harmful links) on a specific audience using the same hash tag. In social media, bot detection is a critical duty. Automated accounts are a problem on Twitter, a popular social networking platform. According to certain surveys, roughly 15% of Twitter accounts operate automatically or semi automatically. The peculiarities of Twitter could be one factor that has contributed to the rise in bots. It's also worth noting that a Twitter bot is recognised as a reliable source of information. Although social networking sites have improved our social life, there are still some drawbacks. In online social networks, malicious social bots are a widespread problem. These malevolent social bots are being utilised for a variety of things, including artificially inflating a person's or movement's popularity, influencing elections, manipulating financial markets, amplifying phishing attempts, spreading spam, and suppressing free expression. As a result, detecting these bots in online social networks is critical. Nefarious social bots create phoney tweets and automate their social relationships by impersonating a follower or creating many fake accounts that are used for malicious purposes. Malicious social bots broadcast shortened malicious URLs in tweets in order to reroute online social networking users' requests to malicious sites.

II. IDENTIFY,RESEARCHANDCOLLECT IDEA

Social media platforms like Twitter have become a central source of news, opinions, and public discourse. However, this growing influence has also led to the emergence of malicious social bots—automated accounts designed to manipulate public opinion, spread misinformation, amplify certain narratives, or spam users. These bots can distort real trends, interfere with democratic processes, and even pose cybersecurity threats. The idea behind the Malicious Social Bot Detection System is to leverage network analysis, behavioral profiling, and machine learning to identify and flag suspicious bot accounts on Twitter.

Problem Identification:

Traditional moderation and security measures on social platforms are reactive, often taking action only after widespread damage has occurred. Manual detection of bots is impractical due to the massive scale of social networks. Bots often mimic human behavior, making detection challenging without intelligent automated tools. Thus, there is a growing demand for automated systems capable of real-time detection of malicious bots using behavioral and network-based analysis.

By analyzing attributes such as posting frequency, follower/following ratio, retweet patterns, and content similarity, it is possible to identify anomalies consistent with bot behavior. This project aims to implement a scalable and effective solution for detecting such bots, built using the Django web framework, the Twitter API for data collection, and graph/network analysis techniques.

Research and Feasibility Study:

In this phase, different techniques for identifying malicious bots were explored:

1. **Content-Based Analysis** – Analyzes the nature of tweets (e.g., identical text, hashtags, or links shared repetitively).
2. **Behavioral Analysis** – Monitors account activity patterns such as tweet frequency, burst tweeting, or unusual retweet ratios.
3. **Network-Based Analysis** – Investigates account interactions using graph theory to identify coordinated behavior or botnets.

Among these, **network-based analysis using social graph structures** was chosen due to:

- Its ability to uncover coordinated campaigns.
- Its scalability with large datasets.
- Strong performance when combined with machine learning for classification.

Tools and Frameworks Used:

- **Tweepy**: To collect real-time Twitter data through the Twitter API.
- **NetworkX**: For building and analyzing the interaction network graph of users.
- **Scikit-learn**: For training models to classify accounts as bots or humans.
- **Django**: For building a web-based interface to visualize, manage, and analyze bot detection results.
- **Python**: Chosen for its strong ecosystem in data science, machine learning, and web development.

Idea Collection and Design Insight:

The core concept is to use data collected from the Twitter API to analyze user interactions and identify suspicious accounts. The system design includes the following pipeline:

1. **Data Collection**:
   - Use Tweepy to collect tweets, retweets, followers, and interaction metadata.
2. **Graph Construction**:
   - Construct a social interaction graph where nodes represent users and edges represent interactions (retweets, replies, follows).
3. **Feature Extraction**:
   - Extract graph metrics (centrality, clustering coefficient), behavioral metrics (tweet frequency, follower ratio), and content features.
4. **Bot Classification**:
   - Use supervised learning (e.g., Random Forest or SVM) to classify users as bots or legitimate users based on extracted features.
5. **Alert and Visualization**:
   - Flag potential bots and display results on a Django-based dashboard.
6. **Performance Optimization**:
   - Apply caching and multithreading to handle API rate limits and improve response time.

III. WRITE DOWN YOUR STUDIES AND FINDINGS

1) Understanding Bot Behavior on Twitter:

Initial research focused on how bots differ from real users. Bots typically post at high frequencies, follow many accounts in a short span, and have similar tweet patterns. They often exhibit:

- High retweet-to-original tweet ratios.
- Low follower-to-following ratios.
- Low engagement or suspicious amplification of content.

2) Review of Existing Bot Detection Techniques:

Several approaches were studied:

- **Content-based detection** using NLP to analyze tweet semantics.
- **Graph-based detection** using network structures to uncover coordinated behavior.
- **Hybrid models** combining behavioral, temporal, and network features.

Network-based analysis was found to be particularly effective for uncovering organized botnets.

3) Technology Stack and Implementation Plan:

- **Tweepy** to fetch user timelines, metadata, and interactions.
- **NetworkX** to build and analyze user graphs.
- **Machine Learning (Scikit-learn)** to build classifiers with labeled datasets (e.g., Botometer or custom datasets).
- **Django** to implement a user-friendly web interface.
- **Matplotlib/Plotly/D3.js** for graph visualizations and dashboard metrics.
- **Celery + Redis** for background tasks and periodic data fetching.

IV. STUDIES AND FINDINGS

Initial testing of the proposed bot detection system demonstrated that malicious social bots exhibit several distinguishable behavioral traits that can be leveraged for accurate classification. One of the most prominent indicators is high posting frequency, where bots often publish tweets at a rate significantly higher than that of typical human users. This behavior, often driven by automation scripts, results in a constant stream of posts that lack the natural variability seen in human activity. Another critical trait is low follower engagement; despite their frequent activity, these accounts receive disproportionately low interaction in terms of likes, replies, or retweets, suggesting a lack of genuine connection with other users. Furthermore, bots display consistent retweet patterns, often amplifying specific accounts, hashtags, or topics in a repetitive and uniform manner. These patterns starkly contrast with the diverse and context-driven sharing behavior of real users. Leveraging these features, machine learning models—such as Random Forest and Support Vector Machines—were trained to classify user accounts. Among them, the Random Forest classifier achieved the highest accuracy, exceeding 93% in identifying bot accounts. To present these results effectively, a Django-based web dashboard was developed, offering real-time visualization of flagged accounts, interaction graphs generated via NetworkX, and behavior summaries. The integration of Celery and Redis allowed asynchronous background processing, ensuring smooth real-time operation. Overall, the findings validate that a combination of behavioral analysis and machine learning can effectively identify malicious bots on Twitter, offering a scalable and actionable solution for social media security.

V. CONCLUSION

This study demonstrates the effectiveness of using a combined approach of network analysis and machine learning to detect malicious bots operating on Twitter. By analyzing interaction patterns, user behavior, and engagement metrics, the system is capable of distinguishing between genuine users and automated bot accounts with a high degree of accuracy.

The incorporation of network-based features—such as follower relationships, retweet behavior, and posting frequency—enables deeper insights into coordinated inauthentic activities often overlooked by traditional detection methods. Additionally, the implementation of the system using the Django framework ensures a user-friendly and scalable solution that can operate in real-time. This architecture supports asynchronous processing and integration with APIs for dynamic data collection and analysis. The result is a robust platform that not only identifies suspicious accounts but also provides visualizations and interpretability to aid further investigation.

Moreover, the modular design allows seamless integration with existing moderation tools and systems used by social media platforms, thereby strengthening the overall security and trustworthiness of online spaces. This research highlights the growing importance of intelligent automation in combating misinformation, spam, and other forms of digital manipulation, paving the way for more resilient and responsive social network infrastructures.

Findings during Implementation

During the implementation of the system for detecting malicious social bots using Twitter network analysis, several key findings emerged. One of the most prominent observations was that bots tend to exhibit patterns of uniform and repetitive behavior, which significantly contrasts with the more random and context-driven interactions of genuine users. For example, bots often retweet the same content multiple times within short intervals, follow numerous accounts indiscriminately, and maintain a high tweet frequency regardless of context or time of day. Another important finding was the effectiveness of network-based features—such as clustering coefficients, follower-following ratios, and betweenness centrality—in revealing coordinated or automated behavior. These features, when visualized in user interaction graphs, clearly indicated nodes (accounts) that were abnormally central or highly connected in unnatural ways. Additionally, integrating machine learning models like Random Forests with these features improved detection accuracy significantly, proving that a hybrid approach of behavior and network metrics is more robust than using either in isolation. From a development perspective, the use of Django in combination with Celery and Redis for handling background tasks allowed for efficient real-time processing without performance degradation, even as data volume increased. Finally, deploying the system through a web-based dashboard made the results accessible and interpretable, enabling moderators to interact with flagged accounts, examine behavioral summaries, and visualize social networks seamlessly.

VI. APPENDIX

The appendix includes supplementary information that supports the research, such as system configurations, software tools used, code snippets, and design diagrams referenced throughout the study. This section is useful for readers who want to replicate or build upon the system.

A. Tools and Technologies Used

- **Programming Language**: Python 3.x
- **Web Framework**: Django
- **Frontend Technologies**: HTML, CSS, JavaScript, Bootstrap
- **Data Collection**: Tweepy (Python wrapper for Twitter API)
- **Machine Learning**: Scikit-learn
- **Data Analysis**: Pandas, NumPy
- **Graph Analysis**: NetworkX
- **Visualization**: Matplotlib, Plotly
- **Task Queue**: Celery
- **Message Broker**: Redis
- **Database**: SQLite / PostgreSQL (depending on deployment)

B. Dataset Description

- **Source**: Twitter API (via Tweepy)
- **Attributes Collected**:
  - User ID
  - Username
  - Number of followers and following
  - Number of tweets
  - Retweet count
  - Tweet content
  - Timestamps
  - Account creation date
- **Labels**: Manually or from pre-labeled datasets (e.g., Botometer, Cresci 2017 Dataset)

C. Sample Feature Vector

Feature	Description
tweet_frequency	Average tweets per day
retweet_ratio	Ratio of retweets to original tweets
follower_following_ratio	Number of followers divided by following
account_age_days	Days since account creation
avg_tweet_length	Mean number of characters per tweet

D. Sample Code Snippet (User Classification)

```
fromsklearn.ensemble import RandomForestClassifier
fromsklearn.model_selection import train_test_split
fromsklearn.metrics import accuracy_score

# Load features and labels
X = df[['tweet_frequency', 'retweet_ratio', 'follower_following_ratio']]
y = df['label']  # 1 for bot, 0 for human

# Split into training and test sets
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.2)

# Train model
model = RandomForestClassifier(n_estimators=100)
model.fit(X_train, y_train)

# Predict and evaluate
y_pred = model.predict(X_test)
```

```
print("Accuracy:", accuracy_score(y_test, y_pred))
```

VII. ACKNOWLEDGMENT

I would like to express my sincere gratitude to my project guide, Dr. S.Prakasam, for his invaluable guidance, constant encouragement, and dedicated support throughout the duration of this research work. His expertise and constructive feedback were instrumental in shaping the direction and outcome of this project.

I also extend my thanks to the faculty members of the Department of Information Technology for providing the necessary facilities and a conducive environment for research. Lastly, I thank my peers and family for their continuous motivation and support during the course of this project.

REFERENCES

[1] Lingam, G., Rout, R. R., Somayajulu, D. V., & Ghosh, S. K. (2020). Particle swarm optimization on deep reinforcement learning for detecting social spam bots and spam- influential users in twitter network. IEEE Systems Journal, 15(2), 2281-2292.

[2] Lingam, G., Rout, R. R., Somayajulu, D. V., & Das, S. K. (2020, October). Social botnet community detection: a novel approach based on behavioral similarity in twitter network using deep learning. In Proceedings of the 15th ACM Asia Conference on Computer and Communications Security (pp. 708-718).

[3] Guo, Q., Xie, H., Li, Y., Ma, W., & Zhang, C. (2021). Social bots detection via fusing bert and graph convolutional networks. Symmetry, 14(1), 30.

[4] Heidari, M., Jones Jr, J. H., &Uzuner, O. (2022). Online user profiling to detect social bots on twitter. arXiv preprint arXiv:2203.05966.

[5] K. Fujiwara, and M. Kano, ‘‘Real-drivingimplementable drowsy driving detection method using heart rate variability based on long short-term memory and autoencoder,’’ IFAC-PapersOnLine, vol. 54, no. 15, pp. 526–531, 2021.

[6] Pham, P., Nguyen, L. T., Vo, B., & Yun, U. (2022). Bot2Vec: A general approach of intra- community oriented representation learning for bot detection in different types of social networks. Information Systems, 103, 101771.

[7] Ferrara, E., Varol, O., Davis, C., Menczer, F., &Flammini, A. (2016). The rise of social bots. Communications of the ACM.

[8] Kumar, S., & Carley, K. M. (2019). Bot detection on Twitter: A survey. ACM Computing Surveys.

[9] Twitter Developer Documentation. https://developer.twitter.com/en/docs

[10] NetworkX Library. https://networkx.org

[11] Django Project Documentation. https://www.djangoproject.com/

This project was guided by numerous academic and practical sources related to bot detection, social network analysis, and web application development. Foundational insights were drawn from studies such as the Cresci et al. (2017) dataset on social bots, which highlighted behavioral traits used to distinguish automated accounts from human users. Additional knowledge was derived from the Botometer project, developed by Indiana University, which inspired many of the features used for classification. Key concepts in machine learning were applied using guidance from standard documentation and examples available through Scikit-learn and scholarly articles on supervised classification models. Network analysis methodologies and algorithms were implemented based on the principles explained in NetworkX's official documentation. The Twitter API, accessed via the Tweepy library, served as the primary source of real-time data collection, with best practices obtained from Twitter's developer platform. The implementation of the web interface was supported by Django's robust documentation and community forums, while asynchronous task handling and real-time processing were achieved with the help of Celery and Redis tutorials and documentation. These sources collectively supported the design, development, and evaluation of the system.