A Better Twitter Bot Detection

Devon Hockley, Wahaj Hassan, Jacob Stanich

December 2021

Abstract

In this paper, we present a Twitter bot detection system that identifies whether a Twitter user account is used by a real person or is an automated software generated and controlled bot account. In social media sites like Twitter, malicious Bot account are problematic because they are able to cause the spread of misinformation, manipulate political information and even influence elections. Therefore, detecting these bots accurately and deleting them is vital in the protection of the Twitter users. In order to help solve this issue, we used a recently developed, high difficulty dataset to evaluate a set of features for a bot detection model. From these features, we selected a subset of them using different methods to create an optimized bot detection system.

1 Introduction

Twitter is a social networking and online news site in which people use 'tweets' (short messages) to communicate with one another. As it is one of the top social networking sites in America, it boasts about 396.5 million users [4], and with those users come many human-made bots. These Twitter bots are software controlled user accounts, that are created for mass tweeting (submitting a post), retweeting (republishing a post), following other users and to make themselves look an average normal Twitter user account, with a purpose to achieve certain goals on huge scale [8].

While these automated user accounts can be used for good things such as promoting your content, broadcasting real time weather emergencies, automated audience engagement, etc., they can also be used for malicious purposes. These malicious purposes include spreading misinformation to instigate panic within the users, spreading fake news to confuse people, spreading malware on a big scale, influencing elections, circulating political propaganda and much more.

Twitter uses its own bot detection software to detect these malicious bots and slowly deletes them. However, the process is often very slow and by the time these bots are detected and deleted, they are likely to have already accomplished their purpose and been replaced by several newer bots. Our solution to this problem is to contribute to a better Twitter bot detecting software, through the use of combining multiple different models and detection techniques to make it more accurate and faster in detecting the software generated bots.

2 Previous Work

In this section we summarize some of the papers used for our project and how they helped improve our bot detection program.

2.1 Bot Detection Techniques (SMU)

Through this paper we are able to learn about certain features that bot accounts have that separate them from real user accounts. From its data collected about the user accounts from Figure 2 page 7 [6], it shows us that real users accounts have on average less than 1000 followers, unlike certain bot accounts which contain a huge number of followers, mostly made up of other bot accounts. In addition, from the paper's Figure 3 [6] it is shown that fake follow bot accounts generally have a relatively low friends to follower's ratio, as they only exist to boost follower counts of other accounts. Meanwhile, genuine users average less than 1000 friends. These ratios help us to differentiate between follower bot accounts and other accounts.

The paper also claims that bots tend to lack certain custom profile details that real users usually have on their accounts, such use a custom profile picture, description on their profiles, a custom profile location, etc. Looking for the absence of these features in a user account help us determine whether they are likely to be a bot account. In addition, the paper analyzed user accounts and found that users that follow less than 30 accounts are most likely bots as well due to them not having a reason to follow other accounts [6](pg.9).

Furthermore, the paper analyzes that if an account has never tweeted or does not have a screen name at all, it is most likely a bot. In addition, a bot doesn't has no reason to enable Geo-location for their account, so it is another indication of whether the account is a bot or not.

From this paper we used the features it suggested, such as: User has a fifty to one friends to followers ratio, User has never tweeted, User has a custom screen name, User has less than 30 followers, Absence of ID, User has more than a 1000 followers, User has a one hundred to one friends to followers ratio, Absence of profile picture, Not geo-located, User has a profile description, and the Levenshtein distance between User's tweets is less than 30. We chose these specific features as they have the strongest impact in detecting bot accounts.

2.2 Spam Profiles Detection (JIS)

This research article [1] discusses the various characteristics of spam profiles(bots) in social networks like Twitter. After analyzing the many profile structures, they decided to study 29 unique user profile features that would help classify whether a user is a bot or not. These 29 features they studied were: Suspicious Words, Following to Followers Ratio, Default Image, Text to Links Ratio, Repeated Words, Comments Ratio, Tweet Time Pattern, Following Interest vs. Tweets, Description vs Tweets, Number of Tweets per Day, Use Emotion, Tweet via Mobile, Multilingual, Tweet via PC, Use Symbol, Pictures in Tweets, Uses Numbers, Uses Hashtag, Uses URL in description, Video in Tweets, Real Picture, Living Place, Retweet, Likes, Uses other Social Media, Account Age, Number of Followers, Trending Topics, Number of Following.

Using some of these features in our program, we are able to enhance our own bot detection and separate real user account from the automated generated bots accounts more easily. The features that we implemented were: Following to Followers Ratio, Default Image Feature, Comments Ratio, Text to Link Ratio, Uses Emotions, Use Symbols, Uses Hashtag, Living Place, Account Age, Numbers of Followers and Number of Following. We were not able to implement other features, as the data required was omitted from the TwiBot-20 [7] dataset, such as tweet date and what device the tweet was sent from.

2.3 BERT

BERT [5] is an advanced sentence prediction machine model developed by Google. It has many features, and has been used in the past for tweet prediction [11]. However,

due to its complexity, and technical limitations stemming from the need to convert the tweet to a vector of integers in order for BERT to comprehend it, we were unable to apply it to our experiments. We instead used simpler measures of predictability, like Levenshtein Distance.

2.4 TwiBot-20

This research paper [7] presents us with a large bot detection dataset that covers diverse bots and user account which better accurately represents the real-world Twittersphere. In order to get this diverse dataset, they studied the users' follow relationships from various different topic spaces, allowing the TwiBot-20 dataset to have diverse interest domains and user data of people from different parts of the world.

The user data was also chosen from four of the major domains such as Politics, Business, Entertainment and Sports, where various big named users and corporations were selected from each domain along with some users from the same communities, increasing the diversity of the TwiBot-20 dataset further.

According to their paper [7], the TwiBot-20 dataset presents a much more difficult challenge for bot detection than previous ones. They used a couple different benchmarks, such as cresci-17 [3] and PAN-19 [10] which both gave much more accurate tests than those run on TwiBot-20, demonstrating TwiBot's difficulty.

2.5 LASSO Text Analysis Model

It's a statistic model that measures the message predictability by converting all messages to features and build linear regression models which will then analyze whether that message belongs to that feed or not. It has been used in the past [11] to measure tweet predictability, in order to detect bots. We were unable to use this model due to it clashing with the other models we were testing and not working well with them.

2.6 Statistical Machine Intelligence and Learning Engine (SMILE)

Statistical Machine Intelligence and Learning Engine (SMILE) is a library for Java, Kotlin and Scala. The machine learning aspects it covers are things like regression, clustering, feature selection, classification, efficient nearest neighbour search, and many more. The implementations from Smile that we used were its algorithms for text normalization, feature selection algorithms, and classification algorithms.

2.7 Temporal Pattern Analysis (DeBot)

This bot detection algorithm [2] explores a method of detection that involves temporal patterns displayed by groups of users. The algorithm, named DeBot, operates on the assumption that if a group of users act in a synchronous fashion for an extended period of time, then that group is unlikely to be comprised of humans. This synchronous activity would include things such as creating posts, but it also includes the deletion of posts. Since our dataset did not include certain pieces of necessary information such as the time a post was created, we were unable to make use of this type of analysis in our project.

3 Developed Solution

In order to make a strong Twitter Bot Detection Program that is better than the current Twitter bot detector, we developed our own features and then combined many algorithms, techniques, and features from various other research papers to enhance it further, attempting to make it much more accurate in detection.

3.1 Features

Using the SMU Data Science Review paper and Journal of Information Science paper, we were able to make many of the features they suggested that helped us differentiate between real user accounts and bot user accounts, through seeing which of the characteristics the accounts had or didn't have.

SMU also used Levenshtein Distance, a string metric algorithm to compare the tweets between users, to check for similarities.[6] The Levenshtein distance algorithm helps the program find patterns in the bot messages which show that they are displaying bot-like behaviour.

Through the use of Statistical Machine Intelligence and Learning Engine (SMILE), we were able to use many of its given algorithms and techniques which allowed us to make a variety of classifiers and models that helped us further in determining differences between an automated generated user account and a real user account.

One feature from the JIS dataset was to check if the user is multilingual. We accomplished this through a library called Lingua [12]. It uses a set of models, one for each language. In order to improve performance, 2 optimizations were made. First, we save the language of the first tweet that has a recognizable language. We then construct a new instance of the language checker, that only checks for that language, otherwise returning Unknown. The other optimization was to cache these single language checkers in a hash map, as they are non-trivial to construct.

3.2 Feature Selection

Two different methods were used to select the optimal features for the Optimized classifiers. First, a simple Signal to Noise ratio selector was used, which returned values found in Table 1. We also used a Sum of Squares selector, which gave us the values found in Table 2. Both of these selectors were provided by the SMILE [9] library.

From these results, we used any features that achieved above a certain cutoff score. For Sum of Squares, we used a cutoff of 0.05, and for Signal to Noise we used a cutoff of 0.1. We found these cutoffs gave us the best results with the final models. The selected features are listed in Table 1.

3.3 Classifiers

We used two types of classifier, both provided by the SMILE library. We used Support Vector Machine (SVM) and k-Nearest Neighbours (KNN), due to their use in the SMU and JIS papers respectively. They both operate under similar principles, mapping a training set into an n-dimensional vector-space, where n is the number of features being evaluated. They both also predict an object's class by projecting it into the same vector space.

However, they differ in how they classify. SVM computes which side of a pre-computed hyper-plane that divides the 2 classes, and kNN finds how many of each class are in the list of the k nearest objects to the one being predicted, choosing whichever class has more representation in that selection.

4 Results

For our results, we measured the accuracy of the models. We also measured the false positives, which is the number of users it flagged as bots when they were not actually bots, and the false negative rate, which is the number of users it labelled as genuine users, when they were actually bots.

4.1 TwiBot-20

When testing the TwiBot-20 dataset, we first had to measure a baseline. We could only use the SMU [6] feature set

Table 1: Results for Signal to Noise Ratio

Feature Name	Score
MissingID	NaN
UsingDefaultProfileImage	0.0764
HasScreenName	NaN
LessThan30Followers	0.0734
HasLocation	0.0310
DescHasLink	0.0906
LessThan50Tweets	0.0709
TwoToOneFriendsToFollowers	0.229
MoreThan1kFollowers	0.302
UsingDefaultProfileImage	0.0764
HasNeverTweeted	0.0381
FiftyToOneFriendsToFollowers	0.0191
OneHundredToOneFriendsToFollowers	0.00959
HasDescription	0.0947
LevenshteinDistanceLessThan30	0.00414
HighFollowingToFollowersRatio	0.456
UsingDefaultProfileImage	0.0764
HighLinksToTweetRatio	0.0107
CommentsRatio	0.0360
IsMultilingual	0.00991
NameContainsSymbols	0.0317
NameContainsNumbers	0.0606
UsesHashtag	0.00631
DescHasLink	0.0906
HasLocation	0.0310
HasProfileLocation	0.0683
RetweetsMoreThanTweets	0.208
LessThan100Likes	0.0649
AgeLessThan2Months	NaN
MoreThan100Followers	0.139
MoreThan50Following	0.0452
EmojiUsage	0.0383
UserIsVerified	1.10
ProfileUsesBackgroundImage	0.0207
BusinessDomain	NaN
EntertainmentDomain	0.135
PoliticsDomain	0.233
SportsDomain	0.0406
DomainCount	0.374
ListingCount	0.205

Table 2: Results for Sum of Squares

Feature Name	Score
MissingID	NaN
UsingDefaultProfileImage	0.00512
HasScreenName	NaN
LessThan30Followers	0.00514
HasLocation	9.46E-4
DescHasLink	0.00822
LessThan50Tweets	0.00482
TwoToOneFriendsToFollowers	0.0498
MoreThan1kFollowers	0.0883
UsingDefaultProfileImage	0.00512
HasNeverTweeted	0.00144
FiftyToOneFriendsToFollowers	3.53E-4
OneHundredToOneFriendsToFollowers	8.97E-5
HasDescription	0.00851
LevenshteinDistanceLessThan30	1.69E-5
HighFollowingToFollowersRatio	0.206
UsingDefaultProfileImage	0.00512
HighLinksToTweetRatio	1.15E-4
CommentsRatio	0.00126
IsMultilingual	9.63E-5
NameContainsSymbols	9.85E-4
NameContainsNumbers	0.00343
UsesHashtag	3.92E-5
DescHasLink	0.00822
HasLocation	9.46E-4
HasProfileLocation	0.00444
RetweetsMoreThanTweets	0.0399
LessThan100Likes	0.00410
AgeLessThan2Months	NaN
MoreThan100Followers	0.0184
MoreThan50Following	0.00204
EmojiUsage	0.00143
UserIsVerified	0.730
ProfileUsesBackgroundImage	4.25E-4
BusinessDomain	NaN
EntertainmentDomain	0.0181
PoliticsDomain	0.0546
SportsDomain	0.00161
DomainCount	0.117
ListingCount	0.0259

to do this, as we could not implement some of the features from the JIS [1] paper. As can be seen in Figure 1, the SMU (SVM) classifier scored an accuracy of around 64%, which is significantly lower than the accuracy claimed [6] for the SMU version. This confirms the difficulty of the TwiBot-20 dataset [7], as they observed much lower performance on their benchmark in comparison to others, so it would make sense that our benchmark for the SMU features was lower than the reported benchmark.

4.2 Selected Features

After testing a variety of feature cutoffs, we decided on using a score of 0.05 and 0.1 for the Sum of Squares and Signal to Noise Ratio methods, respectively. Interestingly, the Sum of Squares method selected a strict subset of the features in Table 4 which were chosen by the Signal to Noise Ratio method in Table 3.

These features have a few common themes, such as evaluating the number of followers a user has in comparison to how many they are following, which makes sense since a Bot is likely to not have many followers. The domains of an account (both which specific domains a user is a part of, and the absolute count of domains they are a part of) can be indicative of a user being a bot. Finally, a user's verification status was an important flag, and was responsible for eliminating the false negative responses.

4.3 Accuracy

Our optimized models were capable of reaching around 80% accuracy on the TwiBot-20 dataset. They achieved this with almost zero false negatives, and around 18-20% false positives. This can be seen alongside the baseline results for the SMU and JIS models in Figure 1.

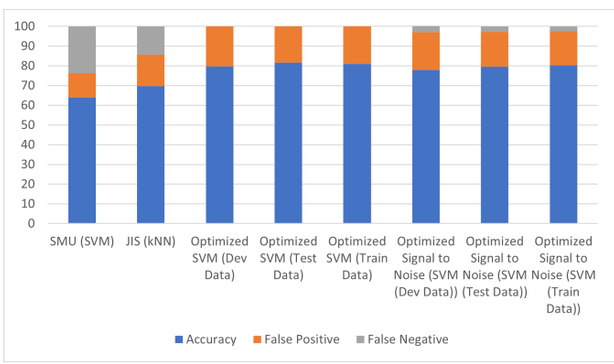

Figure 1: Program Accuracy Results

From just using the SMU paper [6] SVM suggested features we only had a 64 percent accuracy, and by just using the JIS Paper [1] suggested features along with the kNN classifier that we wrote, we had an accuracy of about 70 percent. For our final results With the optimized SVM, we were able to get an average percent of up to 80 percent accuracy from the three datasets given by the Twibot

Table 3: Selected Features (Optimized Signal to Noise)

Feature Name
TwoToOneFriendsToFollowers
MoreThan1kFollowers
HighFollowingToFollowersRatio
RetweetsMoreThanTweets
MoreThan100Followers
UserIsVerified
EntertainmentDomain
PoliticsDomain
DomainCount
ListingCount

Table 4: Selected Features (Optimized Sum of Squares)

Feature Name
MoreThan1kFollowers
HighFollowingToFollowersRatio
UserIsVerified
PoliticsDomain
DomainCount

benchmark.

5 Conclusion

In conclusion, we were able to determine a subset of features from the SVM and JIS papers which provided optimal performance on the TwiBot-20 dataset. In this subset, we found a few segments that provided the most accuracy, such as Follower ratios and Domain. Finally, we were able to confirm the difficulty of the TwiBot-20 dataset in Twitter Bot detection.

References

[1] A. Al-Zoubi, H. Faris, J. Alqatawna, and M. Hassonah. Spam profiles detection on social networks using computational intelligence methods: The effect of the lingual context. *Journal of Information Science*, 47:016555151986159, 08 2019.

[2] N. Chavoshi, H. Hamooni, and A. Mueen. Debot: Twitter bot detection via warped correlation. In *2016 IEEE 16th International Conference on Data Mining (ICDM)*, pages 817–822, 2016.

[3] S. Cresci, R. Di Pietro, M. Petrocchi, A. Spognardi, and M. Tesconi. The paradigm-shift of social spambots. *Proceedings of the 26th International Conference on World Wide Web Companion - WWW '17 Companion*, 2017.

[4] B. Dean. How many people use twitter in 2021? [new twitter stats]. `https://backlinko.com/twitter-users`, 2021.

[5] J. Devlin, M. Chang, K. Lee, and K. Toutanova. Bert: Pre-training of deep bidirectional transformers for language understanding, 2019.

[6] P. G. Efthimion, S. Payne, and N. Proferes. Supervised machine learning bot detection techniques to identify social twitter bots. 2018.

[7] S. Fen, H. Wan, N. Wang, J. Li, and M. Luo. TwiBot-20: A comprehensive twitter bot detection benchmark. *Proceedings of the 30th ACM International Conference on Information & Knowledge Management*, Oct 2021.

[8] A. G. Johansen. What's a twitter bot and how to spot one. `https://us.norton.com/internetsecurity-emerging-threats-what-are-twitter-bo html`, 2020.

[9] H .Li. Smile. `https://haifengl.github.io`, 2014.

[10] PAN. Pan-19. `https://pan.webis.de/clef19/pan19-web/author-profiling.html`, 2019.

[11] P. Przybyła. Detecting bot accounts on twitter by measuring message predictability. `http://ceur-ws.org/Vol-2380/paper_58.pdf`, 2019.

[12] P. M. Stahl. Lingua. `https://github.com/pemistahl/lingua`, 2018.