Twitter Bot Detection

Vatsal Agarwal
University of Maryland
vatsalag99@gmail.com

Davin Park
University of Maryland
dpark354@umd.edu

Nikhil Pateel
University of Maryland
npateel@umd.edu

Domenic Sangiovanni
University of Maryland
domsangio@gmail.com

ABSTRACT

The increasing relevance and use of social media creates a bigger target for widespread manipulation. In particular, bot campaigns have become popular in pushing out political ideologies to a large, unknowing audience. Efficiently identifying and deactivating bot accounts is an important goal. Past attempts at classifying bots have relied on training models via machine learning. Recent advancements in natural language processing help in analyzing short posts on the popular social media site Twitter. However, these same advancements help bots become less susceptible to detection.

By incorporating other related information such as the number of followed accounts and number of tweets over time, we attempt to create a more holistic model that will more accurately identify Twitter bots. We compare the accuracy of our model to other pre-existing models.

1. INTRODUCTION

Social media platforms such as Twitter or Facebook have become one of the most popular ways people communicate and spread information around the world. Although these mediums have a great potential for good, they also have an equally destructive capacity. Today more so than ever, such platforms are exploited to disseminate malicious messages and create a spiral of misinformation. One of the most common strategies to accomplish this task includes the use of bots, which are accounts controlled by automated software. Bots can have a wide variety of purposes from helping mediate discussions or posting motivational sayings to having more malevolent intentions such as spreading false information and stoking conflict. Specifically, on Twitter, bots continue to gain a larger traction in influencing elections. Thus, there is an urgent need to find methods to limit the hold bots have on the general public and ensure that conversations are not determined by them.

Despite these compelling issues, there is still a significant challenge in discerning bots from real users. Recent advances in natural language processing and artificial intelligence have propelled the ability of bots to emulate human behavior and have thereby increased the difficulty of identifying them. This has led to bots being able to coordinate with each other and form more intricate networks to project false narratives. Another major issue in detecting bots is the wide variety of content that needs to be considered. For instance, a classifier trained to detect bots for the 2016 U.S. presidential election [1] would not perform as well in determining bots for the 2019 Spanish election [5]. Thus, it is important to use features that extend beyond the textual patterns in the tweets. Furthermore, it would be important for real time identification such that users can determine when they might be getting fed false information. Many prior attempts also utilize traditional machine learning algorithms and provide handcrafted features to the model.

We aim to show that our classifier would be able to determine if a Twitter use was a bot or not using their account history and activity. Our insight is to combine both deep learning and more traditional machine learning methods for use in the classifier. We shall show that our classifier can perform on a similar level to most current models, and if this process is rebuilt with more training data and fine-tuning, a viable alternative to current classifiers is possible.

2. BACKGROUND AND RELATED WORK

2.1 Non-DL Methods

Much of the previous work done has been focused on feature engineering and more traditional machine learning techniques. The current state of the art bot detector, Botometer, was a project done by Indiana University trying to classify whether a twitter account was a Bot or not[7]. This was accomplished using several hundred features from a Twitter account complemented with a Random Forest Classifier that was trained on a large labeled data set, which the team labeled along with an already labeled data set. These features ranged from common phrases, to frequency of responses, which resulted in the current BotometerAPI which can classify a given account as a bot or not[7]. The current Botometer project is public on Twitter and is free to use for people. This has limitations though due to Twitter's REST API having requests limit, or the fact that detecting bots is just plain hard[7].

There have been other ways to find these twitter bots, though. For example, during the 2016 election, Twitter Bot detection was also very prevalent in 2016, where researchers wanted to see if Russia played a role in the election. To do

this, one team used Natural Language Processing tools, or NLP, in order to look at tweets and see if they were malicious or bots[2]. They did this using Latent Dirichlet allocation, or LDA, to process similar speech patterns between groups. In this example, Bots often talked repetitively about supporting Trump among other similar sentiments like anti-muslim [2]. This method of determining Bots could prove useful as it is yet another way that can be used to anticipate and classify these Twitter Bots.

Another example of a twitter classifier is with a random forest classifier, a very popular means of classification used in Machine Learning. A project team from the University of Nevada implemented a random forest, using several new features including length or bio and age of account in order to predict if an account was a bot or not[6]. They then included some derived characteristics like number of likes or follower ratio, which is not directly associated with the TwitterAPI. These features were able to make this model more accurate than previous models as most previous models used information that was only available directly from the API[6]. They "derived" this information themselves which helped as this classifier was able to give a prediction unlike prior ones that rely solely on the Twitter API[6]. This model was able to score an accuracy of at least 90 percent, though this was done on old information and Bots have changed drastically since then, being able to impersonate people.

2.2 DL Methods

A way of interpreting the semantic language of a bot's tweets will also be needed. First, we need encode each word with a vector whose properties are related to the word's semantic meaning. Word embedding is a well-known encoding in NLP. When two words are close in meaning, their corresponding word embeddings will also be close in space. The approach in [3] expands upon word embeddings, creating a new model called GloVe word vectors. The authors in [8] have employed GloVe word vectors successfully with bidirectional LSTM's to distinguish between Twitter bots and real accounts.

Another example of a word encoding we considered is Bidirectional Encoder Representations from Transformers (BERT) [8]. BERT is a multi-layer bidirectional Transformer encoder, an encoder being a neural network with a self-attention mechanism. BERT attempts to encode each word given some text. Unlike GloVe, BERT takes into account the context of the words around each word. Thus, BERT yields more accurate word encodings for our purposes.

3. APPROACH

3.1 Dataset

For both training and validation, we used the dataset collected in Caverlee 2011. This set contains Twitter users, their account histories, and their tweets, alongside timestamps. The dataset is split into two categories: legitimate users and content polluters (bots). The data that was specifically collected were the timestamps of each tweet, the tweet text alongside user ID, the the number of followers and followings of each user, the number of tweets, length of screen name and length of description in the user profile. This is a more limited set of features compared to the features used in [7, 6, 2], but could be easily expanded with more thorough data collection.

Directly from the Caverlee 2011 dataset, we were provided the following features:

- Length of the Screen Name
- Length of Description
- Number Of Tweets
- Number of Followers
- Number of Followings (Followed Accounts)

Following [4], we added additional features to train on:

- Ratio of Followings to Followers
- Standard Deviation of Unique Numerical ID's of Followers
- Standard Deviation of Unique Numerical ID's of Followings
- Number of Unique mentions per tweet
- Number of mentions per tweet
- Number of Unique URL's per tweet
- Number of URL's per tweet

3.2 Machine Learning Pipeline

First, we wanted to implement a Machine Learning model similar to the previous work already done. Using Python, we were able extract this information from the Caverlee data set and use it in our model. This included multiple features about user's Twitter accounts, and also their tweets themselves. In order to train our model we used a combination of these factors and their tweets, but some of the information discussed in Caverlee's paper was not actually provided itself in the data set, which may cause some problems during validation. First, we ran our random forest classifier on these attributes.

We then were able to test this data set and predict whether or not an account was a human or a content-pollutor. Overall, this was done through traditional methods very similar to previous work done, i.e. training a model on the tweet data set which has already been done prior.

3.3 Deep Learning Pipeline

To further strengthen our work, we added in an additional step, merging some of the prior work that has been done. We implemented a deep-learning model GloVe, as discussed before, in order to change a tweet into a vector. It does this by extracting words from the tweet and transforming it into a vector, which can then run on a machine learning model. This vector was then used in the machine learning model in conjunction to other features like age of the account, frequency of likes, etc. (See Figure 1 for details). GloVe was used as a way to extract feature vectors from individual tweets themselves. A single tweet was sampled from each user due to the large overhead in running word embeddings. The mins, maxes and averages of the word embeddings were concatenated to get a final "textual feature" element for a single tweet. This feature vector was then joined to all of the

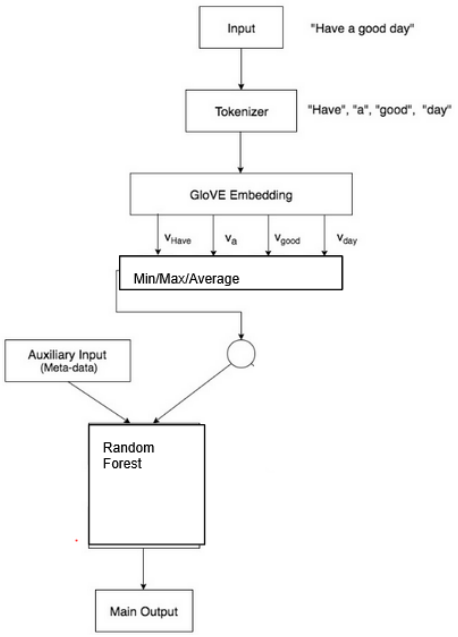

Figure 1: Organization of Model

non-textual Caverlee features, and a random forest classifier was trained.

The idea behind this is that its creates a more robust and holistic view of the data, so in theory, it should be more accurate. In our tests, we had limited resources so the lightweightness of GloVe was much better, for our purposes. If we had the resources, we would expect BERT would yield better results since it gives a much better vector, but it was too expensive to run and prohibited us to use BERT.

With the accompaniment of deep learning, we expect to see a better result and more accurate predictions since we can extrapolate more information and more features, while also using a better model. BERT would have been an even better model since it is bidirectional and actually learns the context of the words, but GloVe is good for our purposes.

4. RESULTS

4.1 Non-Deep Learning

We ran our model with the original set up not including the GloVe component. Using a random forest classifier and the methods described in Caverlee 2011, we were able to get a 96% accuracy in predicting if an account was a bot or not.

4.2 Deep Learning with GloVe

Our model using GloVe instead of handcrafted features performed at 95% accuracy on the Caverlee 2011 data-set. This is likely not a significant change, so we were able to show that we could replace hand-crafted features with word embeddings and get a viable classifier.

5. CONCLUSION

	precision	recall	f1-score	support
Bot	0.96	0.96	0.96	5237
User	0.96	0.96	0.96	4801
accuracy			0.96	10038
macro avg	0.96	0.96	0.96	10038
weighted avg	0.96	0.96	0.96	10038

Figure 2: Classification report of hand-crafted textual features

In this paper, we described a better, modern approach to classifying twitter accounts from Bot or human. To do this, we have built upon prior work such as the already popular decision tree classifier along with some Natural Language Processing. We then further expanded this by implementing some modern methods in Deep Learning and language processing. Our classifier distinguishes itself against previous models because it utilizes a mix of these methods in order to get better results. All of the prior work either exclusively focuses on some deep learning or on more traditional machine learning algorithms. By using both of these methods, we are able to create a better, more robust classifier that utilizes more of the information that is provided.

Comparing our new model's accuracy in Figure 3 to the accuracy of previous methods in Figure 2, our model does not provide much significant advantage. However, we have demonstrated that instead of directly using words, word embeddings can be also be used without any significant de-

	precision	recall	f1-score	support
Bot	0.95	0.96	0.95	5257
User	0.96	0.94	0.95	4781
accuracy			0.95	10038
macro avg	0.95	0.95	0.95	10038
weighted avg	0.95	0.95	0.95	10038

Figure 3: Classification report using GloVe textual embeddings

crease in accuracy or performance. These findings indicate the potential of word embeddings in other problems in computer science where word embeddings or similar vector representations may be more applicable or squeeze out slightly better performance.

We believe that the Twitter Bot detection is a big field that has definitely expanded in the past few years, and with even more Deep Learning tools and more advanced language processing, along with more information from Twitter itself, we could expand this even more.

With more language processing at an even higher level, we think that our model could become even better and be able to classify near perfect accuracy. With more resources and using BERT, we expect that this could build up even further with a higher accuracy of predicting if an account is a content-polluter or not. However the resources required are tremendous.

Acknowledgments

We thank our professors Leilani Battle and Dave Levin for their guidance and smooth transition to online learning during these difficult times. We also thank our fellow CMSC396H classmates for their input during class discussions.

Keywords

Twitter; Bot; Machine Learning

6. REFERENCES

[1] A. Bessi and E. Ferrara. Social bots distort the 2016 us presidential election online discussion. *First Monday*, 21(11-7), 2016.

[2] B. Kong. Analysing russian trolls via nlp tools, 2019.

[3] S. Kudugunta and E. Ferrara. Deep neural networks for bot detection. *Information Sciences*, 467:312–322, Oct 2018.

[4] K. Lee, B. Eoff, and J. Caverlee. Seven months with the devils: A long-term study of content polluters on twitter, 2011.

[5] J. Pastor-Galindo, M. Zago, P. Nespoli, S. L. Bernal, A. H. Celdrán, M. G. Pérez, J. A. Ruipérez-Valiente, G. M. Pérez, and F. G. Mármol. Spotting political social bots in twitter: A use case of the 2019 spanish general election. *arXiv preprint arXiv:2004.00931*, 2020.

[6] J. Schnebly and S. Sengupta. Random forest twitter bot classifier. In *2019 IEEE 9th Annual Computing and Communication Workshop and Conference (CCWC)*, pages 0506–0512, 2019.

[7] O. Varol, E. Ferrara, C. Davis, F. Menczer, and A. Flammini. Online human-bot interactions: Detection, estimation, and characterization, 2017.

[8] F. Wei and U. T. Nguyen. Twitter bot detection using bidirectional long short-term memory neural networks and word embeddings. *2019 First IEEE International Conference on Trust, Privacy and Security in Intelligent Systems and Applications (TPS-ISA)*, Dec 2019.