TwiBot-22: Towards Graph-Based Twitter Bot Detection

Shangbin Feng[1,2*] **Zhaoxuan Tan**[1*] **Herun Wan**[1*] **Ningnan Wang**[1*] **Zilong Chen**[1,3*] **Binchi Zhang**[1,4*]
Qinghua Zheng[1†] **Wenqian Zhang**[1] **Zhenyu Lei**[1] **Shujie Yang**[1] **Xinshun Feng**[1] **Qingyue Zhang**[1]
Hongrui Wang[1] **Yuhan Liu**[1] **Yuyang Bai**[1] **Heng Wang**[1] **Zijian Cai**[1] **Yanbo Wang**[1]
Lijing Zheng[1] **Zihan Ma**[1] **Jundong Li**[4] **Minnan Luo**[1]
Xi'an Jiaotong University[1], University of Washington[2], Tsinghua University[3], University of Virginia[4]
contact: `shangbin@cs.washington.edu`

Abstract

Twitter bot detection has become an increasingly important task to combat misinformation, facilitate social media moderation, and preserve the integrity of the online discourse. State-of-the-art bot detection methods generally leverage the graph structure of the Twitter network, and they exhibit promising performance when confronting novel Twitter bots that traditional methods fail to detect. However, very few of the existing Twitter bot detection datasets are graph-based, and even these few graph-based datasets suffer from limited dataset scale, incomplete graph structure, as well as low annotation quality. In fact, the lack of a large-scale graph-based Twitter bot detection benchmark that addresses these issues has seriously hindered the development and evaluation of novel graph-based bot detection approaches. In this paper, we propose TwiBot-22, a comprehensive graph-based Twitter bot detection benchmark that presents the largest dataset to date, provides diversified entities and relations on the Twitter network, and has considerably better annotation quality than existing datasets. In addition, we re-implement 35 representative Twitter bot detection baselines and evaluate them on 9 datasets, including TwiBot-22, to promote a fair comparison of model performance and a holistic understanding of research progress. To facilitate further research, we consolidate all implemented codes and datasets into the TwiBot-22 evaluation framework, where researchers could consistently evaluate new models and datasets. The TwiBot-22 Twitter bot detection benchmark and evaluation framework are publicly available at `https://twibot22.github.io/`.

1 Introduction

Automated users on Twitter, also known as Twitter bots, have become a widely known and well-documented phenomenon. Over the past decade, malicious Twitter bots were responsible for a wide range of problems such as online disinformation [Cui et al., 2020, Wang et al., 2020, Lu and Li, 2020], election interference [Howard et al., 2016, Bradshaw et al., 2017, Rossi et al., 2020, Ferrara, 2017], extremism campaign [Ferrara et al., 2016, Marcellino et al., 2020], and even the spread of conspiracy theories [Ferrara, 2020, Ahmed et al., 2020, Anwar et al., 2021]. These societal challenges have called for automatic Twitter bot detection models to mitigate their negative influence.

[*] These authors contributed equally to this work.
[†] Corresponding author: Qinghua Zheng, School of Computer Science and Technology, Xi'an Jiaotong University. Email: qhzheng@xjtu.edu.cn

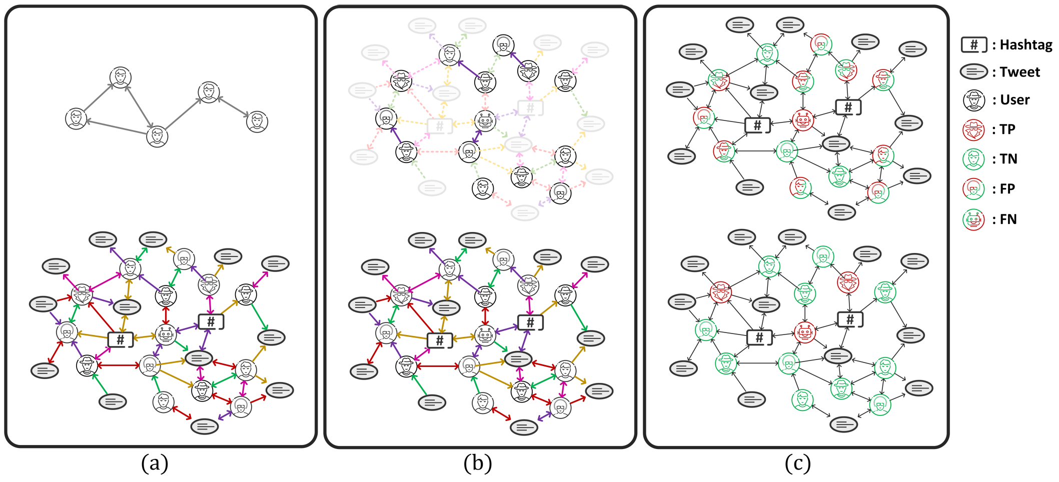


Figure 1: Compared to real-world Twitter (below), existing Twitter bot detection datasets (above) suffer from (a) limited dataset scale, (b) incomplete graph structure, and (c) poor annotation quality.

Existing Twitter bot detection models are generally **feature-based**, where researchers propose to extract numerical features from user information such as metadata [Yang et al., 2020, Lee et al., 2011], user timeline [Mazza et al., 2019, Chavoshi et al., 2016], and follow relationships [Beskow and Carley, 2020, Chu et al., 2012]. However, feature-based approaches are susceptible to adversarial manipulation, *i.e.*, when bot operators try to avoid detection by tampering with these hand-crafted features [Cresci et al., 2017a, Cresci, 2020]. Researchers also proposed **text-based** approaches, where text analysis techniques such as word embeddings [Wei and Nguyen, 2019], recurrent neural networks [Kudugunta and Ferrara, 2018, Feng et al., 2021a], and pre-trained language models [Dukić et al., 2020] are leveraged to analyze tweet content and identify malicious intent. However, new generations of Twitter bots often intersperse malicious content with normal tweets stolen from genuine users [Cresci, 2020, Feng et al., 2021b], thus their bot nature becomes more subtle to text-based methods. With the advent of graph neural networks, recent advances focus on developing **graph-based** Twitter bot detection models. These methods [Ali Alhosseini et al., 2019, Feng et al., 2021b] interpret users as nodes and follow relationships as edges to leverage graph mining techniques such as GCN [Kipf and Welling, 2016], R-GCN [Schlichtkrull et al., 2018], and RGT [Feng et al., 2022] for graph-based bot detection. In fact, recent research have shown that graph-based approaches achieve state-of-the-art performance, are capable of detecting novel Twitter bots, and are better at addressing various challenges facing Twitter bot detection [Feng et al., 2021b, 2022].

However, the development and evaluation of graph-based Twitter bot detection models are poorly supported by existing datasets. The Bot Repository[3] provides a comprehensive collection of existing datasets. Out of the 18 listed datasets, only two of them, TwiBot-20 [Feng et al., 2021c] and cresci-2015 [Cresci et al., 2015], explicitly provide the graph structure among Twitter users. In addition, these two graph-based datasets suffer from the following issues as illustrated in Figure 1:

- **(a) limited dataset scale**. Twibot-20 [Feng et al., 2021c] contains 11,826 labeled users and cresci-15 [Cresci et al., 2015] contains 7,251 labeled users, while online conversations and discussions about heated topics often involve hundreds of thousands of users [Banda et al., 2021].
- **(b) incomplete graph structure**. Real-world Twitter is a heterogeneous information network that contains many types of entities and relations [Feng et al., 2022], while TwiBot-20 and cresci-15 only provide users and follow relationships.
- **(c) low annotation quality**. TwiBot-20 resorted to crowdsourcing for user annotation, while crowdsourcing leads to significant noise [Graells-Garrido and Baeza-Yates, 2022] and is susceptible to the false positive problem [Rauchfleisch and Kaiser, 2020].

In light of these challenges, we propose TwiBot-22, a graph-based Twitter bot detection benchmark that addresses these issues. Specifically, TwiBot-22 adopts a two-stage controlled expansion to

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

sample the Twitter network, which results in a dataset that is 5 times the size of the largest existing dataset. TwiBot-22 provides 4 types of entities and 14 types of relations in the Twitter network, which provides the first (truly) heterogeneous graph for Twitter bot detection. Finally, TwiBot-22 adopts the weak supervision learning strategy for data annotation which results in significantly improved annotation quality. To compare TwiBot-22 with existing datasets, we re-implement 35 Twitter bot detection baselines and evaluate them on 9 datasets, including TwiBot-22, to provide a holistic view of research progress and highlight the advantages of TwiBot-22. We consolidate all datasets and implemented codes into the TwiBot-22 evaluation framework to facilitate further research. Our main contributions are summarized as follows:

- We propose TwiBot-22, a graph-based Twitter bot detection dataset that establishes the largest benchmark to date, provides diversified entities and relations in the Twitter network, and has considerably improved annotation quality.
- We re-implement and benchmark 35 existing Twitter bot detection models on 9 datasets, including TwiBot-22, to compare different approaches fairly and facilitate a holistic understanding of research progress in Twitter bot detection.
- We present the TwiBot-22 evaluation framework, where researchers could easily reproduce our results, examine existing datasets and methods, infer on unseen Twitter data, and contribute new datasets and models to the framework.

2 Related Work

2.1 Twitter Bot Detection

Existing Twitter bot detection methods mainly fall into three categories: **feature-based** methods, **text-based** methods, and **graph-based** methods.

Feature-based methods conduct feature engineering with user information and apply traditional classification algorithms for bot detection. Various features are extracted from user metadata [Kudugunta and Ferrara, 2018], tweets [Miller et al., 2014], description [Hayawi et al., 2022], temporal patterns [Mazza et al., 2019], and follow relationships [Feng et al., 2021a]. Later research efforts improve the scalability of feature-based approaches [Yang et al., 2020], automatically discover new bots [Chavoshi et al., 2016], and strike the balance between precision and recall [Morstatter et al., 2016]. However, as bot operators are increasingly aware of these hand-crafted features, novel bots often try to tamper with these features to evade detection [Cresci, 2020]. As a result, feature-based methods struggle to keep up with the arms race of bot evolution [Feng et al., 2021a].

Text-based methods use techniques in natural language processing to detect Twitter bots based on tweets and user descriptions. Word embeddings [Wei and Nguyen, 2019], recurrent neural networks [Kudugunta and Ferrara, 2018], the attention mechanism [Feng et al., 2021a], and pre-trained language models [Dukić et al., 2020] are adopted to encode tweets for bot detection. Later research combines tweet representations with user features [Cai et al., 2017], learns unsupervised user representations [Feng et al., 2021a], and attempts to address the multi-lingual issue in tweet content [Knauth, 2019]. However, novel bots begin to counter text-based approaches by diluting malicious tweets with content stolen from genuine users [Cresci, 2020]. In addition, Feng et al. [2021b] shows that analyzing tweet content alone might not be robust and accurate for bot detection.

Graph-based methods interpret Twitter as graphs and leverage concepts from network science and geometric deep learning for Twitter bot detection. Node centrality [Dehghan et al., 2022], node representation learning [Pham et al., 2022], graph neural networks (GNNs) [Ali Alhosseini et al., 2019], and heterogeneous GNNs [Feng et al., 2021b] are adopted to conduct graph-based Twitter bot detection. Later research try to combine graph-based and text-based methods [Guo et al., 2021a] or propose new GNN architectures to leverage heterogeneities in the Twitter network [Feng et al., 2022]. Graph-based approaches have shown great promise in tackling various challenges facing Twitter bot detection, such as bot communities and bot disguise [Feng et al., 2021b].

The development and evaluation of these models would not be possible without the many valuable Twitter bot detection datasets that were proposed over the past decade. These datasets mainly focus on politics and elections in the United States [Yang et al., 2020] and European countries [Cresci et al., 2017a]. Cresci-17 [Cresci et al., 2017a] propose the concept of "social spambots" and presents a widely used dataset with more than one type of bots. TwiBot-20 [Feng et al., 2021c] is the latest and most comprehensive Twitter bot detection dataset that addresses the issue of user diversity in previous datasets. However, among 18 datasets presented in the Bot Repository, the go-to place for Twitter bot detection research, only 2 explicitly provide the graph structure of the Twitter network. In addition, these datasets suffer from limited dataset scale, incomplete graph structure, and low annotation quality while increasingly falling short of consistently benchmarking novel graph-based approaches. In light of these challenges, we present TwiBot-22 to alleviate these issues, promote a rethinking of research progress, and facilitate further research in Twitter bot detection.

2.2 Graph-based Social Network Analysis

Users in online social networks interact with each other and become part of the network structure, while the network structure is essential in understanding the patterns of social media [Carrington et al., 2005]. With the advent of geometric deep learning, graph neural networks (GNNs) have become increasingly popular in social network analysis research. Qian et al. [2021] propose to model social media with heterogeneous graphs and leverage relational GNNs for illicit drug trafficker detection. Guo et al. [2021b] propose dual graph enhanced embedding neural network to improve graph representation learning and tackle challenges in click-through rate prediction. Graphs and GNNs are also adopted to detect online fraud [Liu et al., 2021, Li et al., 2021, Wang et al., 2021, Dou et al., 2020], combat misinformation [Cui et al., 2020, Wang et al., 2020, Lu and Li, 2020, Varlamis et al., 2022, Hu et al., 2021], and improve recommender systems [Ying et al., 2018, Wu et al.]. The task of Twitter bot detection is no exception, where novel and state-of-the-art approaches are increasingly graph-based [Ali Alhosseini et al., 2019, Magelinski et al., 2020, Feng et al., 2021b, 2022, Lei et al., 2022]. In this paper, we propose the TwiBot-22 benchmark to better support the development and evaluation of graph-based Twitter bot detection models.

3 TwiBot-22 Dataset

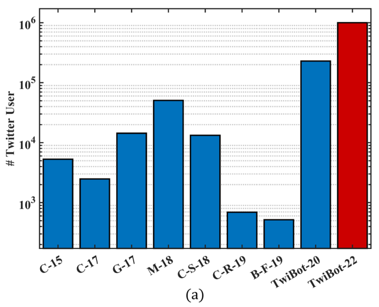
3.1 Data Collection

TwiBot-22 aims to present a large-scale and graph-based Twitter bot detection benchmark. To this end, we adopt a two-stage data collection process. We firstly adopt diversity-aware breadth-first search (BFS) to obtain the user network of TwiBot-22. We then collect additional entities and relations on the Twitter network to enrich the heterogeneity of the TwiBot-22 network.

User network collection. A common drawback of existing Twitter bot detection datasets is that they only feature a few types of bots and genuine users, while real-world Twitter is home to diversified users and bots [Feng et al., 2021c]. As a result, TwiBot-20 proposes to use breadth-first search (BFS) for user collection, starting from "seed users" and expanding with user follow relationships. To ensure that TwiBot-22 includes different types of bots and genuine users, we augment BFS with two diversity-aware sampling strategies:

- **Distribution diversity**. Given user metadata such as follower count, different types of users fall differently into the metadata distribution. Distribution diversity aims to sample users in the top, middle, and bottom of the distribution. For numerical metadata, among a user's neighbors and their metadata values, we select the k users with the highest value, k with the lowest, and k randomly chosen from the rest. For true-or-false metadata, we select k with true and k with false.
- **Value diversity**. Given a user and its metadata, value diversity is adopted so that neighbors with significantly different metadata values are more likely to be included, ensuring the diversity of collected users. For numerical metadata, among expanding user u's neighbors X and their metadata values x^{num}, the probability of user $x \in X$ being sampled is denoted as $p(x) \propto |u^{num} - x^{num}|$. For true-or-false metadata we select k users from the opposite class.

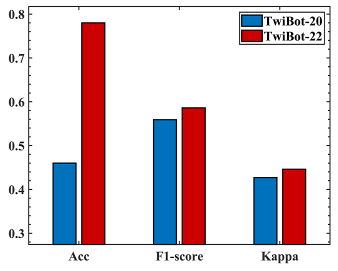
Based on these sampling strategies, TwiBot-22 conducts diversity-aware BFS starting from @*NeurIPSConf*. For each neighborhood expansion, one metadata and one of the sampling strategies are randomly adopted. The BFS process stops until the user network contains a desirable amount of Twitter users. More information about the user collection process is presented in Appendix A.2.



(a)

	Entity	Relation
C-15 & TwiBot-20	user, tweet	follower, following, post
TwiBot-22	user, tweet, list, hashtag	follower, following, post, pin, like, mention, retweet, quote, reply, own, member, follow, contain, discuss

(b)



(c)

Figure 2: Analyzing TwiBot-22. (a) Dataset scale in terms of users. (b) List of entities and relations in TwiBot-22 and graph-based datasets. (c) Accuracy and F1-score of dataset labels compared to expert annotations as well as Randolph's kappa coefficient [Randolph, 2005] of expert agreement.

Heterogeneous graph building. The user network collection process constructs a homogeneous graph with users as nodes and follow relationships as edges. Apart from that, the Twitter network contains diversified entities and relations such as lists and retweets. Based on the user network, we collect the tweets, associated lists, and mentioned hashtags of these users as well as 12 other relations between users and these new entities. As a result, TwiBot-22 presents a heterogeneous Twitter network with 4 types of entities and 14 types of relations. More information about the heterogeneous Twitter network is presented in Appendix A.4.

As a result, we obtain the TwiBot-22 heterogeneous graph that contains 92,932,326 nodes and 170,185,937 edges. We present more dataset statistics in Table 7 in the appendix.

3.2 Data Annotation

Existing bot detection datasets often rely on manual annotation or crowdsourcing, while it is labor-intensive and thus not feasible with the large-scale TwiBot-22. As a result, we adopt weak supervision learning strategy to generate high-quality labels. We firstly invite bot detection experts to annotate 1,000 Twitter users in TwiBot-22. We then generate noisy labels with the help of bot detection models. Finally, we generate high-quality annotations for TwiBot-22 with the Snorkel framework [Ratner et al., 2017].

Expert annotation. We randomly select 1,000 users in TwiBot-22 and assign each user to 5 Twitter bot detection experts to identify if it is a bot. We then create an 8:2 split for these expert annotations as training and test sets. More details about these experts are presented in Appendix A.3.

Generate noisy labels. We employ 8 hand-crafted labeling functions and 7 competitive feature-based and neural network-based bot detection models to generate noisy labels. For handcrafted labeling functions, we adopt spam keywords in tweets and user descriptions as well as user categorical features such as verified and sensitive tweets. For feature engineering models, we select features based on users' metadata such as creation time, follower count and name length. We then adopt Adaboost, random forest, and MLP to result in three feature-based classifiers. For neural network-based models, we follow Feng et al. [2021b] to encode user information and employ MLP, GAT [Veličković et al., 2018], GCN [Kipf and Welling, 2016], and R-GCN [Schlichtkrull et al., 2018] as four classifiers. We train these classifiers on the training set of expert annotations and calculate the uncertainty scores for all users in TwiBot-22 under each classifier as $\phi = -(\hat{y}_0 \log(\hat{y}_0) + \hat{y}_1 \log(\hat{y}_1))$, where $\hat{y}_0$ and $\hat{y}_1$ denote the probability of being genuine users or bots. For each classifier, we then remove model predictions with the top 40% uncertainty scores to alleviate label noises.

Majority voting. After obtaining the noisy labels, we evaluate their plausibility with Snorkel [Ratner et al., 2017] and clean the labels at the same time. The output of the Snorkel system are probabilistic labels, thus we use these labels to train an MLP classifier to obtain the final annotations of TwiBot-22. We further evaluate the annotation quality on the test set of expert annotations and we obtain an 90.5% accuracy. Compared to the 80% accuracy standard in TwiBot-20 [Feng et al., 2021c], TwiBot-22 has considerably imporved annotation quality.

Table 1: Statistics of the 9 datasets. TwiBot-20 contains unlabelled users so that # User $\neq$ # Human $+$ # Bot. C-15, G-17, C-17, M-18, C-S-18, C-R-19, B-F-19 are short for cresci-2015, gilani-2017, cresci-2017, midterm-18, cresci-stock-2018, cresci-rtbust-2019, botometer-feedback-2019. C-17 contains only "post" edges between users and tweets, which is not a graph-based dataset.

Dataset	C-15	G-17	C-17	M-18	C-S-18	C-R-19	B-F-19	TwiBot-20	TwiBot-22
# Human	1,950	1,394	3,474	8,092	6,174	340	380	5,237	860,057
# Bot	3,351	1,090	10,894	42,446	7,102	353	138	6,589	139,943
# User	5,301	2,484	14,368	50,538	13,276	693	518	229,580	1,000,000
# Tweet	2,827,757	0	6,637,615	0	0	0	0	33,488,192	88,217,457
# Human Tweet	2,631,730	0	2,839,361	0	0	0	0	33,488,192	81,250,102
# Bot Tweet	196,027	0	3,798,254	0	0	0	0	33,488,192	6,967,355
# Edge	7,086,134	0	6,637,615	0	0	0	0	33,716,171	170,185,937

3.3 Data Analysis

Existing graph-based Twitter bot detection datasets suffer from limited dataset scale, incomplete graph structure, and low annotation quality. As a result, we examine whether TwiBot-22 has adequately addressed these challenges and present our findings in Figure 2.

Dataset scale. Figure 2(a) illustrates the number of Twitter users in TwiBot-22 and existing datasets. It is illustrated that TwiBot-22 establishes the largest Twitter bot detection benchmark to date, with approximately 5 times more users than the second-largest TwiBot-20.

Graph structure. Figure 2(b) demonstrates that the TwiBot-22 network contains 4 types of entities and 14 types of relations, resulting in significantly enriched graph structure compared to existing graph-based datasets cresci-15 and TwiBot-20 with only 2 entity types and 3 relation types.

Annotation quality. TwiBot-20, the largest graph-based Twitter bot detection benchmark to date, leveraged crowdsourcing for data annotation. To improve label quality, TwiBot-22 uses weak supervision learning strategies and leverages 1,000 expert annotations to guide the process. To examine whether TwiBot-22 has improved annotation quality than TwiBot-20, we ask Twitter bot detection experts to participate in an "expert study", where they are asked to evaluate users in TwiBot-20 and TwiBot-22 to examine how often do experts agree with dataset labels. Figure 2(c) illustrates the results, which shows that these experts find TwiBot-22 to provide more consistent, accurate, and trustworthy data annotations. More details about the expert study are available in Appendix A.5.

4 Experiments

4.1 Experiment Settings

Datasets. We evaluate Twitter bot detection models on all 9 datasets in the Bot Reporistory that contain both bots and genuine users: cresci-2015 [Cresci et al., 2015], gilani-2017 [Gilani et al., 2017], cresci-2017 [Cresci et al., 2017a,b], midterm-18 [Yang et al., 2020], cresci-stock-2018 [Cresci et al., 2018, 2019], cresci-rtbust-2019 [Mazza et al., 2019], botometer-feedback-2019 [Yang et al., 2019], TwiBot-20 [Feng et al., 2021c], and TwiBot-22. We present dataset details in Table 1. We create a 7:2:1 random split as training, validation, and test set to ensure fair comparison.

Baselines. We re-implement and evaluate 35 Twitter bot detection baselines SGBot [Yang et al., 2020], Kudugunta and Ferrara [2018], Hayawi et al. [2022], BotHunter [Beskow and Carley, 2018], NameBot [Beskow and Carley, 2019], Abreu et al. [2020], Cresci et al. [2016], Wei and Nguyen [2019], BGSRD [Guo et al., 2021a], RoBERTa [Liu et al., 2019], T5 [Raffel et al., 2020], Efthimion et al. [2018], Kantepe and Ganiz [2017], Miller et al. [2014], Varol et al. [2017], Kouvela et al. [2020], Ferreira Dos Santos et al. [2019], Lee et al. [2011], LOBO [Echeverrï£¡ a et al., 2018], Moghaddam and Abbaspour [2022], Ali Alhosseini et al. [2019], Knauth [2019], FriendBot [Beskow and Carley, 2020], SATAR [Feng et al., 2021a], Botometer [Yang et al., 2022], Rodríguez-Ruiz et al. [2020], GraphHist [Magelinski et al., 2020], EvolveBot [Yang et al., 2013], Dehghan et al. [2022], GCN [Kipf and Welling, 2016], GAT [Veličković et al., 2018], HGT [Hu et al., 2020], SimpleHGN [Lv

Table 2: Average bot detection accuracy and standard deviation of five runs of 35 baseline methods on 9 datasets. **Bold** and underline indicate the highest and second highest performance. The F, T, and G in the "Type" column indicates whether a baseline is feature-based, text-based, or graph-based. Cresci *et al.* and Botometer are deterministic methods or APIs without standard deviation. "/" indicates that the dataset does not contain enough user information to support the baseline. "-" indicates that the baseline is not scalable to the largest TwiBot-22 dataset.

Method	Type	C-15	G-17	C-17	M-18	C-S-18	C-R-19	B-F-19	TwiBot-20	TwiBot-22
SGBot	F	77.1 (±0.2)	**78.6** (±0.8)	92.1 (±0.3)	99.2 (±0.0)	81.3 (±0.1)	80.9 (±1.5)	75.5 (±1.9)	81.6 (±0.5)	75.1 (±0.1)
Kudugunta *et al.*	F	75.3 (±0.1)	70.0 (±1.1)	88.3 (±0.2)	91.0 (±0.5)	77.5 (±0.1)	62.9 (±0.8)	74.0 (±4.7)	59.6 (±0.7)	65.9 (±0.0)
Hayawi *et al.*	F	84.3 (±0.0)	52.7 (±0.0)	90.8 (±0.0)	84.6 (±0.0)	50.0 (±0.0)	51.2 (±0.0)	77.0 (±0.0)	73.1 (±0.0)	76.5 (±0.0)
BotHunter	F	96.5 (±1.2)	76.4 (±1.0)	88.1 (±0.2)	**99.3** (±0.0)	81.2 (±0.2)	81.5 (±1.7)	74.7 (±1.0)	75.2 (±0.4)	72.8 (±0.0)
NameBot	F	77.0 (±0.0)	60.8 (±0.0)	76.8 (±0.0)	85.1 (±0.0)	55.8 (±0.0)	63.2 (±0.0)	71.7 (±0.0)	59.1 (±0.1)	70.6 (±0.0)
Abreu *et al.*	F	75.7 (±0.1)	74.3 (±0.1)	92.7 (±0.1)	96.5 (±0.1)	75.4 (±0.1)	80.9 (±0.1)	**77.4** (±0.1)	73.4 (±0.1)	70.7 (±0.1)
Cresci *et al.*	T	37.0	/	33.5	/	/	/	/	47.8	-
Wei *et al.*	T	96.1 (±1.4)	/	89.3 (±0.7)	/	/	/	/	71.3 (±1.6)	70.2 (±1.2)
BGSRD	T	87.8 (±0.6)	48.5 (±8.4)	75.9 (±0.0)	82.9 (±1.5)	50.7 (±1.3)	50.0 (±4.9)	59.6 (±3.1)	66.4 (±1.0)	71.9 (±1.8)
RoBERTa	T	97.0 (±0.1)	/	97.2 (±0.0)	/	/	/	/	75.5 (±0.1)	72.1 (±0.1)
T5	T	92.3 (±0.1)	/	96.4 (±0.0)	/	/	/	/	73.5 (±0.1)	72.1 (±0.1)
Efthimion *et al.*	FT	92.5 (±0.0)	55.5 (±0.0)	88.0 (±0.0)	93.4 (±0.0)	70.8 (±0.0)	67.6 (±0.0)	69.8 (±0.0)	62.8 (±0.0)	74.1 (±0.0)
Kantepe *et al.*	FT	97.5 (±1.3)	/	98.2 (±1.5)	/	/	/	/	80.3 (±4.3)	76.4 (±2.4)
Miller *et al.*	FT	75.5 (±0.0)	51.0 (±0.0)	77.1 (±0.2)	83.7 (±0.0)	52.5 (±0.0)	54.4 (±0.0)	**77.4** (±0.0)	64.5 (±0.4)	30.4 (±0.1)
Varol *et al.*	FT	93.2 (±0.5)	/	/	/	/	/	/	78.7 (±0.6)	73.9 (±0.0)
Kouvela *et al.*	FT	97.8 (±0.5)	74.7 (±0.9)	98.4 (±0.1)	97.0 (±0.1)	79.3 (±0.3)	79.7 (±1.2)	71.3 (±0.9)	84.0 (±0.4)	76.4 (±0.0)
Santos *et al.*	FT	70.8 (±0.0)	51.4 (±0.0)	73.8 (±0.0)	86.6 (±0.0)	62.5 (±0.0)	73.5 (±0.0)	71.7 (±0.0)	58.7 (±0.0)	-
Lee *et al.*	FT	98.2 (±0.1)	74.8 (±1.2)	**98.8** (±0.1)	96.4 (±0.1)	81.5 (±0.4)	**83.5** (±1.9)	75.5 (±1.3)	77.4 (±0.5)	76.3 (±0.1)
LOBO	FT	**98.4** (±0.3)	/	96.6 (±0.3)	/	/	/	/	77.4 (±0.2)	75.7 (±0.1)
Moghaddam *et al.*	FG	73.6 (±0.2)	/	/	/	/	/	/	74.0 (±0.8)	73.8 (±0.0)
Alhosseini *et al.*	FG	89.6 (±0.6)	/	/	/	/	/	/	59.9 (±0.6)	47.7 (±8.7)
Knauth *et al.*	FTG	85.9 (±0.0)	49.6 (±0.0)	90.2 (±0.0)	83.9 (±0.0)	**88.7** (±0.0)	50.0 (±0.0)	76.0 (±0.0)	81.9 (±0.0)	71.3 (±0.0)
FriendBot	FTG	96.9 (±1.1)	/	78.0 (±1.0)	/	/	/	/	75.9 (±0.5)	-
SATAR	FTG	93.4 (±0.5)	/	/	/	/	/	/	84.0 (±0.8)	-
Botometer	FTG	57.9	71.6	94.2	89.5	72.6	69.2	50.0	53.1	49.9
Rodrifuez-Ruiz *et al.*	FTG	82.4 (±0.0)	/	76.4 (±0.0)	/	/	/	/	66.0 (±0.1)	49.4 (±0.0)
GraphHist	FTG	77.4 (±0.2)	/	/	/	/	/	/	51.3 (±0.3)	-
EvolveBot	FTG	92.2 (±1.7)	/	/	/	/	/	/	65.8 (±0.6)	71.1 (±0.1)
Dehghan *et al.*	FTG	62.1 (±0.0)	/	/	/	/	/	/	86.7 (±0.1)	-
GCN	FTG	96.4 (±0.0)	/	/	/	/	/	/	77.5 (±0.0)	78.4 (±0.0)
GAT	FTG	96.9 (±0.0)	/	/	/	/	/	/	83.3 (±0.0)	79.5 (±0.0)
HGT	FTG	96.0 (±0.3)	/	/	/	/	/	/	**86.9** (±0.2)	74.9 (±0.1)
SimpleHGN	FTG	96.7 (±0.5)	/	/	/	/	/	/	86.7 (±0.2)	76.7 (±0.3)
BotRGCN	FTG	96.5 (±0.7)	/	/	/	/	/	/	85.8 (±0.7)	**79.7** (±0.1)
RGT	FTG	97.2 (±0.3)	/	/	/	/	/	/	86.6 (±0.4)	76.5 (±0.4)

et al., 2021], BotRGCN [Feng et al., 2021b], RGT [Feng et al., 2022]. These methods leverage different aspects of user information and represent different stages of bot detection research. More details about these baseline methods are available in Appendix B.1.

4.2 Experiment Results

We re-implement 35 baseline methods and evaluate them on 9 Twitter bot detection datasets. We run each baseline method for **five times** and report the average performance and standard deviation. Table 2 presents the benchmarking results. Our main discoveries are summarized as follows:

- Graph-based approaches are generally more effective than feature-based or text-based methods. As a matter of fact, all top 5 models on TwiBot-20 and TwiBot-22 are graph-based. On average, these top-5 graph-based methods outperform the global average of all baselines by 13.8% and 8.2% on TwiBot-20 and TwiBot-22. This trend suggests that future research in Twitter bot detection should further examine how users and bots interact on Twitter and the heterogeneous graph structure thus formed.

- Most existing datasets do not provide the graph structure of Twitter users to support graph-based approaches, while TwiBot-22 supports all baseline methods and serve as a comprehensive evaluation benchmark. As novel and state-of-the-art models are increasingly graph-based, future Twitter bot detection datasets should provide the graph structure of real-world Twitter.

Table 3: Removing the graph-related model component from graph-based methods (w/o G) while comparing to their original versions (Prev.) on TwiBot-20 and TwiBot-22.

Method	**TwiBot-20**						**TwiBot-22**					
	Acc			F1			Acc			F1		
	Prev.	w/o G	Diff.	Prev.	w/o G	Diff.	Prev.	w/o G	Diff.	Prev.	w/o G	Diff.
Ali Alhosseini et al. [2019]	59.9	61.8	+1.9	72.1	70.7	−1.4	70.7	66.9	−3.8	5.7	3.5	−2.2
Moghaddam and Abbaspour [2022]	74.0	72.2	−1.8	77.9	75.8	−2.1	73.8	73.3	−0.5	32.1	32.0	−0.1
Knauth [2019]	81.9	81.4	−0.5	85.2	84.9	−0.3	71.3	71.5	+0.2	37.1	11.3	−25.8
EvolveBot [Yang et al., 2013]	65.8	65.1	−0.7	69.7	69.3	−0.4	71.1	71.0	−0.1	14.1	14.0	−0.1
BotRGCN [Feng et al., 2021b]	85.7	82.6	−3.1	87.3	83.8	−3.5	79.7	75.4	−4.3	57.5	41.2	−16.3
RGT [Feng et al., 2022]	86.6	82.6	−4.0	88.0	83.8	−4.2	76.5	75.4	−1.1	42.9	41.2	−1.7

- TwiBot-22 establishes the largest benchmark while exposing the scalability issues of baseline methods. For example, Dehghan et al. [2022] achieves near-sota performance on TwiBot-20, while failing to scale to TwiBot-22 as our implementation encounters the out-of-memory problem.
- Performance on TwiBot-22 is on average 2.7% lower than on TwiBot-20 across all baseline methods, which demonstrates that Twitter bot detection is still an open problem that calls for further research. This could be attributed to the fact that Twitter bots are constantly evolving to improve their disguise and evade detection, thus bot detection methods should also adapt and evolve.

4.3 Removing Graphs from Baselines

Benchmarking results in Table 2 demonstrate that graph-based approaches generally achieve better performance. To examine the role of graphs in graph-based approaches, we remove the graph component in competitive graph-based methods [Ali Alhosseini et al., 2019, Moghaddam and Abbaspour, 2022, Knauth, 2019, Yang et al., 2013, Feng et al., 2021b, 2022] and report model performance in Table 3. It is demonstrated that:

- All baseline methods exhibit performance drops to different extents on two datasets when the graph component is removed. This indicates that the graph-related components in graph-based approaches contribute to bot detection performance.
- For graph neural network-based approaches BotRGCN [Feng et al., 2021b] and RGT [Feng et al., 2022], the performance drop is generally more severe. This suggests that graph neural networks play an important role in boosting model performance and advancing bot detection research.

More details about how graphs are removed from baseline methods are presented in Appendix B.4.

4.4 Generalization Study

The challenge of generalization [Yang et al., 2020, Feng et al., 2021a], *i.e.*, whether Twitter bot detection models perform well on unseen data, is essential in ensuring that bot detection research translates to effective social media moderation and real-world impact. To evaluate the generalization ability of existing Twitter bot detection approaches, we identify 10 sub-communities in the TwiBot-22 network. We then use these sub-communities as folds and examine the performance of several representative models when trained on fold i and evaluated on fold j. Figure 3 illustrates that:

- **Graph-based methods are better at generalizing to unseen data.** For example, BotRGCN [Feng et al., 2021b] achieves the best avg score among all baseline methods, outperforming the second-highest RGT by 3.66. This suggests that leveraging the network structure of Twitter might be a potential solution to the generalization challenge.
- **Good model performance does not necessarily translate to good generalization ability.** For example, GAT outperforms LOBO by 5.9% and 3.8% on TwiBot-20 and TwiBot-22 respectively in terms of accuracy. However, GAT has lower avg (-2.55) compared to LOBO. This suggests that future bot detection research should focus on generalization in addition to model performance.

More details about the 10 sub-communities are provided in Appendix B.5.

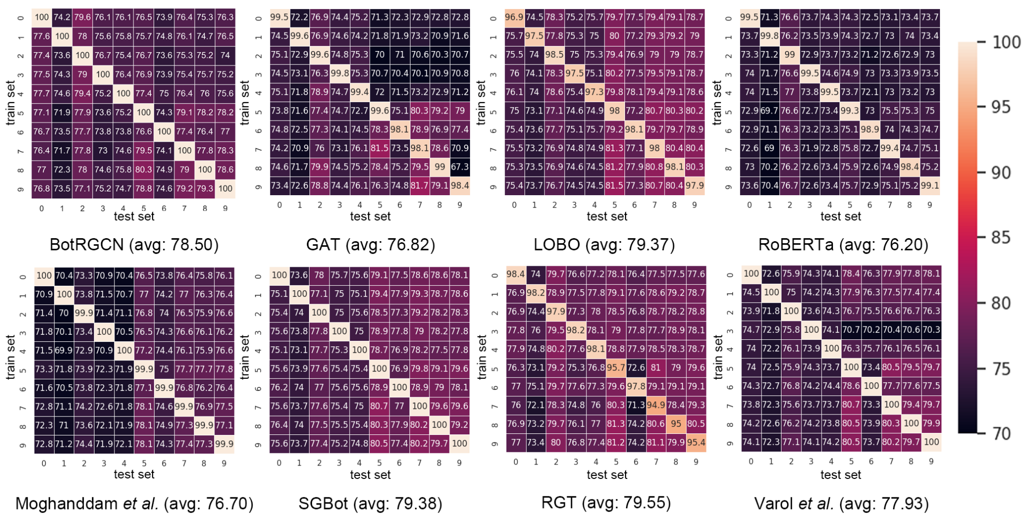


Figure 3: Training models on fold i and testing on fold j. We present model accuracy and report the average value of each heatmap (avg), which serves as an overall indicator of generalization ability.

5 Evaluation Framework

We consolidate Twitter bot detection datasets, data prepossessing codes, and all 35 implemented baselines into the TwiBot-22 evaluation framework and make it publicly available. We hope our efforts would facilitate further research in Twitter bot detection through:

- establishing a unified interface for different types of Twitter bot detection datasets
- providing 35 representative baselines and well-documented implementations
- enriching the evaluation framework with new datasets and methods proposed in future research

Please refer to `https://twibot22.github.io/` for more details.

6 Conclusion and Future Work

In this paper, we propose TwiBot-22, a graph-based Twitter bot detection benchmark. TwiBot-22 successfully alleviates the challenges of limited dataset scale, incomplete graph structure, and poor annotation quality in existing datasets. Specifically, we employ a two-stage data collection process and adopt the weak supervision learning strategy for data annotation. We then re-implement 35 representative Twitter bot detection models and evaluate them on 9 datasets, including TwiBot-22, to promote a holistic understanding of research progress. We further examine the role of graphs in graph-based methods and the generalization ability of competitive bot detection baselines. Finally, we consolidate all implemented codes into the TwiBot-22 evaluation framework, where researchers could easily reproduce our experiments and quickly test out new datasets and models.

Armed with the TwiBot-22 benchmark and the TwiBot-22 evaluation framework, we aim to investigate these research questions in the future:

- **How do we identify bot clusters and their coordination campaigns?** While existing works study Twitter bot detection through individual analysis, novel Twitter bots are increasingly observed to act in groups and launch coordinated attacks. We aim to complement the scarce literature by proposing temporal and subgraph-level bot detection approaches to address this issue.
- **How do we incorporate multi-modal user features for bot detection?** In addition to text and graph, Twitter users and bots generate multi-modal user information such as images and videos. Since TwiBot-22 provides user media while none of the 35 baselines leverage these modalities, we aim to further explore Twitter bot detection with the help of images and videos.

- **How do we evaluate the generalization ability of bot detection methods?** Existing works mainly focus on bot detection performance while generalization is essential in ensuring that bot detection research generates real-world impact. We aim to complement the scarce literature by proposing measures to quantitatively evaluate bot detection generalization.
- **How do we improve the scalability of graph-based models?** Existing graph-based bot detection methods demand significantly more computation resources and execution time than feature-based models. Given that the Twitter network is rapidly expanding, we aim to further explore scalable and graph-based bot detection methods.

Acknowledgements

This work was supported by the National Key Research and Development Program of China (No. 2020AAA0108800), National Nature Science Foundation of China (No. 62192781, No. 62272374, No. 61872287, No. 62250009, No. 62137002), Innovative Research Group of the National Natural Science Foundation of China (61721002), Innovation Research Team of Ministry of Education (IRT_17R86), Project of China Knowledge Center for Engineering Science and Technology and Project of Chinese academy of engineering "The Online and Offline Mixed Educational Service System for 'The Belt and Road' Training in MOOC China".

We would like to thank the reviewers and area chair for the constructive feedback. We would also like to thank all LUD lab members for our collaborative research environment and for making our bot detection research series possible. As Shangbin concludes his term as the director and Zhaoxuan begins his term, we hope the LUD lab will continue to thrive in the years to come.

References

Limeng Cui, Haeseung Seo, Maryam Tabar, Fenglong Ma, Suhang Wang, and Dongwon Lee. Deterrent: Knowledge guided graph attention network for detecting healthcare misinformation. In *Proceedings of the 26th ACM SIGKDD international conference on knowledge discovery & data mining*, pages 492–502, 2020.

Youze Wang, Shengsheng Qian, Jun Hu, Quan Fang, and Changsheng Xu. Fake news detection via knowledge-driven multimodal graph convolutional networks. In *Proceedings of the 2020 International Conference on Multimedia Retrieval*, pages 540–547, 2020.

Yi-Ju Lu and Cheng-Te Li. Gcan: Graph-aware co-attention networks for explainable fake news detection on social media. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 505–514, 2020.

Philip N Howard, Bence Kollanyi, and Samuel Woolley. Bots and automation over twitter during the us election. *Computational propaganda project: Working paper series*, 21(8), 2016.

Samantha Bradshaw, Bence Kollanyi, Clementine Desigaud, and Gillian Bolsover. Junk news and bots during the french presidential election: What are french voters sharing over twitter? Technical report, COMPROP Data Memo, 2017.

Sippo Rossi, Matti Rossi, Bikesh Upreti, and Yong Liu. Detecting political bots on twitter during the 2019 finnish parliamentary election. In *Proceedings of the 53rd Hawaii International Conference on System Sciences*, 2020.

Emilio Ferrara. Disinformation and social bot operations in the run up to the 2017 french presidential election. *arXiv preprint arXiv:1707.00086*, 2017.

Emilio Ferrara, Wen-Qiang Wang, Onur Varol, Alessandro Flammini, and Aram Galstyan. Predicting online extremism, content adopters, and interaction reciprocity. In *International conference on social informatics*, pages 22–39. Springer, 2016.

William Marcellino, Madeline Magnuson, Anne Stickells, Benjamin Boudreaux, Todd C Helmus, Edward Geist, and Zev Winkelman. Counter-radicalization bot research using social bots to fight violent extremism. Technical report, Rand Corp Santa Monica CA United States, 2020.

Emilio Ferrara. What types of covid-19 conspiracies are populated by twitter bots? *arXiv preprint arXiv:2004.09531*, 2020.

Wasim Ahmed, Francesc López Seguí, Josep Vidal-Alaball, Matthew S Katz, et al. Covid-19 and the "film your hospital" conspiracy theory: social network analysis of twitter data. *Journal of medical Internet research*, 22(10):e22374, 2020.

Ahmed Anwar, Haider Ilyas, Ussama Yaqub, and Salma Zaman. Analyzing qanon on twitter in context of us elections 2020: Analysis of user messages and profiles using vader and bert topic modeling. In *DG. O2021: The 22nd Annual International Conference on Digital Government Research*, pages 82–88, 2021.

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

Kyumin Lee, Brian Eoff, and James Caverlee. 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, pages 185–192, 2011.

Michele Mazza, Stefano Cresci, Marco Avvenuti, Walter Quattrociocchi, and Maurizio Tesconi. Rtbust: Exploiting temporal patterns for botnet detection on twitter. In *Proceedings of the 10th ACM conference on web science*, pages 183–192, 2019.

Nikan Chavoshi, Hossein Hamooni, and Abdullah Mueen. Debot: Twitter bot detection via warped correlation. In *2016 IEEE 16th International Conference on Data Mining (ICDM)*, pages 817–822. IEEE Computer Society, 2016.

David M Beskow and Kathleen M Carley. You are known by your friends: Leveraging network metrics for bot detection in twitter. In *Open Source Intelligence and Cyber Crime*, pages 53–88. Springer, 2020.

Zi Chu, Steven Gianvecchio, Haining Wang, and Sushil Jajodia. Detecting automation of twitter accounts: Are you a human, bot, or cyborg? *IEEE Transactions on dependable and secure computing*, 9(6):811–824, 2012.

Stefano Cresci, Roberto Di Pietro, Marinella Petrocchi, Angelo Spognardi, and Maurizio Tesconi. 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*, pages 963–972, 2017a.

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

Feng Wei and Uyen Trang Nguyen. 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)*, pages 101–109. IEEE, 2019.

Sneha Kudugunta and Emilio Ferrara. Deep neural networks for bot detection. *Information Sciences*, 467:312–322, 2018.

Shangbin Feng, Herun Wan, Ningnan Wang, Jundong Li, and Minnan Luo. Satar: A self-supervised approach to twitter account representation learning and its application in bot detection. In *Proceedings of the 30th ACM International Conference on Information & Knowledge Management*, pages 3808–3817, 2021a.

David Dukić, Dominik Keča, and Dominik Stipić. Are you human? detecting bots on twitter using bert. In *2020 IEEE 7th International Conference on Data Science and Advanced Analytics (DSAA)*, pages 631–636. IEEE, 2020.

Shangbin Feng, Herun Wan, Ningnan Wang, and Minnan Luo. 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*, pages 236–239, 2021b.

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

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

Michael Schlichtkrull, Thomas N Kipf, Peter Bloem, Rianne van den Berg, Ivan Titov, and Max Welling. Modeling relational data with graph convolutional networks. In *European semantic web conference*, pages 593–607. Springer, 2018.

Shangbin Feng, Zhaoxuan Tan, Rui Li, and Minnan Luo. Heterogeneity-aware twitter bot detection with relational graph transformers. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 36, pages 3977–3985, 2022.

Shangbin Feng, Herun Wan, Ningnan Wang, Jundong Li, and Minnan Luo. Twibot-20: A comprehensive twitter bot detection benchmark. In *Proceedings of the 30th ACM International Conference on Information & Knowledge Management*, pages 4485–4494, 2021c.

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

Juan M Banda, Ramya Tekumalla, Guanyu Wang, Jingyuan Yu, Tuo Liu, Yuning Ding, Ekaterina Artemova, Elena Tutubalina, and Gerardo Chowell. A large-scale covid-19 twitter chatter dataset for open scientific research—an international collaboration. *Epidemiologia*, 2(3):315–324, 2021.

Eduardo Graells-Garrido and Ricardo Baeza-Yates. Bots don't vote, but they surely bother! a study of anomalous accounts in a national referendum. *arXiv preprint arXiv:2203.04135*, 2022.

Adrian Rauchfleisch and Jonas Kaiser. The false positive problem of automatic bot detection in social science research. *PloS one*, 15(10):e0241045, 2020.

Zachary Miller, Brian Dickinson, William Deitrick, Wei Hu, and Alex Hai Wang. Twitter spammer detection using data stream clustering. *Information Sciences*, 260:64–73, 2014.

Kadhim Hayawi, Sujith Mathew, Neethu Venugopal, Mohammad M Masud, and Pin-Han Ho. Deeprobot: a hybrid deep neural network model for social bot detection based on user profile data. *Social Network Analysis and Mining*, 12(1):1–19, 2022.

Fred Morstatter, Liang Wu, Tahora H Nazer, Kathleen M Carley, and Huan Liu. A new approach to bot detection: striking the balance between precision and recall. In *2016 IEEE/ACM International Conference on Advances in Social Networks Analysis and Mining (ASONAM)*, pages 533–540. IEEE, 2016.

Chiyu Cai, Linjing Li, and Daniel Zeng. Detecting social bots by jointly modeling deep behavior and content information. In *Proceedings of the 2017 ACM on Conference on Information and Knowledge Management*, pages 1995–1998, 2017.

Jürgen Knauth. Language-agnostic twitter-bot detection. In *Proceedings of the International Conference on Recent Advances in Natural Language Processing (RANLP 2019)*, pages 550–558, 2019.

Ashkan Dehghan, Kinga Siuta, Agata Skorupka, Akshat Dubey, Andrei Betlen, David Miller, Wei Xu, Bogumil Kaminski, and Pawel Pralat. Detecting bots in social-networks using node and structural embeddings. 2022.

Phu Pham, Loan TT Nguyen, Bay Vo, and Unil Yun. Bot2vec: A general approach of intra-community oriented representation learning for bot detection in different types of social networks. *Information Systems*, 103:101771, 2022.

Qinglang Guo, Haiyong Xie, Yangyang Li, Wen Ma, and Chao Zhang. Social bots detection via fusing bert and graph convolutional networks. *Symmetry*, 14(1):30, 2021a.

Peter J Carrington, John Scott, and Stanley Wasserman. *Models and methods in social network analysis*, volume 28. Cambridge university press, 2005.

Yiyue Qian, Yiming Zhang, Yanfang Ye, and Chuxu Zhang. Distilling meta knowledge on heterogeneous graph for illicit drug trafficker detection on social media. *Advances in Neural Information Processing Systems*, 34, 2021.

Wei Guo, Rong Su, Renhao Tan, Huifeng Guo, Yingxue Zhang, Zhirong Liu, Ruiming Tang, and Xiuqiang He. Dual graph enhanced embedding neural network for ctr prediction. In *Proceedings of the 27th ACM SIGKDD Conference on Knowledge Discovery & Data Mining*, pages 496–504, 2021b.

Can Liu, Li Sun, Xiang Ao, Jinghua Feng, Qing He, and Hao Yang. Intention-aware heterogeneous graph attention networks for fraud transactions detection. In *Proceedings of the 27th ACM SIGKDD Conference on Knowledge Discovery & Data Mining*, pages 3280–3288, 2021.

Zhao Li, Haishuai Wang, Peng Zhang, Pengrui Hui, Jiaming Huang, Jian Liao, Ji Zhang, and Jiajun Bu. Live-streaming fraud detection: A heterogeneous graph neural network approach. In *Proceedings of the 27th ACM SIGKDD Conference on Knowledge Discovery & Data Mining*, pages 3670–3678, 2021.

Li Wang, Peipei Li, Kai Xiong, Jiashu Zhao, and Rui Lin. Modeling heterogeneous graph network on fraud detection: A community-based framework with attention mechanism. In *Proceedings of the 30th ACM International Conference on Information & Knowledge Management*, pages 1959–1968, 2021.

Yingtong Dou, Zhiwei Liu, Li Sun, Yutong Deng, Hao Peng, and Philip S Yu. Enhancing graph neural network-based fraud detectors against camouflaged fraudsters. In *Proceedings of the 29th ACM International Conference on Information & Knowledge Management*, pages 315–324, 2020.

Iraklis Varlamis, Dimitrios Michail, Foteini Glykou, and Panagiotis Tsantilas. A survey on the use of graph convolutional networks for combating fake news. *Future Internet*, 14(3):70, 2022.

Linmei Hu, Tianchi Yang, Luhao Zhang, Wanjun Zhong, Duyu Tang, Chuan Shi, Nan Duan, and Ming Zhou. Compare to the knowledge: Graph neural fake news detection with external knowledge. In *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, pages 754–763, 2021.

Rex Ying, Ruining He, Kaifeng Chen, Pong Eksombatchai, William L Hamilton, and Jure Leskovec. Graph convolutional neural networks for web-scale recommender systems. In *Proceedings of the 24th ACM SIGKDD international conference on knowledge discovery & data mining*, pages 974–983, 2018.

Shiwen Wu, Fei Sun, Wentao Zhang, Xu Xie, and Bin Cui. Graph neural networks in recommender systems: a survey. *ACM Computing Surveys (CSUR)*.

Thomas Magelinski, David Beskow, and Kathleen M Carley. Graph-hist: Graph classification from latent feature histograms with application to bot detection. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 34, pages 5134–5141, 2020.

Zhenyu Lei, Herun Wan, Wenqian Zhang, Shangbin Feng, Zilong Chen, Qinghua Zheng, and Minnan Luo. Bic: Twitter bot detection with text-graph interaction and semantic consistency. *arXiv preprint arXiv:2208.08320*, 2022.

Alexander Ratner, Stephen H Bach, Henry Ehrenberg, Jason Fries, Sen Wu, and Christopher Ré. Snorkel: Rapid training data creation with weak supervision. In *Proceedings of the VLDB Endowment. International Conference on Very Large Data Bases*, volume 11, page 269. NIH Public Access, 2017.

Justus J Randolph. Free-marginal multirater kappa (multirater k [free]): An alternative to fleiss' fixed-marginal multirater kappa. *Online submission*, 2005.

Petar Veličković, Guillem Cucurull, Arantxa Casanova, Adriana Romero, Pietro Liò, and Yoshua Bengio. Graph attention networks. In *International Conference on Learning Representations*, 2018.

Zafar Gilani, Reza Farahbakhsh, Gareth Tyson, Liang Wang, and Jon Crowcroft. Of bots and humans (on twitter). In *Proceedings of the 2017 IEEE/ACM International Conference on Advances in Social Networks Analysis and Mining 2017*, pages 349–354, 2017.

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

Stefano Cresci, Fabrizio Lillo, Daniele Regoli, Serena Tardelli, and Maurizio Tesconi. Fake: Evidence of spam and bot activity in stock microblogs on twitter. In *Twelfth international AAAI conference on web and social media*, 2018.

Stefano Cresci, Fabrizio Lillo, Daniele Regoli, Serena Tardelli, and Maurizio Tesconi. Cashtag piggybacking: Uncovering spam and bot activity in stock microblogs on twitter. *ACM Transactions on the Web (TWEB)*, 13(2):1–27, 2019.

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

David M Beskow and Kathleen M Carley. Bot-hunter: a tiered approach to detecting & characterizing automated activity on twitter. In *Conference paper. SBP-BRiMS: International Conference on Social Computing, Behavioral-Cultural Modeling and Prediction and Behavior Representation in Modeling and Simulation*, volume 3, page 3, 2018.

David M Beskow and Kathleen M Carley. Its all in a name: detecting and labeling bots by their name. *Computational and Mathematical Organization Theory*, 25(1):24–35, 2019.

Jefferson Viana Fonseca Abreu, Célia Ghedini Ralha, and João José Costa Gondim. Twitter bot detection with reduced feature set. In *2020 IEEE International Conference on Intelligence and Security Informatics (ISI)*, pages 1–6. IEEE, 2020.

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

Yinhan Liu, Myle Ott, Naman Goyal, Jingfei Du, Mandar Joshi, Danqi Chen, Omer Levy, Mike Lewis, Luke Zettlemoyer, and Veselin Stoyanov. Roberta: A robustly optimized bert pretraining approach. *arXiv preprint arXiv:1907.11692*, 2019.

Colin Raffel, Noam Shazeer, Adam Roberts, Katherine Lee, Sharan Narang, Michael Matena, Yanqi Zhou, Wei Li, Peter J Liu, et al. Exploring the limits of transfer learning with a unified text-to-text transformer. *J. Mach. Learn. Res.*, 21(140):1–67, 2020.

Phillip George Efthimion, Scott Payne, and Nicholas Proferes. Supervised machine learning bot detection techniques to identify social twitter bots. *SMU Data Science Review*, 1(2):5, 2018.

Mücahit Kantepe and Murat Can Ganiz. Preprocessing framework for twitter bot detection. In *2017 International conference on computer science and engineering (ubmk)*, pages 630–634. IEEE, 2017.

Onur Varol, Emilio Ferrara, Clayton Davis, Filippo Menczer, and Alessandro Flammini. Online human-bot interactions: Detection, estimation, and characterization. In *Proceedings of the international AAAI conference on web and social media*, volume 11, 2017.

Maria Kouvela, Ilias Dimitriadis, and Athena Vakali. Bot-detective: An explainable twitter bot detection service with crowdsourcing functionalities. In *Proceedings of the 12th International Conference on Management of Digital EcoSystems*, pages 55–63, 2020.

Eric Ferreira Dos Santos, Danilo Carvalho, Livia Ruback, and Jonice Oliveira. Uncovering social media bots: a transparency-focused approach. In *Companion Proceedings of The 2019 World Wide Web Conference*, pages 545–552, 2019.

Juan Echeverrï£¡ a, Emiliano De Cristofaro, Nicolas Kourtellis, Ilias Leontiadis, Gianluca Stringhini, and Shi Zhou. Lobo: Evaluation of generalization deficiencies in twitter bot classifiers. In *Proceedings of the 34th Annual Computer Security Applications Conference*, pages 137–146, 2018.

Samaneh Hosseini Moghaddam and Maghsoud Abbaspour. Friendship preference: Scalable and robust category of features for social bot detection. *IEEE Transactions on Dependable and Secure Computing*, 2022.

Kai-Cheng Yang, Emilio Ferrara, and Filippo Menczer. Botometer 101: Social bot practicum for computational social scientists. *arXiv preprint arXiv:2201.01608*, 2022.

Jorge Rodríguez-Ruiz, Javier Israel Mata-Sánchez, Raúl Monroy, Octavio Loyola-González, and Armando López-Cuevas. A one-class classification approach for bot detection on twitter. *Computers & Security*, 91:101715, 2020.

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

Ziniu Hu, Yuxiao Dong, Kuansan Wang, and Yizhou Sun. Heterogeneous graph transformer. In *Proceedings of The Web Conference 2020*, pages 2704–2710, 2020.

Qingsong Lv, Ming Ding, Qiang Liu, Yuxiang Chen, Wenzheng Feng, Siming He, Chang Zhou, Jianguo Jiang, Yuxiao Dong, and Jie Tang. 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*, pages 1150–1160, 2021.

Lynnette Hui Xian Ng, Dawn C Robertson, and Kathleen M Carley. Stabilizing a supervised bot detection algorithm: How much data is needed for consistent predictions? *Online Social Networks and Media*, 28:100198, 2022.

Tomas Mikolov, Kai Chen, Greg Corrado, and Jeffrey Dean. Efficient estimation of word representations in vector space. *arXiv preprint arXiv:1301.3781*, 2013.

Adam Paszke, Sam Gross, Francisco Massa, Adam Lerer, James Bradbury, Gregory Chanan, Trevor Killeen, Zeming Lin, Natalia Gimelshein, Luca Antiga, Alban Desmaison, Andreas Kopf, Edward Yang, Zachary DeVito, Martin Raison, Alykhan Tejani, Sasank Chilamkurthy, Benoit Steiner, Lu Fang, Junjie Bai, and Soumith Chintala. Pytorch: An imperative style, high-performance deep learning library. In H. Wallach, H. Larochelle, A. Beygelzimer, F. d'Alché-Buc, E. Fox, and R. Garnett, editors, *Advances in Neural Information Processing Systems*, volume 32. Curran Associates, Inc., 2019. URL `https://proceedings.neurips.cc/paper/2019/file/bdbca288fee7f92f2bfa9f7012727740-Paper.pdf`.

William Falcon and The PyTorch Lightning team. PyTorch Lightning, 3 2019. URL `https://github.com/PyTorchLightning/pytorch-lightning`.

Kay Liu, Yingtong Dou, Yue Zhao, Xueying Ding, Xiyang Hu, Ruitong Zhang, Kaize Ding, Canyu Chen, Hao Peng, Kai Shu, et al. Pygod: A python library for graph outlier detection. *arXiv preprint arXiv:2204.12095*, 2022.

Thomas Wolf, Lysandre Debut, Victor Sanh, Julien Chaumond, Clement Delangue, Anthony Moi, Pierric Cistac, Tim Rault, Remi Louf, Morgan Funtowicz, Joe Davison, Sam Shleifer, Patrick von Platen, Clara Ma, Yacine Jernite, Julien Plu, Canwen Xu, Teven Le Scao, Sylvain Gugger, Mariama Drame, Quentin Lhoest, and Alexander Rush. Transformers: State-of-the-art natural language processing. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing: System Demonstrations*, pages 38–45, Online, October 2020. Association for Computational Linguistics. doi: 10.18653/v1/2020.emnlp-demos.6. URL `https://aclanthology.org/2020.emnlp-demos.6`.

Matthias Fey and Jan Eric Lenssen. Fast graph representation learning with pytorch geometric. *arXiv preprint arXiv:1903.02428*, 2019.

F. Pedregosa, G. Varoquaux, A. Gramfort, V. Michel, B. Thirion, O. Grisel, M. Blondel, P. Prettenhofer, R. Weiss, V. Dubourg, J. Vanderplas, A. Passos, D. Cournapeau, M. Brucher, M. Perrot, and E. Duchesnay. Scikit-learn: Machine learning in Python. *Journal of Machine Learning Research*, 12:2825–2830, 2011.

Radim Řehůřek and Petr Sojka. Software Framework for Topic Modelling with Large Corpora. In *Proceedings of the LREC 2010 Workshop on New Challenges for NLP Frameworks*, pages 45–50, Valletta, Malta, May 2010. ELRA. `http://is.muni.cz/publication/884893/en`.

Matthew Honnibal, Ines Montani, Sofie Van Landeghem, and Adriane Boyd. spaCy: Industrial-strength Natural Language Processing in Python. 2020. doi: 10.5281/zenodo.1212303.

Joshua Roesslein. tweepy documentation. *Online http://tweepy. readthedocs. io/en/v3*, 5, 2009.

Wes McKinney et al. pandas: a foundational python library for data analysis and statistics. *Python for high performance and scientific computing*, 14(9):1–9, 2011.

Charles R Harris, K Jarrod Millman, Stéfan J Van Der Walt, Ralf Gommers, Pauli Virtanen, David Cournapeau, Eric Wieser, Julian Taylor, Sebastian Berg, Nathaniel J Smith, et al. Array programming with numpy. *Nature*, 585(7825):357–362, 2020.

Clayton Hutto and Eric Gilbert. Vader: A parsimonious rule-based model for sentiment analysis of social media text. In *Proceedings of the international AAAI conference on web and social media*, volume 8, pages 216–225, 2014.

Guillaume Lemaître, Fernando Nogueira, and Christos K. Aridas. Imbalanced-learn: A python toolbox to tackle the curse of imbalanced datasets in machine learning. *Journal of Machine Learning Research*, 18(17):1–5, 2017.

Checklist

1. For all authors...
   (a) Do the main claims made in the abstract and introduction accurately reflect the paper's contributions and scope? [Yes]
   (b) Have you read the ethics review guidelines and ensured that your paper conforms to them? [Yes]
   (c) Did you discuss any potential negative societal impacts of your work? [Yes] Section A.6
   (d) Did you describe the limitations of your work? [Yes] Section A.6
2. If you are including theoretical results...
   (a) Did you state the full set of assumptions of all theoretical results? [N/A]
   (b) Did you include complete proofs of all theoretical results? [N/A]
3. If you ran experiments...
   (a) Did you include the code, data, and instructions needed to reproduce the main experimental results (either in the supplemental material or as a URL)? [Yes] Please refer to the TwiBot-22 GitHub repository listed in Section A.6.
   (b) Did you specify all the training details (e.g., data splits, hyperparameters, how they were chosen)? [Yes] Please refer to the TwiBot-22 GitHub repository listed in Section A.6.
   (c) Did you report error bars (e.g., with respect to the random seed after running experiments multiple times)? [Yes]
   (d) Did you include the amount of compute and the type of resources used (e.g., type of GPUs, internal cluster, or cloud provider)? [Yes] Section B.6
4. If you are using existing assets (e.g., code, data, models) or curating/releasing new assets...
   (a) If your work uses existing assets, did you cite the creators? [Yes]
   (b) Did you mention the license of the assets? [Yes] Section A.6
   (c) Did you include any new assets either in the supplemental material or as a URL? [Yes] We provide the TwiBot-22 dataset as a new asset and provide URLs in Section A.6.
   (d) Did you discuss whether and how consent was obtained from people whose data you're using/curating? [Yes] We strictly follow the original license of existing datasets and rules in the Twitter Developer Agreement and Policy.
   (e) Did you discuss whether the data you are using/curating contains personally identifiable information or offensive content? [Yes] Our dataset does not contain the information of private and protected users. Since TwiBot-22 aims to facilitate bot detection research and certain bots are designed to be offensive, there might be offensive content.
5. If you used crowdsourcing or conducted research with human subjects...
   (a) Did you include the full text of instructions given to participants and screenshots, if applicable? [N/A]
   (b) Did you describe any potential participant risks, with links to Institutional Review Board (IRB) approvals, if applicable? [N/A]
   (c) Did you include the estimated hourly wage paid to participants and the total amount spent on participant compensation? [N/A]

Table 4: Entities in the TwiBot-22 heterogeneous graph.

Entity Name	Description	Main Metadata
User	Users are the most important entity on Twittersphere.	created at, description, entities location, name, profile image url protected, url, username verified, withheld, followers count following count, tweet count, listed count
Tweet	Users post tweets to share their thoughts and interact with other users.	attachments, context annotations, entities created at, geo, lang, possibly sensitive referenced tweets, reply settings source, text, withheld, retweet count reply count, like count, quote count
List	A list is curated feeds from selected users that allow you to listen to relevant discussions or influencers.	private, created at, description name, follower count, member count
Hashtag	A hashtag is a metadata tag that is prefaced by "#". It is used to link tweets with the same theme together.	hashtag name

Table 5: Relations in the TwiBot-22 heterogeneous graph.

Relation	Source Entity	Target Entity	Description
following	user	user	user A follows user B
follower	user	user	user A is followed by user B
post	user	tweet	user A posts tweet B
pin	user	tweet	user A pins tweet B
like	user	tweet	user A likes tweet B
mention	tweet	user	tweet A mentions user B
retweet	tweet	tweet	tweet A retweets tweet B
quote	tweet	tweet	tweet A quotes tweet B with comments
reply	tweet	tweet	tweet A replies to tweet B
own	user	list	user A is the creator of list B
member	user	list	user A is a member of list B
follow	user	list	user A follows list B
contain	list	tweet	list A contains tweet B
discuss	tweet	hashtag	tweet A discussed hashtag B

A TwiBot-22 Details

A.1 Entities and Relations

TwiBot-22 collects four types of entities on the Twitter social network: user, tweet, list, and hashtag. The detailed information of these entities is shown in Table 4 while a complete list of all relation types in TwiBot-22 is presented in Table 5.

A.2 Data Collection Details

The complete process of data collection. For the first stage of user network collection, we adopt *@NeurIPSConf* as the starting user. We use the Twitter API to retrieve 1,000 followers and 1,000 followees as the user's neighborhood for BFS expansion. We randomly adopt one of the two sampling strategies (distribution diversity or value diversity) and randomly select one metadata from Table 6 to include 6 users from its neighborhood into the TwiBot-22 dataset. We then randomly select one unexpanded user in TwiBot-22 for a new round of neighborhood expansion. For the second stage of heterogeneous graph building, we first collect 1,000 tweets for the user in the user network, and

Table 6: User metadata adopted in diversity-aware sampling.

Metadata Name	Description	Type
active days	days between user creation time and collected time	numerical
following count	number of user followings	numerical
followers count	number of user followers	numerical
tweet count	number of user tweets	numerical
listed count	number of user lists	numerical
verified	whether the user is verified or not	true-or-false
homepage url	whether user has urls in homepage or not	true-or-false

Table 7: Statistics of TwiBot-22.

Item	Value	Item	Value	Item	Value
entity type	4	post	88,217,457	following	2,626,979
relation type	14	pin	347,131	follower	1,116,655
user	1,000,000	like	595,794	contain	1,998,788
hashtag	5,146,289	mention	4,759,388	discuss	66,000,633
list	21,870	retweet	1,580,643	bot	139,943
tweet	88,217,457	quote	289,476	human	860,057
user metadata	17	reply	1,114,980	entity	92,932,326
hashtag metadata	2	own	21,870	relation	170,185,937
list metadata	8	member	1,022,587	max degree	270,344
tweet metadata	20	follow	493,556	verified user	95,398

200 tweets for the user expanded from the user network. We collect the pinned tweet and the recent 100 liked tweet of each user in TwiBot-22. For each tweet we collect now, we collect the tweets it retweets, quotes, or replies and the users it mentions. We collect a user's recent 100 lists with the newest 100 members, followers, and tweets. We collect all hashtags in the tweets and search for more tweets related to a hashtag using Twitter API. Finally, we make sure that the creator of each tweet is collected and collect 40 tweets of these users according to Ng et al. [2022] for stable benchmarking

Data collection time. The first stage of user network collection is conducted from January 20th, 2022 to February 1st, 2022. The second stage of heterogeneous graph building is conducted from February 1st, 2022 to March 15th, 2022.

A.3 Expert Annotation Details

We invite 17 researchers in our group who are active Twitter users, are familiar with bot detection literature, and have conducted experiments with the TwiBot-20 datasets. We then assign each Twitter user in TwiBot-22 to 5 different experts and ask them to evaluate whether the user is a human, a bot, or not sure. We use majority voting to obtain the expert annotations of these 1,000 users, which are then leveraged to guide the weak supervision leaning process.

A.4 Dataset Statistics

Table 7 presents important statistics about the TwiBot-22 benchmark.

A.5 Annotation Quality Study Details

To compare the annotation quality of TwiBot-22 and TwiBot-20, we ask 6 researchers to participate in an expert study. They are familiar with Twitter bot detection research and most of them have previously published on this topic. Specifically, we randomly select 500 users from TwiBot-20 and TwiBot-22 respectively and assign each user to 3 experts. We then ask them to evaluate each user as "definitely bot", "likely bot", "not sure", "likely human", and "definitely human". Based on their evaluations, we calculate the accuracy and F1-score between expert opinions and dataset labels. We also report the Randolph's Kappa Coefficient [Randolph, 2005], which models agreement between experts in Figure 2(c).

A.6 Other TwiBot-22 Details

Dataset documentation. We encourage the readers to refer to the TwiBot-22 evaluation framework (`https://github.com/LuoUndergradXJTU/TwiBot-22`) for the documentation of TwiBot-22 and the TwiBot-22 evaluation framework.

Intended use. TwiBot-22 should be used for research in Twitter bot detection and social network analysis.

Relevent URLs. We list all TwiBot-22 URLs in the following.

- **Official TwiBot-22 website** (`https://twibot22.github.io/`) is the main reference of TwiBot-22, presenting our dataset, paper, evaluation framework, and contact information.
- **TwiBot-22 repository** (`https://github.com/LuoUndergradXJTU/TwiBot-22`) hosts implemented codes for dataset preprocessing and 35 Twitter bot detection methods. The repository is well documented to facilitate reproducible research.
- **TwiBot-22 dataset** will be permanently hosted on our group Google Drive account (`https://drive.google.com/drive/folders/1YwiOUwtl8pCd2GD97Q_WEzwEUtSPoxFs?usp=sharing`).

Hosting and maintenance. The TwiBot-22 dataset will be hosted via Google Drive and regularly maintained by our research group. The TwiBot-22 evaluation framework will be hosted on GitHub. Our team will continue to add new datasets and baselines with the help of the research community.

Licensing. The TwiBot-22 dataset uses the CC BY-NC-ND 4.0 license. Implemented code in the TwiBot-22 evaluation framework uses the MIT license.

Author statement. We bear all responsibility in case of violation of rights, etc., and confirmation of the data license.

Limitations. One minor limitation of TwiBot-22 is that we do not download and store user media (images and videos) in TwiBot-22, while these multimedia content might be useful for bot detection. However, if researchers do deem multimedia content as necessary for bot detection, they can download with media links in TwiBot-22 by themselves.

Potential negative societal impact. Although TwiBot-22 and the TwiBot-22 evaluation framework are designed to facilitate bot detection research and improve bot detection models, it might be abused by bot operators to examine the characteristics of bots that evade detection, and thus designing bot algorithms that are more evasive. We need to make sure that the TwiBot-22 dataset and evaluation framework should not be abused to design advanced Twitter bots.

B Experiment Details

B.1 Baseline Details

We briefly describe each of the 35 Twitter bot detection baseline methods:

- **SGBot** [Yang et al., 2020]. SGBot is proposed to address the scalability and generalization problem in Twitter bot detection. SGBot leverages 8 types of user metadata such as status count and 12 derived features such as tweet frequency and adopts random forest to identify bot.
- **Kudugunta *et al.*** [Kudugunta and Ferrara, 2018]. The baseline addresses two challenges, account-level classification and tweet-level classification. In the task of account-level classification, the baseline introduces a technique that combines synthetic minority oversampling (SMOTE) with undersampling techniques. In the task of tweet-level classification, the baseline introduces an architecture called contextual LSTM.

- **Hayawi *et al.*** [Hayawi et al., 2022]. DeeProBot, which is short for Deep Profile-based Bot detection Framework, utilizes different types of features including user's numerical or binary metadata and user's description, making the model more comprehensive. DeeProBot uses GLoVe word embeddings to get the embedding of the textual information. LSTM and dense layers are then used to learn user representations for bot detection.
- **BotHunter** [Beskow and Carley, 2018]. In this work, the features are composed of user attributes, network attributes, contents, and timing information. After extracting the above features, random forest is exploited as the classifier.
- **NameBot** [Beskow and Carley, 2019]. NameBot utilizes twitter username as the only classification basis and extracts numerical features such as TF-IDF. The extracted numeric features are then exploited as input of a linear regression classifier. While achieving great good accuracy on training set, this basline shows poor transferbility on new datasets.
- **Abreu *et al.*** [Abreu et al., 2020]. This model chooses five essential Twitter user features to conduct relative experiments. They calculated accuracy, AUC, recall and F1-score on several existing datasets with four machine learning algorithms.
- **Cresci *et al.*** [Cresci et al., 2016]. This method encodes different action types with different characters, thus representing usernames by strings. Users who share the longest common substring are considers as bots have similar behaviors.
- **Wei *et al.*** [Wei and Nguyen, 2019]. This paper use pre-trained GloVe word vectors on Twitter as word embedding. Multiple layers of bidirectional LSTMs are used for bot detection.
- **BGSRD** [Guo et al., 2021a]. This model utilizes BERT and GCN to achieve social bots detection. In specific, this paper use word and user description as graph nodes, there are no connection within users and words. In training steps, BGSRD first uses BERT (Roberta) to process users' description, the output of BERT layer is the initial feature of graph nodes, and the initial feature of words node is zeros. The final prediction is the combination of BERT and GAT output.
- **RoBERTa** [Liu et al., 2019]. This baseline leverages pre-trained language model RoBERTa to encode user tweets and descriptions, then feed them to an MLP to distinguish bots from human.
- **T5** [Raffel et al., 2020]. In this baseline, we use pretained language model T5-small to encode tweets and descriptions, then feed them into an MLP classifier.
- **Efthimion *et al.*** [Efthimion et al., 2018]. This paper leverages a wide range of users' features including length of user names, reposting rate, temporal patterns, sentiment expression, followers-to-friends ratio, and message variability for bot detection. Logistic regression and support vector machine are applied successively for profile and account activity analysis. Levenshtein distance is applied for text mining.
- **Kantepe *et al.*** [Kantepe and Ganiz, 2017]. This paper explores the importance of features by comparing sixty-two different features related to Twitter account properties and tweet contents, and selects the most important eleven features through experimental comparison. The classification task was then performed using 62 or 11 features as input with classifier like GBDT or SVM.
- **Miller *et al.*** [Miller et al., 2014]. Miller et al. proposed a clustering based method to detect spam accounts in twitter. Specifically, they exploit classic clustering algorithms such as DBSCAN and K-MEANS on human accounts to obtain human clusters. In test phase, users whose clusters can not be filed under any of the existing human clusters are consider as bots.
- **Varol *et al.*** [Varol et al., 2017]. User metadata, tweet content, friends, sentiment and network statistics are adapted as user's features, then the extracted features are utilized as inputs of a random forest model.
- **Kouvela *et al.*** [Kouvela et al., 2020]. This baseline leverages user features and content features from each user and classifies users with random forest. Specifically, it uses 36 features from each account and the content features are extracted from the latest 20 tweets.
- **Santos *et al.*** [Ferreira Dos Santos et al., 2019]. This baseline extracts 16 features from users' tweets and descriptions and feed the features into a decision tree for classification.
- **Lee *et al.*** [Lee et al., 2011]. This method introduces the social honeypots to attract bot users by manipulating the honeypot users' tweets frequency and social network structure. After analyzing the data collected by social honeypots, 18 features are selected and fed into the random forest classifier.

- **LOBO** [Echeverrï£¡ a et al., 2018]. This baseline extracts 19 features (26 on Twibot-22) from each user and adopts random forest for classification.
- **Moghaddam *et al.*** [Moghaddam and Abbaspour, 2022]. This model combines profile-based features and friendship preference features, which compares the distribution of followers' features and sub-population of accounts.
- **Alhosseini *et al.*** [Ali Alhosseini et al., 2019]. This model uses age, statuses_count, account length name, followers_count, friends_count and favourites_count as user features and feed them into a GCN layer to identify bot users.
- **Knauth *et al.*** [Knauth, 2019]. This paper extracts features from user's meta data, tweets, user behavior, and feeds these features into Adaboost classifier.
- **FriendBot** [Beskow and Carley, 2020]. This paper introduces network metrics into twitter bot detection tasks. Specifically, they construct a 2-hop ego network for each twitter account based on four types of relations: following, retweet, mention, and reply. They collect account features based on metrics of these ego networks. Finally, they exploit random forest algorithm for classification.
- **SATAR** [Feng et al., 2021a]. SATAR is a self-supervised representation learning framework of Twitter users. SATAR jointly uses semantic, property, and neighborhood information and adopts a co-influence module to aggregate these information. SATAR considers the follower count as self-supervised label to pre-train parameters and fine-tune parameters in bot detection task.
- **Botometer** [Yang et al., 2022]. Botometer (formerly BotOrNot) is a public website to check the activity of a Twitter account and give it a score, where higher scores mean more bot-like activity. Botometer's classification system leverages more than 1,000 features using available meta-data and information extracted from interaction patterns and content.
- **Rodriguez-Ruiz *et al.*** [Rodríguez-Ruiz et al., 2020]. This paper designs a one-class classification model, which uses the social network and tweet information of Twitter users to extract 13 features for feature engineering, and the model has also achieved good classification results.
- **GraphHist** [Magelinski et al., 2020]. The authors design a new graph classifier based on histogram and customized backward operator. By exploiting the ego-graph of twitter users, bot detection can be solved by utilizing the proposed graph-level classifier.
- **EvolveBot** [Yang et al., 2013]. This method designs 11 robust features, together with 7 efficient features to combat evasion tatics of spammers.
- **Dehghan *et al.*** [Dehghan et al., 2022]. This baseline combines the profile features, text features, and graph features for bot detection. After obtaining account representations through Deepwalk and struc2vec, XGBclassifier is applied to identify bot users.
- **GCN** [Kipf and Welling, 2016]. GCN aggregates features from neighbors equally and learns representation for each user. These representations are passed to an MLP for classification. The initial user features are identical with BotRGCN.
- **GAT** [Veličković et al., 2018]. Graph Attention Network (GAT) introduces attention mechanism to GNN models, making it capable of distinguishing the importance of neighboring users in aggregation. Same as GCN, it can learn user representations and feed them into an MLP for classification. The initial user features are identical with BotRGCN.
- **HGT** [Hu et al., 2020]. Heterogeneous Graph Transformers (HGT) is a dedicated heterogeneous GNN that mainly consists of two modules, heterogeneous mutual attention and heterogeneous message passing. Heterogeneous mutual attention considers the edge type and source and target node type when calculating attention scores. Heterogeneous message passing module incorporates the source node type and the edge dependency in passed messages. The initial user features are identical with BotRGCN.
- **simpleHGN** [Lv et al., 2021]. SimpleHGN is a simple yet effective GNN for heterogeneous graph inspired by the GAT. SimpleHGN adopts three strategies to enhance GAT, learnable embedding for each edge-type, node-level and edge-level residual connections as well as the L2 regularization on output representations. The initial user features are identical with BotRGCN.
- **BotRGCN** [Feng et al., 2021b]. BotRGCN utilizes the text information from user descriptions and tweets, as well as numerical and categorical user property information. Then BotRGCN constructs a heterogeneous graph from the Twitter network based on user relationships and relational graph convolutional networks (R-GCN) is applied to learn user representations for bot detection tasks.

Table 8: Average model performance (F1-score) and standard deviation of 35 baseline methods on 9 datasets. **Bold** and underline indicate the highest and second highest performance. The F, T, and G in the "Type" column stands for feature, text, and graph. Cresci et al. and Botometer are deterministic methods without standard deviation. / indicates that the dataset could not support the baseline. - indicates that the baseline could not scale to the largest TwiBot-22 dataset.

Method	Type	C-15	G-17	C-17	M-18	C-S-18	C-R-19	B-F-19	TwiBot-20	TwiBot-22
SGBot	F	77.9 (±0.1)	72.1 (±1.2)	94.6 (±0.2)	99.5 (±0.0)	82.3 (±0.1)	82.7 (±1.7)	49.6 (±3.4)	84.9 (±0.4)	36.6 (±0.2)
Kudugunta *et al.*	F	75.3 (±0.2)	49.8 (±2.1)	91.7 (±0.2)	94.5 (±0.3)	50.9 (±0.4)	49.2 (±1.3)	49.6 (±8.2)	47.3 (±1.4)	51.7 (±0.0)
Hayawi *et al.*	F	85.6 (±0.0)	34.7 (±0.1)	93.8 (±0.0)	91.5 (±0.0)	60.8 (±0.1)	60.9 (±0.0)	20.5 (±0.1)	77.1 (±0.0)	24.7 (±0.1)
BotHunter	F	97.2 (±1.0)	69.2 (±1.0)	91.6 (±0.1)	**99.6** (±0.0)	82.2 (±0.2)	82.9 (±1.9)	49.6 (±3.1)	79.1 (±0.4)	23.5 (±0.1)
NameBot	F	83.4 (±0.0)	44.8 (±0.0)	85.7 (±0.0)	91.6 (±0.0)	61.1 (±0.0)	67.5 (±0.0)	38.5 (±0.0)	65.1 (±0.1)	0.5 (±0.0)
Abreu *et al.*	F	76.4 (±0.1)	66.7 (±0.1)	95.0 (±0.1)	97.9 (±0.1)	76.9 (±0.1)	**83.5** (±0.1)	**53.8** (±0.1)	77.1 (±0.1)	53.4 (±0.1)
Cresci *et al.*	T	1.17	/	22.8	/	/	/	/	13.7	-
Wei *et al.*	T	82.7 (±2.2)	/	78.4 (±1.7)	/	/	/	/	57.3 (±3.1)	53.6 (±1.3)
BGSRD	T	90.8 (±0.6)	35.7 (±32.6)	86.3 (±0.0)	90.5 (±1.0)	58.2 (±12.0)	41.1 (±13.0)	13.0 (±13.0)	70.0 (±2.6)	21.1 (±29.0)
RoBERTa	T	95.8 (±0.1)	/	94.3 (±0.1)	/	/	/	/	73.1 (±0.5)	20.5 (±1.7)
T5	T	89.3 (±0.2)	/	92.3 (±0.1)	/	/	/	/	70.5 (±0.3)	20.2 (±2.0)
Efthimion *et al.*	FT	94.1 (±0.0)	5.2 (±0.0)	91.8 (±0.0)	95.9 (±0.0)	68.2 (±0.0)	71.7 (±0.0)	0.0 (±0.0)	67.2 (±0.0)	27.5 (±0.0)
Kantepe *et al.*	FT	78.2 (±1.4)	/	79.4 (±1.3)	/	/	/	/	62.2 (±2.1)	**58.7** (±1.6)
Miller *et al.*	FT	83.8 (±0.0)	59.9 (±0.0)	86.8 (±0.1)	91.1 (±0.0)	56.8 (±0.0)	43.6 (±0.0)	0.0 (±0.0)	74.8 (±0.3)	45.3 (±0.0)
Varol *et al.*	FT	94.7 (±0.4)	/	/	/	/	/	/	81.1 (±0.5)	27.5 (±0.3)
Kouvela *et al.*	FT	98.2 (±0.4)	66.6 (±1.7)	99.1 (±0.1)	98.2 (±0.1)	80.4 (±0.2)	81.1 (±1.0)	28.1 (±5.3)	86.5 (±0.3)	30.0 (±0.0)
Santos *et al.*	FT	78.8 (±0.0)	14.5 (±0.0)	83.0 (±0.0)	92.4 (±0.0)	65.2 (±0.0)	75.7 (±0.0)	21.0 (±0.0)	60.3 (±0.0)	-
Lee *et al.*	FT	98.6 (±0.1)	67.8 (±1.8)	**99.3** (±0.0)	97.9 (±0.1)	82.5 (±0.4)	82.7 (±1.8)	50.3 (±3.2)	80.0 (±0.5)	30.4 (±0.2)
LOBO	FT	**98.8** (±0.3)	/	97.7 (±0.2)	/	/	/	/	80.8 (±0.2)	38.6 (±0.2)
Moghaddam *et al.*	FG	73.9 (±0.2)	/	/	/	/	/	/	79.9 (±0.7)	32.1 (±0.0)
Alhosseini *et al.*	FG	92.2 (±0.4)	/	/	/	/	/	/	72.0 (±0.5)	38.1 (±5.9)
Knauth *et al.*	FTG	91.2 (±0.0)	39.1 (±0.0)	93.4 (±0.0)	91.3 (±0.0)	**94.0** (±0.0)	54.2 (±0.0)	41.3 (±0.0)	85.2 (±0.0)	37.1 (±0.0)
FriendBot	FTG	97.6 (±0.8)	/	87.4 (±0.5)	/	/	/	/	80.0 (±0.3)	-
SATAR	FTG	95.0 (±0.3)	/	/	/	/	/	/	86.1 (±0.7)	-
Botometer	FTG	66.9	**77.4**	96.1	46.0	79.6	79.0	30.8	53.1	42.8
Rodrifuez-Ruiz *et al.*	FTG	87.7 (±0.0)	/	85.7 (±0.0)	/	/	/	/	63.1 (±0.1)	56.6 (±0.0)
GraphHist	FTG	84.5 (±8.2)	/	/	/	/	/	/	67.6 (±0.3)	-
EvolveBot	FTG	90.1 (±2.0)	/	/	/	/	/	/	69.7 (±0.5)	14.1 (±0.1)
Dehghan *et al.*	FTG	88.3 (±0.0)	/	/	/	/	/	/	76.2 (±0.0)	-
GCN	FTG	97.2 (±0.0)	/	/	/	/	/	/	80.8 (±0.0)	54.9 (±0.0)
GAT	FTG	97.6 (±0.0)	/	/	/	/	/	/	85.2 (±0.0)	55.8 (±0.0)
HGT	FTG	96.9 (±0.2)	/	/	/	/	/	/	**88.2** (±0.2)	39.6 (±2.1)
SimpleHGN	FTG	97.5 (±0.4)	/	/	/	/	/	/	**88.2** (±0.2)	45.4 (±0.4)
BotRGCN	FTG	97.3 (±0.5)	/	/	/	/	/	/	87.3 (±0.7)	57.5 (±1.4)
RGT	FTG	97.8 (±0.2)	/	/	/	/	/	/	88.0 (±0.4)	42.9 (±0.5)

- **RGT** [Feng et al., 2022]. Relational Graph Transformers is a GNN framework that uses graph transformers and semantic attention network to model the intrinsic influence heterogeneity and relation heterogeneity in Twittersphere. Specifically, RGT first learns users' representation under each relation with graph transformers, then it aggregate representations from all relations using the semantic attention network.

B.2 F, T, or G?

We categorize baseline methods into F, T, or G with the following rules:

- If the baseline leverages user metadata and conduct feature engineering, the baseline is F.
- If the baseline leverages the content of tweets and user description texts, the baseline is T.
- If the baseline leverages the network structure of Twitter, the baseline is G.

Baseline methods may check multiple boxes and have multiple types. For exmample, BotRGCN [Feng et al., 2021b] is F since it selects user metadata and encode users as feature vectors. BotRGCN is T since it encodes user tweets and description with pre-trained RoBERTa. BotRGCN is G since it constructs a relational graph and adopts relational graph neural networks for bot detection. As a result, BotRGCN has the type of FTG.

Table 9: Example hashtags in the five hashtag-based sub-communities.

ID	Example Hashtags
1	#Christ, #Taliban, #Kabul, #Germany, #EU, #manufacturer, #Manchester, #Covid, #covid, #bitcoin, #Ukraine, #Kyiv, #Iowa, #farm, #health, #bullying, #Putin, #gerrymandering, #Covid19, #Labour
2	#cybersecurity, #CVE, #GCP, #marketing, #datacenter, #OSINT, #SMEs, #aerospace, #innovation, #science, #exoplanet, #log4j, #conservation, #farming, #biology, #chemistry, #agriculture, #growth, #aging, #dementia
3	#Curitiba, #Colombia, #inversiones, #emprender, #emprendedor, #negocios, #liderazgo, #bici, #ciclismo, #RRSS, #correr, #sueños, #metas, #familia, #inversión, #Fortalecimiento, #emprendimiento, #Familia, #éxito, #ventas
4	#Vimeo, #Industry, #Anonymous, #iHeartRadio, #Biomass, #Contest, #Books, #Humor, #Memoir, #Storytelling, #Butterfly, #Art, #Canvas, #Handbag, #Tshirt, #Kidney, #Passion, #Quarantine, #Whitelist, #PMC
5	#UCL, #coach, #FF, #USMNT, #Coventry, #Orpheus, #CRO, #Sydney, #Houston, #Jordan, #Buffalo, #UBC, #writer, #Shona, #Christchurch, #Antigua, #Sonny, #Gladstone, #500th, #Philips

Table 10: Statistics of the 10 sub-communities.

Sub-communities	0	1	2	3	4	5	6	7	8	9
# Human	5,000	5,000	5,000	5,000	5,000	5,000	5,000	5,000	5,000	5,000
# Bot	5,000	5,000	5,000	5,000	5,000	5,000	5,000	5,000	5,000	5,000
# User	10,000	10,000	10,000	10,000	10,000	10,000	10,000	10,000	10,000	10,000
# Tweet	969,979	942,020	1,099,962	989,536	1,083,655	1,156,640	1,333,018	1,138,480	1,151,362	1,142,717
# Edge	1,116,208	1,120,637	1,245,190	1,167,285	1,249,535	1,535,397	1,924,616	1,508,054	1,511,824	1,526,627

B.3 F1-score Results

We re-implement 35 Twitter bot detection baselines and evaluate them on 9 representative datasets and benchmarks. We present their detection accuracy in Table 2 and F1-score in Table 8.

B.4 Graph Component Removal Details

We remove the graph component in graph-based methods to examine the role of graphs in Twitter bot detection and present results in Table 3. We provide details about how graphs are removed from each baseline as follows:

- **Alhosseini *et al.***. We remove the GCNs while using two MLP layers with user features.
- **Moghaddam *et al.***. We remove 11 graph-based features from the "friend preference" section.
- **Knauth *et al.***. We remove 2 graph-based features, namely friend count and follower count.
- **EvolveBot**. We remove 4 graph-based feature extracted with the help of neighbor information.
- **BotRGCN** and **RGT**. We remove the R-GCN and RGT while using two MLP layers with user features for bot detection. BotRGCN and RGT use the same user features so that their "w/o graph" results are identical.

B.5 Generalization Study Details

To evaluate existing methods and their ability to generalize on unseen data, we identify 10 sub-communities in the TwiBot-22 network and conduct experiments in Figure 3. Specifically, we firstly select 5 closely connected sub-communities around @*BarackObama*, @*elonmusk*, @*CNN*, @*NeurIPSConf*, and @*ladygaga*. These five users feature different interest domains and their neighborhood represents different aspects of the Twitter network. In addition, we use K-means to cluster the word2vec [Mikolov et al., 2013] representations of hashtags and identify users tweeting about similar hashtags into 5 sub-communities. Examples of these hashtags in these sub-communities are presented in Table 9. The statistics of the 10 sub-communities are presented in Table 10.

B.6 Computation Details

We ran all experiments on a server with 8 GeForce RTX 2080 Ti GPUs. We run each experiment for five times and report the average model performance as well as standard deviation.

Table 11: We remove labeling functions in the annotation process and compare their results with the full annotation model.

labeling function	bot->bot	bot->human	human->human	human->bot	changed percentage
w/o adaboost	77,050	49,965	869,317	3,986	5.36%
w/o random forest	117,191	9,524	841,531	31754	4.13%
w/o MLP	109,152	17,563	796,159	77,126	9.46%
w/o GCN	120,925	5,790	833,776	39,509	4.54%
w/o GAT	123,397	3,318	824,436	48,849	5.22%
w/o RGCN	123,676	3,042	819,293	53,970	5.70%
w/o verify	122,177	4,538	873,101	184	0.47%
w/o keywords	123,880	2,835	840,142	33,143	3.60%

Table 12: We use the training set and validation set in TwiBot-22 while using the expert labels as the test set. We used 6 baseline methods for a quick evaluation. Test 1 indicates using only the 1,000 manually annotated users in Section 3.2, and test 2 indicates using only the 500 manually annotated users in Section 3.3.

Model	test set 1				test set 2			
	Accuracy	F1-score	Precision	Recall	Accuracy	F1-score	Precision	Recall
Moghaddam *et al.*	89.41 (±0.30)	24.98 (±2.72)	16.57 (±1.97)	50.79 (±4.25)	83.93 (±0.28)	18.49 (±0.95)	11.58 (±0.59)	45.94 (±3.35)
SGBot	91.87 (±0.11)	47.43 (±1.21)	76.16 (±2.31)	34.48 (±1.56)	87.42 (±0.31)	26.00 (±2.80)	54.55 (±2.80)	17.11 (±2.28)
BotHunter	91.44 (±0.12)	40.39 (±0.32)	78.28 (±3.11)	27.24 (±0.52)	85.63 (±0.31)	23.38 (±1.55)	73.67 (±9.81)	13.95 (±1.18)
GAT	91.14 (±0.45)	47.00 (±2.92)	64.83 (±4.31)	36.95 (±3.04)	84.93 (±0.23)	30.47 (±2.64)	55.64 (±2.02)	21.05 (±2.46)
BotRGCN	88.74 (±0.29)	65.89 (±1.62)	79.82 (±2.53)	56.23 (±3.24)	85.59 (±0.68)	55.45 (±2.77)	67.45 (±2.74)	47.17 (±3.65)
RGT	92.80 (±0.45)	23.39 (±4.61)	58.33 (±11.78)	16.44 (±2.98)	87.10 (±1.19)	38.02 (±7.21)	58.50 (±10.18)	28.57 (±6.68)

B.7 Annotation Bias Test

To study the effect of individual labeling function, we remove each of them and examine how many labels have changed in the snorkel-based annotation process, as is shown in Table 11.

Experiment results on using the expert labels as the test set is presented in Table 12.

B.8 Implementation Details

The TwiBot-22 evaluation framework is built with help of many valuable scientific artifacts, including pytorch [Paszke et al., 2019], pytorch lightning [Falcon and The PyTorch Lightning team, 2019], pygod [Liu et al., 2022], transformers [Wolf et al., 2020], pytorch geometric [Fey and Lenssen, 2019], sklearn [Pedregosa et al., 2011], gensim [Řehůřek and Sojka, 2010], spacy [Honnibal et al., 2020], tweepy [Roesslein, 2009], pandas [McKinney et al., 2011], numpy [Harris et al., 2020], vaderSentiment [Hutto and Gilbert, 2014], and imbalanced-learn [Lemaître et al., 2017].