Leveraging node neighborhoods and egograph topology for better bot detection in social graphs

Björn Bebensee
Dept. of Comp. Science and Eng.
Seoul National University
bebensee@snu.ac.kr

Nagmat Nazarov
Dept. of Comp. Science and Eng.
Seoul National University
nagmat@snu.ac.kr

Abstract

Due to their popularity online social networks are a popular target for spam, scams, malware distribution and more recently state-actor propaganda. In this paper we review a number of recent approaches to fake account and bot classification. Based on this review and our experiments, we propose our own method which leverages the social graph's topology and differences in ego graphs of legitimate and fake user accounts to improve identification of the latter. We evaluate our approach against other common approaches on a real-world dataset of users of the social network Twitter.

Keywords: Fake account detection, social graph, network topology

1 Introduction

Today people all around the world use online social networks (OSNs) not only for personal connections but also for entertainment, to share opinions, to read news and inform themselves and to exchange knowledge and information. With their rise in popularity OSNs in the past decade however, they have become a target for abuse by malicious actors who are spamming the network, attempting to scam users, distribute malware, boost a legitimate user's popularity or increase the visibility of certain content. Furthermore with the 2016 United States presidential election the focus has been on social media rather than traditional media for the first time and the concern over widespread *fake news* influencing public opinion as well as social bots pushing state-actor agendas has been growing [1, 9] with some recent research focusing on identifying fake news using data mining methods [15].

Many OSNs spend a considerable amount of money and time in the form of manual labor on identifying and removing fake accounts. Our goal is to build on previous research and to improve the classification process for more effective and efficient identification of

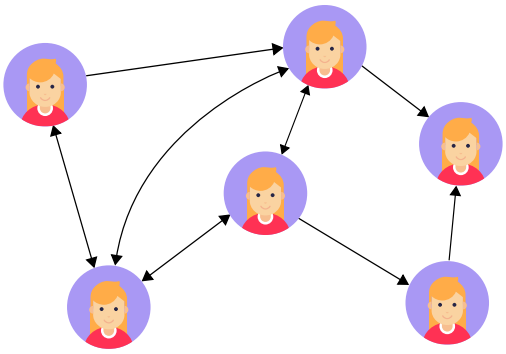


Figure 1: A social graph

fake accounts. Unlike other approaches which focus only on basic profile features [6, 14] or temporal patterns in account activity [3, 8, 10], our approach belongs to a category of graph-based approaches to fake account identification.

Given the directed graph $G = (V, E)$ induced by the social network's structure as well as additional classification features m_v for every $v \in V$, we want to identify all nodes $v \in V$ that are likely to correspond to fake accounts. We call this graph a social graph (figure 1). For social networks with bidirectional connections i.e. through friend requests an undirected graph may be used to describe the network structure. It is however vital for the classification task that the false-positive rate be kept low as suspensions of legitimate user accounts can degrade the experience enormously.

In this paper we propose a novel approach to identification of fake accounts in OSNs by exploiting differences between the ego networks of legitimate users and fake accounts as well as weak links between communities of real users and communities of fake accounts. We first obtain a labelled dataset of legitimate and fake accounts on Twitter. As Twitter's developer policy limits the ways in which scraped user data may be published, these datasets typically contain a list of user IDs which we will then have to scrape to construct the social graph. We will explore this social graph and run experiments to find predictive features based on which we develop a novel approach. Finally, we evaluate our approach in experiments against other popular approaches for fake account detection in OSNs.

The rest of this paper is structured the following way: first we do a short survey of related work and their approaches (Section 2), based on the findings in these papers we will explore a dataset of Twitter users and fake accounts and their social graph, propose our own classification method, and evaluate our approach in experiments.

2 Related Work

In this section we will list papers that each member has read and reviewed as part of a first survey into the topic, along with a summary of their main ideas, how they can be of use for our own approach and possible shortcomings of the approach or important points not addressed by the paper.

2.1 Papers read by Björn Bebensee

2.1.1 Detecting Clusters of Fake Accounts in Online Social Networks

Main idea: As opposed to previous literature which approaches the fake account classification problem on a per-account basis, Xiao, Freeman and Hwa [16] suggest a different approach which uses an approach based on clustering instead. They suggest that as more efficient way of identifying a set of spam accounts made by a single spammer, one might classify entire clusters of users to be legitimate or fake instead of single user accounts. Furthermore their approach focuses on identifying and removing fake accounts before they can interact with legitimate users and spam the network so as to prevent damaging the experience of legitimate users. As they want to stop fake accounts as early as possible and only limited information becomes available during registration Xiao et al. focus on these few features which are available at registration time.

The authors divided their machine learning pipeline into three major parts: a cluster builder, a profile featurizer and an account scorer. The cluster builder takes a raw list of accounts and builds clusters of accounts where the clustering criteria can be simple (i.e. share a common feature) or more complex (like k-means). These clusters, along with the features needed for the profile featurizer, are then labelled as real or fake. If there are accounts of both groups in one cluster, it is labelled according to a threshold x. The featurizer extracts features from the set of accounts in one cluster to find a numerical representation (an *embedding*) which can then be used by the account scorer to score the cluster. The authors test a number of models for the account scorer, specifically logistic regression, random forests and support vector machines. They find that for their use-case random forests perform best, with a recall slightly better than SVMs. Overall the model showed good performance in tests on in-sample data as well as a newer out-of-sample dataset. The authors have sinced deployed it in production at linked in and restricted more than 250,000 accounts.

Use for our project: This paper is closely related to the approach we want to take to classification of accounts as genuine or as fake. Xiao et al. suggest classifying entire clusters of users rather than single users to leverage similarities between fake accounts. This technique could prove useful for our approach and can be used in combination with features from each user's social graph. It might be possible to cluster users based on graph features such as degree, number of triangles a node participates in and others.

Shortcomings: A classification of entire clusters as proposed by the authors may not perform as well if fake accounts are less homogeneous. Such a distribution of fake accounts will lead to clusters being more mixed and therefore to a higher number of false-negatives and false-positives. Additionally, this approach uses many features that only the social network operator has access to and that are not available in public datasets to cluster accounts. Unfortunately, we do not have access to this type of data and different features may prove less effective for clustering.

2.1.2 Botnet detection using graph-based feature clustering

Main idea: In this paper Chowdhury et al. [4] explore the use of graph-based features for clustering in computer networks to detect botnets. As much prior literature has focused on flow-based or rule-based detection, the authors suggest using clustering to first identify clusters of suspicious nodes. The authors are using a self-organizing map (SOM) for dimensionality reduction and clustering by assigning each node to a different cluster according to the output of the SOM. The features used for clustering are node in-degree, out-degree, in-degree weight (i.e. how many packets are received), out-degree weight (i.e. number of outgoing packets), clustering coefficient, node betweenness, and eigenvector centrality. Finally they are classifying nodes in each cluster (except the largest as it is unlikely to contain bots) starting from the smallest cluster using their own bot-search algorithm which only requires examination of few nodes for classification. Chowdhury et al. show that their approach performs better than SVM classification on the CTU-13 dataset (a dataset of botnet traffic) using the same graph features.

Use for our project: Although the approach presented in the paper operates on an entirely different set of data, it is very similar to our goal in its nature. The authors want to identify a set of bad actors in a network given interactions between devices and given the network structure. As we are attempting to classify users in a social network according to the structure and topology of the social graph, we aim to use a set of graph-based features, similar to the features used in the paper, to cluster groups of users which we may subsequently classify jointly.

Shortcomings: Calculating all given graph features for all nodes in the graph will not scale very well. For the CTU-13 dataset used by the authors the computation took 30 hours on a supercomputer cluster. This is not an acceptable amount of processing power and time to detect social bots in social networks in (near) real-time in order to prevent interactions with real users. However, as the CTU-13 dataset contains much data and information that is not contained or necessary for an application on social graphs, some of the ideas from these paper may still be viable in our use-case. Further experimentation is required.

2.1.3 Aiding the detection of fake accounts in large scale social online services

Main idea: Cao, Sirivianos, Yang and Pregueiro [2] build on previous work in *sybil detection* that aims to use random walks to identify fake accounts (*sybils*) based on key observations made on the structure of social graphs. Specifically a main assumption in these fake account detection schemes is that the connectivity between real users and fake accounts is limited and lower than the number of inter-user and inter-bot connections. In this work Cao et al. propose a new algorithm called SybilRank which, unlike previous work in the field, does not aim to make a binary classification of each user account but instead focuses on creating a ranking which allows for a measure of confidence in classifications as well as further challenges like *captchas* for suspicious accounts. The key idea behind this algorithm is that

in a social network an early-terminated random walk starting from a real user account has a higher probability of landing at another real-user than at a fake account. Early termination is necessary for these random walks as the probability of landing at any node converges to a uniform distribution for random walks of sufficient length. The authors can thus use the degree-normalized landing probability of early-terminated random walks to rank nodes and leverage the fact that connections between real users and fake accounts are limited. They further propose a more effecient way of calculating the landing probability of random walks using power iteration.

Use for our project: Use for our project: Cao et al. show that it is possible to leverage the topology of the social graph, specifically the weak links between fake accounts and real users, to identify these fake accounts. As we plan to use binary classification for this task, it could prove helpful to include the degree-normalized landing probability for random walks as an additional graph feature either in the machine learning algorithm or for clustering of similar nodes, given that it can be computed efficiently enough which may not be the case for large-scale social networks.

Shortcomings: The authors suggest running the SybilRank algorithm periodically, i.e. once every month, which would give fake accounts a window of time that is big enough to interact with and impact real users' experience on the social network unlike the approach introduced by Xiao et al. [16].

2.2 Papers read by Nagmat Nazarov

2.2.1 Towards a language independent Twitter bot detector

Main idea: Lundberg, Nordqvist and Laitinen [13] present a language-independent approach to classify single tweets as either auto-generated (AGT) or human-generated (HGT). Their classifier consists of 10 tweet features:

a) *isReply* $\in \{0, 1\}$ indicates if a tweet is a reply

b) *isRetweet* $\in \{0, 1\}$ indicates if a tweet is a retweet

c) *accountReputation* given by number of followers divided by number of friends and followers

d) *hashtagdensity*,*urldensity*, *mentiondensity* given by number of occurrences divided by number of words in the tweet

e) *statusesPerDay* is the number of status updates per day

f) *favoritesPerDay* is the number of tweets favorited per day

g) *deviceType* $\in$ {web, mobile, app, bot, ...}

The authors find that decision tree-based supervised learning algorithms work particularly well on this type of problem. Out of the evaluated algorithms, random forests (RF) perform best.

Use for our project: We may focus on decision tree-based supervised learning algorithms and particularly RF for a set of tweet features (or similarly basic profile features) like this for classification of fake accounts in online social networks.

Shortcomings: The proposed algorithm does not perform as well as single-language classifiers. If enough resources are available it may be more sensible to train a single-language classifier for each language one wants to identify auto-generated tweets in rather than using a multi-language model. The model has only been trained on a small dataset of tweets in two languages and may perform better if other languages are used as well. Furthermore, the authors evaluated the model in only one other language, more extensive evaluation may be necessary.

2.2.2 A network topology approach to bot classification

Main idea: Cornelissen, Barnett, Schoonwinkel, Eichstadt and Magodla are proposing a graph-based network topology approach to the bot classification problem [5]. They propose utilizing the surrounding network topology of an ego in the social graph to determine whether the user is an automated agent or human. The ego graph of node n is a K-2 graph obtained by a crawler, that is a graph with all nodes i of distance $d(n, i) \leq 2$. Using clustering on features from the ego's graph such as the density, clustering coefficient and centrality among others, they achieve an accuracy of 70%. The authors suggest using such network analysis in conjunction with other methods for better accuracy.

Use for our project: The authors suggest to use centrality graph measure, for example celebrities have high indegree and low outdegree. For instance, celebrities on Twitter tend to have more people following them than they follow themselves. The authors also propose that the bots must have high outdegree but very low indegree, since most people will not follow back. We can use this proposal to distinguish the bots with non bots. [5]

Shortcomings: The false-positive rate is very high at around $\sim$45% which is not acceptable in a real-world setting as it can potentially lead to many falsely removed accounts belonging to legitimate users, impacting the user experience negatively. However, this may be in part due to obvious outliers that have not been removed as the authors state.

2.2.3 Bot Classification for Real-Life Highly Class-Imbalanced Dataset

Main idea: Typically the research on bot detection is based on particular botnet characteristics, but in this paper Sarah Harun, Tanveer Hossain Bhuiyan, Song Zhang, Hugh

Medal and Linkan Bian develop three generic features to detect different types of bots regardless of specific botnet characteristics [11]. They suggest five classification models based on those features to classify bots from a large, real-life class-imbalanced network dataset. The authors show that the generalized bot detection methods perform better than the botnet specific methods.

They first filter out unnecessary data and then extract features from the rest of data by computing the previously developed three generic features for each source-destination pair of IP addresses. In the filtering step the authors remove any IPs which never act as a source and those which perform only single communication with another device. In the feature extraction step the following features are computed: 1) Falling rate of communication frequency, 2) median communication frequency and 3) source bytes per packet for highest communication frequency. Using the extracted features the authors explore a number of different supervised learning algorithms like Quadratic Discriminant Analysis (QDA), Gaussian Naïve Bayes (GNB), Support Vector Machine (SVM), K-Nearest neighbors (KNN) and Random Forests (RF) for this highly class-imbalanced dataset. In their experiments they find that RF, KNN and even SVM perform poorly on imbalanced data and that QDA and GNB perform best for imbalanced datasets.

Use for our project: QDA and GNB perform much better than the other supervised learning algorithms on class-imbalanced data. Given such an imbalanced data distribution we should avoid using RF and KNN as these perform extremely poorly. Another important takeaway from this paper is that for accurate training of supervised models a balanced dataset is important and we should thus use such a dataset if possible.

Shortcomings: The biggest shortcoming of this paper is that it ignores passive or less active bots from the beginning. Furthermore, it is not a real-time detection system and works primarily by observing activity patterns. Although a similar approach is possible in social networks (i.e. accounts posting very frequently, sending spam links or only replies are likely to be fake accounts) but such activities can only be observed after the fact which makes this type of approach less useful in preventing fake account interactions with real users.

3 Proposed Method

In this section we will briefly describe and analyze the dataset used in this work. In section 3.2 describe our novel approach and how we train the classification model in detail.

3.1 Dataset

For analysis, training and testing we use the labelled data collected in the Cresci-2018 dataset [7]. The dataset contains 25,987 user accounts, each of which is assigned a binary label $l \in \{\text{human}, \text{bot}\}$ and is the biggest one of its kind that we are aware of. Due to

	Ours		Cresci-2018	
Humans	6,082	46.46%	7,479	28.78%
Bots	7,009	54.54%	18,508	71.22%
Total	**13,091**		**25,987**	

Table 1: Dataset distribution after retrieving available user profiles using the Twitter API

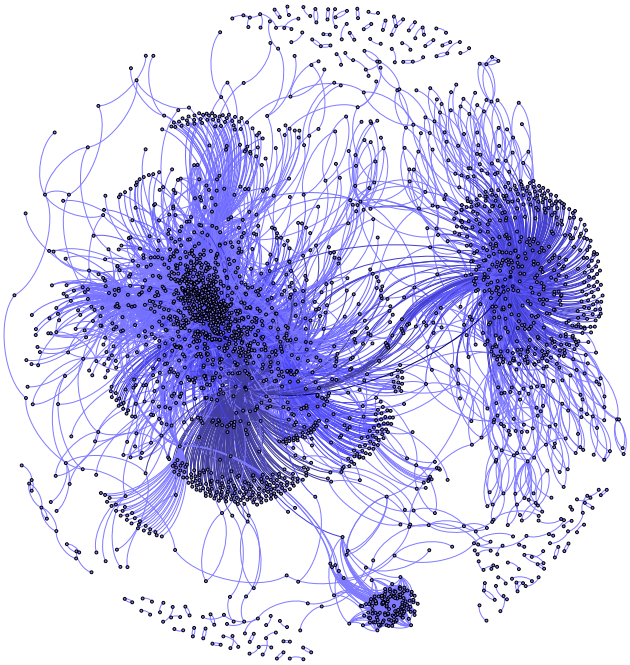


Figure 2: A visualization of the dataset. Each vertex corresponds to a Twitter user and each edge corresponds to a "following" relationship between two users. A darker node has a higher degree. We can see many disconnected components.

Twitter's developer policy[1] which restricts the ways in which user data collected using the Twitter API may be shared, it is difficult to find sizable high-quality datasets. As the rules allow for profile data to be shared online only under specific circumstances and with imposed restrictions, these datasets of fake accounts are typically shared as a set of (user_id, label) pairs. The associated accounts can then be retrieved again using the Twitter API at a later time. However, as many of the labelled accounts are bots might have been deleted or changed to a "protected" status which hides most of the profile information from the public. In our case, even though the dataset is relatively recent, only 13,091 of the originally 25,987 labelled accounts were still available (a reduction of about 49.6%). Of these 6,082 accounts are labelled to be human and 7,009 accounts are labelled as bots (see table 1).

[1]https://developer.twitter.com/en/developer-terms/agreement-and-policy.html

In addition to the labelled user accounts we have collected all profiles of users followed by or following a user in the dataset. As the Twitter API limits us to a maximum of the first 5,000 users when retrieving these lists we only get those first 5,000 users for users following or followed by more than 5,000 accounts. Including these user profiles we have a total of 4.6 million user profiles. For each user we have the following attributes:

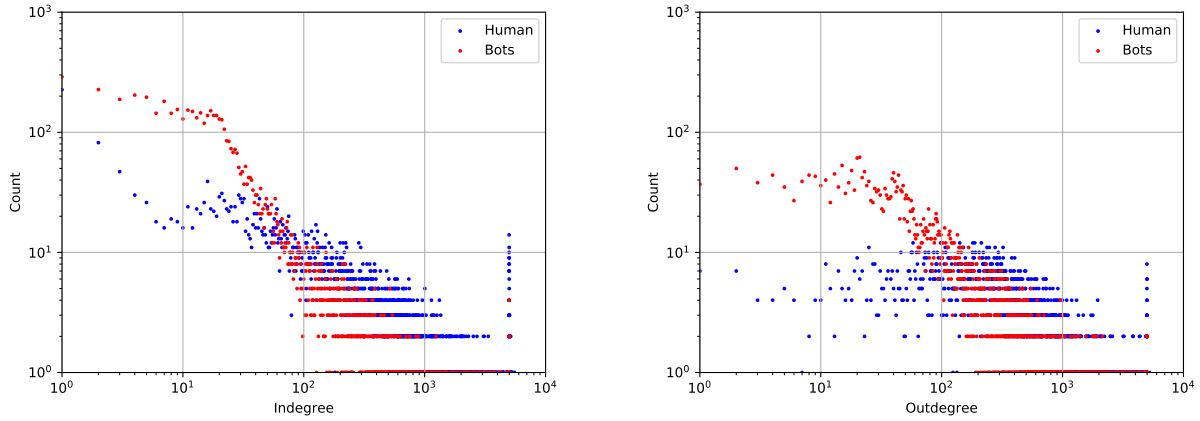
- user ID
- label
- username
- screen name
- number of "followers" (in-degree)
- number of "following" (out-degree)
- location
- profile URL
- profile description
- number of times the user is listed
- number of favourites
- number of status
- date created
- default profile (boolean)
- default profile image (boolean)
- list of users followed (up to 5000)
- list of followers (up to 5000)

From this data we can recreate the social graph with distance $d \leq 2$ from the users in the Cresci-2018 dataset as we have the number of followers and users followed for all users of distance $d = 1$ which allows us to simply create "dummy" nodes without any other attributes in the graph. Unfortunately scraping user profiles of distance $d \geq 3$ from the original dataset does not seem feasible for us as this would seem to include a large part of all Twitter accounts and given the Twitter API restrictions might take months. For reference the 90^{th}-percentile effective diameter of Twitter in 2010 was 4.8 [12]. A visualization of the graph of all labelled nodes can be seen in figure 2.

Given this data we perform some basic analysis. When plotting the in-degree-count and out-degree-count graphs we can easily see that bot users tend to have a lower in-degree (less followers) and also a lower out-degree (following less users) than human users. It is worth noting that while the bot accounts seem to mostly follow a power-law, the out-degree distribution for human accounts does not follow a power-law and many nodes in the dataset appear to have a surprisingly high out-degree that may not be perfectly representative of the general user population on Twitter.

When plotting the frequency of favorite counts and status counts for human and bot accounts, we notice that their distributions differ greatly (figure 4). While human accounts appear to follow a unimodal distribution that is greatly shifted towards zero, bots appear to follow a bimodal distribution and greatly outnumber human accounts in the ranges of their peaks. This clearly visible difference makes these features especially valuable for classification.

Further analysis shows that the clustering coefficient of both human and bot accounts appears to be rather low overall which can be explained to the lack of edges and paths that are not included due to the limitations of our crawl of the Twitter network (see figure 5).



(a) In-degree-count graph

(b) Out-degree-count graph

Figure 3: Degree-count graphs for human and bot accounts

However, we find that the clustering coefficient of human accounts is still slightly higher than that of bot accounts. Regarding the eigenvector centrality we find that human accounts have a slightly higher value too (see also figure 5). We believe that these findings may be stronger in a more comprehensive dataset and warrants further investigation in future work.

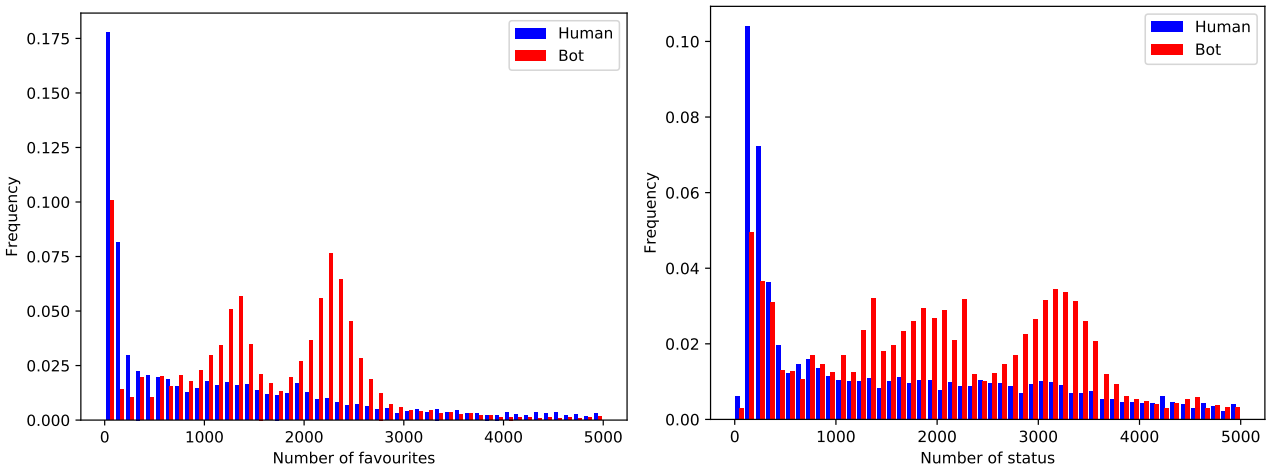
3.2 Neighborhood-based classification

We introduce a novel graph topology and neighborhood-based approach to bot detection. Human and bot accounts typically differ in the way they use Twitter and as a result their structure in the social graph, or their ego graph, is typically different. Instead of classifying a node in the graph just by its immediate features (such as follower count, number of tweets, default profile image etc.) we propose to utilize the graph structure and the features of neighboring nodes in bot classification. We introduce an additional set of features computed as an aggregate over the entire set of predecessors and/or successors.

Reputation. A user's reputation is often measured as the following/follower ratio. We define the reputation r of a user u by

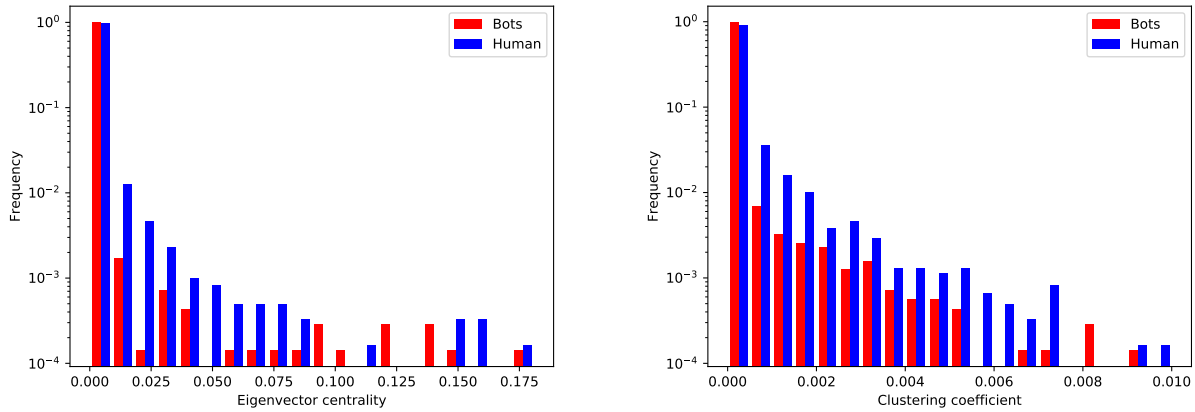
$$r_u = \frac{|u_{\text{followers}}|}{|u_{\text{followers}}| + |u_{\text{following}}|}.$$

Median indegree. We compute median indegree of successors and the median indegree of predecessors. Intuitively, we believe humans tend to follow more high-indegree users (i.e.) news sites, celebrities, while being followed by other low-indegree users.



(a) Favourites-count-frequency graph (b) Status-count-frequency graph

Figure 4: We find that the distributions of favourite counts and status counts among bot and human accounts vary greatly.



(a) Eigenvector centrality-frequency graph (b) Clustering coefficient-frequency graph

Figure 5: We compare the eigenvector centrality and clustering coefficient distributions among human and bot accounts.

Median outdegree. We compute median outdegree of successors and the median outdegree of predecessors. The intuition is that bots may be more likely to be followed by bots or users that follow a large amount of people.

Median reputation. The above measure of reputation, that is $r_u = \frac{|u_{\text{followers}}|}{|u_{\text{followers}}|+|u_{\text{following}}|}$, aggregated for predecessors and successors in the social graph each. Intuitively, real accounts

Median status count. Median of status updates for predecessors and successors in the social graph.

Median favourites count. Median number of favourites or likes by a user's predecessors and successors.

Median listed count. Median number of times a user's predecessors and successors appear in other users lists on Twitter.

Median account age. The median account age of predecessors and successors in the social graph. Intuitively, older accounts tend to be more likely to be real users whereas recently created accounts may be bot accounts.

Number of egograph edges. The total number of edges in a user's egograph. Used to measure how connected the egograph is.

Number of egograph nodes. The total number of nodes in a user's egograph. Used to measure how connected a user is in the social graph.

Density. The density of a user's egograph. A graph's density describes the number of edges in relation to the number of nodes. A complete graph has density 1 while a graph with an empty set of edges has density 0. For n nodes and m edges, it is computed as

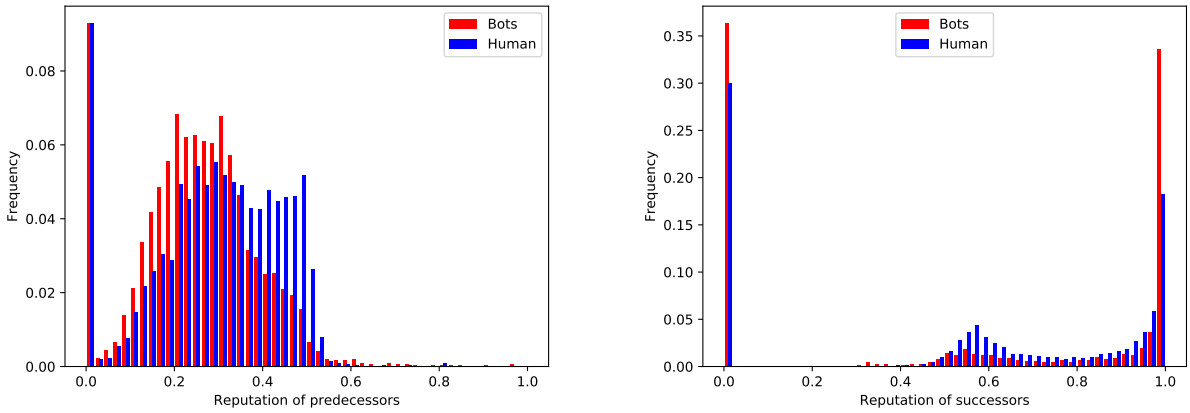
$$d_G = \frac{m}{n(n-1)}.$$

Reciprocity. We measure reciprocity in each user's egograph by computing the ratio of edges that are reciprocated (i.e. that point in both directions) to the total number of edges in the egograph. That is

$$\text{rec}_G = \frac{|\{(u,v) \mid (u,v) \in E \wedge (v,u) \in E\}|}{|\{(u,v) \mid (u,v) \in E\}|}$$

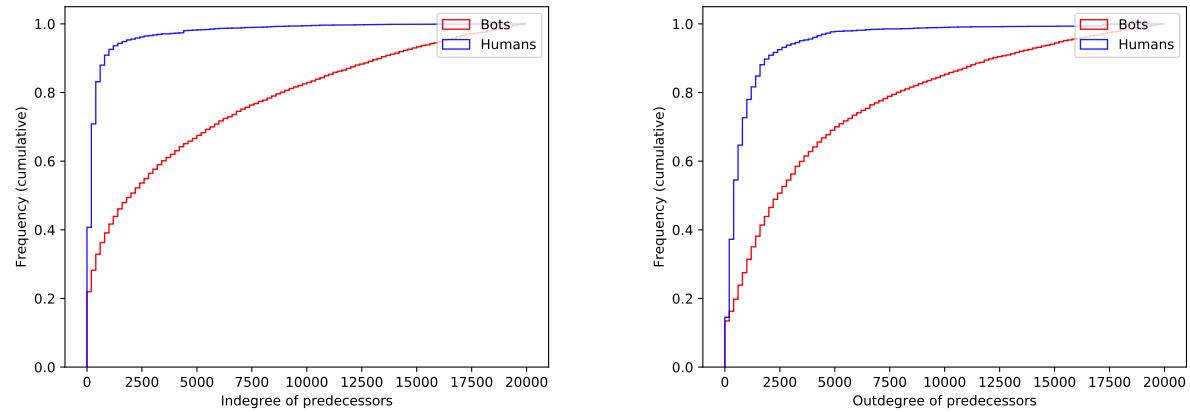
Assortativity. A graph's assortativity measures the similarity of nodes in a graph given a certain attribute. In this case we choose to use (indegree assortativity (i.e. follower assortativity) to describe how similar a user's egograph is in terms of indegrees.

We will first compute such aggregate features for the dataset using the social graph. In addition to the features scraped using the Twitter API, we computed and evaluated the following features during our analysis. An aggregate of a feature for a user's neighborhood (a *neighborhood feature*) can be computed by taking the median value across all predecessors or successors (see figure 6). An aggregate of this measure can be computed across all predecessors/successors by taking the median (see figure 6).



(a) Aggregated reputation of predecessors (b) Aggregated reputation of successors

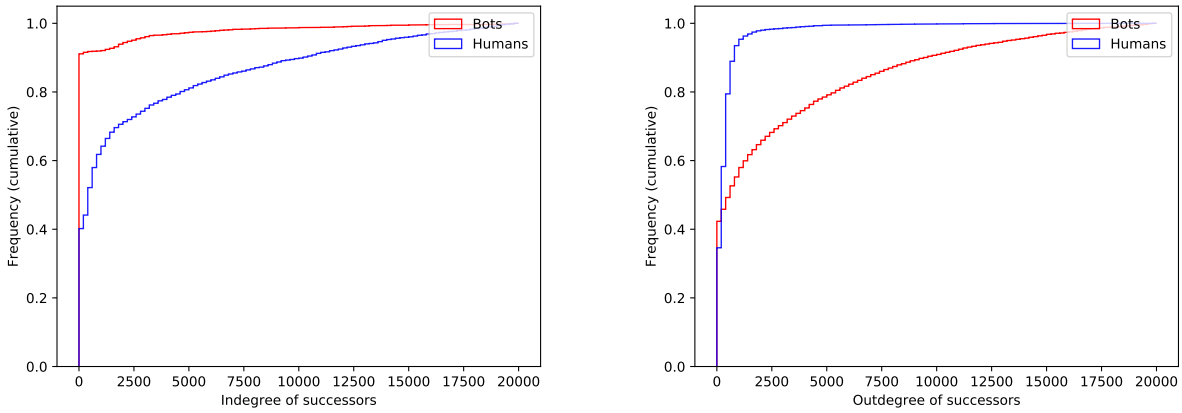
Figure 6: We compare the aggregated reputation of users following (predecessors) or followed by (successors) a given user over all human and bot accounts.



(a) in-degree (b) out-degree

Figure 7: Cumulative aggregated degree distribution of predecessor nodes, i.e. median degree of users following by a given user over all bot and human accounts.

We find that overall successors appear to have a higher reputation than predecessors. This observation makes sense as many users follow high-degree, high-reputation nodes such as news sources, which in turn are not likely to be following them back. We can also make the observation that human nodes tend to have more reputable predecessors. Additionally, the distribution of reputation of successors is extremely skewed to both extremes for bots which indicates most of them they either follow very reputable nodes or nodes with very low reputation. While the distribution is also extreme for humans, it is not skewed quite as strongly and has many more nodes with a reputation $0.1 < r < 0.9$.



(a) in-degree

(b) out-degree

Figure 8: Cumulative aggregated degree distribution of successors nodes, i.e. median degree of users followed by a given user over all bot and human accounts.

When looking at the cumulative degree distribution among predecessors (figure 7) and successors (figure 8) we find that many bot accounts in the dataset have predecessors with very high in-degree and out-degree whereas most genuine accounts seem to have lower predecessor degree distributions. For successors however, human accounts tend to have successors with a higher in-degree and lower out-degree, which is in line with our observations regarding the reputation and again indicating that human users tend to follow more accounts with many followers such as news accounts or celebrities.

We analyze all features in terms of their correlation (see appendix A.1) to find features that correlate strongly with the label and thus might be valuable for use with a classification algorithm while avoiding redundancies and features that have strong correlations to each other. Redundant features add noise to the data distribution and may make training a classifier harder and the end result less effective. Further, we analyze features in terms of their *feature importance* in a classification algorithm. We train a random forest classifier on all of the features. Then, we can extract the feature importance in the classifier for the features (see figure 9). We focus on features that rank highly and experiment with different classification algorithms by adding and removing features and observing the change in the accuracy of predictions. We find that the most effective neighborhood and egograph features are:

- median out-degree of predecessors
- median favourites of predecessors
- median status count of predecessors
- median account age of predecessors
- median favourites of successors

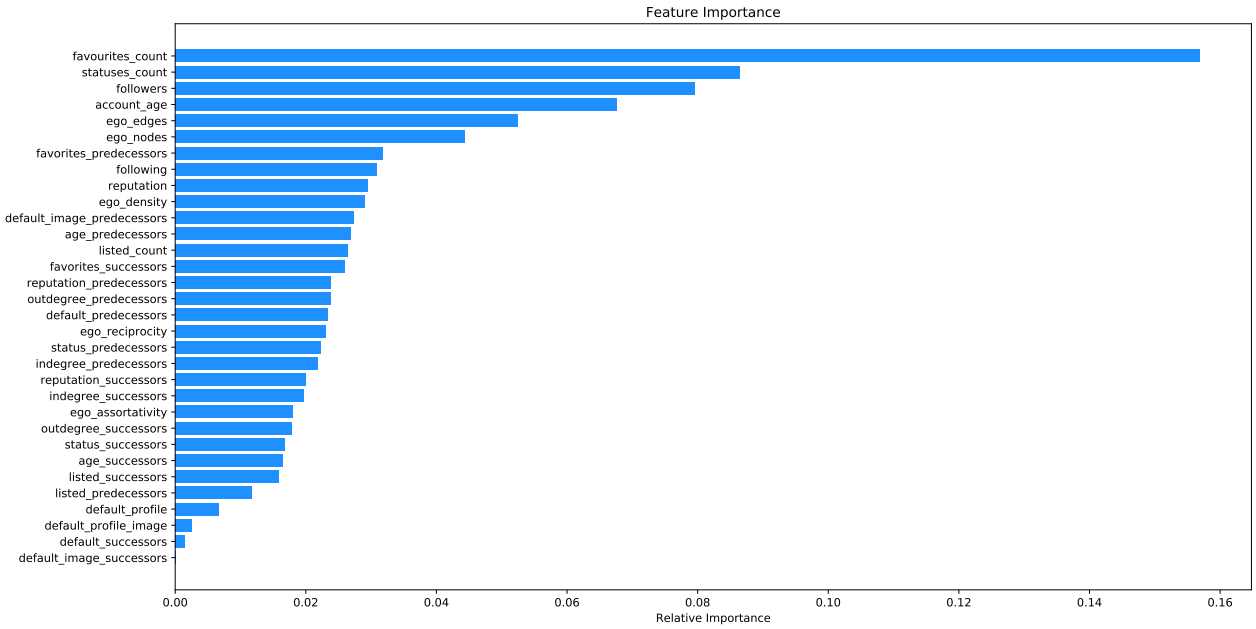


Figure 9: Feature importance of all features, including neighborhood and egograph features, extracted from a random forest classifier trained on these features.

- egograph density
- egograph reciprocity

We find that neighborhood features based on the predecessors in the graph are more effective than those based on the successors in the graph. Intuitively this makes sense. Twitter users choose who they follow and can thus make features based on successors effectively useless by simply following users a genuine user would follow. It is much harder on the other hand to *be followed* by users that seem genuine and have properties of human users.

4 Experiments

In this section we describe our model and compare it to a number of different baseline models. We implement our model as well as the baselines in Python 3.7 using NumPy, Pandas, Scikit-learn and PyTorch. We split the training data into training and test datasets randomly while ensuring that the distributions of human/bot labels in both datasets represent the overall distribution accurately.

We evaluate the model against Gaussian Naïve Bayes (GNB), Quadratic Discriminant Analysis (QDA), a Support Vector Machine (SVM), a k-Nearest Neighbors (KNN), a Random Forest (RF) and a neural network (NN) model. For these baseline models we use the

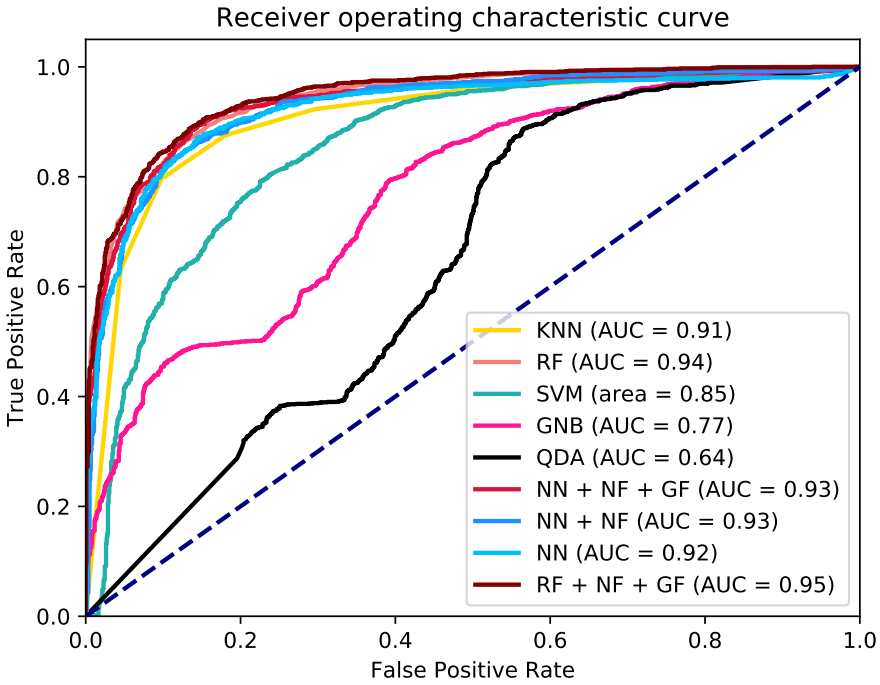


Figure 10: ROC curve for our models and the baseline models

following features:

$$X_{base} = \{\texttt{default\_profile}, \texttt{profile\_image}, \texttt{favourites\_count}, \texttt{followers}, \texttt{following}, \texttt{listed\_count}, \texttt{statuses\_count}, \texttt{account\_age}, \texttt{reputation}\}$$

After evaluating linear, polynomial and RBF kernels for the SVM model, we found the model to perform best with the RBF kernel. The RF model is trained using 100 trees, and splits are performed according to the Gini coefficient. GNB and QDA models are trained using the standard settings in Scikit-learn.

The neural network classifier is a fully-connected feedforward neural network. After much experimentation we chose to use 3 hidden layers with $(500, 200, 200)$ neurons respectively, using batch normalization and dropout with $p = 0.5$. The model is trained using Adam and a learning rate of $\eta = 0.001$ for 500 epochs. We find that the model achieves better performance and generalizes better when we remove outliers that deviate from the mean by more than three standard deviations from the training dataset.

We experimented with different sets of features and found a set of neighborhood features (NF) and graph features (GF) described in section 3.2 that performs best along with the baseline features. We also found that removing some of the original features along with adding the NF and GF improved the performance of our classification model. The final set

Model	Accuracy	TPR	FPR	F_1 score	AUC
GNB	0.6399	**0.9622**	0.7313	0.741	0.7652
QDA	0.6751	0.8937	0.5768	0.7465	0.6405
SVM	0.7797	0.7746	0.2145	0.7901	0.7801
KNN	0.8496	0.8752	0.18	0.8617	0.9074
RF	0.8675	0.8745	0.1405	0.876	0.9426
NN	0.8648	0.8673	0.138	0.8729	0.9154
NN + NF (ours)	0.8702	0.8623	**0.1208**	0.8767	0.9355
NN + NF + GF (ours)	0.8751	0.8894	0.1413	0.8841	0.9367
RF + NF + GF (ours)	**0.8774**	0.888	0.1348	**0.8858**	**0.9468**

Table 2: Results on the CRESCI-2018 dataset for various baseline models (top) and our model using neighborhood features (bottom).

of features used in our neural network classifier are:

$$X_{NF} = \{\texttt{favourites\_count}, \texttt{statuses\_count}, \texttt{outdegree\_predecessors}, \texttt{favorites\_predecessors}, \texttt{status\_predecessors}, \texttt{age\_predecessors}, \texttt{account\_age}, \texttt{followers}\}$$

$$X_{GF} = \{\texttt{following}, \texttt{ego\_density}, \texttt{ego\_reciprocity}\}$$

Note that despite not being a graph feature, we find that including `following` along with the graph features improves classification performance. For the random forest classifier we find the following features to work best:

$$X_{RF} = \{\texttt{favourites\_count}, \texttt{followers}, \texttt{statuses\_count}, \texttt{account\_age}, \texttt{outdegree\_predecessors}, \texttt{favorites\_predecessors}, \texttt{favorites\_successors}, \texttt{status\_predecessors}, \texttt{age\_predecessors}, \texttt{ego\_density}, \texttt{ego\_reciprocity}\}$$

We denote the number of correctly positively classified samples as TPR (true-positive rate) whereas FPR is the number of falsely positively classified samples (false-positive rate). The models are evaluated in terms of:

1. Accuracy
2. TPR
3. FPR

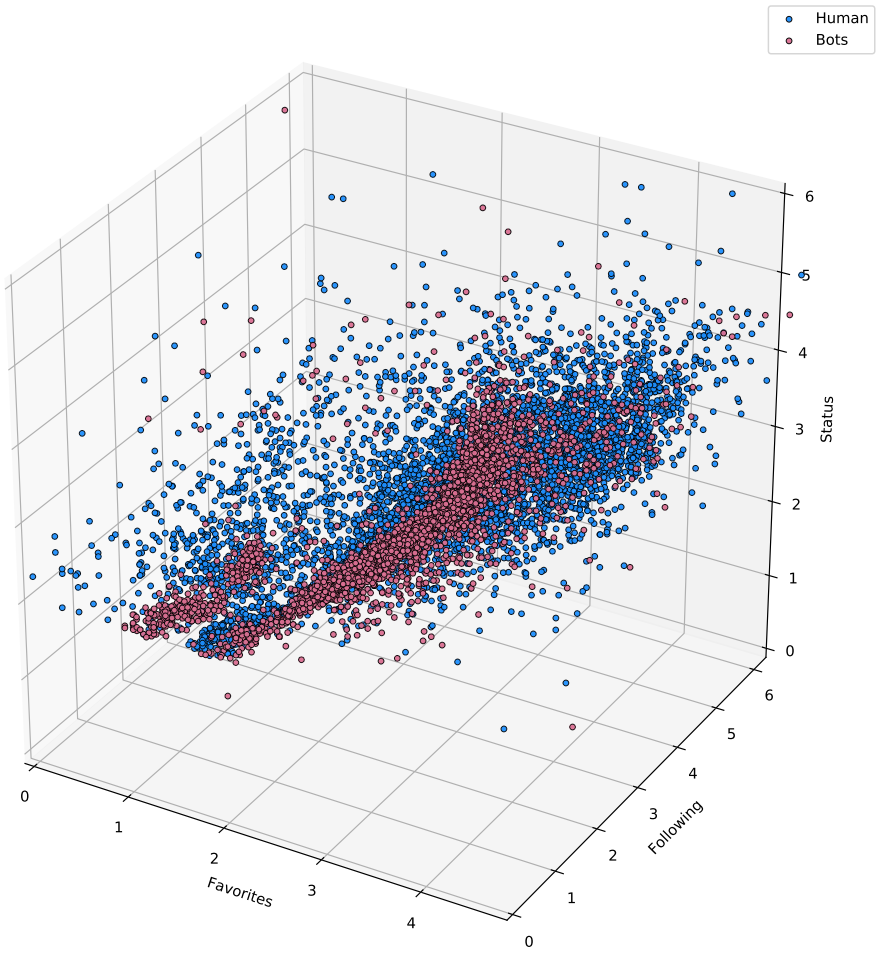


Figure 11: We can clearly see the different distributions of humans (blue) and bots (red). All axes are plotted in log scale.

4. F_1 score
5. area under the ROC curve (AUC)

Our model outperforms the baseline models on the Cresci-2018 dataset (table 2). We notice that both the GNB and QDA model are not very good at identifying bot accounts on this dataset accurately and have a very high false positive rate. The SVM model is slightly better and achieves an F_1 score of 0.79. Interestingly, KNN is surprisingly effective on this dataset and only slightly less effective than the RF and NN models. Both RF and NN turn out to be very effective and achieve the highest scores. Due to this, we chose to focus only on the RF and NN models for our proposed extensions and sets of features. By using the proposed NF we see an overall improvement both for the RF and NN models.

It is interesting to note that only a set of certain, very few features is sufficient to effectively separate bots and humans in multi-dimensional space. We find that the number of a user's favourites, followers and statuses actually suffice to identify very clear differences

in the distribution of human and bot accounts (figure 11). Indeed, a RF classifier is able to achieve an F_1 score of 0.8738 and AUC of 0.8659 using only those three features and actually performs better than all of the baseline models which were trained on all features retrieved using the Twitter API.

Lastly, we would like to note the fact that Cresci-2018 may not have been the ideal dataset and we believe that this method could work much better on higher quality data and particularly well for data with bots who deviate stronger from the norm in terms of social structure in the graph. This dataset was collected from so called "stock Twitter", a subset of Twitter users that mainly talk about stocks and market movements, and the data was focused on bots that aim to influence the market by shaping the perception of public opinion on Twitter through retweeting (i.e. sharing with their followers) of suspicious "peak tweets" [7]. We have noticed that this is also reflected in the users contained in the datasets and that these users may not be easily and reliably identified using only profile, and graph neighborhood data. Additionally, many users in the dataset that are labelled as bots do not immediately strike us as bots upon manual review which brings the quality of the dataset into question. In general the in- and out-degrees of users in the dataset (for both human and bot accounts) seem much higher than we would expect from a random sample of Twitter users. Despite these circumstances we find that our approach works and is able to improve upon baseline classification models for "free": it does not require any additional labelled data and simply augments existing labelled data by use of data from neighbors in the social graph.

5 Conclusion

We have proposed a novel way of generating aggregate neighborhood features in an unsupervised manner from unlabelled data of nodes adjacent to a user in the network's social graph. Apart from data collection efforts these features are free and lead to more accurate classification results.

We conducted experiments on the expanded Cresci-2018 dataset. In order to generate the neighborhood features we have collected a novel dataset of 4.6 million Twitter user profiles based on the users followed by or following users in the Cresci-2018 dataset. In our analysis of the retrieved data we have found a number of egograph features and aggregate features across user neighborhoods that can help in bot classification. We found the most valuable neighborhood features to be the ones based on a node's predecessors. Specifically median out-degree of predecessors, median favourites of predecessors, median status count of predecessors, median account age of predecessors, median favourites of successors and the egograph density and reciprocity seemed to be most effective on this dataset.

We compared our method to different baseline models. We focus on the best performing baseline classifiers, namely the random forest and neural network classifiers. By adding these neigborhood- and egograph-based features to the data the classification algorithm is trained on, we can improve classification performance. Our method outperforms the baseline on the Cresci-2018 dataset. However, as we believe this method to have potential for bigger

improvements over the baseline than shown in this work, we plan to test our method on higher quality data in the future.

Lastly, we do note that our method has its limitations as it only focuses on classifying users based on profile and social graph features. When it comes to classification of accounts that behave suspiciously, our method may not be effective on its own, but we believe in can be used in conjunction with existing methods to improve overall performance.

References

[1] Hunt Allcott and Matthew Gentzkow. Social media and fake news in the 2016 election. *Journal of economic perspectives*, 31(2):211–36, 2017.

[2] Qiang Cao, Michael Sirivianos, Xiaowei Yang, and Tiago Pregueiro. Aiding the detection of fake accounts in large scale social online services. In *Proceedings of the 9th USENIX conference on Networked Systems Design and Implementation*, pages 15–15. USENIX Association, 2012.

[3] Nikan Chavoshi, Hossein Hamooni, and Abdullah Mueen. Temporal patterns in bot activities. In *Proceedings of the 26th International Conference on World Wide Web Companion*, pages 1601–1606. International World Wide Web Conferences Steering Committee, 2017.

[4] Sudipta Chowdhury, Mojtaba Khanzadeh, Ravi Akula, Fangyan Zhang, Song Zhang, Hugh Medal, Mohammad Marufuzzaman, and Linkan Bian. Botnet detection using graph-based feature clustering. *Journal of Big Data*, 4(1):14, 2017.

[5] Laurenz A Cornelissen, Richard J Barnett, Petrus Schoonwinkel, Brent D Eichstadt, and Hluma B Magodla. A network topology approach to bot classification. In *Proceedings of the Annual Conference of the South African Institute of Computer Scientists and Information Technologists*, pages 79–88. ACM, 2018.

[6] 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.

[7] 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.

[8] Alceu Ferraz Costa, Yuto Yamaguchi, Agma Juci Machado Traina, Caetano Traina Jr, and Christos Faloutsos. Rsc: Mining and modeling temporal activity in social media. In *Proceedings of the 21th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, pages 269–278. ACM, 2015.

[9] Nir Grinberg, Kenneth Joseph, Lisa Friedland, Briony Swire-Thompson, and David Lazer. Fake news on twitter during the 2016 us presidential election. *Science*, 363(6425):374–378, 2019.

[10] Supraja Gurajala, Joshua S White, Brian Hudson, and Jeanna N Matthews. Fake twitter accounts: profile characteristics obtained using an activity-based pattern detection approach. In *Proceedings of the 2015 International Conference on Social Media & Society*, page 9. ACM, 2015.

[11] Sarah Harun, Tanveer Hossain Bhuiyan, Song Zhang, Hugh Medal, and Linkan Bian. Bot classification for real-life highly class-imbalanced dataset. In *2017 IEEE 15th Intl Conf on Dependable, Autonomic and Secure Computing, 15th Intl Conf on Pervasive Intelligence and Computing, 3rd Intl Conf on Big Data Intelligence and Computing and Cyber Science and Technology Congress (DASC/PiCom/DataCom/CyberSciTech)*, pages 565–572. IEEE, 2017.

[12] Haewoon Kwak, Changhyun Lee, Hosung Park, and Sue Moon. What is twitter, a social network or a news media? In *Proceedings of the 19th international conference on World wide web*, pages 591–600. AcM, 2010.

[13] Jonas Lundberg, Jonas Nordqvist, and Mikko Laitinen. Towards a language independent twitter bot detector. In *DHN*, pages 308–319, 2019.

[14] Anshu Malhotra, Luam Totti, Wagner Meira Jr, Ponnurangam Kumaraguru, and Virgilio Almeida. Studying user footprints in different online social networks. In *Proceedings of the 2012 International Conference on Advances in Social Networks Analysis and Mining (ASONAM 2012)*, pages 1065–1070. IEEE Computer Society, 2012.

[15] Kai Shu, Amy Sliva, Suhang Wang, Jiliang Tang, and Huan Liu. Fake news detection on social media: A data mining perspective. *ACM SIGKDD Explorations Newsletter*, 19(1):22–36, 2017.

[16] Cao Xiao, David Mandell Freeman, and Theodore Hwa. Detecting clusters of fake accounts in online social networks. In *Proceedings of the 8th ACM Workshop on Artificial Intelligence and Security*, pages 91–101. ACM, 2015.

A Appendix

A.1 Feature correlation heatmap

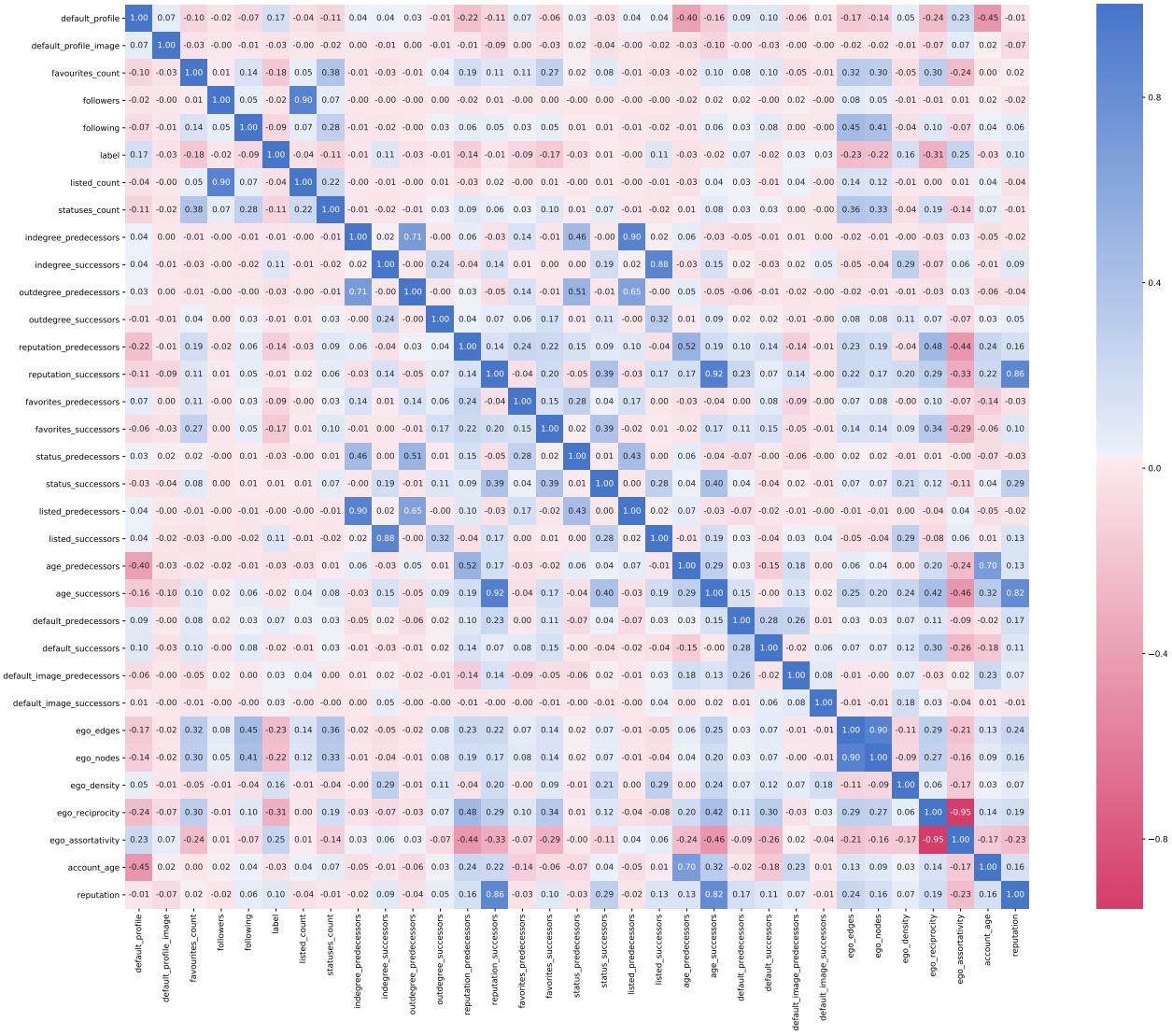


Figure 12: Correlation heatmap of all features scraped or generated (neighborhood features and egograph features)

A.2 Full disclosure with regards to dissertations/projects

Bebensee: His research and thesis is not related to this project in any way. He is working on neural machine translation and visual question answering.

Nazarov: His research and thesis is not related to this project in any way. He is working on designing an open framework for designing, implementing, and evaluating hardware and software components for solid-state drives.

Contents