You Are Known by Your Friends: Leveraging Network Metrics for Bot Detection in Twitter

David M. Beskow and Kathleen M. Carley

Abstract Automated social media *bots* have existed almost as long as the social media platforms they inhabit. Although efforts have long existed to detect and characterize these autonomous agents, these efforts have redoubled in the recent months following sophisticated deployment of bots by state and non-state actors. This research will study the differences between human and bot social communication networks by conducting an account snow ball data collection, and then evaluate network, content, temporal, and user features derived from this communication network in several bot detection machine learning models. We will compare this model to the other models of the *bot-hunter* toolbox as well as current state of the art models. In the evaluation, we will also explore and evaluate relevant training data. Finally, we will demonstrate the application of the *bot-hunter* suite of tools in Twitter data collected around the Swedish National elections in 2018.

1 Introduction

Automated and semi-automated social media accounts have been thrust into the forefront of daily news as they became associated with several publicized national and international events. These automated accounts, often simply called *bots* (though at times called *sybils*), have become agents within the increasingly global marketplace of beliefs and ideas. While their communication is often less sophisticated and nuanced than human dialogue, their advantage is the ability to conduct timely informational transactions effortlessly at the speed of algorithms. This advantage has led to a variety of creative automated agents deployed for beneficial as well as harmful effects. While their purpose, characteristics, and "puppet masters" vary widely, they are undeniably present and active. Their effect, while difficult if not impossible to measure, is tangible.

D. M. Beskow (✉) · K. M. Carley
School of Computer Science, Carnegie Mellon University, Pittsburgh, PA, USA
e-mail: dbeskow@andrew.cmu.edu; kathleen.carley@cs.cmu.edu

M. A. Tayebi et al. (eds.), *Open Source Intelligence and Cyber Crime*, Lecture Notes in Social Networks, https://doi.org/10.1007/978-3-030-41251-7_3

Automated and semi-automated accounts are used for a wide variety of reasons, creating effects that can be positive, nuisance, or malicious. Examples of positive bots include personal assistants and natural disaster notifications. Nuisance bots are typically involved in some type of 'spam' distribution or propagation. The spam content ranges from commercial advertising to the distribution of adult content. Malicious bots are involved in propaganda [52], suppression of dissent [64], and network infiltration/manipulation [8].

Malicious bots have recently gained wide-spread notoriety due to their use in several major international events, including the British Referendum known as "Brexit" [43], the American 2016 Presidential Elections [13], the aftermath of the 2017 Charlottesville protests [35], the German Presidential Elections [55], the conflict in Yemen [7], and recently in the Malaysian presidential elections [4]. These accounts attempt to propagate political and ideological messaging, and at times accomplish this through devious cyber maneuver.

As these bots are used as one line of effort in a larger operation to manipulate the marketplace of information, beliefs, and ideas, their detection and neutralization become one facet of what is becoming known as *social cyber security*. Carley et al. is the first to use this term, and defines it as:

> Social Cyber-security is an emerging scientific area focused on the science to characterize, understand, and forecast cyber-mediated changes in human behavior, social, cultural and political outcomes, and to build the cyber-infrastructure needed for society to persist in its essential character in a cyber-mediated information environment under changing conditions, actual or imminent social cyber-threats. [18]

Within *social cyber security*, bot detection and neutralization are quickly becoming a *cat and mouse* cycle where detection algorithms continuously evolve trying to keep up with ever-evolving *bots*. Early detection algorithms exploited the automated timing, artificial network structure, and unoriginal meta-data of automated accounts in order to identify them. These features are relatively easy for bot puppet-masters to manipulate, and we are now seeing automated accounts that have meaningful screen names, richer profile meta-data, and more reasonable content timing and network characteristics.

We are also seeing an increasing number of accounts that we call "bot assisted" or "hybrid" accounts (also at times called "cyborg" accounts). Although researchers often attempt a binary classification of *bot* or *human*, the reality is that there is a spectrum of automated involvement with an account. Many accounts are no longer strictly automated (all content and social transactions executed by a computer). These accounts will have human intervention to contribute nuanced messaging to two-way dialogue, but will have a computer executing a variety of tasks in the background. Grimme et al. [38] discusses this spectrum in detail, describing how 'social bots' are created, used, and how 'hybridization' can be used to bypass detection algorithms (in their case successfully bypassing the 'Botornot' algorithm discussed later in this paper).

We hypothesize that bots are not involved in social networks and social communication in the same way that humans are, and that this difference is measurable. Like other complex systems (natural ecosystems, weather systems, etc), social

interaction and relationships are the result of myriads of events and stimuli in both the real and virtual worlds. Because bots lack real world engagement and social environments, they embed in different networks than humans.

Many bots are programmed to interact with each other as a bot network, and attempt to interact with humans, but many features of these interactions will be 'robotic'. Even 'hybrid' accounts will have some level of artificial and inorganic structure and substance in their communications. This area of bot detection in Twitter is largely unexplored, primarily because the rich network data (both the friends/followers network as well as their conversational network) are very time consuming to collect. We therefore set out to collect the data to characterize the social network(s) and social conversation(s) that a twitter account participates in, describe these networks with various network metrics, leverage these rich network metrics in traditional machine learning models, and evaluate whether the time involved creates substantial value.

1.1 Research Questions

1. Do bot Twitter accounts have fundamentally different conversational network structures than human managed accounts?
2. Do the conversations that surround bot accounts diverge from human conversations in general substance and timing?
3. Can the measured differences between bot and human conversation networks lead to increased accuracy in bot detection?

This paper will begin by discussing past bot detection techniques, as well as summarize historical techniques for extracting features from network structures. Next we discuss our data collection, data annotation, and methodology for creating ego-network metrics. We describe training and testing our bot-hunter machine learning algorithms and present our results. We construct an evaluation to compare all bot-hunter models against the state of the art. Finally, we will demonstrate the application of the bot-hunter suite of tools in the 2018 Swedish National elections, providing a possible workflow to open source intelligence practitioners.

This chapter is an extension of [10], with a focus of extending the feature space beyond network metrics to include content and temporal metrics of the larger ego network. Several of these features are novel, including a cascaded classifier that identifies portion of alters that are likely bots, portion of alters that don't have normal daily rhythms, as well as portion of ego network that produces tweets that are more popular than the account itself. All of these have been documented as attributes of bots, and we've coded them into features in this algorithm. Additionally, we used the larger models to explore several new bot data sets. Finally, this extension will compare all of the bot-hunter suite of tools against state of the art models.

2 Related Work

2.1 Understanding Data Tiers

In earlier research our team proposed a tiered approach to bot detection [11] that mirrors the data tiers introduced below. This tiered approach creates a flexible bot-detection "tool-box" with models designed for several scenarios and data granularities. Tier 0 builds models on a single entity (usually a tweet text or *user* screen name). Tier 1 builds models based on features extracted from the basic Tweet object (and associated *user* object). Tier 2 extracts features from a users' timeline, and Tier 3 (explained in this paper) builds features from the conversation surrounding a user. Higher tier models are generally more accurate, but consume more data and are therefore computationally expensive. Some research requires bot detection at such a scale, that models based on Tier 0 or Tier 1 are the only feasible option. At other times, highly accurate classification of a few accounts is required. In these cases, models based on Tier 2 or Tier 3 data are preferred. This paper proposes an approach to Tier 2–3 bot detection that builds on the previous Tier 0 [12] and Tier 1 [11] research and relies heavily on network metrics collected through single seed snowball sampling. We will view past research in bot detection through the lens of these Tiers (Table 1).

Since the early efforts to conduct bot/spam detection, numerous teams have developed a variety of models to detect these. While similar, these models will differ based on the underlying data they were built on (for example many community detection and clickstream models were developed for Facebook, while the overwhelming majority of models built on Twitter data use Supervised and Unsupervised Machine learning [1]). Even in Twitter bot detection, these models can be grouped by either the models/methods or by the data that they use. We have provided Table 2 to outline the connection between past models and the data that they use.

Adewole et al. [1] reviewed 65 bot detection articles (articles from 2006–2016) and found that 68% involved machine learning, 28% involved graph techniques (note that these include some machine learning algorithms that rely heavily on

Table 1 Four *tiers* of Twitter data collection to support account classification (originally presented in [11])

Tier	Description	Focus	Collection time per 250 accounts	# of Data entities (i.e. tweets)
Tier 0	Tweet text only	Semantics	N/A[a]	1
Tier 1	Account + 1 Tweet	Account meta-data	~1.9 s	2
Tier 2	Account + Timeline	Temporal patterns	~3.7 min	200+
Tier 3	Account + Timeline + Friends Timeline	Network patterns	~20 h	50,000+

[a]This tier of data collection was presented by Kudugunta and Ferrara [47] and assumes the status text is acquired outside of the Twitter API

Table 2 Table of Twitter Bot detection models and the data that they use

Data	Community detection	Machine learning		Crowd sourcing
		Supervised	Unsupervised	
Tier 0 Text		[12, 47]	[48]	
Tier 1 + Profile		[22, 49]	[33]	
Tier 2 + History		[63]	[20]	
Tier 3 + Snowball	[8]	No known research		[66]
Stream		[3, 14]		

network metrics), and 4% involved crowd-sourcing. Below we will summarize the salient works under each of these modeling techniques.

2.2 Machine Learning Techniques

As noted above, Twitter bot detection has primarily used Machine Learning models. The *supervised* machine learning models used for bot detection include Naïve Bayes [22], Meta-based [49], SVM [48], and Neural Network [47]. The *unsupervised* machine learning models used include hierarchical [48], partitional [33], PCA-based [65], Stream-based [53], and correlated pairwise similarity [20]. Most of these efforts leverage data collected from the basic tweet object or user object (what we would define as a *Tier 0* or *Tier 1* model).

In 2014, Indiana University launched one of the more prominent supervised machine learning efforts with the *Bot or Not* online API service [25] (the service was recently rebranded to *Botometer*). This API uses 1150 features with a random forest model trained on a collage of labeled data sets to evaluate whether or not an account is a bot. *Botometer* leverages network, user, friend, temporal, content, and sentiment features with Random Forest classification [28].

In 2015 the Defense Advanced Research Projects Agency (DARPA) sponsored a Twitter bot detection competition that was titled "The Twitter Bot Challenge" [59]. This 4 week competition pitted four teams against each other as they sought to identify automated accounts that had infiltrated the informal Anti-Vaccine network on Twitter. Most teams in the competition tried to use previously collected data (mostly collected and tagged with *honey pots*) to train detection algorithms, and then leverage tweet semantics (sentiment, topic analysis, punctuation analysis, URL analysis), temporal features, profile features, and some network features to create a feature space for classification. All teams used various techniques to identify initial bots, and then used traditional classification models (SVM and others) to find the rest of the bots in the data set.

2.3 Other Techniques

Several other novel bot detection methods exist outside of machine learning and network based approaches. Wang et al. [66] investigated the idea of Crowd Sourcing bot detection. While showing limited success, it was costly at scale, and usually required multiple workers to examine the same account. Another unique type of unsupervised learning involves algorithms that find and label correlated accounts. Most bots are not deployed by themselves. Even if not deployed as a united bot-net, many *bot herders* often task multiple bots to perform the same operations. Chavoshi et al. [20] has leveraged the semantic and temporal similarity of accounts to identify bots in an unsupervised fashion, creating the *Debot* model which we will compare against in our results section.

2.4 Network Based Techniques

Networks are an extremely important part of bots, bot behavior and bot detection. Aiello et al. [2] discusses the impact of bots on influence, popularity, and network dynamics. Adewole et al. [1] highlights that network features are robust to criminal manipulation.

One approach to leveraging network structure involves community based bot/sybil detection. While community detection has been effectively implemented on Facebook [67] and Seino Weibo [51], it has only recently been used on Twitter Data due the strict friend/follower rate limiting discussed above. Only recently has Benigni et al. [9] used dense subgraph detection to find extremists and their supporting bots in Twitter.

Most research that uses networks for bot detection with Twitter Data are in fact creating network based metrics and introducing these features in traditional machine learning models. As discussed below, the most challenging part of this type of research is focused on how to build networks from limited data. The closest works to ours were performed by Bhat and Abulaish [14] in 2013 and [3] in 2016. Both research efforts used network features along with profile and temporal features from a Twitter Sample Stream without any snowball sampling enrichment. They created an egocentric network that involved ego, alters, with links between alters for both following and mention ego centric networks. Having done this, they calculated content, profile, and social interaction features. Their network features were restricted to centrality measures, density measures, and weak and strongly connected components. A similar earlier work by Bhat and Abulaish [14] attempts to use community features (number of communities, core/periphery, foreign in/out degree, etc). This was applied to both Facebook data and the Enron email data (not to Twitter).

Additionally, the *Botometer* algorithm leverages some network features extracted from the user timeline. This includes metrics on the *retweet* network, *mention* network, and *hashtag* co-occurrence network. The metrics include density, degree

distributions, clustering coefficient, and basic network characteristics. The *Botometer* algorithm does not conduct a snowball collection of *friends* or *followers*, but does appear to collect user objects for accounts found in the timeline as a *retweet* or *mention* [28].

2.5 Building Networks with Twitter Data

As noted above, however, it is difficult to quickly build comprehensive network structure with Twitter data due the Twitter API rate limits, primarily associated with collecting friend/follower ties. Researchers have generally used one of two methods to build limited networks.

The first method is used if the research team has a large sample or stream. These samples may be random (collected from the 1% Twitter Sample) or they may be associated with an event or theme (i.e. collecting all Tweets that have a given hashtag like #hurricanesandy). These researcher then build ego-centric networks from this stream, without collecting any additional data from the Twitter API. This has the advantage of speed, and doesn't suffer from issues getting data for suspended accounts. This method, however, will only model a small portion of an account's activity and network. A 1% sample will arguably contain marginal activity for given account, and even topical streams will only contain a small part of an account's activity, given that they are involved in multiple topics and discussions. These small samples may not be rich enough to serve as strong features for machine learning.

The second method that researchers use is to only collect the users *timeline* (history of tweets, up to last 3200). They then build an ego-centric network from this data (variously using replies, retweets, hashtags, urls, and mentions to build networks). This is much richer than the first method, contain all of the users activity, but still lacks any information beyond that individual, providing the limited star graph illustrated in Fig. 1. It doesn't contain the larger conversation(s) that they are participating in. Additionally, a bots's *timeline* is completely managed by the bot *puppet master*, and therefore can be manipulated to avoid detection.

To date our team has not found supervised learning bot detection research that leverages extensive snowball sampling to build ego networks.

2.6 Extracting Features from Social Networks

Evaluating network centrality measures, started by Bavelas in 1948 [6] and effectively clarified by Freeman in 1978 [31], has long been an important metric for evaluating both nodes and networks. According to Freeman, network-level centrality metrics measure the "compactness" of the network. Our model includes several network centrality measures: degree centrality, k-betweenness [5], and eigenvector centrality [45] are used to measure differing "compactness" between human and bot conversation networks.

Several seminal works describe the importance of triadic relationships in social networks [19, 40] and as a foundation for measuring network clustering and groups [42]. The fact that the study of triadic relationship has almost exclusively been contained within the study of social interaction provides evidence that these observed triadic relationships are unique to human behavior. We have therefore included several features based on these triadic relationships, including the full triadic census [41], number of Simmelian Ties [26, 46], and clustering coefficient. We also included reciprocity based on Mislove et al.'s [54] examination of reciprocity in online social networks.

In addition to finding network centrality and triadic structures, network community detection has been an important aspect of network characterization, and is still an active research area. Current group detection techniques generally fall into traditional methods, divisive methods, modularity based methods, statistical inference methods, and dynamic methods [29]. Our community detection features leverage Louvain Clustering [16], which is based on modularity optimization.

Our approach uses network sampling in order to restrict the time of computation. While research in network sampling started in the 1970s with work from [30] and others, the emergence of Online Social Networks (OSN's) increased the size of networks and the need for sampling. Our approach to sampling ego networks was informed by Gjoka [34]. Our sampling uses breadth-first-search (BFS) on the target node. The known bias of BFS is eliminated because we are only conducting two hops from the target (only includes friends of alters).

Finally, the study of ego networks is a special branch of social network analysis that is relevant to our study. In 1972, [37] presented the classic concept of the "Strength of Weak Ties" in ego networks, which [17] clarified is more due to the structural location of ties, and can be measured by effective size, efficiency, and constraint. This informed our use of ego network effective size in our features space. Additionally, Centrality of ego-networks was explored by Freeman [32] in 1982 informing our use of betweenness in the feature space.

2.7 Contributions of This Work

While we discuss above several other research attempts to use network metrics in a bot detection feature space, these have largely relied on the mention network extracted from any Twitter query/stream. Ego-centric networks built on a single stream/query arguably contain only a small subset of the overall account ego network. Researchers have not attempted to build this ego network based on snowball sampling [36] with a seed node since this requires significant time given the extent of the data and the strict API rate limits that Twitter imposes on friend/follower data. Our research has taken the time to build this rich conversational network in a novel way, and then evaluate whether the time and effort render sufficient value.

Having built this extensive network for every account in question, this work attempts to fully exploit all available features, going above and beyond just

structural features. These additional features include content, temporal, and user summary features. Adding the full range of additional features allows us to fully evaluate the increased accuracy against the additional computational cost.

This work additionally creates and explores bot detection metrics that require greater effort and sophistication to circumvent. Currently, bot-herders can circumvent current algorithms by changing their screen name, adding account meta-data, spending additional time selecting a unique profile picture, and creating a more realistic tweet inter-arrival time. They can also deploy bots in bot networks, therefore artificially manipulating friend/follower values to appear like they are popular. However, it will arguably require significantly more sophistication to change the centrality, components, or triadic relationships in the conversations that they participate in. By increasing the cost to deploy and operate bots, it may economically force "bot-herders" out of their devious market.

Finally, the *bot-hunter* framework builds on the multi-tiered bot detection approach that we introduced in [11]. This multi-tiered approach provides researchers and government or non-governmental agencies with a "tool-box" of models designed for different classes of bots as well as different scales of data (designed for either high volume of high accuracy). This multi-tiered approach acknowledges that there is not a one-size fits all model/approach that will work for all bot detection requirements. By merging and expanding on past bot detection research, we can create an easy to use "tool box" that can address several bot-detection requirements. The evaluation provided later in this paper will demonstrate that key models in the bot-hunter suite of tools are equivalent or better than state of the art models.

3 Data

Our team used the Twitter REST and Streaming API's to access the data used in this research effort. Details of this process are provided below.

3.1 Overview of Available Data

Research is loosely divided between account-focused data collection strategies and topical or stream based collection strategies. Account based approaches will only use data objects directly tied to the user (user JSON object, user time-line object, etc). Stream-based approaches extract features from a given topical stream or twitter stream sample. These stream based features are often network features, but represent a small fraction of the ego-centered network of a given account. Our research therefore pursues an account based approach to build a fuller representation of the account's ego network.

Researchers must find a balance between speed and richness of data. Past account focused research generally falls into four *tiers*. Table 1 provides a description for each *tier* of data collection, the estimated time it would require to collect this data for 250 accounts, and the amount of data that would be available for feature engineering per account.

3.2 Data Required for Account Conversation Networks

Detailed ego network modeling of a Twitter account's social interactions requires Tier 3 data collection, but to date our team has not found any research that has conducted that level of data collection to model the network structures and social conversations that an automated Twitter account interacts with. In fact, few teams go beyond basic in-degree (follower count) and out-degree (friend count) network metrics found in Tier 1 meta-data. The closest effort to date is the *Botometer* model, which arguably operates at *Tier 2*. By adding the user timeline, *Tier 2* provides limited network dynamics, to include being able to model hashtag and URL co-mentions in a meta-network (see Fig. 1). The resulting *timeline* based network, however, lacks comprehensive links between alters. While the time-line can provide rich temporal patterns, we found that it lacked sufficient structure to model the ego network of an actor.

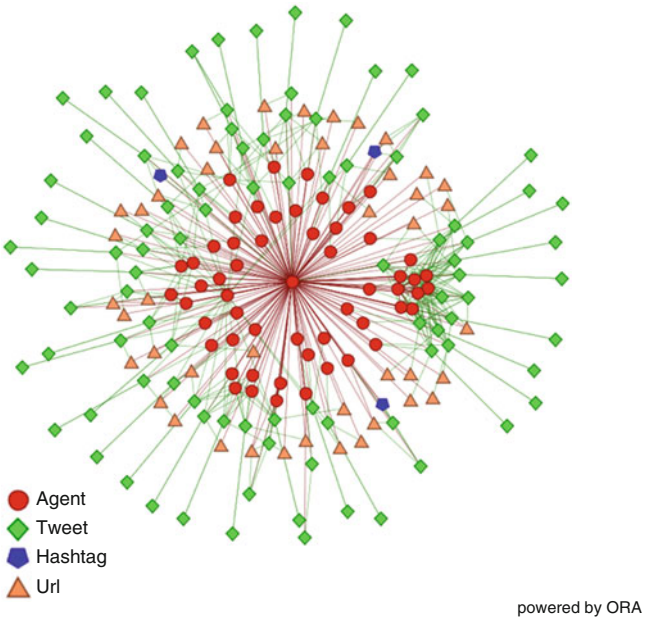

Fig. 1 Leveraging only user *timeline* provides limited network features in a *star graph*

We set about to build the social network and social conversations that a twitter account is interacting with. We also tried to do this in a way that would expedite the time it takes to collect the data and measure network metrics. Our initial goal was to collect data, build the feature space, and classify an account within 5 min. We selected the 5 min limit in an attempt to process ~250 accounts per day with a single thread

To collect the necessary data, we executed the following steps sequentially:

1. Collect user data object
2. Collect user timeline (last 200 tweets)
3. Collect user followers (if more than 250, return random sample of 250 followers)
4. Collect follower timelines (last 200 tweets)

When complete, this data collection process (illustrated in Fig. 2) creates up to 50,000 events (tweets) that represent the conversation and virtual social interaction that the user and their followers participate in.

The resulting network, while partially built on social network structure (the initial *following* relationship), is primarily focused on the larger conversation they participate in. We initiated the single seed snowball by querying *followers* rather

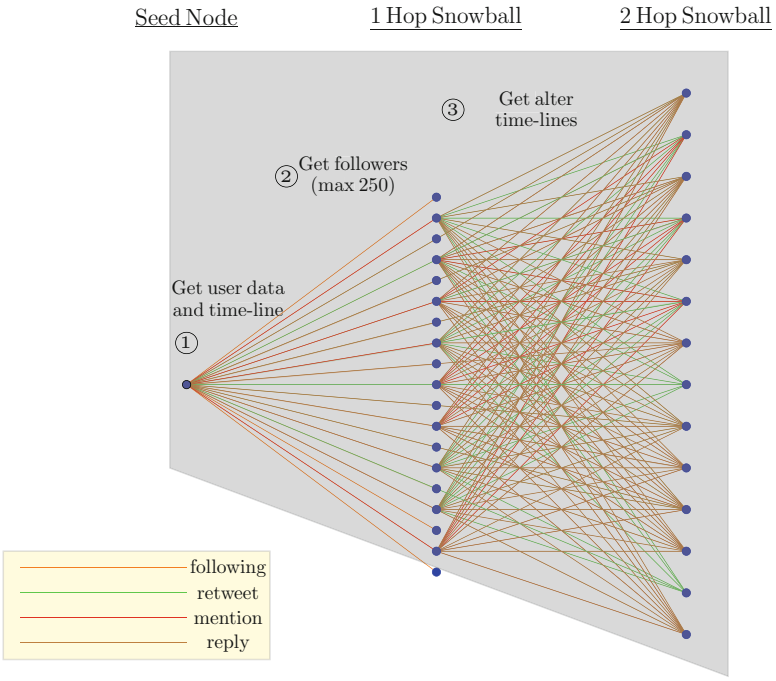

Fig. 2 Illustration of 2-hop snowball sampling: conversation of target node and followers. First get followers of target node (if more than 250, sample followers). Then get timelines of alters. Use timelines to draw connections to accounts that alters *retweet*, *reply*, and *mention*

than *friends* since *followers* are much less controlled by the bot-herder, and contain fewer news and celebrity accounts. We conducted a *timeline* rather than *followers* search for the 2nd hop of the snowball to overcome rate-limiting constraints and to model the conversation network rather than directly model the social network. This single seed snowball process conducts a limited breadth-first-search starting with a single seed and terminating at a depth of 2.

Artificially constraining the max number of alters at 250 was a modeling compromise that facilitates the self-imposed 5 min collect/model time horizon. The choice of 250 allows our process to stay under 5 min, and also represents the upper bound of Dunbar's number (the number of individuals that one person could follow based on extrapolations of neocortex size) [27]. Additionally, in evaluating a sample of 22 million twitter accounts, we found that 46.6% had less than 250 followers. This means that approximately 50% of accounts will have their entire ego network modeled. Bots tend to have fewer followers than human accounts and from the 297,061 annotated bot accounts that we had available for this research, 72.5% of them had fewer than 250 followers. Given that this compromise will only affect 25% of the bot accounts and 50% of all accounts, we felt that it was appropriate.

We used this data to create an agent to agent network where links represent one of the following relationships: mention, reply, retweet. These collectively represent the paths of information and dialogue in the twitter "conversation". We intentionally did not add the *follow/friend* relationships in the network (collected in the first hop of the snowball) since *follow/friend* relationships are an easy metric for bot herders to simulate and manipulate with elaborate bot nets. Complex conversations, however, are much harder to simulate, even in a virtual world. Additionally, adding the *following* links between the ego and alters would have created a single large connected graph. By leaving them out, we were able to easily identify the natural fragmentation of the social interaction.

3.3 Visualizing Conversations

During our initial exploration, we visualized these *conversations* for both human accounts and bot accounts. A comparison of these conversations is provided in Fig. 3. Note that bots tend to get involved in isolated conversations, and the followers of the bot are very loosely connected. The network created from a human virtual interaction on Twitter, is highly connected due to shared friendship, shared interests, and shared experiences in the real world.

3.4 Annotated Data

For annotated bot data, we combined several legacy annotated bot data sets as well as some that our team has annotated during the development of the *bot-hunter*

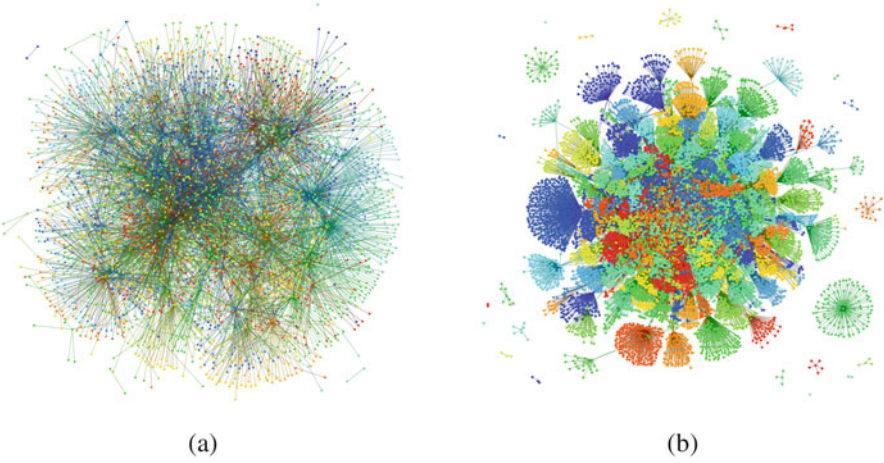

Fig. 3 Differences between a human Twitter conversation(s) and a bot Twitter interactions (networks colored by Louvain group) [10]. (**a**) Human *conversation*. (**b**) Bot *conversation*

toolbox. Note that Tier 3 model requires additional collection of friends, followers, and followers timeline, and therefore requires accounts that are not suspended. Several rich annotated bot data sets were used for our Tier 1 and Tier 2 models have a high number of suspended accounts, and therefore were not used for the development of a Tier 3 model. These datasets will still be discussed in the results and evaluation sections since they were used in the development of Tier 1 and Tier 2 models.

The first data set used for Tier 3 training data is a large diverse bot data set that was annotated by detecting 15 digit random alpha-numeric strings as indicated in [12] (a data annotation method using a Tier 0 model). This method provided 1.7 million annotated bot accounts. From this data we built network metrics on 6874 of these accounts. The second data set is from the Debot bot detection system [21] which includes bots that were found due to correlated activity. Using the Debot API, our team extracted 6949 of these accounts, from which we built network metrics on 5939 accounts. Additionally, we used the bot data manually annotated by Cresci et al. in 2015 [23] and again in 2017 [24].

In the results section we will discuss several other data sets that were used to train our Tier 1 and Tier 2 models. These include the annotated data our team captured in a bot attack on the NATO and the Digital Forensic Labs [11]. This data will be referred to as NATO in the results. We also used the suspended Russian bot data set that Twitter released in October 2018 [62]. This data set primarily contains bot/cyborg/troll activity generated by the Russian Internet Research Agency (IRA) during the 2016 US National Elections. In our results sections, this data set is referred to as the IRA data. Finally, we used a large data set of suspended accounts. To acquire this data, our team streamed the 1% Twitter Sample for 7 months, and

Table 3 Data description

Training data	Description	Tier1	Tier2	Tier3
Cresci 2017	Manually annotated by Cresci et al. in 2017 [24]	X	X	X
Cresci 2015	Manually annotated by Cresci et al. in 2015 [23]	X	X	X
Debot data	Accounts labeled as bots by the Debot bot detection system [21]	X	X	X
NATO	Data our team captured in a bot attack on the Digital Forensic Labs and NATO [11]	X	X	
Suspended accounts	These are accounts that were suspended by Twitter	X		
Random string accounts	Accounts with 15 digit random alpha-numeric strings as screen names [12]	X	X	X
IRA data	Suspended Russian bot dataset that Twitter released in October 2018	X		
Combined data	Combination of data listed above	X		

then went back to discover which of the accounts had been suspended. A similar data collection technique was used by Thomas et al. in [61].

The IRA and *suspended* data sets were only used for Tier 1, since timeline and followers were not available for Tier 2 and Tier 3. For the NATO accounts, 96% of the accounts in this dataset have been suspended. We were able to collect sufficient data for Tier 2, but not Tier 3. A summary of each data set is provided in Table 3 and cross walked with the models that it was used with. Note that the Varol data set is not provided here and was not used in our latest bot-hunter models since it is dated and did not perform well.

These data sets contain a wide variety of bots. The Varol data set was founded on the original 2011 Caverlee [50] Honey Pot data, but was supplemented with manual annotations (we leveraged only the manually annotated data). The Cresci data contains both traditional spambots (largely commercial spambots) as well as social spambots (both commercial and political). The random string data contains a large variety of bots ranging from political bots focused on the Middle East to hobby bots focused on Japanese Anime. The Debot data is also fairly diverse, with the one unifying feature that they are all have content and timing correlated with other accounts. The differences in these bots are demonstrated in the t-Distributed Stochastic Neighbor Embedding (t-SNE) dimensionality reduction that we conducted on 2000 randomly sampled accounts from the combined data set (see Fig. 4). Here we see that the Debot Data appears to be separate and different from the Varol, Cresci, and Random String data, which appear to be more uniformly distributed in this 2 dimensional representation of the data.

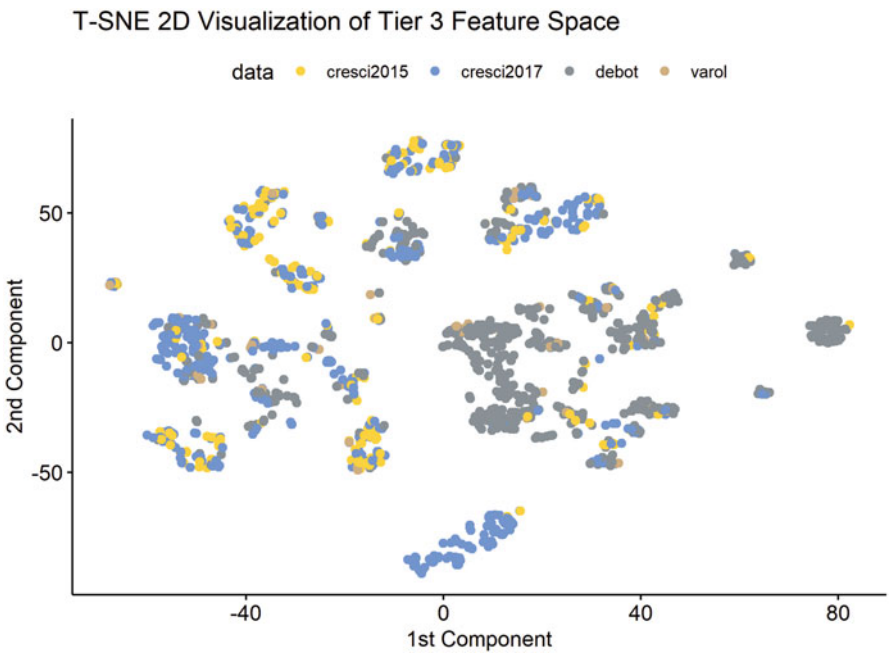

Fig. 4 t-SNE dimentsionality reduction of Tier 3 feature space (by bot dataset)

In order to train a model, we also needed accounts annotated as *human*. We used the Twitter Streaming API to collect a sample of *normal* Twitter data, intentionally collecting both weekend and weekday data. This provided 149,372 accounts to tag as *human* Twitter accounts. Of these accounts, we were able to collect/measure network metrics on 7614 accounts.

Past research has estimated that 5–8% of twitter accounts are automated [63]. If this is true, then we mis-labeled a small amount of our accounts as *human*. We believe this is acceptable noise in the data, but will limit the performance of supervised machine learning models.

Many other research efforts attempt to annotate human accounts. We chose not to do this because, in the process, these efforts create a biased sample of Twitter, heavily skewed toward average users and under sampling Celebrity, Organizational, Political, Commercial, and other accounts that make up a sizable portion of Twitter discourse. We want our 'human' annotated data to match all non-bot accounts, without biasing it towards any part of this space. Our classification is binary, and the model will be forced to classify all accounts, even those that are under-represented in training data. Our approach therefore attempts to create a truly random sample of Twitter, at the cost of having some bot accounts labeled as 'human'.

4 Feature Engineering

In this section we will introduce our feature engineering for user, content, temporal, and network features. We extracted features from Tier 0 through Tier 3, with a focus on measuring the importance of features extracted from Tier 3. The table of proposed features is provided in Table 4. All new features (beyond the features we presented in [10]) are in bold, and from our research most of these have not been used with an ego-network collected with snowball sampling.

Note that our Tiered approach is cumulative, meaning Tier 3 feature space includes features from Tier 0, Tier 1, and Tier 2. The Tier 3 model therefore includes the Tier 2 network features created by building an entity (mention, hashtag, and URL) co-mention network based only on the user's time-line (last 200 tweets). These Tier 2 network features are distinguished in our results section by the *entity* prefix.

Table 4 Features by data collection tier (new features not presented in [10] highlighted in bold)

Source	User attributes	Network attributes	Content	Timing
User object (Tier 1)	Screen name length	Number of friends	Is last status retweet?	Account age
	Default profile image?	Number of followers	Same language?	Avg tweets per day
	Default profile image?	Number of followers	Same language?	Avg tweets per day
	Entropy screen name	Number of favorites	Hashtags last status	
	Has location?		Mentions last status	
	Total tweets		Last status sensitive?	
	Source (binned)		'bot' reference?	
Timeline (Tier 2)		Number nodes of E	Mean/max mentions	Entropy of inter-arrival
		Number edges	Mean/max hash	Max tweets hour
		Density	Number of languages	Max tweets per day
		Components	Fraction retweets	Max tweets per month
		Largest compo		
		Degree/between centrality		

(continued)

Table 4 (continued)

Source	User attributes	Network attributes	Content	Timing
Snowball sample (Tier 3)	**% w/ default image**	# of bot friends	**# of languages**	**Mean tweets/min**
	Median # tweets	Number of nodes	**Mean emoji per tweet**	**Mean tweets/hour**
	Mean age	Number of links	**Mean mention per tweet**	**Mean tweets/day**
	% w/ description	Density	**Mean hash per tweet**	**% don't sleep**
	% many likes and Few followers	Number of isolates	**% retweets**	
		Number of dyad isolates	**Mean jaccard similarity**	
		Number of triad isolates	**Mean cosine similarity**	
		Number of components >4		
		Clustering coefficient		
		Transitivity		
		Reciprocity		
		Degree centrality		
		K-betweenness centrality		
		Mean eigen centrality		
		Number of simmelian ties		
		Number of Louvain groups		
		Size of largest Louvain group		
		Ego effective size		
		Full triadic census		
		Median followers		
		Median friends		

We hypothesize that the network metrics for human conversations will have different distributions than those made by bot accounts. We also believe that these differences would provide increased performance in traditional machine detection algorithms.

We have not found research that has built a snowball sampling network for bot detection, and believe that all of the Snowball Sampling ego network features in our model are novel. To collect these at scale, our team built a Python package that wrapped around the *networkx* package [39]. We leveraged known network metrics, which are provided in Table 4 with references.

4.1 Network Features

We constructed an ego network from the data collected from snowball sampling, extracting metrics from this network in an effort to develop robust features for bot detection. As discussed earlier, this network consisted of the conversation of the account in question and up to 250 of their followers. All nodes were Twitter accounts, and links were means of directed communication in the Twitter ecosystem (retweet, mention, reply). From this network we developed basic network metrics, component level statistics, centrality metrics, triadic relationship metrics, and clustering related metrics. The basic network metrics are widely used and listed in Table 4. The other categories of metrics are described below.

Given that we did not include the *following* link in our network construction, these networks were not fully connected. As seen in Fig. 5, information from these disconnected components could be valuable in distinguishing real human networks from networks dominated by bots. Our features therefore contain multiple metrics measuring number and size of network components.

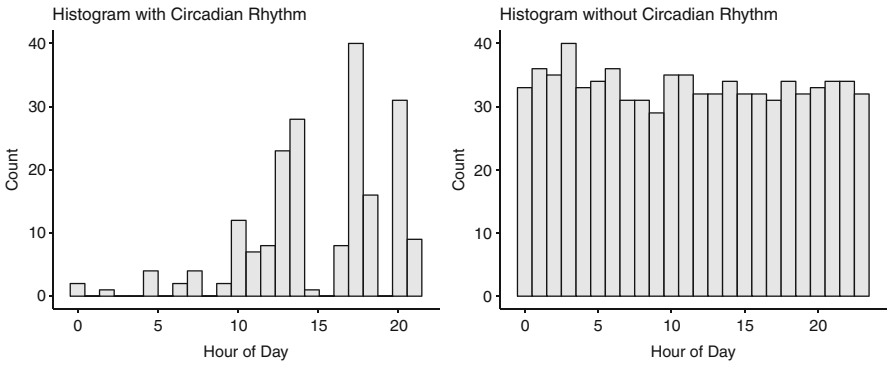

Fig. 5 Differences between a human and bot 24 h circadian rhythms. (**a**) Human *circadian rhythms*. (**b**) Bot *without* circadian rhythms

We included several network centrality metrics in our feature space, and found that they were routinely strong bot predictors. These metrics included mean degree centrality, mean eigenvector centrality, and mean K-betweenness centrality where $K = \min(500, N_{nodes})$.

In addition to analyzing the components, we also computed Louvain grouping and developed metrics based on these groups. We chose the Louvain grouping algorithm given its proven performance on larger data sets. Having computed the Louvain groups, we included metrics such as number of groups and size of largest Louvain group.

Given the importance of triadic relationships in social networks discussed above, we have included several features based on these relationships. These include a full triadic census, number of Simmelian ties, and the clustering coefficient. Calculation of Simmelian ties [46] was not available in the *networkX* package. Our team therefore created a Python implementation of Dekker's version [26] of the original algorithm [46].

4.2 Content Features

We felt we could leverage the large amount of content available from the snowball sample to develop predictive features. This was not done in [10], and was added in recent version of the *bot-hunter* framework.

These features include the number of languages used in the network, as well as some key summary statistics on entities, including mean emojiis, mentions, and hashtags per tweet, as well as the percentage of retweets.

We also wanted to have several measures of similarity of text between the various communicators in the network. This search for similarity measures was motivated by the fact that many bot networks post very similar or conversely very diverse content, and we felt that these measures of similarity may be distinguishing.

To compute similarity, tweet content in the network was aggregated by user. Once aggregated, the content was cleaned and parsed (cleaning included conversion to lower case and removal of punctuation). We did not remove stopwords. The parsed data was then converted to a document term matrix with raw counts (we chose not to normalize the data since the variance on tweet length is artificially constrained to 280 characters). The document term matrix was then used to compute both the Jaccard and Cosine Similarity, which were used as features.

4.3 User Features

The newest version of the Tier 3 classifier also includes several aggregate user attributes that were not leveraged in earlier versions. While many of these are self explanatory, we did want to describe two novel metrics that have not been used before.

Recently, several experts in online disinformation have highlighted how recent online bots seem to produce tweets that are far more popular than the account itself [56]. This phenomena is the result of accounts in large bot-nets that create messages that are then *pushed* by the entire network, resulting in reach that far exceeds expectations given its modest beginning.

To find this phenomena, we devised a simple heuristic that determines if any original (non-retweet) tweet is more popular than its account. This heuristic is defined as:

$$P_{user} = retweets > 2 \times max(followers, friends)$$

where the Boolean measure for a user is defined as $True$ if any tweet receives two time more retweets than the highest value of its in-degree or out-degree. This metric is leveraged in two new features, one at the user level (Tier 2) and one at the Network Level (Tier 3). The user level flags the user if any tweet is flagged as True, and the network metric measures the fraction of tweets produced by the network that are flagged by this heuristic.

4.4 Timing Features

Like user features, most of the temporal features listed in Table 4 are self explanatory. We did develop a heuristic method that measures whether or not an account has daily rhythms. Most human users will have surges in activity based on their daily routines, and will have a measurable drop in activity that aligns with their sleep activity. Bots, on the other hand, do not require these circadian rhythms, and some bots are programmed to produce content spread uniformly across the hours of the day. We developed the heuristic described below to flag these accounts.

To measure whether an account has human circadian rhythms, we first aggregate their tweets by hour of day after ensuring that the account has produced enough data (at least 50 tweets). Given there is sufficient data, we next determine whether this hourly distribution is uniformly distributed by normalizing it and conducting the Kolmogorov-Smirnov non-parametric test for uniformity. A p-value greater than 0.5 provides strong evidence of non-human circadian rhythms.

It is important to note that, while some bots exhibit this lack of circadian rhythm, it only takes a few lines of code for a bot manipulator to give a more realistic temporal pattern. Nonetheless, this remains a strong indicator of bot activity.

5 Modeling

As indicated above, all feature engineering was conducted in Python using several custom Python packages that were developed for the *bot-hunter* framework. These packages build the feature space for Tier 1, Tier2, and Tier 3 models, which is then trained using the steps outlined below.

Table 5 Comparing algorithms for Tier 3 Bot detection

Model	Accuracy	Precision	Recall	AUC	F1
Naïve Bayes	0.562	0.541	0.864	0.563	0.665
Decision tree	0.950	0.949	0.952	0.950	0.951
SVM	0.952	0.969	0.933	0.952	0.952
Logistic regression	0.951	0.940	0.965	0.983	0.952
Random forrest	0.955	0.955	0.956	0.986	0.956

Table 6 Table of results for combined data (Tier 3)

Tier	Accuracy	F1	Precision	Recall	ROC AUC
Tier 1	0.7964	0.7729	0.8677	0.6969	0.8680
Tier 2	0.8335	0.8181	0.8970	0.7522	0.9179
Tier 3	0.8577	0.8478	0.9042	0.7983	0.9410

For training all data sets, human data was sampled so that the classes were balanced. The random forest algorithm was used because of its superior performance on Tier 1 data [11] and its use in other bot detection algorithms [63]. In Table 5 we revisit model comparison in order to verify that the random forest model is still appropriate for Tier 3 feature space. We see that random forest still provides superior performance, and in general is not as computationally expensive as some of the other models. Training, evaluation, and testing were conducted in the *scikit-learn* Python package [57]. Tuning of the Random Forest algorithm was conducted through random search of parameter options while using three fold cross-validation.

The *bot-hunter* behavior returns both a binary classification and an estimate of probability. The estimate of probability is provided by the Random Forest classifier by measuring the proportion of votes by trees in the ensemble. The binary classification result is evaluated by classifying accounts based on a probability threshold of 0.5. The binary classification feature of the results allows researchers to have a consistent threshold to compare results, while the probability allows users to tune a threshold for a given use case.

6 Results

After building the network metrics for all bot data sets as well as the annotated *human* data, we built and evaluated Random Forest models for each of the data sets. Training, evaluating, and testing were conducted at Tier 1, Tier 2, and Tier 3 where possible. We evaluated in-sample performance with 10 fold cross-validation measuring multiple evaluation metrics, which are provided in Table 6 and Fig. 6.

From the results presented in Table 6 and Fig. 6, we see that Tier 1 models continue to provide solid performance, even with basic features extracted from the user profile and last status. We also observe improvement between Tier 1

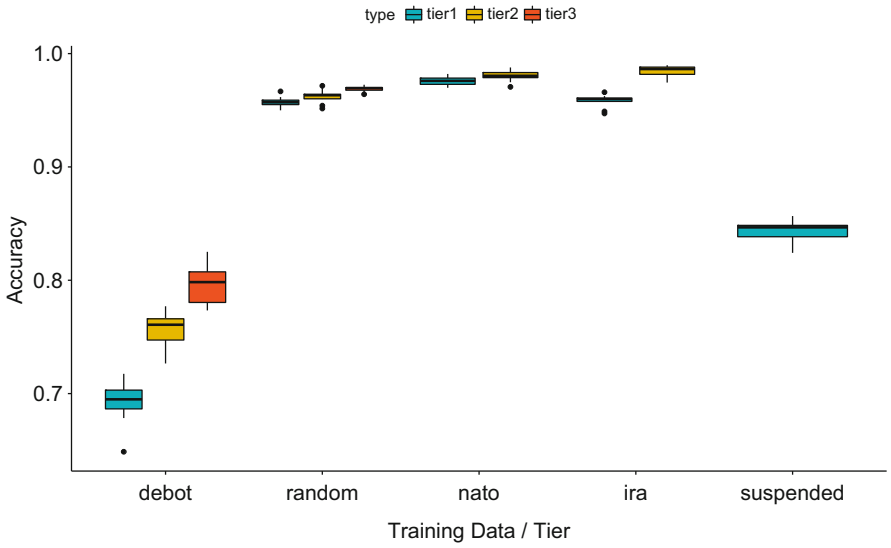

Fig. 6 Results by training data and by *Tier*

and Tier 2 and between Tier 2 and Tier 3 for all models. Using a combined data model we found that the Tier 2 improvement over Tier 1 is statistically significant ($p-\text{value} = 1.303e-10$), as is the Tier 3 improvement over Tier 2 ($p-\text{value} = 1.101e-06$). In Fig. 6 we also see that the Random, NATO, and IRA data provide the highest in sample cross validation performance, while models trained on Debot Data and Suspended data offer lower in data cross validation performance. This likely indicates a wider variety of bot types in the Debot and Suspended data.

Further, in Fig. 7 we see the top features for all Tier 1, Tier 2, and Tier 3 models in the *bot-hunter* suite of tools. These figures represent the percentage that each feature contributed to the model predictions. We see that network features provide strong features in the model. This demonstrates that these values, while tedious to collect, transform, and model, provide strong predictive features that are difficult for bot *puppet master* to manipulate. In these data sets network centrality, network connection, network timing, and network content all provide predictive value.

7 Evaluating Against State of the Art

Given that this is the last Tier of the *bot-hunter* suite of tools, we wanted to evaluate the models as well as various training data that is available. We also wanted to compare the models in the *bot-hunter* suite of tools to existing models, namely the Botometer and Debot models. To do this, we set out to find a test that wasn't biased

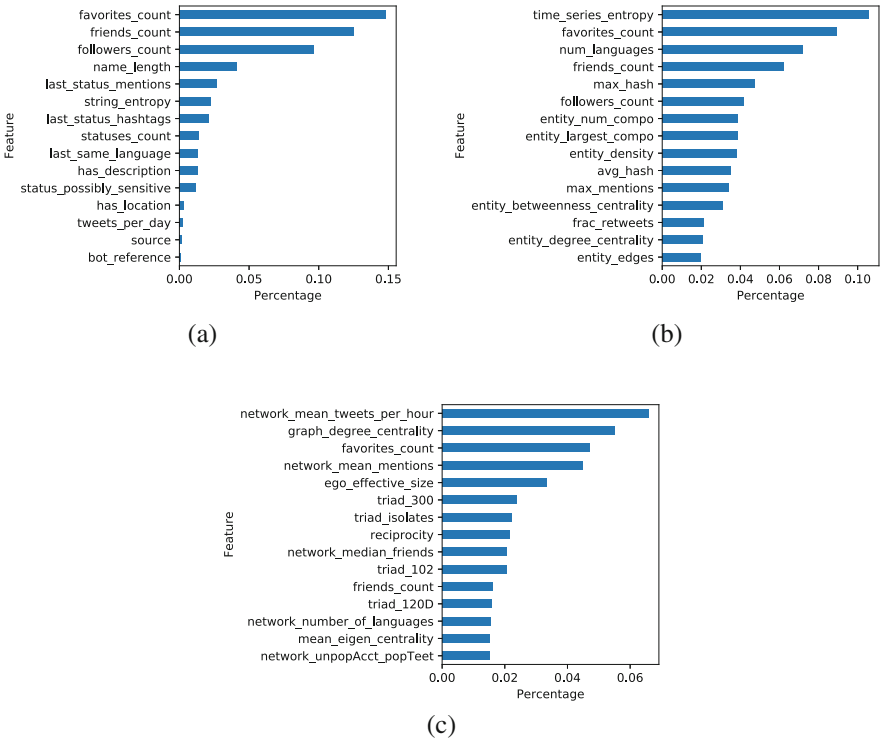

Fig. 7 Comparison of top features for all three Tiers of *bot-hunter*. (**a**) Tier 1 Top Predictive Features. (**b**) Tier 2 Top Predictive Features. (**c**) Tier 3 Top Predictive Features

toward any given model, meaning the test data could not be derived from the training data of any of the models being compared.

To find an unbiased data set, we manually annotated 337 bot accounts. To do this, we started by manually finding several seed bots related to the Swedish elections, separate Russian propaganda bots, and bots found in Middle East conversations. We then manually snowballed out on the followers and followers of followers, manually identifying additional bots. In this evaluation we leveraged the visualizations and metrics provided in the TruthNest tool to aid in making our determination. The TruthNest Tool originally was an EU-funded Reveal project developed to evaluate Twitter accounts for automated activity. While this tool was not evaluated in our test, it was used to assist in labeling bot accounts. TruthNest has instituted a paywall since our use of it. Human users were sampled from the Twitter stream and manually verified. The test data was balanced (337 bots, 337 users).

In evaluating our Tier1, Tier2, and Tier3 models, we also wanted to evaluate which training data and model combination generalizes to new data. Our models were trained on the data and at the tiers described in Table 1. All *bot-hunter* and Botometer thresholds were set at 0.5. F1 performance for all models is provided in Fig. 8 and detailed results are provided in Table 7.

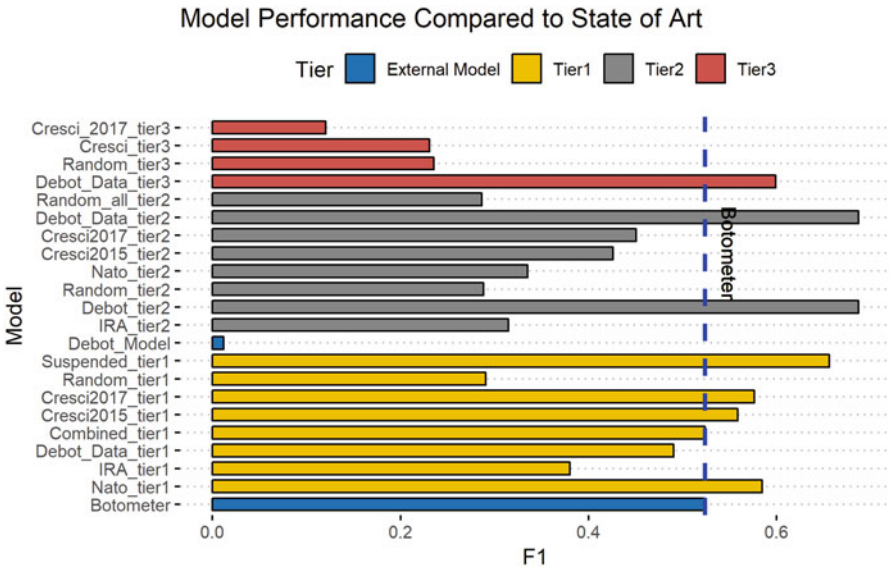

Fig. 8 Results by training data and by *Tier*

Table 7 Detailed results by Tier and training data

Tier	Training data	F1	Accuracy	Precision	Recall	ROC-AUC	TN	FP	FN	TP
Botometer model		0.524	0.657	0.858	0.377	0.587	256	55	200	108
Debot model		0.012	0.502	1.000	0.006	0.503	336	0	335	2
Tier1	NATO	0.584	0.634	0.678	0.513	0.635	254	82	164	173
Tier1	IRA	0.380	0.597	0.830	0.246	0.598	319	17	254	83
Tier1	Combined	0.524	0.657	0.858	0.377	0.657	315	21	210	127
Tier1	Cresci2015	0.559	0.404	0.444	0.754	0.404	18	318	83	254
Tier1	Cresci2017	0.576	0.419	0.454	0.789	0.418	16	320	71	266
Tier1	Debot	0.490	0.527	0.533	0.454	0.528	202	134	184	153
Tier1	Random	0.291	0.572	0.855	0.175	0.573	326	10	278	59
Tier1	Suspended	0.656	0.713	0.821	0.546	0.713	296	40	153	184
Tier2	IRA	0.315	0.567	0.903	0.191	0.584	305	7	276	65
Tier2	Random	0.288	0.547	0.800	0.176	0.564	297	15	281	60
Tier2	NATO	0.335	0.574	0.909	0.205	0.591	305	7	271	70
Tier2	Cresci2015	0.426	0.596	0.824	0.287	0.610	291	21	243	98
Tier2	Cresci2017	0.451	0.600	0.799	0.314	0.614	285	27	234	107
Tier2	Debot	0.687	0.675	0.691	0.683	0.675	208	104	108	233
Tier2	Random	0.286	0.550	0.831	0.173	0.567	300	12	282	59
Tier3	Debot	0.599	0.674	0.837	0.466	0.683	281	31	182	159
Tier3	Random	0.236	0.533	0.810	0.138	0.551	301	11	294	47
Tier3	Cresci2015	0.231	0.541	0.918	0.132	0.560	308	4	296	45
Tier3	Cresci2017	0.120	0.507	0.880	0.065	0.527	309	3	319	22

In these results we first see Botometer demonstrates consistent solid performance in predicting new bots across all metrics. The Debot algorithm provides high precision but extremely low recall, resulting in a low F1 score overall. The value of the Debot algorithm may indirectly lie in the data that it produces. Note that *bot-hunter* algorithms trained on Debot data performed well at all three Tiers, meaning that the Debot algorithm for finding correlated accounts produces great labeled data for other supervised bot detection endeavors.

For the bot-hunter family of models, we see that Tier 1 consistently performs well and seems to generalize to new data better than Tier 2 and Tier 3. Tier 2 still has high performance, given its ability to identify anomalies in content and in temporal statistics. Across the data sets, Tier 1 has a higher mean Accuracy and ROC AUC than Tier 1. Tier 3 has very high precision but low recall. It therefore produces predictions that are more reliable, but fails to find a large portion of the bots in the data. Additionally, this model may become increasingly important in identifying sophisticated emerging bots.

As we look at the various training data used for training these models, we see that the models trained on suspended accounts or on data produced by the Debot model had the highest performance. As indicated earlier, this is likely due to these data sets containing a wide variety of bot "genres." We also see that the NATO data captured in the deliberate attack against NATO and the DFR labs continues to provide strong performance across all metrics. We found that few of the annotated data sets released by other researchers provided strong performance, especially when considering accuracy and ROC-AUC metrics. The Cresci data (both 2015 and 2017) appears to have high recall but low precision, with many false negatives. The models trained on the random string data also have low accuracy and ROC-AUC metrics, in this case caused by high precision but low recall. These random string accounts probably represent a limited band in the spectrum of bot types, and therefore do not generalize well to new data and different bot types.

The Venn Diagram of predicted bots is provided in Fig. 9a. This diagram shows the overlap of the predicted bots, but does not provide any information on predicted humans. We see significant overlap for all three models. We also notice that the Tier 2 model predicted the most accounts (330 accounts), while Tier 1 predicted 260 accounts and Tier 3 predicted 183 bot accounts. The 95 accounts in the intersection contain 20 false positives (78.9% precision).

The Venn Diagram of predicted bots for Tier 1 and 2 compared to the *real* labeled bots is provided in Fig. 9b. This shows that Tier 2 is adding something to Tier 1, finding 94 additional accounts while only missing 21 of the accounts that Tier 1 found.

Figure 9c provides an *upset* visualization to fully explore the intersection of sets. This visualization demonstrates that our largest intersection is the intersection of all four sets. We also see in the upset graph the Tier 1 and in Particular Tier 2 is important to the prediction success, thought Tier 3 is also able to find 32 accounts that neither Tier 1 or 2 could find. These visualizations illustrate the importance of having a tool-box of models that can be used for predicting bots in any given scenario.

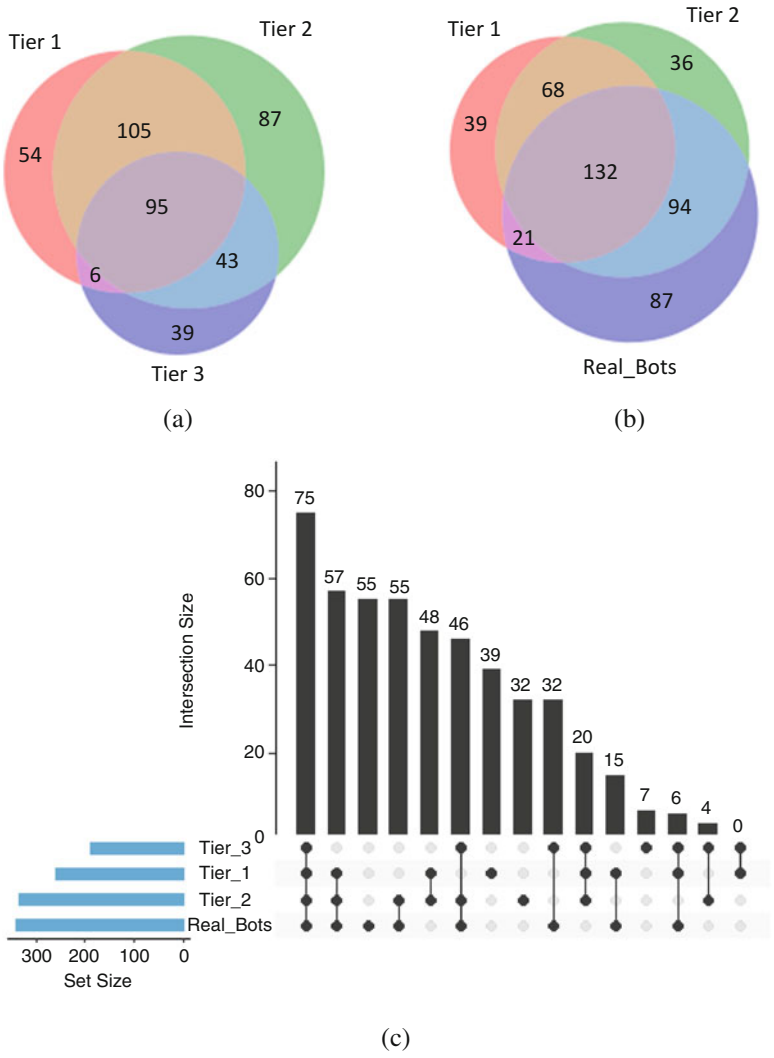

Fig. 9 Understanding the overlap of predicted Bots with Tier 1, 2, and 3 models trained on Debot data. (**a**) Predicted Bots (Tier 1, 2, and 3) . (**b**) Predicted Bots (Tier 1 and 2) with Real Labeled Bots. (**c**) Upset Plot with Predicted Bots (Tier 1, 2, and 3) and Real Labeled Bots

While we believe this evaluation is informative, there are several limitations in our evaluation method. We aknowledge that we were not able to completely remove bias, given that the mental heuristics we used to manually annotate accounts may have unintentionally mirrored the bot-hunter algorithms. Additionally, we acknowledge that the test set is still modest in size and, while somewhat diverse, does not represent the full spectrum of bot types. Finally, we acknowledge that any

given model may perform better if the threshold is tuned for a given data set. Even with these limitations, we believe this test and evaluation is informative for our team and for the greater community.

7.1 Evaluating Bot Classification Thresholds

The random forest model used in the bot-hunter suite of tools (and Botometer) provides a probability estimate rather than just a label. This allows researchers to estimate how strong a given prediction is. Every use case will require the analyst to determine the best threshold for establishing whether or not an account is likely a bot. To evaluate the best threshold for a given data set, a research team should explore several thresholds, each time sampling 50–100 accounts and manually labeling them to estimate a rate of true/false positives, true/false negatives. If possible they should attempt to construct a precision recall curve and/or ROC Curve, as demonstrated in Fig. 10 using the Suspended, NATO, and Botometer models. Note that recall is always monotonically decreasing, but precision is not required to monotonically increase.

As seen in Fig. 10, we generally recommend bot-hunter thresholds between 0.6 and 0.8. The exact choice in this range will need to be made by the research team, and is dependent on the data as well as the team's prioritization of precision vs. recall.

8 Applying Bot Detection to Swedish Election

Having completed the bot-hunter suite of tools, we wanted to leverage this toolbox in analyzing a stream of data from the 2018 Swedish elections. This is done as a case study to illustrate that bot-detection is not a "turn-key" solution, and also to provide practitioners with an example of an open source intelligence workflow.

Sweden held national elections on 9 September 2018 for its equivalent of a Parliament, known as the Riksdag. Swedish elections have historically lacked much drama or suspense, with the center-left Social Democrat Party dominating politics since 1914. In the 2018 election, however, their dominance was challenged by various nationalistic factions that capitalized on anti-immigrant sentiment.

Some of the political discourse surrounding the election transpired on Twitter, as seen in many recent national elections across the world. As this discourse grew, multiple researchers and news agencies saw rising disinformation and associated bot activity [58]. Simultaneously, the Swedish Defence Research Agency reported increased bot activity, primarily supporting right leaning, nationalistic, and anti-immigrant views [60].

As these bots grew in activity in this marketplace of beliefs and ideas, our team began collecting and analyzing streams from this discourse. To collect Twitter

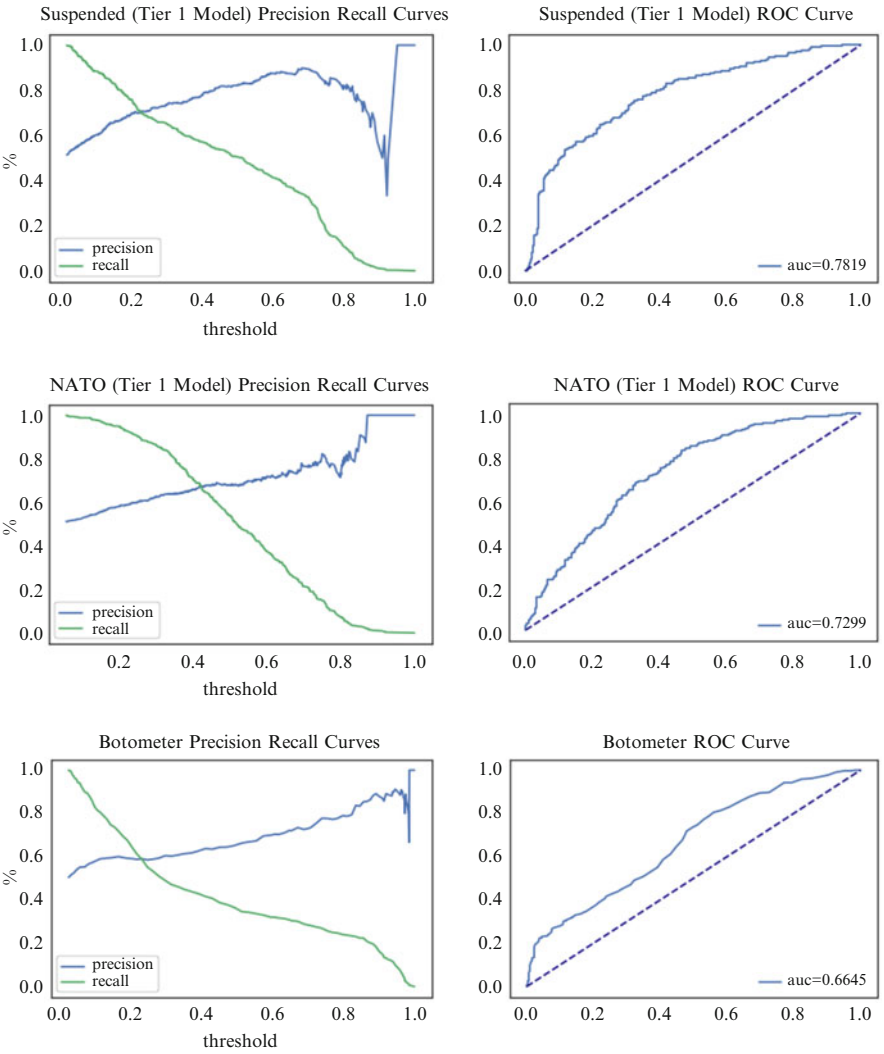

Fig. 10 Using precision-recall curves and ROC curves to determine threshold

data around the Swedish National elections discourse, our team leveraged a spiral collection methodology, starting with content and geographic streaming, and then 'spiraling' into more thorough data collection around the important parts of the discussion. All collection was done through the Twitter Streaming and REST API's using the Tweepy Python Package.

We started by identifying Swedish political hashtags through open source research, eventually identifying #svpol, #Val2018, #feministisktInitiativ, #migpol, #valet2018, #SD2018, #AfS2018, and #MEDval18. These hashtags were not

selected because they cover the full spectrum of Swedish politics, but rather because there was open source reporting of some bot campaigns using these hashtags. We started collecting on these hashtags using both the Streaming and REST API's (the streaming API allows us to easily collect going forward while the REST API allows us to retroactively collect past data). Simultaneously we collected data that was 'geo-associated' with the Scandinavian peninsula, using a bounding box search method.

As we began to collect content and geo-referenced data, we monitored other trending hashtags and added them to the collection query. After launching the exploratory data analysis discussed below, we would also collect users friend and follower relationships as well as user historical timelines for accounts of interest. This continual return to the Twitter API creates the spiral nature of our collection process.

For the Swedish Election Event we collected 661,317 tweets produced by 88,807 unique users. This creates a political *conversation* that contains 104,216 nodes, 404,244 links with a density of 0.000037.

For bot detection in the Swedish Election stream our team found that a 65% probability was appropriate. Given that we were performing this evaluation on 104,216 nodes, we used the Tier 1 model. This model is our best model for getting an accurate prediction on high volume of accounts.

Note that we usually conduct other data enhancement as well, including sentiment analysis with NetMapper as well as geo-inference based on [44]. All enrichments are made available in easy formats that allow tools to merge them with existing event data.

8.1 Exploratory Data Analysis

Our exploratory data analysis focuses on narratives, time, place, groups, and individuals. Our analysis typically starts with some type of temporal analysis. This allows us to see distributions over time. We try to look at overall temporal distribution, bot activity over time, as well as changing narratives over time (Fig. 11).

Our exploration of content and narratives starts with analysis of words and hashtags across the entire corpus, and then we explore narratives associated with topic groups (these are groups that talk about the same thing but may not be connected in the social network or conversational network) and social network group (these are groups that are connected, but may not talk about the same thing). We leverage latent dirichlet allocation [15] for topic group analysis, and content analysis by Louvain group [16] as a way to "triage" network groups. Table 8 provides the top 8 words by Louvain Group for the Swedish elections. In this we already start to see groups that are focused on immigration, particularly immigration from Muslim countries. We also see at least one group that is mixing conversation about religious beliefs with political discourse. Finally and just as

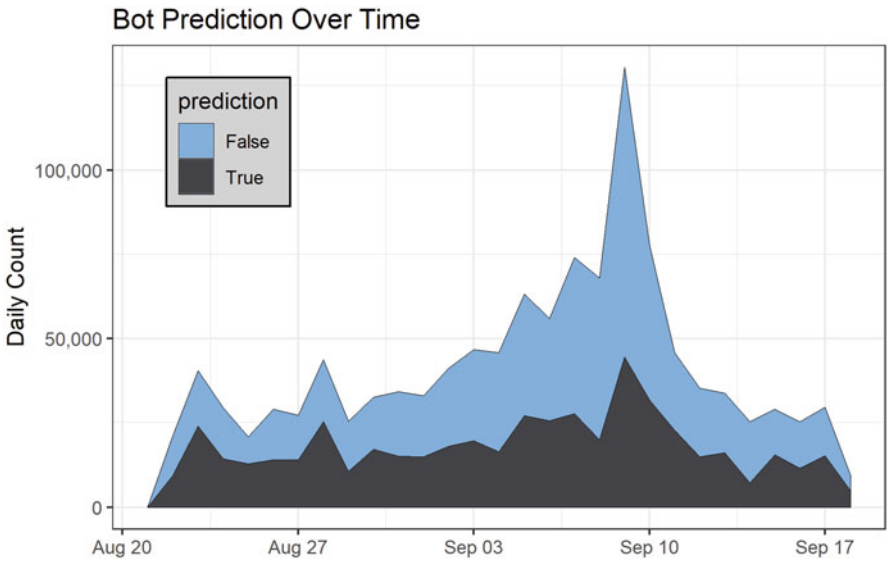

Fig. 11 Bots as a proportion of total volume over time

important, "triaging" the data like this allows us to identify groups like Group 0 that don't appear to have any topics of interest.

Network analysis of groups and individuals is done almost exclusively in the ORA Network Analysis Tool. We typically start by visualizing a reduced conversational network. Nodes in this network represent Twitter accounts, and links represent a conversational action in the Twitter ecosystem (reply, retweet, mention). These network are typically too large to visualize, so we reduce the network by taking the K-core so that we have the core 15,000–20,000 nodes. Once this is done, we color the network by *bot* or *human*, by language, and by Louvain grouping (see Fig. 12). This coloration helps us better understand the groups and their relationship to each other. Finally, we reduce the network to only include reciprocal links. This usually reduces the network significantly, and in Twitter provides the best proxy for a true social network.

We then explore the influential accounts and influential bots in the network. The ORA Network analysis tool provides several reports that analyze nodes by a variety of centrality measures, and assists translating their role in the network. For the Swedish network, we found several bots with high *betweenness*, indicating that these bots were influential in that they connect individuals and groups. With further exploration, it appeared that these bots, in connection with other accounts, were trying to bridge several communities with nationalistic and anti-immigrant groups and narratives.

We leverage the *bot-hunter* Tier 2 and Tier 3 models during this phase of analysis. As we identify influential accounts, we check them in a Tier 2/3 *bot-hunter* web application that allows us to thoroughly explore the account and conduct a more

Table 8 Content analysis by Louvain group

Group	# Tweets	# Nodes	Top 8 words by Louvain group			
0	15,708	4675	Video 2018	Gillade fortnite	Lade world	Spellista part
1	31,059	5688	Country n	Voters number	Refugees capita	82 reported
2	102,146	14,538	Sweden Swedish	Election results	epp left	sd poll
3	306,352	17,600	m6aubkudbg sverige	Jesus gud	Kristus namn	Varnar fader
4	8353	3137	Sweden amp	Swedish vote	Muslim democrats	Election gang
5	40,585	9110	Sverige valet	sd jimmie	Svenska år	åkesson svt
7	82,708	12,300	sd valet	Sverige jimmie	Rösta parti	åkesson val
8	17,675	4000	sd politik	Friend claeson	American tånkt	Rösta frågar
9	7144	5217	Sverige stefan	Löfven kristersson	sd amp	Moderaterna rösta
10	7569	5214	Sverige afs	Riks Sweden	sd svenska	Alternativ hahne

accurate Tier 2 or Tier 3 bot prediction. These applications also allow us to explore in depth visualizations of the activity of the account.

Bot-detection is therefore a part of the overall open source intelligence workflow, trying to identify relevant information about how the world works to inform decision maker situational understanding and decisive action. In this case, our research validated research of large bot activity within the Swedish political discourse on Twitter and provided identification of narratives (primarily nationalistic and anti-immigrant, anti-Muslim, and some anti-Semitism). We were also able to identify influential accounts that were attempting to connect individuals and online communities with extremist content. This type of information informs leaders of current dis-information strategies allowing them to better prepare their government and their populace for similar disinformation campaigns in their country.

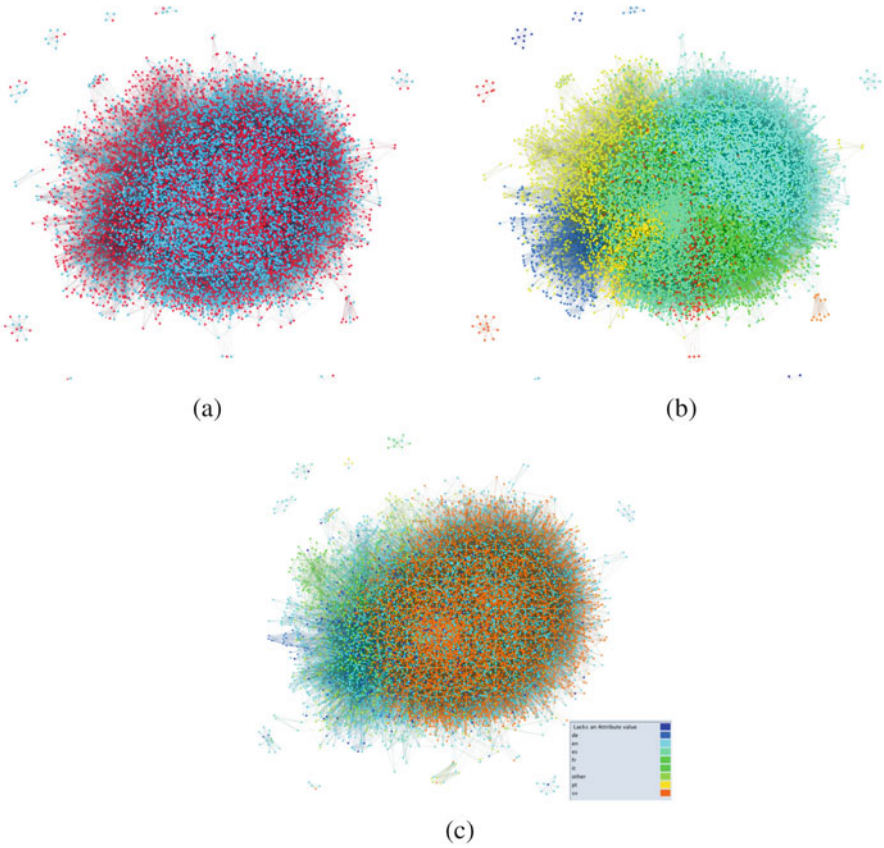

Fig. 12 Exploring the Twitter conversational network surrounding online discourse on Swedish politics. (**a**) Bots (red) in conversation. (**b**) Louvain Groups. (**c**) Language Distribution in Network

9 Conclusion and Future Work

In our pursuit of a multi-model bot detection toolbox, this paper builds on past research by adding a model that leverages a feature space extracted from 50,000+ entities collected with single seed snowball sampling. This model is developed for high accuracy but low volume applications. Our research shows that supervised machine learning models are able to leverage these rich structural, content, and temporal features associated with the target ego-network to increase model precision. Additionally, these network features offer an approach for modeling and detecting bot behavior that is difficult for bot *puppet-masters* to manipulate and evade.

Our evaluation of the bot-hunter suite of tools demonstrates that these models provide performance equivalent to or better than the state of the art. The Tier 1 model in particular is valuable to the community because it is accurate and

can scale to large data (meaning researchers aren't required to sample their data). Additionally, because the Tier 1 model was designed to predict existing data, there isn't a requirement to return to the Twitter API to re-collect account data. This also means that it can be used to predict existing data sets that contain suspended or otherwise missing accounts.

Our analysis of Swedish political discourse on Twitter illustrates how bot-detection tools can support a typical open source intelligence workflow. The bot-hunter suite provides a way to enrich the data which can then be imported into other analysis tools for visualization and further analysis. Bot detection is not a "turn-key" solution, and does require some work to set the right parameters, particularly the appropriate threshold level.

Future work will focus on creating a labeling methodology that will allow us to better characterize bot accounts and the various methods they employ. Binary prediction assists in understanding fake versus real, but does not help us in triaging the hundreds of thousands of bot accounts that exist. Some spam content, others intimidate users. Developing heuristics to label these methods and attributes is essential for characterizing these accounts and the disinformation campaigns they propagate.

Acknowledgements This work was supported in part by the Office of Naval Research (ONR) Multidisciplinary University Research Initiative Award N000140811186 and Award N000141812108, the Army Research Laboratory Award W911NF1610049, Defense Threat Reductions Agency Award HDTRA11010102, and the Center for Computational Analysis of Social and Organization Systems (CASOS). The views and conclusions contained in this document are those of the authors and should not be interpreted as representing the official policies, either expressed or implied, of the ONR, ARL, DTRA, or the U.S. government.

References

1. K.S. Adewole, N.B. Anuar, A. Kamsin, K.D. Varathan, S.A. Razak, Malicious accounts: dark of the social networks. J. Netw. Comput. Appl. **79**, 41–67 (2017)
2. L.M. Aiello, M. Deplano, R. Schifanella, G. Ruffo, People are strange when you're a stranger: Impact and influence of bots on social networks. Links **697**(483,151), 1–566 (2012)
3. A. Almaatouq, E. Shmueli, M. Nouh, A. Alabdulkareem, V.K. Singh, M. Alsaleh, A. Alarifi, A. Alfaris, et al., If it looks like a spammer and behaves like a spammer, it must be a spammer: analysis and detection of microblogging spam accounts. Int. J. Inf. Secur. **15**(5), 475–491 (2016)
4. A. Ananthalakshmi, *Ahead of Malaysian Polls, Bots Flood Twitter with Pro-government...* (2018)
5. D.A Bader, S. Kintali, K. Madduri, M. Mihail, Approximating betweenness centrality, in *International Workshop on Algorithms and Models for the Web-Graph* (Springer, Berlin, 2007), pp. 124–137
6. A. Bavelas, A mathematical model for group structures. Hum. Organ. **7**(3), 16–30 (1948)
7. A.I. Bawaba, The Loop, in *Thousands of Twitter Bots are Attempting to Silence Reporting on Yemen* (2017)

8. M. Benigni, K.M. Carley, From tweets to intelligence: understanding the islamic jihad supporting community on twitter, in *Proceedings of Social, Cultural, and Behavioral Modeling: 9th International Conference, SBP-BRiMS 2016, Washington, DC, USA, June 28-July 1, 2016* (Springer, Berlin, 2016), pp. 346–355
9. M.C. Benigni, K. Joseph, K.M. Carley, Online extremism and the communities that sustain it: detecting the ISIS supporting community on twitter. PLoS One **12**(12), e0181405 (2017)
10. D. Beskow, K.M. Carley, Bot conversations are different: Leveraging network metrics for bot detection in twitter, in *2018 International Conference on Advances in Social Networks Analysis and Mining (ASONAM)* (IEEE, Piscataway, 2018), pp. 176–183
11. D. Beskow, K.M. Carley, Introducing bothunter: A tiered approach to detection and characterizing automated activity on twitter, in *International Conference on Social Computing, Behavioral-Cultural Modeling and Prediction and Behavior Representation in Modeling and Simulation*, ed. by H. Bisgin, A. Hyder, C. Dancy, R. Thomson (Springer, Berlin, 2018)
12. D. Beskow, K.M. Carley, Using random string classification to filter and annotate automated accounts, in *International Conference on Social Computing, Behavioral-Cultural Modeling and Prediction and Behavior Representation in Modeling and Simulation*, ed. by H. Bisgin, A. Hyder, C. Dancy, R. Thomson (Springer, Berlin, 2018)
13. A. Bessi, E. Ferrara, *Social Bots Distort the 2016 US Presidential Election Online Discussion* (2016)
14. S.Y. Bhat, M. Abulaish, Community-based features for identifying spammers in online social networks, in *2013 IEEE/ACM International Conference on Advances in Social Networks Analysis and Mining (ASONAM)* (IEEE, Piscataway, 2013), pp. 100–107
15. D.M. Blei, A.Y. Ng, M.I. Jordan, Latent dirichlet allocation. J. Mach. Learn. Res. **3**(Jan), 993–1022 (2003)
16. V.D. Blondel, J.-L. Guillaume, R. Lambiotte, E. Lefebvre, Fast unfolding of communities in large networks. J. Stat. Mech: Theory Exp. **2008**(10), P10008 (2008)
17. R.S. Burt, *Structural Holes: The Social Structure of Competition* (Harvard University Press, Cambridge, 2009)
18. K.M. Carley, G. Cervone, N. Agarwal, H. Liu, Social cyber-security, in *International Conference on Social Computing, Behavioral-Cultural Modeling and Prediction and Behavior Representation in Modeling and Simulation*, ed. by H. Bisgin, A. Hyder, C. Dancy, R. Thomson (Springer, Berlin, 2018)
19. D. Cartwright, F. Harary, Structural balance: a generalization of Heider's theory. Psychol. Rev. **63**(5), 277 (1956)
20. N. Chavoshi, H. Hamooni, A. Mueen, Debot: Twitter bot detection via warped correlation, in *IEEE International Conference on Data Mining (ICDM)* (2016), pp. 817–822
21. N. Chavoshi, H. Hamooni, A. Mueen, On-demand bot detection and archival system, in *Proceedings of the 26th International Conference on World Wide Web Companion*. International World Wide Web Conferences Steering Committee (2017), pp. 183–187
22. C.-M. Chen, D.J. Guan, Q.-K. Su, Feature set identification for detecting suspicious urls using bayesian classification in social networks. Inform. Sci. **289**, 133–147 (2014)
23. S. Cresci, R. Di Pietro, M. Petrocchi, A. Spognardi, M. Tesconi, Fame for sale: efficient detection of fake twitter followers. Decis. Support. Syst. **80**, 56–71 (2015)
24. S. Cresci, R. Di Pietro, M. Petrocchi, A. Spognardi, M. Tesconi, Social fingerprinting: detection of spambot groups through DNA-inspired behavioral modeling. IEEE Trans. Dependable Secure Comput. **15**(4), 561–576 (2018)
25. C.A. Davis, O. Varol, E. Ferrara, A. Flammini, F. Menczer, Botornot: a system to evaluate social bots, in *Proceedings of the 25th International Conference Companion on World Wide Web*. International World Wide Web Conferences Steering Committee (2016), pp. 273–274
26. D.J. Dekker, Measures of Simmelian Tie Strength, Simmelian Brokerage, and, the Simmelianly Brokered (2006)
27. R.I.M. Dunbar, Coevolution of neocortical size, group size and language in humans. Behav. Brain Sci. **16**(4), 681–694 (1993)

28. E. Ferrara, Measuring social spam and the effect of bots on information diffusion in social media (2017). arXiv preprint:1708.08134
29. S. Fortunato, Community detection in graphs. Phys. Rep. **486**(3–5), 75–174 (2010)
30. O. Frank, Sampling and estimation in large social networks. Soc. Networks **1**(1), 91–101 (1978)
31. L.C. Freeman, Centrality in social networks conceptual clarification. Soc. Networks **1**(3), 215–239 (1978)
32. L.C. Freeman, Centered graphs and the structure of ego networks. Math. Soc. Sci. **3**(3), 291–304 (1982)
33. K. Gani, H. Hacid, R. Skraba, Towards multiple identity detection in social networks, in *Proceedings of the 21st International Conference on World Wide Web* (ACM, New York, 2012), pp. 503–504
34. M. Gjoka, M. Kurant, C.T. Butts, A. Markopoulou, Practical recommendations on crawling online social networks. IEEE J. Sel. Areas Commun. **29**(9), 1872–1892 (2011)
35. A. Glaser, *Russian Bots are Trying to Sow Discord on Twitter After Charlottesville* (2017)
36. L.A. Goodman, Snowball sampling, in *The Annals of Mathematical Statistics* (1961), pp. 148–170
37. M.S. Granovetter, The strength of weak ties, in *Social Networks* (Elsevier, Amsterdam, 1977), pp. 347–367
38. C. Grimme, M. Preuss, L. Adam, H. Trautmann, Social bots: human-like by means of human control? Big Data **5**(4), 279–293 (2017)
39. A. Hagberg, P. Swart, D.S. Chult, *Exploring Network Structure, Dynamics, and Function Using Networkx* (Los Alamos National Lab.(LANL), Los Alamos, 2008). Technical report
40. F. Heider, Attitudes and cognitive organization. J. Psychol. **21**(1), 107–112 (1946)
41. P.W. Holland, S. Leinhardt, Transitivity in structural models of small groups. Compar. Group Stud. **2**(2), 107–124 (1971)
42. P.W. Holland, S. Leinhardt, A method for detecting structure in sociometric data, in *Social Networks* (Elsevier, Amsterdam, 1977), pp. 411–432
43. P.N. Howard, B. Kollanyi, Bots,# strongerin, and# brexit: computational propaganda during the uk-eu referendum, in *Browser Download This Paper* (2016)
44. B. Huang, K.M. Carley, On predicting geolocation of tweets using convolutional neural networks, in *International Conference on Social Computing, Behavioral-Cultural Modeling and Prediction and Behavior Representation in Modeling and Simulation* (Springer, Berlin, 2017), pp. 281–291
45. L. Katz, A new status index derived from sociometric analysis. Psychometrika **18**(1), 39–43 (1953)
46. D. Krackhardt, The ties that torture: Simmelian tie analysis in organizations. Res. Sociol. Organ. **16**(1), 183–210 (1999)
47. S. Kudugunta, E. Ferrara, Deep neural networks for bot detection (2018). arXiv preprint:1802.04289
48. S. Lee, J. Kim, Early filtering of ephemeral malicious accounts on twitter. Comput. Commun. **54**, 48–57 (2014)
49. K. Lee, J. Caverlee, S. Webb, Uncovering social spammers: social honeypots+ machine learning, in *Proceedings of the 33rd International ACM SIGIR Conference on Research and Development in Information Retrieval* (ACM, New York, 2010), pp. 435–442
50. K. Lee, B.D. Eoff, J. Caverlee, Seven months with the devils: a long-term study of content polluters on twitter, in *ICWSM* (2011)
51. D. Liu, B. Mei, J. Chen, Z. Lu, X. Du, Community based spammer detection in social networks, in *International Conference on Web-Age Information Management* (Springer, Berlin, 2015), pp. 554–558
52. C. Lumezanu, N. Feamster, H. Klein, # bias: Measuring the tweeting behavior of propagandists, in *Sixth International AAAI Conference on Weblogs and Social Media* (2012)
53. Z. Miller, B. Dickinson, W. Deitrick, W. Hu, A.H. Wang, Twitter spammer detection using data stream clustering. Inf. Sci. **260**, 64–73 (2014)

54. A. Mislove, M. Marcon, K.P. Gummadi, P. Druschel, B. Bhattacharjee, Measurement and analysis of online social networks, in *Proceedings of the 7th ACM SIGCOMM conference on Internet measurement* (ACM, New York, 2007), pp. 29–42
55. L.M. Neudert, B. Kollanyi, P.N. Howard, *Junk News and Bots During the German Federal Presidency Election: What Were German Voters Sharing Over Twitter?* (2017)
56. B. Nimmo, *#botspot: Twelve Ways to Spot a Bot—dfrlab—medium* (2017). https://medium.com/dfrlab/botspot-twelve-ways-to-spot-a-bot-aedc7d9c110c (Accessed on 11/03/2018).
57. 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. J. Mach. Learn. Res. **12**, 2825–2830 (2011)
58. J. Stubbs, J. Ahlander, *Exclusive: Right-Wing Sites Swamp Sweden with 'Junk News' in Tight Election Race | Reuters* (2018). https://www.reuters.com/article/us-sweden-election-disinformation-exclus/exclusive-right-wing-sites-swamp-sweden-with-junk-news-in-tight-election-race-idUSKCN1LM0DN (Accessed on 11/20/2018).
59. V.S. Subrahmanian, A. Azaria, S. Durst, V. Kagan, A. Galstyan, K. Lerman, L. Zhu, E. Ferrara, A. Flammini, F. Menczer, The darpa twitter bot challenge. Computer **49**(6), 38–46 (2016)
60. Swedish Defence Reserch Agency, Antalet botar på twitter ökar inför valet—totalförsvarets forskningsinstitut (2018). https://www.foi.se/press--nyheter/nyheter/nyhetsarkiv/2018-08-29-antalet-botar-pa-twitter-okar-infor-valet.html (Accessed on 11/20/2018)
61. K. Thomas, C. Grier, D. Song, V. Paxson, Suspended accounts in retrospect: an analysis of twitter spam, in *Proceedings of the 2011 ACM SIGCOMM conference on Internet measurement conference* (ACM, New York, 2011), pp. 243–258
62. Twitter. *Elections Integrity Data Archive*. https://about.twitter.com/en_us/values/elections-integrity.html#us-elections (Accessed on 03/30/2019)
63. O. Varol, E. Ferrara, C.A. Davis, F. Menczer, A. Flammini, Online human-bot interactions: Detection, estimation, and characterization (2017). arXiv preprint:1703.03107
64. J.-P. Verkamp, M. Gupta, Five incidents, one theme: Twitter spam as a weapon to drown voices of protest, in *Presented as part of the 3rd USENIX Workshop on Free and Open Communications on the Internet* (2013).
65. B. Viswanath, M.A. Bashir, M. Crovella, S. Guha, K.P. Gummadi, B. Krishnamurthy, A. Mislove, Towards detecting anomalous user behavior in online social networks, in *USENIX Security Symposium* (2014), pp. 223–238
66. G. Wang, M. Mohanlal, C. Wilson, X. Wang, M. Metzger, H. Zheng, B.Y. Zhao, Social turing tests: crowdsourcing sybil detection (2012). arXiv preprint:1205.3856
67. H. Yu, M. Kaminsky, P.B. Gibbons, A. Flaxman, Sybilguard: defending against sybil attacks via social networks, in *ACM SIGCOMM Computer Communication Review*, vol. 36 (ACM, New York, 2006), pp. 267–278